Hewlett Packard
Enterprise

MapR 6.1 Documentation

MapR Converged Data Platform

MapR 6.1.x

Contents

last-updated: Apr 11, 2022

last-updated: Apr 11, 2022

last-updated: Apr 11, 2022

last-updated: Apr 11, 2022

last-updated: Apr 11, 2022

last-updated: Apr 11, 2022

last-updated: Apr 11, 2022

last-updated: Apr 11, 2022

last-updated: Apr 11, 2022

last-updated: Apr 11, 2022

last-updated: Apr 11, 2022

About MapR 6.1

This site contains the main documentation for Version 6.1 of the MapR Converged Data Platform, including installation, configuration, administration, and reference information.

Related concepts

Products Covered in the MapR Data Platform Documentation on page 5390

This section lists the products covered in the MapR Data Platform documentation portal and provides links to the related product documentation.

Version 6.1.0 Release Notes

These release notes contain information about Version 6.1.0 of the MapR Converged Data Platform.

CAUTION: New installations of MapR 6.1.0 are no longer recommended. Because of a known issue with MapR 6.1.0, HPE recommends installing MapR 6.1.1 or MapR 6.2.0 for new installations. The MapR Installer no longer supports installing MapR 6.1.0 and automatically installs MapR 6.1.1. To review the support advisory announcing release 6.1.1, see Announcement of HPE Ezmeral Data Fabric maintenance release 6.1.1 to address 6.1.0 critical defects.

Upgrading to MapR 6.1.0 also is no longer recommended. If you need to upgrade to MapR 6.1.x, upgrade to MapR 6.1.1. See Installation and Upgrade Notes (MapR 6.1.1) on page 74.

If your cluster is currently installed with MapR 6.1.0, HPE recommends that you install the latest 6.1.0 patch. See Downloading a Patch on page 437.

What's New in Version 6.1.0

The 6.1.0 MapR release supplies substantial new features for the components of the data platform.

To see new features delivered as part of the MapR Ecosystem Pack, see What's New in EEP 6.0.0 on page 6532 There are multiple new features for MapR-Drill. See the Drill release notesfor details.

MapR 6.1.0 Is "Secure by Default"

Because MapR 6.1.0 in its default configuration is more secure than MapR 6.0.1, documentation references to "built-in security" have been changed to "secure by default." See Security for Ecosystem Components on page 702.

Streaming Security and Critical Data Asset Protection

ZooKeeper Upgraded	MapR 6.1.0 includes ZooKeeper 3.4.11 (upgraded from Zookeeper 3.4.5 in MapR 6.0.1) to support ZooKeeper server-to-server authentication.
Zookeeper Supports Server-to-Server Authentication	As of MapR 6.1.0, ZooKeeper is automatically configured for server-to-server authentication with new installations of MapR Core. The following ZooKeeper security parameters are set to "true" whenever you use `configure.sh` to perform a new installation (including when `configure.sh` is invoked by the MapR Installer or MapR Installer Stanzas). • `quorum.auth.enableSasl` • `quorum.auth.learnerRequireSasl`

last-updated: Apr 11, 2022

- `quorum.auth.serverRequireSasl`

These parameters enable secure communication between peer servers in the ZooKeeper quorum using SASL.

Simplified Development and Deployment of AI and Analytics Applications

Lightweight Client Architecture

EEP 6.0.0 introduces Node.js and Python OJAI clients that enable you to write MapR Database JSON applications using a language other than Java. These clients use the MapR Data Access Gateway to access your MapR cluster. The Data Access Gateway performs some data processing that otherwise runs in the client. This keeps the clients lightweight and simplifies their installation and use.

See Understanding the MapR Data Access Gateway on page 750 for more details about the Data Access Gateway. See What's New in EEP 6.0.0 on page 6532 for more details about the new clients.

New Features in MapR Filesystem

Storage Tiers Support for Files

MapR Filesystem v6.1.0 includes rule-based automated tiering functionality for businesses looking to leverage low-cost storage solutions either on low-cost hardware or on the cloud to gain limitless storage capacity. MapR's tiering functionality can seamlessly integrate with the following:

- Low-cost hardware to store data that is less frequently accessed, which is referred to as "warm" data
- Cloud resources to store data that is rarely accessed or archived, which is referred to as "cold" data

Enabling warm or cold tiering allows you to use valuable on-premise storage resources for more active or "hot" file data and applications. You can use "warm" or "cold" tiering for file data retained for compliance, historical, or other business reasons.

See Data Tiering on page 469 for more information.

NFSv4 Protocol Support

As of MapR v6.1, the MapR Filesystem includes support for NFSv4 protocol. NFSv4 provides end-to-end secure filesystem access and a much higher throughput compared to NFSv3, improving performance. MapR uses NFS Ganesha, which is an Open Source userspace implementation of the NFS server, for supporting NFSv4 features. See Accessing Data with NFS v4 on page 1193 for more information.

Also supported are mixed-mode NFS configurations in which some nodes of a cluster use NFSv3 and other nodes use NFSv4. See Installing MapR NFS on page 386.

When installed through the MapR Installer, there is no security between the client and the NFS gateway whether or not the cluster is secure. On a secure cluster, the connection between NFSv4 server and

	MapR File System is secure. The following options are supported for configuring Kerberos security for NFSv4: • Kerberos only (users are in the local node database) • LDAP and Kerberos (users are in LDAP) • Active Directory and Kerberos (users are in the Active Directory) For more information, see Configuring NFSv4 Server for Kerberos on page 1209. When configuing NFSv4 for security, make a note of the following: • If NFSv4 is installed on (existing or new) edge nodes, the edge nodes must be part of the Kerberos setup. • If NFSv4 is installed on cluster nodes, then all the cluster nodes must be part of the Kerberos setup. For more information, see Accessing Data with NFS v4 on page 1193.
Secure by Default	The MapR Data Platform v6.1 and EEP v6.0 components are secure out-of-the-box on all new installations, ensuring all network connections require authentication and all data in motion is protected with wire-level encryption. Without requiring an external security manager server or a particular security plug-in for each ecosystem component, MapR provides the ability to apply security protection directly for data as it comes into and moves out of the platform. The security semantics are applied automatically on data being retrieved or stored by any ecosystem component, application, or users. See Security for more information. Because MapR 6.1.0 in its default configuration is more secure than MapR 6.0.1, documentation references to "built-in security" have been changed to "secure by default." See Security for Ecosystem Components on page 702.
Encryption of Data at Rest	MapR v6.1 includes support for encryption of data at rest. Data on disk (or data-at-rest) in a secure MapR cluster can be encrypted, enabling you to protect the data if a disk is compromised. Encryption of data-at-rest not only prevents unauthorized users from accessing sensitive data, but it also protects against data theft via sector-level disk access. See Security for more information.
Client-side Port Binding	MapR now requires only a single source port on the client side. MapR will bind multiple sockets to the same port for establishing connections to all the nodes on the MapR cluster.

Core Data Services Innovations to Speed AI and Analytics and Lower TCO

Complex Types Support in MapR Database JSON	In MapR Database 6.1.0, you can write more expressive queries on complex types, which includes

	arrays of scalar types and arrays of nested documents. MapR Database JSON introduces the notion of a *container field path*. Using a container field path, you can access a field that is either a single value or an arbitrary array element. This is useful if you want to perform one of the following operations: • Perform comparisons on a field path that is either a single value or an arbitrary array element • Access subfields in a nested document, where the nested document is either an arbitrary array element or a single nested document • Access arbitrary elements in an array For example, suppose your JSON document contains an `addresses` field that is an array of nested documents with address details like `street`, `city`, and `state`. You can use a container field path to write a query that filters on *any* element in the `addresses` array that matches a specific `city` and `state`. You can also create secondary indexes using container field paths, improving the performance of these more expressive queries. See the following topics for more details about the feature: • Container Field Paths on page 518 • OJAI Query Conditions Using Container Field Paths on page 2615 • Using Container Field Paths as Indexed Fields on page 562 In MapR Database 6.1.0, you also are no longer restricted to creating secondary indexes on scalar data fields. You can now create indexes on fields with arrays and nested documents. See Data Types and Secondary Index Fields on page 561 for more information. To understand how indexes behave, depending on the version you are using, see Secondary Indexes and Upgrades.
Complex Type Support in Drill	The query planner in Drill can leverage indexes created on MapR Database JSON document fields with complex data types. You must write queries using specific SQL syntax for the query planner in Drill to leverage indexes on complex fields.

New Features in MapR Event Store For Apache Kafka

Idempotent (Exactly-Once) Producer	An "exactly-once" message delivery semantic produces messages without duplication. Each message is delivered once and only once. Exactly-once is insured by uniquely identifying a group of messages that are atomically persisted. Exactly-once message delivery is set with the producer idempotence option. See Enabling an Idempotent Producer on page 2766 for more information.

last-updated: Apr 11, 2022

Log Compaction

Log compaction purges previous, older messages that were published to a topic-partition and retains the latest version of the record. See Log Compaction on page 641 for more information.

Apache Kafka 1.1 Support

MapR 6.1.0 introduces support for the Apache Kafka 1.1 API.

Other Enhancements

Security Certificate Expiry Alarm

A new alarm has been added to alert users to security certificates that are expiring within a configurable time period. See Security Certificate Expiry Alarm on page 2233.

MapR Database Enhancements

MapR Database Table Metrics

MapR Database 6.1.0 supports table metrics and enhanced node metrics. Table metrics provide more granular metrics. They enable you to detect and diagnose bottlenecks and performance issues that are specific to individual tables.

MapR enables table metrics by default. You cannot disable metrics on individual tables. During installation, you can disable table metrics across your entire cluster by selecting a minimal configuration for metrics collection.

See MapR Database Metrics on page 1352 for more details. For details about table metrics, including how to disable them, see MapR Database Table Metrics on page 1355. To learn about how to view these metrics in the MapR Control System (MCS), see Visualizing Table Metrics in the Control System on page 1303.

OJAI 3.0 Support

MapR 6.1.0 introduces support for the OJAI 3.0 API. This version includes extensions for new complex types support in MapR Database JSON. See Complex Types Support in MapR Database JSON for an overview of this feature.

last-updated: Apr 11, 2022

Documentation Enhancements	**New Component Versions Matrix**	A new matrix has been added to the list of Interoperability Matrices. The Component Versions for Released EEPs matrix lets you compare the versions of ecosystem and MapR Monitoring components across different EEPs.
	Revised Product Naming	Some components of the MapR Data Platform have been changed. MapR-Streams is now called MapR Event Store For Apache Kafka. MapR-DB is now MapR Database. MapR-FS is now MapR File System.
	Removal of Third-Party Solutions Content	Third-Party Solutions information has been removed from the MapR 6.1 documentation. Installation and usage information for the following products is no longer current and will not be updated: • Datameer • HParser • Pentaho • Vertica Analytics Platform

Installation and Upgrade Notes (MapR 6.1.0)

CAUTION: New installations of MapR 6.1.0 are no longer recommended. Because of a known issue with MapR 6.1.0, HPE recommends installing MapR 6.1.1 or MapR 6.2.0 for new installations. The MapR Installer no longer supports installing MapR 6.1.0 and automatically installs MapR 6.1.1. Installation and upgrade considerations can be found in the support advisory announcing release 6.1.1. See Announcement of HPE Ezmeral Data Fabric maintenance release 6.1.1 to address 6.1.0 critical defects.

Upgrading to MapR 6.1.0 also is no longer recommended. If you need to upgrade to MapR 6.1.x, upgrade to MapR 6.1.1. See Installation and Upgrade Notes (MapR 6.1.1) on page 74.

If your cluster is currently installed with MapR 6.1.0, HPE recommends that you install the latest 6.1.0 patch. See Downloading a Patch on page 437.

List of YARN Enhancements

MapR 6.1 runs the 2.7.0 version of Hadoop. A number of new features have been incorporated into the version of Hadoop that MapR Converged Data Platform 6.1 uses.

- Support for Azure Data Lake Store.
- Support for YARN Resource Calculation Based on Labels on page 1278.

last-updated: Apr 11, 2022

Support for ADLS
Starting with MapR 6.1, you can use Azure Data Lake Store (ADLS) as a data source or destination for all applications.

Prerequisites for Using ADLS
Setting up Azure Data Lake Store (ADLS) on the Azure portal enables you to access ADLS from any application.

- Create an account on the Azure portal.
- Create an Azure Data Lake Store (get started with Azure Data Lake Storage).

Authenticating ADLS Account
To access data stored in Azure Data Lake Store (ADLS), you must first authenticate your ADLS account using your ADLS credentials.

1. Obtain the following properties from your Azure application:
   - `dfs.adls.oauth2.access.token.provider.type`

     `ClientCredential`, `Refresh Tokens`, or `Client Keys` to obtain the authentication type.
   - `dfs.adls.oauth2.client.id`

     Create an Azure Active Directory application and get your application ID and authentication key.
   - `dfs.adls.oauth2.refresh.url`

     Navigate to Azure Active Directory and click on `Endpoints`. Use the `OAUTH 2.0 TOKEN ENDPOINT` value.
   - `dfs.adls.oauth2.credential`

     Obtain the access token key value from `App Registrations` in your Azure account.

2. Add the properties obtained in step 1 to the `core-site.xml` file:

```
<!--ADL-->
<property>
   <name>dfs.adls.oauth2.access.token.provider.type</name>
   <value>ClientCredential</value>
</property>

<property>
   <name>dfs.adls.oauth2.client.id</name>
   <value>f377fab9-c0a3-4531-a1c9-77345105</value>
</property>

<property>
   <name>dfs.adls.oauth2.refresh.url</name>
   <value>https://login.microsoftonline.com/25735fb/oauth2/token</
value>
</property>

<property>
   <name>dfs.adls.oauth2.credential</name>
   <value>WTkn4xS0ISsqyzo4R6bu/OW2oPyGNMzWRw/d2z2CGiw=</value>
</property>
```

Note: The `core-site.xml` file can be overwritten using the command line. You can also specify these properties at runtime. The syntax for overwriting ADLS properties at runtime using the command line is as follows:

```
yarn jar /opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/mapreduce/
hadoop-mapreduce-examples-2.7.0-mapr-1710-SNAPSHOT.jar wordcount
-Ddfs.adls.oauth2.access.token.provider.type=ClientCredential
-Ddfs.adls.oauth2.client.id=f377fab9-c0a3-4531-a1c9-77345105
-Ddfs.adls.oauth2.refresh.url=https://login.microsoftonline.com/
25735fb/oauth2/token
-Ddfs.adls.oauth2.credential=WTkn4xS0ISsqyzo4R6bu/OW2oPyGNMzWRw/
d2z2CGiw= adl://testhue.azuredatalakestore.net/some_folder/testfile
adl://testhue.azuredatalakestore.net/some_folder/wordcountout
```

To provide your ADLS credentials securely, see Securely Providing ADLS Credentials on page 42.

3. Provide your application with file access.

4. For secure clusters, MapR-SASL (Simple Authentication and Security Layer), and Kerberos, import the required CA certificate.

- Open source documentation
- Azure documentation
- Azure documentation on authorization and access control
-

last-updated: Apr 11, 2022

Securely Providing ADLS Credentials
You can provide your ADLS credentials securely by hiding the open, readable configuration on the command line using the Hadoop credential provider.

1. Generate a `jceks` file for ADLS authorization:

```
hadoop credential create dfs.adls.oauth2.client.id -provider jceks://
hdfs/user/USER_NAME/adlskeyfile.jceks -value client ID
hadoop credential create dfs.adls.oauth2.credential -provider jceks://
hdfs/user/USER_NAME/adlskeyfile.jceks -value client secret
hadoop credential create dfs.adls.oauth2.refresh.url -provider jceks://
hdfs/user/USER_NAME/adlskeyfile.jceks -value refresh URL
```

2. Run the `DistCp` example using the `jceks` file:

```
hadoop distcp
[-D hadoop.security.credential.provider.path=localjceks://hdfs/user/
USER_NAME/adlskeyfile.jceks]
hdfs://<NameNode Hostname>:9001/user/foo/007020615
adl://<Account Name>.azuredatalakestore.net/testDir/
```

3. Configure the `core-site.xml` file to use the `jceks` file:

```
<property>
  <name>hadoop.security.credential.provider.path</name>
  <value>localjceks://hdfs/user/USER_NAME/adlskeyfile.jceks</value>
  <description>Path to interrogate for protected credentials.</
description>
</property>
```

Using ADLS for Data Input or Output
You can use Azure Data Lake Store (ADLS) as a source or destination for your application data.

For general information about the features of ADLS, refer to the Azure Data Lake Store documentation.

For information about configuring ADLS as storage for a Hadoop cluster, refer to the official Apache documentation.

The Azure Data Lake Storage access path syntax is:

```
adl://<Account Name>.azuredatalakestore.net/
```

You can use ADLS the same way as you use MapR File System, substituting an `adl` scheme instead of `maprfs`, `hdfs`, `webhdfs`, and so on.

1. Create a directory and read data:

```
[mapr@node4 ~]$ hadoop fs -mkdir adl://<username>.azuredatalakestore.net/
testdir

[mapr@node4 ~]$ hadoop fs -ls adl://<username>.azuredatalakestore.net/

Found 1 items
drwxr-xr-x - 9d3f4f74-8337-4dae-ad77-f63459438553
331c9f66-6875-4e13-a74f-458dd23e4bde 0 2018-04-16 09:09
adl://<username>.azuredatalakestore.net/testdir
```

2. Put data into ADLS from your local MapR File System:

```
[mapr@node4 ~]$ hadoop fs -put testfile adl://
<username>.azuredatalakestore.net/testdir

[mapr@node4 ~]$ hadoop fs -ls adl://<username>.azuredatalakestore.net/
testdir

Found 1 itemsrw-rr- 1 9d3f4f74-8337-4dae-ad77-f63459438553
331c9f66-6875-4e13-a74f-458dd23e4bde 0 2018-04-16 09:10
adl://<username>.azuredatalakestore.net/testdir/testfile
```

3. Delete data from ADLS:

```
[mapr@node4 ~]$ hadoop fs -rm -r adl://<username>.azuredatalakestore.net/
testdir

[mapr@node4 ~]$ hadoop fs -ls adl://<username>.azuredatalakestore.net/
```

4. Run YARN jobs with your input and output stored in ADLS:

```
yarn jar /opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/mapreduce/
hadoop-mapreduce-examples-2.7.0-mapr-1710-SNAPSHOT.jar wordcount
adl://<username>.azuredatalakestore.net/testdir/testfile adl://
<username>.azuredatalakestore.net/wordcountout
```

Deleting Data from ADLS
You can delete your data from Azure Data Lake Store (ADLS).

- To delete data from ADLS:

```
[mapr@node4 ~]$ hadoop fs -rm -r adl://<username>.azuredatalakestore.net/
testdir

[mapr@node4 ~]$ hadoop fs -ls adl://<username>.azuredatalakestore.net/
```

Operational Changes (MapR 6.1.0)

Lists the functional changes made to existing commands in MapR version 6.1.0.

Note the following functional changes to existing commands in Version 6.1.

MapR Database

OJAI Support	MapR 6.1.0 introduces support for OJAI 3.0. For a list of classes and methods deprecated in OJAI 3.0, see https://docs.datafabric.hpe.com/apidocs/61/ojai/java/deprecated-list.html. MapR 6.1.0 deprecates support for OJAI 1.0. The next release after 6.1.0 will no longer support OJAI 1.0.
Permissions on Arrays in MapR Database JSON	Starting from MapR 6.1.0, when you grant permissions on a field using array syntax (for example, `person[]`), you no longer have to grant separate permissions if the field also contains non-array values. However, this can result in errors if you attempt to define a new permission that conflicts with an existing one.

For example, suppose you have the following two documents in a JSON table:

```
{
    "_id" : "id001",
    "person" : [
        {"name" : {"last" : "Smith",
"first" : "John"}},
        {"name" : {"last" :
"Subramanium", "first" : "Ananya"}}
    ]
}
{
    "_id" : "id002",
    "person" : {"name" : {"last" :
"Doe", "first" : "Jane"}}
}
```

In document `id001`, `person` is an array of nested documents and in `id002`, it is a single nested document.

The following table summarizes the behavior in 6.1 versus pre-6.1 when you grant permissions in the sequence shown:

Permission Grant Sequence	6.1 Behavior	Pre-6.1 Behavior	Key Differences
1. `person[]` **2.** `person`	**1.** Permission granted on `person` in documents `id001` and `id002` **2.** Error - conflicts with permission on `person[]` You do not need to grant this permission. The previous is sufficient.	**1.** Permission granted on `person` in document `id001` **2.** Permission granted on `person` in document `id002`	In 6.1, having permission on `person[]` also gives you permission on `person`. Prior to 6.1, you need two separate permissions.

last-updated: Apr 11, 2022

Permission Grant Sequence	6.1 Behavior	Pre-6.1 Behavior	Key Differences
1. `person` 2. `person[]`	1. Permission granted on `person` in document `id002` 2. Error - conflicts with permission on `person` You (or an administrator with appropriate permissions) must drop the permission on `person` before you grant permission on `person[]`.	1. Permission granted on `person` in document `id002` 2. Permission granted on `person` in document `id001`	In 6.1, if an array permission conflicts with an existing permission, you must remove the conflicting permission.

The new 6.1 behavior applies to permissions granted in earlier versions after you upgrade your MapR cluster.

Note: If you upgrade your cluster using rolling upgrades, whether you encounter pre-6.1 or 6.1 behavior depends on whether the MapR node enforcing the permission has been upgraded.

See Permissions on Arrays on page 538 for more details about the new 6.1 behavior.

Data Types and Secondary Indexes

In MapR Database 6.1.0, you are no longer restricted to creating secondary indexes on scalar data fields. You can now create indexes on fields with arrays and nested documents. As a consequence, a 6.1 index may contain additional rows that are missing in an equivalent pre-6.1 index. If a query uses a covering

last-updated: Apr 11, 2022

index, MapR Database 6.1 returns these additional rows. See Data Types and Secondary Index Fields on page 561 for more information about data type support.

Secondary Indexes and Upgrades

Starting with MapR Database 6.1, you can define secondary indexes using container field paths. Pre-existing indexes created in earlier releases do not support this new functionality, even after you upgrade your MapR cluster to 6.1. After you upgrade your cluster to 6.1, any new indexes you create support this functionality. You can use these indexes in your application if you are using a 6.1 (or later) client.

Note: If your client is running on a MapR cluster node, after you upgrade the cluster node, the client becomes a 6.1 client and supports the new indexing functionality. Your pre-6.1 client can use an index created in a 6.1 cluster if the index does not use the new container field path functionality.

For more information about indexes on complex types, see Complex Types Support in MapR Database JSON.

MapR File System

Data-on-Wire Encryption

Beginning with MapR 6.1, data-on-wire encryption is enabled by default for newly created volumes for secure clusters. Data-on-wire encryption encrypts data in a volume during transmission over the wire. Data-on-wire encryption is not supported for non-secure clusters. You can disable data-on-wire encryption for individual volumes using MCS, the `maprcli`, or REST API commands. For more information, see the `-wiresecurityenabled` parameter of volume create on page 1931 and volume modify on page 2005. See also Creating a Volume on page 864 or Modifying a Volume on page 892.

Impersonation

Beginning with MapR 6.1:

- You cannot generate a ticket with impersonated UID and/or GID as the following:
  - UID `0` and/or GID `0` (user root)
  - UID `mapr_uid` and/or GID `mapr_gid` (user mapr)
- User mapr can impersonate anyone, including user root.

For more information, see Managing Impersonation on page 1476 and `maprlogin on page 2130`.

MapR Tickets

Beginning with MapR 6.1, if the UID and GID in the ticket (without impersonation capability) is different from the UID and GID of the logged-in user, all operations are performed using the UID and GID of the ticket and not that of the logged-in user.

MapR Event Store For Apache Kafka (Streams)

Caution: Do Not Remove mapr-librdkafka In MapR 6.1 and later, the `mapr-core` package has a dependency on `mapr-librdkafka`. If the `mapr-librdkafka` package is installed, do not remove it manually. Doing so might result in the removal of MapR core packages, rendering the node unusable.

Using the **Incremental Install** function of the MapR Installer, you can safely deselect the **Streams Tools** and **Streams Clients** service options. The MapR Installer ensures that the `mapr-librdkafka` package is left intact when the services are removed. For manual operations, removing the `mapr-librdkafka` package is not recommended unless you also plan to remove the MapR software.

Partition Maximum As of MapR 6.1, the MapR Event Store For Apache Kafka API enforces a maximum of 4096 partitions for a topic. If you create an application with the MapR Event Store For Apache Kafka 6.1 API, the maximum number of partitions is 4096. If you previously created an application with MapR Event Store For Apache Kafka 6.0.1 API (or older) and you've upgraded, the original number of partitions can be used. For example, if you were using more than 4096 partitions in MapR 6.0.1 or earlier, you can continue to use the same number of partitions after upgrading.

MapR Installer

Off-Cluster Elasticsearch and OpenTSDB With MapR Installer 1.10, the installer no longer includes an option to support off-cluster Elasticsearch and OpenTSDB when security is turned on. For non-secure clusters, MapR continues to support the option for off-cluster Elasticsearch and OpenTSDB.

Deprecated Features

- None

Known Issues at Release (MapR 6.1.0)

You may encounter the following known issues after upgrading to Version 6.1. This list is current as of the release date. For a dynamic list of all known issues for all MapR product releases, refer to Support notices of known issues

Where available, the workaround for an issue is also documented in this topic. MapR regularly releases maintenance releases and patches to fix issues. We recommend checking the release notes for any subsequent maintenance releases to see if one or more of these issues are fixed.

Installation and Configuration Issues

You can see generic installation issues here: MapR Installer Known Issues.

Keytool error on SLES 12 SP4 Node	Running the `configure.sh` script on a SLES 12 SP 4 node can fail with the following error:
	`keytool error: java.security.ProviderException: Could not initialize NSS`
	The installation fails because the `mozilla-nss` dependency is not installed.
	Workaround: Install the `mozilla-nss` package using the following command, and rerun `configure.sh`:
	`zypper install mozilla-nss`

IN-2637	After a manual installation, Oozie and Hive services can fail to connect to a MySQL or MariaDB database because the server time-zone value is unrecognized

	or represents more than one time zone. The issue affects your installation if you applied the `mapr-patch` released on or after February 21, 2021 (including the latest `mapr-patch`). This issue affects manual installations but is fixed in Installer 1.14.0.0. **Workaround:** For manual installations, you must configure either the server or JDBC driver (using the `serverTimezone` configuration property) to use a more specific time-zone value if you want to utilize time-zone support. After running `configure.sh` but before starting the Oozie or Hive services, update the `serverTimezone` parameter in the `hive-site.xml` or `oozie-site.xml`. For more information, see MySQL Bug #95036.
MFS-6772	Installation of MapR 6.1 fails on Red Hat or CentOS 8.2 and later due to a missing dependency on sdparm. Getting dependency error as sdparm is removed from RHEL8. **Workaround:** See support advisory 4730.
MFS-10783	After a manual installation of release 6.1.0 or 6.1.1 on RHEL or CentOS 8.x, Collectd fails to start. Error messages indicate that `libcrypto.so.10` cannot open a shared object file. **Workaround:** Install the `compat-openssl10` package using the following command, and restart Collectd: `yum install compat-openssl10` Alternatively, you can install the `compat-openssl10` package during the course of manual installation before installing the `mapr-*` packages. See Step 3: Install Cluster Service Packages on page 150.
SPYG-1136	During a manual installation or upgrade, Collectd provided in core 6.1.0 won't start on RHEL / CentOS 8.2 because it expects the Python 2 libraries to be installed, and RHEL / CentOS 8.2 provides the Python 3 libraries instead. This issue does not affect installations or upgrades performed using the Installer. **Workaround:** Before installing the monitoring components, check to see if Python 2 is installed. If the following error is generated, try installing Python 2 on RHEL / CentOS 8.2: `failed: libpython2.7.so.1.0: cannot open shared object file`

Upgrade Issues

INFO-960	**Description:** The Kibana log (`/opt/mapr/kibana/kibana-<version>/var/log/kibana/kibana.*.<#>`) can show the following error if you upgrade Kibana without capturing a snapshot of the Kibana index, as described in Pre-Upgrade Steps for MapR Monitoring on page 342. This issue was

noticed after an upgrade from MapR 6.0.1 with EEP 5.0.3 to EEP 6.1.0 with EEP 6.2.0:

```
{"type":"error","@timestamp":"2019-0
4-25T01:29:26Z","tags":
["fatal","root"],"pid":28300,"level":"
fatal","error":
{"message":"Index .kibana belongs to
a version of Kibana that
cannot be automatically migrated.
Reset it or use the X-Pack upgrade
assistant.","name":"Error","stack":"Er
ror: Index .kibana belongs to a
version of Kibana that cannot be
automatically migrated. Reset it or
use the X-Pack upgrade
assistant.\n
at assertIsSupportedIndex (/opt/mapr/
kibana/kibana-6.5.3/src/server/
saved_objects/migrations/core/
elastic_index.js:237:15)\n
at Object.fetchInfo (/opt/mapr/kibana/
kibana-6.5.3/src/server/saved_objects/
migrations/core/
elastic_index.js:48:12)\n
at
<anonymous>"},"message":"Index .kibana
 belongs to a version of Kibana that
cannot be automatically migrated.
Reset it or use the X-Pack upgrade
assistant."}
```

Resolution: Use this procedure to migrate the Kibana index to version 6.0. Then run Kibana from `/opt/mapr/kibana/kibana-6.5.3/bin`.

MAPRYARN-129

Issue: For some clusters that are upgraded from MapR 6.0.1 to 6.1.0, using any Hadoop command generates the following `java.lang.ClassNotFoundException`:

```
hadoop fs -ls /
log4j:ERROR Could not instantiate
class
[org.apache.hadoop.yarn.WorldWritableL
ogAppender].
java.lang.ClassNotFoundException:
org.apache.hadoop.yarn.WorldWritableLo
gAppender
at
java.net.URLClassLoader.findClass(URLC
lassLoader.java:381)
at
java.lang.ClassLoader.loadClass(ClassL
oader.java:424)
at
sun.misc.Launcher$AppClassLoader.loadC
lass(Launcher.java:349)
at
java.lang.ClassLoader.loadClass(ClassL
oader.java:357)
at java.lang.Class.forName0(Native
```

```
Method)
at
java.lang.Class.forName(Class.java:264
)
at
org.apache.log4j.helpers.Loader.loadCl
ass(Loader.java:198)
at
org.apache.log4j.helpers.OptionConvert
er.instantiateByClassName(OptionConver
ter.java:327)
at
org.apache.log4j.helpers.OptionConvert
er.instantiateByKey(OptionConverter.ja
va:124)
at
org.apache.log4j.PropertyConfigurator.
parseAppender(PropertyConfigurator.jav
a:785)
at
org.apache.log4j.PropertyConfigurator.
parseCategory(PropertyConfigurator.jav
a:768)
at
org.apache.log4j.PropertyConfigurator.
parseCatsAndRenderers(PropertyConfigur
ator.java:672)
at
org.apache.log4j.PropertyConfigurator.
doConfigure(PropertyConfigurator.java:
516)
at
org.apache.log4j.PropertyConfigurator.
doConfigure(PropertyConfigurator.java:
580)
at
org.apache.log4j.helpers.OptionConvert
er.selectAndConfigure(OptionConverter.
java:526)
at
org.apache.log4j.LogManager.<clinit>(L
ogManager.java:127)
at
org.apache.log4j.Logger.getLogger(Logg
er.java:104)
at
org.apache.commons.logging.impl.Log4JL
ogger.getLogger(Log4JLogger.java:262)
at
org.apache.commons.logging.impl.Log4JL
ogger.<init>(Log4JLogger.java:108)
at
sun.reflect.NativeConstructorAccessorI
mpl.newInstance0(Native Method)
at
sun.reflect.NativeConstructorAccessorI
mpl.newInstance(NativeConstructorAcces
sorImpl.java:62)
at
sun.reflect.DelegatingConstructorAcces
sorImpl.newInstance(DelegatingConstruc
torAccessorImpl.java:45)
```

```
at
java.lang.reflect.Constructor.newInsta
nce(Constructor.java:423)
at
org.apache.commons.logging.impl.LogFac
toryImpl.createLogFromClass(LogFactory
Impl.java:1025)
at
org.apache.commons.logging.impl.LogFac
toryImpl.discoverLogImplementation(Log
FactoryImpl.java:844)
at
org.apache.commons.logging.impl.LogFac
toryImpl.newInstance(LogFactoryImpl.ja
va:541)
at
org.apache.commons.logging.impl.LogFac
toryImpl.getInstance(LogFactoryImpl.ja
va:292)
at
org.apache.commons.logging.impl.LogFac
toryImpl.getInstance(LogFactoryImpl.ja
va:269)
at
org.apache.commons.logging.LogFactory.
getLog(LogFactory.java:657)
at
org.apache.hadoop.fs.FsShell.<clinit>(
FsShell.java:43)
```

This issue only affects clusters that use scheduler debug logging as described in **31038** of Known Issues at Release (MapR 6.0.1). The issue occurs because the `hadoop-yarn-common-2.7.0-mapr-1808.jar` in MapR 6.1.0 does not contain the `org.apache.hadoop.yarn.WorldWritableLogAppender` Java class.

Workaround: Remove the following lines from the `/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/log4j.properties` file:

```
yarn.debug.controller.logger = INFO,
ED
yarn.debug.controller.log.file=yarn-ma
pr-scheduling-debug.log
yarn.debug.controller.log.maxfilesize=
256MB
yarn.debug.controller.log.maxbackupind
ex=20

log4j.logger.org.apache.hadoop.yarn.se
rver.resourcemanager.scheduler.
DebugController=$
{yarn.debug.controller.logger}
log4j.additivity.org.apache.hadoop.yar
n.server.resourcemanager.scheduler.
DebugController=false

log4j.appender.ED=org.apache.hadoop.ya
rn.WorldWritableLogAppender
log4j.appender.ED.File=$
```

```
{hadoop.log.dir}/$
{yarn.debug.controller.log.file}
log4j.appender.ED.MaxFileSize=$
{yarn.debug.controller.log.maxfilesize
}
log4j.appender.ED.layout=org.apache.lo
g4j.PatternLayout
log4j.appender.ED.layout.ConversionPat
tern=%d{ISO8601} : %m%n
```

30422

Issue: Using the `configure.sh -R` command after an upgrade from 5.2.x to 6.0 or later can return error messages for some MapR Monitoring components. For example:

```
/opt/mapr/server/configure.sh -R
Configuring Hadoop-2.7.0 at /opt/mapr/
hadoop/hadoop-2.7.0 Done configuring
Hadoop Node setup configuration:
drill-bits fileserver hbinternal nfs
nodemanager opentsdb Log can be found
at: /opt/mapr/logs/configure.log
ls: cannot access /opt/mapr/
hbinternal: No such file or directory
Configuring opentsdb
getopt: unrecognized option
'--unsecure'
getopt: unrecognized option '-EC'
usage: /opt/mapr/opentsdb/
opentsdb-2.3.0/bin/configure.sh
[-nodeCount <cnt>] [-nodePort
<port> -nodeZkCount <zkCnt>]
[-nodeZkPort <zkPort>]
[-R] -OT "ip:port,ip1:port," -Z
"ip:port,ip1:port,"

/opt/mapr/server/configure.sh -R
Configuring Hadoop-2.7.0
at /opt/mapr/hadoop/hadoop-2.7.0
Done configuring Hadoop
Node setup configuration:
drill-bits fileserver grafana
hbinternal historyserver
nfs nodemanager spark-historyserver
Log can be found
at: /opt/mapr/logs/configure.log
ls: cannot access /opt/mapr/
hbinternal: No such file or directory
ls: cannot access /opt/mapr/
spark-historyserver: No such
file or directory
Configuring grafana
getopt: unrecognized option
'--unsecure'
getopt: unrecognized option '-EC'
usage: /opt/mapr/grafana/
grafana-4.1.2/bin/configure.sh
[-nodeCount <cnt>] [-nodePort
<port>] [-grafanaPort
<port>] [-secureCluster]
```

```
[-loadDataSourceOnly] [-R] -OT
"ip:port,ip1:port,"
```

These errors are generated because a newer version of the server `configure.sh` uses certain options that do not exist with older ecosystem components.

Workaround: None. The upgrade can succeed in spite of the errors, and the current ecosystem configuration is left unchanged. The error messages can be ignored.

File System

MFS-14890

Issuing `hadoop mfs -lsfid` with a non-`mapr` user ticket returns an `Operation not permitted` error. For example:

```
2022-03-15 18:12:23,6663 DEBUG
Cidcache fc/cidcache.cc:4315 Thread:
18743 Enter GetVolumeMountPoint
138102135
2022-03-15 18:12:23,6663 DEBUG
Cidcache fc/cidcache.cc:4754 Thread:
18743 Sending RPC to CLDB:
10.163.172.130:7222
2022-03-15 18:12:23,6672 DEBUG
Cidcache fc/cidcache.cc:4193 Thread:
18743 GetVolumeProperties returns: 0,
numTries:0
2022-03-15 18:12:23,6673 ERROR
Cidcache fc/cidcache.cc:4277 Thread:
18743 VolumePropertiesLookupRequest
failed, cldb returned err
Operation not permitted(1), CLDB:
10.163.172.130:7222
2022-03-15 18:12:23,6673 DEBUG Client
fc/client.cc:11139 Thread: 18743
GetVolumeMountPoint returns: 0
```

In a secure cluster, this happens because the `hadoop mfs -lsfid` command works only when the UID associated with the ticket has either READ permission or full permission at the cluster level.

Workaround: Assign read permission to the UID by using an ACL to give the non-`mapr` user login permission. For example:

```
# maprcli acl  edit -type
cluster -user user1:login
maprcli acl show -type cluster
Allowed actions            Principal
[login, ss, cv, a, fc, cp]  User mapr
[login, ss, cv, a, fc, cp]  User root
[login]                    User user1
```

Indexing and MapR Drill

MD-2218 (previously 28041)

Running multiple queries in sequence can cause Drillbits to run out of direct memory. Limiting records in large batches to 5KB may prevent this issue

from occurring. Should Drill return an out of memory exception, issue the following command to restart the Drillbits:

```
$ maprcli node services -name
drill-bits -action restart -nodes
<node host names separated by a space>
```

28096

If you use the Google Guava library in your application and encounter the following error:

```
Caused by:
java.lang.NoSuchMethodError:
com.google.common.base.Stopwatch.creat
eStarted()Lcom/google/common/base/
Stopwatch;
```

You must use version 18.0 of the library. If you use Maven, you can use the following dependency:

```
            <dependency>

<groupId>com.google.guava</groupId>
              <artifactId>guava</
artifactId>
              <version>18.0</version>
              </dependency>
```

MapR Database

MAPRDB-1375

If you have a MapR Database JSON table with multiple column families and the table has a secondary index, after you perform a series of update operations (for example: set, increment, decrement, replace, append, etc.) on the table, the index may have duplicate rows.

MAPRDB-1520

If you have an OR condition that references a container field path more than once, and MapR Database uses an index to process the query, then the query may return incorrect results. You can workaround this problem by forcing MapR Database to not use indexes. See Forcing Secondary Index Usage in OJAI on page 2589 for details.

27057

When you issue a maprcli table changelog add command, during the initial sync when the existing data is copied to the change log, the end-to-end latency during the initial 5 minutes can be high. After the initial sync, latency will be small, specifically 2-4 seconds.

MAPRDB-2092

In the MapR Database, adding a table index or replicating a table fails if the cluster administrator (MAPR_USER) does not have write access to the parent volume of the table.

Workaround: For more information, see this support advisory. If a patch is not available or the available patch has not been applied, another workaround is to add the MAPR_USER to the writeACE for the table parent volume.

MapR Client

23624

The same MapR client can access both secure and nonsecure clusters; however, a MapR client that is configured to access a secure cluster can access a nonsecure cluster only if these conditions are met:

- The secure cluster must be listed first in `mapr-clusters.conf`.
- A user must obtain a ticket for the secure cluster even if the user wants to access only the nonsecure cluster.

MapR Control System

MON-4891

Description: The MCS displays an Oops error on the **Admin** > **User Settings** > **SMTP** page when the `mapr.smtp.provider` is not any of the 3 allowed options (Gmail, Office 365, SMTP). This impacts all upgrade from 5.x to 6.x where your SMTP provider setting is "other".

Resolution: Run the following command to resolve this error if your SMTP provider is not Gmail or Office 365:

```
maprcli config save -values
{"mapr.smtp.provider":"smtp"}
```

If your SMTP provider is Office 365, run the following command:

```
maprcli config save -values
{"mapr.smtp.provider":"office"}
```

MON-4874

Description: When MAPR_EXTERNAL environment variable is set, the MCS UI might not be accessible and might return the following error:

```
HTTP ERROR 503
           Problem accessing /app/mcs/
```

Resolution: Remove entries for `tsdbService` and `amazonS3Service` from the `/opt/mapr/apiserver/conf/applicationContext.xml` file. For example:

1. Remove the following and save and close the file:

```
<bean id="tsdbService"
class="com.mapr.admin.service.impl.
TsdbServiceImpl"/>
<bean id="amazonS3Service"
class="com.mapr.admin.service.Amazo
nS3Service"/>
```

2. Restart apiserver.

MON-4862

Description: The MCS does not start after an upgrade to MapR 6.1 if the installed license does not

last-updated: Apr 11, 2022

have table support enabled. See Support Portal for additional information.

MON-4788

Description: When running large number of table (get, scan, index) operations, the API server may become unresponsive and you may see following error: GC overhead limit exceeded.

Resolution: Check the heap size and increase it based on available system resources or set up a load balancer to distribute the HTTP requests (load) to other API servers. To increase heap size:

1. Open `/opt/mapr/apiserver/bin/mapr-apiserver.sh` file.
2. Modify the value for `JVMXMS` and `JVMXMX` (under JVM tuning).

   For example, modify the following in the file:

   ```
   # JVM tuning
   JVMXMS=${JVMXMS:-1024m}
   JVMXMX=${JVMXMX:-1024m}
   ```

3. Save and close the file.
4. Restart apiserver for the changes to take effect.

   For example, run the following command to restart the apiserver:

   ```
   maprcli node services -action
   restart -name apiserver -nodes
   `hostname`
   ```

MON-4776

Description: You cannot mount or unmount multiple volumes at the same time by selecting them from different pages in the **Volumes** pane (of the **Data** > **Volumes** page).

Workaround: Increase the number of rows, if possible, in the **Volumes** pane and select the volumes from the same page to mount or unmount.

MON-4701

Description: The table metrics list view displays metric values calculated using endTime-startTime. If you select a time range like **Today**, **This Week**, **This Month**, where the end time is in the future, no data is displayed in the columns.

Workaround: Choose **Today so far**, **This week so far**, **This month so far** instead to view the most current data.

MON-3922

Description: After upgrading to v6.1.0, editing volumes that have no volumes ACEs is not supported in MCS. You cannot view or edit volumes that were created using MapR Data Platform version <=5.1.

Workaround: To resolve the issue, do the following:

- Use the `volume modify` on page 2005 command to set or modify volume ACEs.
- Download and install the following patches:

last-updated: Apr 11, 2022

- `mapr-apiserver-6.1.0.20190301101733.GA-1.noarch`
- `mapr-webserver-6.1.0.20190301101733.GA-1.noarch`

MON-3817	**Description:** The **Node Health by Topology** pane does not reflect the most recent topology of the node if the node was just moved to a different/new topology. **Workaround:** Navigate to another tab or page and then return to the page to view the updated topology for the node in the **Node Health by Topology** pane.
MON-3709	**Description:** The list of services in the **Services** page do not refresh per the configured refresh rate settings. **Workaround:** To view the latest status for the services, reload the page in the browser.
MON-3476	**Issue:**In the MapR Control System, table permissions can only be modified by the owner/creator of the table. **Workaround:** User, who is not the owner but has table admin permissions, must use the CLI to modify table permissions.
MON-3452	**Issue:** After selecting the **Include System Volumes** checkbox to display system volumes in the **Volumes** list pane, if you set quota (using the link in the **Quota** column), the system volumes are no longer displayed in the list of volumes even though the system volumes checkbox is still selected. **Workaround:** Deselect and then re-select the **Include System Volumes** checkbox to display the system volumes in the list of volumes.
MON-3438	**Issue:** You may experience some latency in the **Admin** > **Cluster Settings** > **Alarms** page when you click the name of an alarm to display the **Alarm Settings** window for the alarm.
MON-3425	**Issue:** When you filter the list of alarms in the **Alarms** summary page, the page does not refresh and display the correct list of alarms if there are muted alarms.
MON-3407	**Issue:** If you are upgrading from a pre-6.0 version, the MapR Control System might display 0 instances of webserver running in the **Services** page. This is because starting from v6.0, the UI static files are in `mapr-webserver` and the `mapr-apiserver` runs the server that sends the queries. For more information on webserver in v6.0, see Setting Up the Control System on page 423.
MON-3404	**Issue:** The topic search field in the **Stream (detail)** > **Topic** tab does not display correctly in Firefox. **Workaround:** Use Chrome (v58 and later) to perform a topic search.
MON-3247	**Issue:** The MapR Control System does not display NFS edge nodes and POSIX client nodes that are currently connected to the cluster when you sort the list of nodes by topology. **Workaround:** Use the `node list` on page 1705 command to retrieve the list of client nodes currently connected to the cluster.

last-updated: Apr 11, 2022

MON-2409	**Issue:** When you try to navigate to another cluster by clicking on the cluster name from the drop-down list at the top of the page, the **Overview** page for the current cluster displays and it takes a few minutes to display the MCS login page for the other cluster.
MON-2374	**Issue:** If a node has multiple IP addresses, the log viewer link for the services in the nodes information page is broken because the Kibana URL contains multiple IP addresses. **Workaround:** Remove all irrelevant IP addresses in the URL and use the modified URL to view logs for the service.
MON-2321	**Issue:** If Elasticsearch fails, the **Services** page does not show the nodes on which Elasticsearch has failed.
MON-2219	**Issue:** If you add Streams license separately, you may not be able to view the **Streams** page. **Workaround:** Contact MapR Support.
MON-2218	**Issue:** When you preserve snapshots, there is no notification (success message) or indication (**Expires On** column still does not show *No Expiration*) even though the operation succeeded. **Workaround:** Refresh the page after preserving a snapshot to see the *No Expiration* setting in the **Expires On** column.
MON-2076	**Issue:** When you select topics to remove, deselect all the topics in the **Remove Topic** confirmation window and click **Cancel**, then select a topic in the page by clicking on the checkbox, the **Remove Topic** window opens again even if you did not click on the **Remove Topic** button. **Workaround:** Refresh the page.
MON-2018	**Issue:** In the **Add Change Log** page, after entering field to add in the **Publish Selected Field Path** text field, the **Add Field** button is disabled. **Workaround:** Click outside the text field to enable the **Add Field** button.
MON-2011	**Issue:** When adding or editing the compression setting on a replica, selecting **Inherited** does not have any effect. The default value is lz4. **Workaround:** Do not select **Inherited** as the compression setting on a replica.
MON-2009	**Issue:** When modifying replica of a binary table with three or more column families, if you select different set of columns to replicate for different column families, the operation might result in an error.
MON-1994	**Issue:** If you adjust the system time, the time stamp associated with the alarm in the **Active Alarms** pane might show negative value.
MON-1968	**Issue:** In the **User Disk Usage** page, you cannot remove the email address after setting an email address for an entity by editing the properties.
MON-1929	**Issue:** The **Save Changes** button in the **Access Control Expression** builder page is disabled after entering user, group, or role name in text field.

last-updated: Apr 11, 2022

	Workaround: Click outside the field to enable the **Save Changes** button.
MON-1810	**Issue:** When editing multiple volumes, you cannot undo changes to **Auditing** and **Accountable Entity** fields by clicking ↶.
MON-1755	**Issue:** When creating a mirror volume, after entering the name of the source volume on the remote cluster and selecting the remote cluster on which the volume resides, you may see an error that the source volume does not exist. **Workaround:** Re-enter the name of the source volume on the remote cluster.
MON-1221	**Issue:** In the **Alarm Settings** window, if you click the cursor in the **Additional email recipients** text field and then press the return/enter key in the keyboard without entering an email address, the **Cluster Settings** page displays once again.

MapR Monitoring

ES-58	**Issue:** After an upgrade from MapR 5.2.x with EEP 3.0.x to MapR 6.1.0 using the MapR Installer, Elasticsearch and Fluentd fail to start, and the Elasticsearch roles file (`/opt/mapr/roles/elasticsearch`) is missing. This issue has been seen on upgrades from MapR 5.2.1 with EEP 3.0.3 to MapR 6.1.0 with EEP 6.0.0 but can affect all upgrades from EEP 3.0.x.
	Workaround: To work around the issue, you must re-create the roles file. On the node where Elasticsearch was upgraded and the roles file was removed:
	1. Re-create the roles file by using this command:
	`touch /opt/mapr/roles/elasticsearch`
	2. Run configure.sh with the `-R` option:
	`/opt/mapr/server/configure.sh -R`
	3. Restart Fluentd using the MapR Control System (MCS) or a terminal.

SPYG-1083	**Issue:** On initial startup, Kibana 6.5.3.0 takes longer to complete its optimization than previous Kibana versions. During this time, if you try to connect to Kibana, the browser can show: `Kibana server is not ready yet` This issue affects EEP 6.1.0 installations. (Kibana 6.5.3.0 was introduced with EEP 6.1.0.) **Workaround:** Wait for the Kibana optimization to complete before trying to connect. To check the progress of startup operations, see the startup log at `/opt/mapr/kibana/kibana-6.5.3/var/log/`

last-updated: Apr 11, 2022

`kibana/kibana_startup.log`. Messages such as the following indicate that startup is still in progress:

```
2019-01-12 13:10:26: Waiting on
Kibana daemon to start
2019-01-12 13:11:26: kibana server
optimizing - waiting for it to
complete
```

The following message indicates that optimization is complete:

```
2019-01-12 13:16:47: kibana server
optimizing completed
```

OTSDB-72

Issue: The memory allocated to OpenTSDB can be insufficient, resulting in empty graphs and out-of-memory or `GC overhead limit exceeded` errors.

Workaround: Increase the default memory for OpenTSDB by making the following changes on all OpenTSDB nodes:

1. Edit the `/opt/mapr/conf/conf.d/warden.opentsdb.conf` file to change:

   ```
   service.heapsize.max=2000
   service.heapsize.min=2000
   ```

   to

   ```
   service.heapsize.max=6000
   service.heapsize.min=6000
   ```

2. Edit the `/opt/mapr/opentsdb/opentsdb-*/etc/init.d/opentsdb` file to change:

   ```
   $
   {JVMXMX:=-Xmx2000m -Xss1m -XX:MaxMe
   taspaceSize=128m}
   ```

   to

   ```
   $
   {JVMXMX:=-Xmx6000m -Xss1m -XX:MaxMe
   taspaceSize=128m}
   ```

3. Restart the OpenTSDB service:

   ```
   maprcli node services -name
   opentsdb -nodes <space-separated
   list of OpenTSDB nodes> -action
   restart
   ```

OTSDB-60

Issue: EEPs 6.0.x or earlier with MapR Installer 1.10 or earlier can generate numerous instances of the following streams error in the `opentsdb.err` file:

```
2018-06-22 21:33:55,5510 ERROR
StreamsClient fs/client/marlin/cc/
marlinclient.cc:402 Thread: 2958 /var/
mapr/mapr.monitoring/streams/0 does
not exist
```

The error might be valid for users upgrading from a previous release, but is generated even when users have not upgraded.

Workaround: To stop the error from being generated, use the following workaround:

1. Create the `/var/mapr/mapr.monitoring/streams` directory:

   ```
   hadoop fs -mkdir /var/mapr/
   mapr.monitoring/streams
   ```

2. Use `maprcli` to create a stream that will prevent the error from being generated:

   ```
   maprcli stream create -path /var/
   mapr/mapr.monitoring/steams/0 -ttl
   86400
   ```

SPYG-994

Issue: The time-to-live for MapR Event Store For Apache Kafka metrics is fixed at 30 days and cannot be reduced to lower disk space usage.

SPYG-934

Issue: On secure Ubuntu 14.04 or 16.04 clusters, Elasticsearch fails to generate a keystore password if the `uuid-runtime` package is not installed. `uuid-runtime` is one of the MapR Installer prerequisites.

Workaround: Install the `uuid-runtime` package from the default Ubuntu repository:

```
apt-get install uuid-runtime
```

MapR PACC

CORE-498:

Issue: After installation of the `mapr-thin-client*.tar.gz`, the `mapr-posix-client-container` fails to start, and the following message appears in the `/opt/mapr/logs/posix-client-container.log`:

```
Failed to call init on library /tmp/
libMapRClient_c.so.0
```

This issue is fixed in release 6.2 but might affect earlier PACC releases.

last-updated: Apr 11, 2022

Workaround: Create the following file (the file can be empty), and retry the operation:

```
/opt/mapr/hadoop/hadoop-2.7.4/etc/
hadoop/core-site.xml
```

See also MapR PACC Known Issues on page 417.

NFSv4

32364

Issue: Any running IO on NFSv4 mount (with kerberos) is stuck if the krb5 ticket expires for the current user. The mount point also hangs and becomes inaccessible.

Workaround: Restart rpcgssd service with the new ticket to make the mount point accessible and re-trigger the IO to proceed.

Oozie

29634

Issue: On a secure cluster, Oozie jobs can fail when the timeline server for the Hive-on-Tez user interface is configured with MapR-SASL.

Workaround: Restart the resource manager and timeline server. For more information, see Configuring the Timeline Server to Use the Hive-on-Tez User Interface on page 3507.

YARN

MAPRYARN-185

YARN cgroups are not supported on RHEL 7.x. In RHEL 7.x, libcgroup is deprecated and systemd maintains cgroups. Moving default hierarchies that systemd manages can generate unexpected results.

PJDTest Suite Compliance

28912

Issue: When trying to open a FIFO on a FUSE mounted filesystem, permissions to perform the operation are not checked.

29234

Issue: The FUSE-based POSIX can only trace at millisecond level and the test suite utimensat in pjdtestsuite fails because the test suite sets time in nanoseconds.

29237

Issue: Any user can set time using `touch -t` for any file on a FUSE mounted filesystem.

Timeline Server

See Hive-on-Tez User Interface Known Issues on page 3510.

Resolved Issues

Lists the issues that were resolved in MapR version 6.1.0

The following MapR issues, which were reported by customers, are resolved in Version 6.1.

Component	Number	Description	Resolution

last-updated: Apr 11, 2022

CLDB	31346	Number of resync are large for an extended time after a couple of nodes are stopped/restarted in the cluster	The issue occurs when there are a large number of snapshots on a cluster. There are new alarms to warn about this issue, and are configurable. See the following resources: • Cluster Alarms on page 2222 • cldb.conf on page 2182 • Configuring the Alarm Threshold Using the CLI on page 786
	31620	Volume mirroring floods log with misleading error	The bug occurs when mirroring happens between clusters that have both internal and external IPs set in `mapr-clusters.conf`. The getIPTypeForCluster method in CLDBRpcCommonUtils is unable to determine whether the IP type is internal or external. The workaround is to put in the internal IP in `mapr-clusters.conf` and keep the external IP in the `env.sh` file. See: Starting the Mirror on page 919
Upgrade	31038	Prior to 6.1, the log4j.properties file was not automatically updated in an upgrade	All files update automatically.
Upgrade	29752	Prior to 6.1, if Oozie was not upgraded to the EEP 4.0.0 version, the Oozie process would fail following a manual upgrade from MapR 5.2.x/ EEP 3.0.1 to MapR 6.0	The upgrade process works correctly.

FileClient	31024	When `hadoop fs -rmr` command is run to remove a list of files/directories, if a file/directory is not present at the time of removal, the command returns an error, 'no such file or directory', and terminates without removing remaining files and/or directories from the given list.	When `hadoop fs -rmr` command is run to remove a list of files and/or directories, if a file/directory from the list is not present on the system, the command now removes remaining files and/or directories from the list.
	30987	Memory leak in DoPathWalk	This leak is fixed.
	31026	FileClient should use one source port to connect to any server	This issue occurred because FileClient was using multiple source ports to connect to file servers, thus exhausting all available ports. This issue is now fixed to let FileClient connect on a single source port. See Client Side Port Binding in What's New in Version 6.1.0 on page 34
	31129	File Client crashed at mapr::fs::CidCache::GetBinding	The issue occurred because the assumption was that the number of volume replicas will not exceed 7. This issue is fixed.
	31146	The SQL queries are failing intermittently	The issue occurred because user impersonation was faulty. This issue is fixed.
	31738	Create request should take setattr credentials from ticket	To avoid this issue, all requests should take the credentials from the ticket, and not use the context user. This problem is fixed. See Managing the FUSE-Based POSIX Client on page 1255
	31804	FileClient hangs at WaitUntilEnqueued	This is a backport of the fix in #24266, where the Idle Flusher needed to be disabled to resolve hangs, for MapR version 5.2.2. This fix is backported.

FileServer	26792	createTTVolume.sh needs to reliably determine the MFS state before deciding to recreate the local NM volume	This bug caused rolling updates to fail. This issue is fixed.
	30063	FCR sent during disk I/O error causes all 3 copies of container to be unavailable	This bug occurred because the primary filesystem instance reported incorrect replica chain management information. This issue is fixed.
	30917	EIO on a read op of a local volume cid deletes/offlines the container but local volume in not recreated	This issue was caused because Volume IDs were not being passed in HandleDamagedVolumes() when called from the ContainerOffline case, causing the volumes to be deleted and not recreated automatically. This bug is fixed.
	31007	FUSE clients do not honor impersonation constraints in servicewithimpersonation tickets	This issue was caused when FUSE failed to honor constraints for a servicewithimpersonation ticket which includes impersonatedgids constraints. This issue is fixed, and FUSE now enforces such constraints. A support advisory is available at https://support.datafabric.hpe.com/s/article/FUSE-Clients-do-not-honor-impersonation-constraints-in-servicewithimpersonation-tickets?language=en_US
	31301	Some snapcids on mirror volumes that are marked for delete never get deleted	This issue is fixed.
	31361	Rename operation fails on tables and streams	This issue occurred when a table or stream already existed in a rename operation... that is, in mv x y -> y already exists. This issue is fixed.
	31365	MFS Rpc thread on CLDB node is running out of CPU	This issue occurred because the process that reads from, and writes data to the key-value store, was not offloaded to the compression thread. This issue is fixed.
	31453	MFS on CLDB secondary instances crashed due to failure in kvstorerangedelete (ENOENT)	This bug occurred due to corruption of the structure that holds information about volume containers. This issue occurred due to a large number of containers being present in the volume. This issue is fixed. Appropriate documentation that depicts how to set the alarm for too many containers is available at Configuring the Alarm Threshold Using the CLI on page 786 (see CLUSTER_ALARM_TOO_MANY_SNAPSHOT_CONTAINERS), and cldb.conf on page 2182
	31981	Disk failure on Master (A) container when reporting loss of B to CLDB, can cause all copies to be unavailable	The problem arose because there was no sanity check to check for the validity of a replica, when the reconnection timer expires. This check has been added.
FS::ACE	26280	"hadoop mfs -setace" does not accept groups with spaces in the name	Spaces in AD group names caused the issue. This issue is fixed. Spaces in AD group names are correctly parsed.
	30245	Permission denied error due to aceCache when readdir served by the primary node of NC	This issue occurred due to a null character in the ACE expression, as the ACE for execute file was null. This issue has been resolved.

last-updated: Apr 11, 2022

FS::Audit	30928	expandaudit does not resolve most of fids in a huge volume	This issue is fixed by making logging more explanatory.
FS::Fuse	31662	Fuse : OPEN with O_TRUNC fails with permission denied error	This issue is fixed.
	31730	Fuse: mkdir fails with EBUSY	The issue is caused by inode number reuse. The work around is to set the `fuse.use.compressed.inode.format` parameter to 1 as documented in Configuring the MapR FUSE-Based POSIX Client on page 1240
	32030	Fuse Assert in fs/client/fuse/cc/fuse_special_ll.c	This issue was caused by Fuse reading debugs_assert() from the wrong location. This issue is now fixed.
	32050	Fuse needs to honor prod_build	This issue was caused by Fuse reading debugs_assert() from the wrong location. This issue is now fixed.
	32074	Memory leak in fuse cache	This leak is fixed.
FS::Snapshot	31051	MCS & maprcli command are not showing correct Volume size	This issue occurred because some snapshot containers might have had a delayed deletion. This issue is fixed.
hoststats	31858	Hoststats process is not coming up after changing the default value (5660) of "mfs.server.port" in /opt/mapr/conf/mfs.conf	The problem occurred because the ports were hadcorded in MapR settings. Using a port other than the default, causes the shared memory key to fail. This issue is fixed.
MCS	MON-3709	The list of services in the **Services** page do not refresh per the configured refresh rate settings.	The list of services in the **Services** page refresh per the configured refresh rate settings.
MapR Monitoring	SPYG-1010	The Grafana dashboard shows "No data points" for Volume metrics	Dashboard entries are correct.
	SPYG-916	MapR monitoring index is not loaded correctly.	The MapR monitoring index loads correctly.
MapR Event Store for Apache Kafka	31074	[Kafka 1.0] incorrect behavior for offsetsForTimes when streams.rpc.timeout.ms is configured	Behavior corrected for offsetsForTimes when streams.rpc.timeout.ms
Elasticsearch	ES-27	Elasticsearch fails to start correctly	Elasticsearch starts correctly.

MapR Database	29278	Puts to tablets that failed with out of space errors, continue to fail on the first put, even though there is sufficient space; subsequent puts succeed.	When space becomes available, the very first put no longer returns an error.
	30489	Client rpc trace messages for slow operations are not printing tablet `fid`	The messages include tablet `fid`.
	31092	MapR filesystem nodes are crashing when there is a burst of read requests	Corrected batching of read requests to avoid the condition that was causing the crash.
	31297	Using Python `happyhbase` module to scan a MapR Database table via HBase Thrift causes the Thrift server to hang	Addressed underlying issue that was causing the hang.
	31766	MapR filesystem node crash causes client RPC timeouts during get and put operations	Corrected the underlying logic for notification calls when creating new MapR Database buckets.
	31901	Memory corruption occurs when deleting/ freeing memory	Corrected race conditions that were causing the corruption.
	32262	Incremental bulkload via MapReduce fails to load data, even though it does not report an error	Corrected the underlying logic that was causing the incremental load to fail without returning an error.
NFS	31810	Unable to mount specific volume, when `cldb.reject.root` is enabled	Enabling `cldb.reject.root` caused NFS mounts to fail. This issue is fixed.
security	31935	All versions of MapR have Serious Ticket Vulnerability Enabling Authority Escalation	This issue was caused by CLDB generating tickets from falsified credentials. This issue is fixed. A security bulletin is available at https://support.datafabric.hpe.com/s/article/MapR-Ticket-Credentials-can-become-compromised?language=en_US and the vulnerabilities list is updated and available at Security Vulnerabilities on page 6569
Warden	31628	maprcli urls show the wrong value for hbmaster after shutdown of current HBmaster	This issue occurred because maprcli had stale information on a HBase Master process that was killed. This issue is fixed.

YARN	25473	Aggregated log is written with wrong ownership	Addressed the underlying issue that was causing writing the aggregated log to the wrong user.
	31082	ResourceManager address does not resolve after redirect via a proxy request	Fixed incorrect condition that updated the current ResourceManager address.
	31174	In the primary application log, the following error repeatedly occurred: "Tez AM is trying to access Timeline server and it fails"	TEZ AM is released after job or session is completed.
	31200	YARN job submission using server side uid resolution fails with ownership exception	The problem occurred because the username was not read from the submitted ticket. This issue is fixed.
	31487	Containers fail in LOCALIZING state	Implemented clean up of subprocesses spawned by Shell when the process exits. Localization failures are available in the container diagnostics.
	31679	CVE-2016-6811 user who can escalate to Yarn user can possibly run arbitrary commands as a root user	Fixed the underlying security vulnerability.
	32178	Thread leak when executing a Scoop job	Fixed thread leak.
	32304	Fails to refresh labels for nodes in the cluster	DefaultContainerExecutor sets proper permissions.

Packages and Dependencies for MapR Software

Package and dependency details for MapR 6.1 platform and ecosystem components.

For downloadable packages, see these links:

- MapR Core Packages
- EEP Packages
- MapR Installer Packages

For MapR Installer package dependencies, see:

- MapR Installer Prerequisites and Guidelines

This table shows the MapR 6.1 package dependencies for Red Hat / CentOS and SUSE. A dash (—) indicates no dependency.

	Depends on . . .	
This package . . .	**These MapR packages**	**And these non-MapR packages**
`mapr-apiserver`	`mapr-core`	—
`mapr-cldb`	`mapr-core` `mapr-fileserver`	—

`mapr-client`	`mapr-librdkafka`	`glibc`
		`libgcc`
		`libstdc++`
		`syslinux`

last-updated: Apr 11, 2022

This package . . .	Depends on . . .	
	These MapR packages	**And these non-MapR packages**
mapr-compat-suse	—	/etc/default/useradd /sbin/rpcinfo /usr/sbin/acpidump libffi4 libgcc_s1 libsnappy1 libstdc++6 openssl
mapr-core	mapr-core-internal mapr-hadoop-core mapr-librdkafka (version 0.11.3 or later) mapr-mapreduce2	—
mapr-core-internal	—	dmidecode glibc hdparm initscripts irqbalance libgcc libstdc++ nss (version 3.19 or later) redhat-lsb-core sdparm shadow-utils syslinux
mapr-fileserver	mapr-core	—
mapr-hadoop-core	mapr-core-internal	—
mapr-historyserver	mapr-mapreduce2	—
mapr-loopbacknfs	—	iputils nfs-utils redhat-lsb-core rpcbind
mapr-mapreduce2	mapr-hadoop-core	—
mapr-nfs	mapr-core	/usr/sbin/rpcinfo iputils nfs-utils
mapr-nodemanager	mapr-mapreduce2	—

last-updated: Apr 11, 2022

This package . . .	Depends on . . .	
	These MapR packages	**And these non-MapR packages**
mapr-posix-client	mapr-client	—
mapr-resourcemanager	mapr-mapreduce2	—
mapr-single-node	mapr-cldb mapr-fileserver mapr-nfs mapr-webserver mapr-zookeeper	—
mapr-upgrade	mapr-core	rpmrebuild
mapr-webserver	mapr-apiserver	—
mapr-zk-internal	—	—
mapr-zookeeper	mapr-core mapr-zk-internal	—

These MapR 5.x packages were removed for MapR 6.x:

- mapr-jobtracker
- mapr-mapreduce1
- mapr-metrics
- mapr-tasktracker

This table shows the MapR 6.1 package dependencies for Ubuntu:

This package . . .	Depends on . . .	
	These MapR packages	**And these non-MapR packages**
mapr-client	mapr-librdkafka (version 0.11.3 or later)	awk bash (version 2.05a-11 or later) coreutils (version 5.0-5 or later) grep (version 2.4.2-3 or later) libc6 libgcc1 libstdc++6 perl procps (version 1:2.0.7-8 or later) sed (version 3.02-8 or later) syslinux

This package . . .	Depends on . . .	
	These MapR packages	**And these non-MapR packages**

`mapr-core-internal`	—	`adduser` (version 3.11 or later)
		`awk`
		`bash` (version 2.05a-11 or later)
		`coreutils` (version 5.0-5 or later)
		`dmidecode`
		`grep` (version 2.4.2-3 or later)
		`hdparm`
		`libc6`
		`libgcc1`
		`libstdc++6`
		`lsb-base`
		`irqbalance`
		`perl`
		`procps` (version 1:2.0.7-8 or later)
		`sdparm`
		`sed` (version 3.02-8 or later)
		`syslinux`
		`sysvinit-utils`

`mapr-loopbacknfs`	—	`iputils-arping`
		`lsb-base`
		`rpcbind`

`mapr-nfs`	`mapr-core`	`awk`
		`bash` (version 2.05a-11 or later)
		`coreutils` (version 5.0-5 or later)
		`grep` (version 2.4.2-3 or later)
		`iputils-arping`
		`nfs-common`
		`perl`
		`procps` (version 1:2.0.7-8 or later)
		`sed` (version 3.02-8 or later)

Patches and Documentation

Describes important considerations for patches and patch documentation.

Whenever possible, keep your software up to date by applying the latest patches available on the MapR Support Portal. This practice can help you to resolve issues and minimize downtime.

Some patches enable new features or behaviors that are described in the documentation. However, the MapR documentation does not typically include patch numbers or identify the features or behaviors that are delivered by specific patches. If you see a fix or feature in the documentation that is not available on your platform, you might need to apply a patch in order to use the fix or feature.

last-updated: Apr 11, 2022

To understand which patches apply to your platform, contact your MapR support representative, or visit the Support notices of known issues. For information about applying a patch, see Applying a Patch on page 437.

Version 6.1.1 Release Notes

These release notes contain information about Version 6.1.1 of the MapR Converged Data Platform.

To review the support advisory announcing release 6.1.1, see Announcement of HPE Ezmeral Data Fabric maintenance release 6.1.1 to address 6.1.0 critical defects.

What's New in Version 6.1.1

Describes fixes and new features contained in release 6.1.1.

MapR version 6.1.1 supports SuSE Linux Enterprise Server version 12 SP5 and EEP 6.3.3. In addition to the information on this page, see What's New in EEP 6.3.3 on page 6524.

Filesystem Additions

CORE-517	Added the `-no-auto-permission-update` option to prevent MapR from silently altering the `/etc/shadow` file during reconfiguration.
MFS-2343	Enhanced the node list command. Using the node list command without the `-clientsonly true` or the `-nfsnodes true` option, does not list edge nodes. To include edge nodes, use the `-nfsnodes true` or the `-clientsonly true` option.
MFS-2344	Added a new CLDB parameter cldb.ignore.posix.only.hb.alarm that controls whether this alarm is raised for edge nodes.
MFS-2749	Added a parameter cldb.disable.alarm.history to disable alarm history.
MFS-11243	Added a FUSE tunable (fuse.max.cache.pages) to limit the amount of memory that each FUSE process can use when working with a large number of open files.
MFS-11674	Added a new `-D|--crc` option to enable gfsck to perform CRC checks without blocking the operations on the EC frontend volume.

MCS Additions

MON-5511	Added Snapshot Size as a configurable column in the Volumes List view.

Operating System Support

MapR version 6.1.1 supports SuSE Linux Enterprise Server version 12 SP5.

Performance

MAPRDB-2250	Added a parameter `mfs.db.max.concurrent.internal.ops` to regulate the number of parallel *BatchGet* operations.

last-updated: Apr 11, 2022

MAPRMR-8	Added the parameter mapreduce.input.fileinputformat.split.maxblocknum that determines the number of blocks that can be added to one split.
MFS-2078	Added the `fuse.negative.timeout` parameter to cache negative lookup results, and speed up FUSE path lookups.
MFS-4750	Added a tunable - prevent.volume.skew.by.diskbalancer to let the Disk Balancer allow or prevent volume skew.
MFS-4805	Added the following entities in `/opt/mapr/conf/nfsserver.conf`: • `MemDebugEnable` - Set to true to enable Memory Debugging. • `HighMemLimitMB` - Sets the maximum amount of memory that the NFS Server can use.

Security

CORE-384	Added a parameter `JMXDISABLE` to enable or disable loading ZooKeeper JMX parameters.
MFS-3310	Added the Security Certificate Expiry Alarm.
MFS-5236	Added a new parameter warden.enable.jmxremote that must be explicitly set to `true` to enable the Warden JMX Server.

YARN

MAPRMR-19	Two options have been added to YARN to retry fetching job statuses. • `yarn.app.mapreduce.job.update-status-max-retries` - The number of times to retry. • `yarn.app.mapreduce.job.update-status-retry-interval` - The interval to wait before each retry attempt.
MAPRYARN-210	Added the Local Log Aggregation Feature.
MAPRYARN-221	Added two options: • `yarn.nodemanager.timeout-localizing-container` - The maximum time to wait to localize resources for containers. • `yarn.nodemanager.check-interval-localizing-container.ms` - The frequency at which the ApplicationMaster checks the running time of the localizing container.
MAPRYARN-250	Added a parameter, mapreduce.jobhistory.intermediate-done-scan-timeout

last-updated: Apr 11, 2022

to set the timeout in milliseconds for rescanning the `done_intermediate` user directory.

Other Features

For a list of other features, see What's New in Version 6.1.0 on page 34

Installation and Upgrade Notes (MapR 6.1.1)

Describes considerations for installing and upgrading to MapR 6.1.1.

If you are upgrading from MapR 5.x, be sure to review the Installation and Upgrade Notes (MapR 6.0.0) first. Installation and upgrade considerations can also be found in the support advisory announcing release 6.1.1. See Announcement of HPE Ezmeral Data Fabric maintenance release 6.1.1 to address 6.1.0 critical defects.

Installation Considerations

Installation of MapR 6.1.1 is the same as for other core releases. You may install the packages manually or by using the MapR Installer, as described in:

- Installing with the MapR Installer on page 141
- Installing without the MapR Installer on page 141

However, unlike other releases, MapR 6.1.1 supports only one EEP: EEP 6.3.3. For EEP support information, see EEP Support and Lifecycle Status on page 5531.

Upgrade Considerations (All Upgrade Methods)

Note these considerations for upgrading to MapR 6.1.1, which apply regardless of the method you use to upgrade:

Supported Upgrade Paths	You can upgrade to MapR 6.1.1 from the following releases: • MapR 5.2.2 • MapR 6.0.x • MapR 6.1.0 The upgrade from MapR 6.1.0 to 6.1.1 is a full package upgrade and not simply a patch update. As with any upgrade, HPE recommends applying the latest patches for the release to which you are upgrading. For upgrade instructions, see Upgrading MapR Core or EEP Components on page 284.
Support Advisory for Upgrading to MapR 6.1.x	If you are upgrading a cluster with volumes created from MapR 4.0.1 or earlier, you might be subject to the issue described in Advisory 4176. Be sure to review the advisory before upgrading to MapR 6.1.x.
Upgrading to MapR 6.1.x Might Require an OS Upgrade	If you want to upgrade to MapR 6.1.x but your cluster does not use a supported Linux operating system, you must upgrade your operating system before starting the upgrade. For the list of operating system versions supported by MapR 6.1.x, see Operating System Support Matrix (MapR 6.x).
Patch Required Before Upgrading to EEP 6.3.4	**CAUTION:** EEP 6.3.4 requires a core patch to resolve a Warden defect. The defect is fixed in `mapr-patch-6.1.0.20180926230239.GA-20210512163609.x86_64` and later. Before upgrading to EEP 6.3.4, you must apply the patch. See Patches and Documentation for MapR 6.1.1 on page 96.

Data-on-wire-encryption	Beginning with MapR 6.1.0, data-on-wire encryption is enabled by default for newly created volumes on secure clusters. Data-on-wire encryption encrypts data in a volume during transmission over the wire. Data-on-wire encryption is *not* supported for non-secure clusters. When you upgrade a cluster to a MapR 6.1.1 secure cluster, data-on-wire encryption is enabled by default. You can disable data-on-wire encryption for individual volumes using MCS, the maprcli, or REST API commands.
MapR 6.1.1 and EEP 6.3.3	MapR 6.1.1 requires EEP 6.3.3. EEP 3.0.1 or later can coexist with MapR 6.1.x only temporarily in the context of an upgrade. For EEP compatibility information, see EEP Support and Lifecycle Status on page 5531.
Metrics Monitoring	To support metering, MapR 6.1.1 requires a minimal level of metrics monitoring to be configured. If metrics monitoring is already configured before the upgrade, you must upgrade it as part of the MapR Ecosystem Pack upgrade. If metrics monitoring is not installed, you must install it after the upgrade. When you upgrade to MapR 6.1.1 using the MapR Installer, metrics monitoring is configured and enabled by default. See Installing Metering Using the MapR Installer on page 283. When you upgrade to MapR 6.1.1 using manual steps, you must install metrics monitoring after the upgrade. See Installing Metering Using Manual Steps on page 284.
Log Monitoring Upgrades	If log monitoring is configured on your cluster and you plan to use either the MapR Installer or manual steps to upgrade to MapR 6.1.1, you might need to migrate or back up the Kibana and Elasticsearch indexes to version 6 before starting the upgrade. For more information, see Pre-Upgrade Steps for MapR Monitoring on page 342. Migrating or backing up the indexes is not needed if log monitoring is not configured or if you are performing a new installation.
Regenerating the `maprusserticket` File	Changes to the `CanImpersonate` parameter of the `maprusserticket` file in MapR 6.1.1 require users who upgrade manually to regenerate the file before restarting Warden. See Step 1: Restart and Check Cluster Services on page 323. The file needs to be regenerated to ensure that impersonation works correctly for non-`mapr` users. Prior to MapR 6.1.0, all `maprusserticket` files were generated with `CanImpersonate = false`. MapR 6.1.0 and later enforce the `CanImpersonate` parameter and sets the parameter to `true` for freshly installed clusters. For upgraded clusters, if `CanImpersonate` is not set to `true`, some services will not be able to impersonate.
Upgrade Workflows	The following upgrade workflows can help you understand the scope of activities involved in an upgrade based on the method you use: • Workflow: Manual Rolling Upgrade on page 287 • Workflow: Offline Manual Upgrade on page 289 • Workflow: MapR Installer Upgrade on page 292
OpenTSDB and Upgrades to EEP 6.0.1 or Later	In EEPs 6.0.1 and later, the OpenTSDB service is configured to use a default heap size of 6 GB. In earlier EEPs, the default was 2 GB. Upgrading to EEPs 6.0.1 and later causes the default setting to change to 6 GB. If you configured a custom value for the OpenTSDB service heap size and you want to reinstate the custom value after upgrading to EEP 6.0.1 or later, see Configure the OpenTSDB Service Heap Size on page 1380.

last-updated: Apr 11, 2022

Professional Support for Upgrades	Upgrading can be time-consuming and complicated. Consider engaging HPE professional support services to assist in planning and executing your upgrade. For more information, contact your HPE support representative.

Upgrade Considerations for the MapR Installer

Note these considerations for upgrading to MapR 6.1.1 that are specific to the MapR Installer:

MapR Installer Version	Before upgrading, update the MapR Installer to the latest MapR Installer. Only MapR Installer 1.15.0.1 and later can be used with MapR 6.1.1 and EEP 6.3.3.
Maintenance Update	If your cluster is installed with MapR 6.1.0 and EEP 6.3.3, you can use the maintenance update feature of the MapR Installer to perform the upgrade. Otherwise you must use the MapR Installer version upgrade feature to perform the upgrade. For more information about maintenance updates, see Performing a Maintenance Update on page 5447.
Version Upgrade	Except in cases where a maintenance update is supported, upgrading to MapR 6.1.1 using the MapR Installer requires you to perform a version upgrade.
Changes to Auto-Provisioning Templates	MapR 6.1.0 introduced changes to some auto-provisioning templates. See the "MapR Installer 1.10 Updates" described in Auto-Provisioning Templates.
Special Consideration for MySQL Database	If you are upgrading from MapR 5.x to MapR 6.1.x and your cluster is configured with a MySQL database, the installer prompts you for the MySQL password during the upgrade. You must enter the MySQL password that you specified when you first installed the cluster. If you did not specify a password for MySQL, you must leave the password field blank during the upgrade operation. If you specify the wrong value for the password, the upgrade can fail. See the known issue for IN-1972 in MapR Installer Known Issues.

List of YARN Enhancements

Describes the enhancements in YARN, if any, for MapR version 6.1.1.

No new enhancements for MapR version 6.1.1.

For a list of YARN enhancements in MapR 6.1.0, see List of YARN Enhancements.

Operational Changes (MapR 6.1.1)

Lists the functional changes, if any, made to existing commands in MapR version 6.1.1.

There are no functional changes in MapR 6.1.1.

For a list of functional changes in MapR 6.1.0, see Operational Changes (MapR 6.1.0) on page 43.

Known Issues at Release (MapR 6.1.1)

Lists the known issues that users should be aware of before installing or using release 6.1.1.

You may encounter the following known issues after upgrading to Version 6.1.1. This list is current as of the release date. For a dynamic list of all known issues for all MapR product releases, refer to Support notices of known issues

Where available, the workaround for an issue is also documented in this topic. MapR regularly releases maintenance releases and patches to fix issues. We recommend checking the release notes for any subsequent maintenance releases to see if one or more of these issues are fixed.

last-updated: Apr 11, 2022

Installation and Configuration Issues

You can see generic installation issues here: MapR Installer Known Issues on page 5460.

Keytool error on SLES 12 SP4 Node

Running the `configure.sh` script on a SLES 12 SP 4 node can fail with the following error:

```
keytool error:
java.security.ProviderException:
Could not initialize NSS
```

The installation fails because the `mozilla-nss` dependency is not installed.

Workaround: Install the `mozilla-nss` package using the following command, and rerun `configure.sh`:

```
zypper install mozilla-nss
```

IN-2637

After a manual installation, Oozie and Hive services can fail to connect to a MySQL or MariaDB database because the server time-zone value is unrecognized or represents more than one time zone. The issue affects your installation if you applied the `mapr-patch` released on or after February 21, 2021 (including the latest `mapr-patch`). This issue affects manual installations but is fixed in Installer 1.14.0.0.

Workaround: For manual installations, you must configure either the server or JDBC driver (using the `serverTimezone` configuration property) to use a more specific time-zone value if you want to utilize time-zone support. After running `configure.sh` but before starting the Oozie or Hive services, update the `serverTimezone` parameter in the `hive-site.xml` or `oozie-site.xml`. For more information, see MySQL Bug #95036.

Monitoring

ES-75

Elasticsearch purge jobs fail in MapR 6.1.0 with EEP 6.3.1 because certain curator files are missing.

Workaround: A patch is required for this issue. See Known Issue article 4812 in the Support notices of known issues. To download a patch, see Downloading a Patch on page 437. EEP 6.3.3 and later include the fix for this issue.

ES-76

Running the curator purge script in Elasticsearch in EEP 6.3.2 or EEP 7.0.1 returns a syntax error.

Workaround: A patch is required for this issue. See Known Issue article 4812 in the Support notices of known issues. To download a patch, see Downloading a Patch on page 437. EEP 6.3.3 and later include the fix for this issue.

ES-77

During an upgrade from release 6.1.0 and EEP 6.3.x to release 6.2.0 and EEP 7.0.1, Elasticsearch, Kibana, and Grafana packages do not get upgraded and remain at the EEP 6.3.x package version. The issue occurs because of a misnumbered fourth digit in the EEP 7.0.1 packages for Elasticsearch, Kibana, and Grafana. This issue affects manual upgrades and

last-updated: Apr 11, 2022

	version upgrades using the Installer. This issue does not affect new installations. **Workaround**: After the upgrade, manually uninstall the `mapr-elasticsearch` and `mapr-kibana` packages using a command such as `yum remove`. Then reinstall the packages. If you installed using the Installer, run an **Incremental Install** to reinstall the packages. If you installed the cluster manually, use a command such as `yum install` to reinstall the `mapr-elasticsearch` and `mapr-kibana` packages. For a guide to the component package versions, see Component Versions for Released EEPs on page 5586.
NFSv4	
MFS-11919	`nfs-ganesha` crashes when starting NFSv4 server on SLES 12 sp4/sp5 nodes. However, subsequently NFSv4 is able to come up and work fine. **Workaround**: None.
Upgrade	
IN-2824	During a version upgrade using the Installer, the upgrade fails during the Verification phase with a message that the OS is not supported. Hovering over the error in the right-navigation pane indicates that your OS is not supported for the core version you selected, and the core version is displayed as 5.1.0. Troubleshooting indicates that the `install.json` file includes EEP components but no core services in the services list. This can happen because of a timing issue in the Installer user interface. **Workaround**: You might be able to avoid this issue by incorporating a brief delay when you use the **Maintenance Update** or **Version Upgrade** function. After specifying the core version and the EEP version, wait a minute or two before clicking **Next** to advance to the next screen. This delay gives the Installer time to process the selections that you made. To recover from the upgrade failure: **1.** Reset the Installer database as described in Resetting the Installer Database on page 5459. **2.** Import the last known state by following the steps for importing the cluster state in Importing or Exporting the Cluster State on page 5447. **3.** Retry the upgrade through the Installer.
Impersonation	
MFS-11943	Impersonation does not work by default on insecure clusters because the admin file is not automatically created in the `/opt/mapr/conf/proxy` directory upon initial cluster configuration. **Workaround:** Manually create an empty admin file in `/opt/mapr/conf/proxy`, for example `/opt/mapr/conf/proxy/mapr`, and then restart services.

File System

MFS-14890

Issuing `hadoop mfs -lsfid` with a non-`mapr` user ticket returns an `Operation not permitted` error. For example:

```
2022-03-15 18:12:23,6663 DEBUG
Cidcache fc/cidcache.cc:4315 Thread:
18743 Enter GetVolumeMountPoint
138102135
2022-03-15 18:12:23,6663 DEBUG
Cidcache fc/cidcache.cc:4754 Thread:
18743 Sending RPC to CLDB:
10.163.172.130:7222
2022-03-15 18:12:23,6672 DEBUG
Cidcache fc/cidcache.cc:4193 Thread:
18743 GetVolumeProperties returns: 0,
numTries:0
2022-03-15 18:12:23,6673 ERROR
Cidcache fc/cidcache.cc:4277 Thread:
18743 VolumePropertiesLookupRequest
failed, cldb returned err
Operation not permitted(1), CLDB:
10.163.172.130:7222
2022-03-15 18:12:23,6673 DEBUG Client
fc/client.cc:11139 Thread: 18743
GetVolumeMountPoint returns: 0
```

In a secure cluster, this happens because the `hadoop mfs -lsfid` command works only when the UID associated with the ticket has either READ permission or full permission at the cluster level.

Workaround: Assign read permission to the UID by using an ACL to give the non-`mapr` user login permission. For example:

```
# maprcli acl  edit -type
cluster -user user1:login
maprcli acl show -type cluster
Allowed actions             Principal
[login, ss, cv, a, fc, cp]  User mapr
[login, ss, cv, a, fc, cp]  User root
[login]                     User user1
```

Resolved Issues (MapR 6.1.1)

Lists the issues that were resolved in MapR version 6.1.1.

The following MapR issues, which were reported by customers, are resolved in Version 6.1.1.

Apache Kafka

MS-791

KafkaConsumer position does not honor TTL.

MS-946

Consumer poll returns messages before a subscription or re-balance operation with *ConsumerRebalanceListener* on *PartitionsAssigned* callback is complete.

MS-947

Kafka message headers should not be Null but can be empty.

last-updated: Apr 11, 2022

Control System

MON-4844	Navigating to Usage tab when editing a volume causes OOPS error.
MON-4845	API server fails to start if the user is not *mapr*.
MON-4902	Default snapshot policies and times are incorrect.
MON-5075	Cannot modify existing volume quota due to insufficient permissions.
MON-5102	MCS does not allow configuring SMTP without a username and password.
MON-5131	OOPS error when adding a SMTP provider other than smtp, Office 365, and Gmail.
MON-5169	MCS call to schedule/list returns an error.
MON-5194	Incorrect scale on Y-axis for CPU Utilization graph on Overview page.
MON-5206; MON-5141	Volume modification fails when the metrics feature is not enabled.
MON-5270	Memory leak in API server.
MON-5271	Remove the limit on the number of results returned for MAC addresses when configuring a virtual IP.
MON-5272	User name should not be hardcoded as `mapr` in MCS.
MON-5275	MCS displays MAC addresses incorrectly.
MON-5338	I/O errors occur and the table summary page is not loaded when huge MapR DB tables are queried.
MON-5489	Volume filter options should not be pre-defined but should allow filtering on any volume attribute.
MON-5491	Detailed alarm descriptions were missing in the alarm popup view and the volume details view.
MON-5510	The Disk Space Available column uses the wrong units in the Nodes view.
MON-5517	Enhance the alarm summary command to return the alarm occurrences for each alarm type, indicate whether the alarm is a warning or an error, and specify the total number of occurrences for each alarm type.
MON-5672	The Security Certificate Expiry Alarm (*NODE_ALARM_CERTIFICATE_NEAR_EXPIRATION*) is incorrectly labeled.
MON-5774	API Server crashes when the *tmp* directories are mounted with the `noexec` option.
MON-5900	Resource Manager URL is invalid on Ubuntu.

last-updated: Apr 11, 2022

Flume

MS-770	Array Index Out of Bounds Exception.

Filesystem

CORE-387	MapR user tickets get overwritten when Kerberos authentication is enabled.
CORE-416	The configure.sh script must support environments where root activity is not allowed.
CORE-427	Local Spark shuffle volumes if damaged, need to be automatically recreated when NodeManager starts.
CORE-472	The disk list command runs very slowly and fails to identify the right MapR-FS disk when more than one symlink points to the same disk.
CORE-476	Running configure.sh with the `-R` option, erases the value of the `MAPR_JMXAUTH` variable in the *env_override.sh* file. This prevents NodeManager from starting.
CORE-480	Volume rack path is not updated for volumes with local path set when moving nodes.
CORE-517	The configure.sh script should not silently alter permissions in the `/etc/shadow` file. Added the `-no-auto-permission-update` option as the fix.
CORE-566	The *hoststats* process crashes continuously and causes cluster auditing to fail.
CORE-571	Oozie server does not start because of a missing jar file.
MFS-1984	The maprcli dashboard info command returns incorrect compression statistics.
MFS-1985	POSIX Client service (FUSE) does on auto start on system reboot on Ubuntu 16.
MFS-2015	The maprcli dashboard info command returns incorrect memory statistics.
MFS-2019	API Server hangs intermittently and fails to access CLDB servers, in a multi-NIC environment.
MFS-2051	Improve the clarity of NFS logs.
MFS-2055	CLDB crashes when processing alarms.
MFS-2062	Remote mirroring fails repeatedly even after a source CLDB that went down is restarted and operational.
MFS-2143	MFS does not preserve excluded volume audit data operations on restart.
MFS-2144	Spark streaming tasks are stuck indefinitely when looking up tablets.

last-updated: Apr 11, 2022

MFS-2209	Exception occurs when calling the getVolName() function.
MFS-2211	Name Container master freezes during resync of orphan entries and causes MFS and the resync operation to restart frequently.
MFS-2218	MFS randomly crashes if errors occur when reading data.
MFS-2260	MapR jobs fail since the file client fails to check the MapR filesystem to determine the status of the RPCs sent previously to MFS, before resending them.
MFS-2266	Spark encounters RPC errors when reading files from volumes with `wiresecurity` enabled.
MFS-2273	Storage Pools fail randomly when MFS is restarted, and many containers go offline without a valid replica.
MFS-2275	Spark jobs fail intermittently when trying to retrieve rows from MapR DB tables.
MFS-2294	CLDB crashes when registering NFS version 4 servers.
MFS-2298	CRC errors occur randomly in Storage Pools and cause them to go offline.
MFS-2306	Master CLDB crashes when MFS nodes are added or removed frequently.
MFS-2307	Add an internal cluster level flag to prevent storage pools from going offline when Read CRC errors are encountered.
MFS-2323	NFS Server version 4 boot script should not contain hardcoded user and group (*mapr:mapr*).
MFS-2343	The node list command should not display nodes which contain only the POSIX client (edge node). Using the node list command without the `-clientsonly true` or the `-nfsnodes true` option, does not list edge nodes. To include edge nodes, use the `-nfsnodes true` or the `-clientsonly true` option.
MFS-2344	The NODE_ALARM_NO_HEARTBEAT (No Heartbeat) alarm should not be raised for POSIX clients (edge nodes). CLDB has a new parameter cldb.ignore.posix.only.hb.alarm that controls whether this alarm is raised for edge nodes.
MFS-2392	`gfsck` fails on secure clusters due to a missing library.
MFS-2423	POSIX only clients should be immediately removed when marked dead.
MFS-2444	FUSE process does not remove shared memory segments resulting in volumes failing to mount.
MFS-2462	Crash in MapR DB when looking up role memberships.

last-updated: Apr 11, 2022

MFS-2498	FUSE does not honor the product build value.
MFS-2573	Mirroring fails with a CLBD internal error when an invalid container ID is found on the source cluster.
MFS-2610	Persistent Volume mounts hang when tickets expire.
MFS-2628	Null Pointer Exception in CLDB Server.
MFS-2630	The mrconfig info threads command crashes the MFS process when attempting to retrieve volume aces, when Extended Attributes are not enabled.
MFS-2631	CLDB shuts down when adding or removing NFS servers.
MFS-2632	CLDB operations fail with the Server Retry error.
MFS-2659	MFS process hangs intermittently in environments with multiple NICs.
MFS-2694	Stack overflow in MFS when deleting a container chain.
MFS-2695	Cross cluster mirroring fails after enabling the Snapshot Lite feature.
MFS-2725	NFS Server version 3 crashes randomly when trying to satisfy mount requests.
MFS-2731	MFS does not automatically retry reconnecting to CLDB after a connection reset request.
MFS-2732	RPC connections between MFS and CLDB fail intermittently with Connection Reset by Peer errors.
MFS-2757	MapR service status is displayed incorrectly due to change in `systemd`.
MFS-2767	File client tries to connect to the same failed CLDB node repeatedly.
MFS-4480	NFS Server version 4 crashes intermittently.
MFS-4485	FUSE client fails to work with a scoped impersonation ticket.
MFS-4531	Snapshots of mirrors are not deleted after mirroring completes.
MFS-4551	`loopbacknfs` does not log any messages to the `loopbacknfs.log` file.
MFS-4562	File stat on the FUSE mount indicates the block size as a fixed value (512) instead of the client's block size.
MFS-4585	CLDB exception occurs when a NFS heartbeat reports a failed Virtual IP.
MFS-4597	Warden and the maprcli command intermittently cannot start dependent services.

last-updated: Apr 11, 2022

MFS-4605	CLDB shuts down when ACL size exceeds the threshold value.
MFS-4667	Replicated operations fail and cause frequent resyncs.
MFS-4776	FUSE RPCs fail intermittently.
MFS-5356	The `getAces()` API raises a Null Pointer Exception when called on a non-existing object.
MFS-5405	Client sends the `NODE_ALARM_SERVICE_NODEMANAGER_DOWN` alarm but CLDB raises the `NODE_ALARM_SERVICE_OPENTSDB_DOWN` alarm.
MFS-5422	The `create()` API does not create files with the same permissions as the parent directory.
MFS-5430	NFS Server is unable to parse lines exceeding 8K characters in the exports file.
MFS-5482	Memory leak in CLDB master instance.
MFS-5488	Jobs on random nodes fail to create FileClient.
MFS-5502	Path lookup error occurs when client nodes run a newer version of MapR than the CLDB server nodes.
MFS-5711	Unable to access files on EC tier due to I/O error and StripeletIO failure.
MFS-6585	Applications intermittently fail to detect updates to the ticket file.
MFS-6587	Avoid flooding the CLDB log with invalid snapshot ID messages.
MFS-6667	`hoststats` creates defunct Python processes.
MFS-6748	Automatic offload does not trigger EC offload.
MFS-6873	Cross cluster move operation fails on FUSE.
MFS-6874	Node Manager fails repeatedly during log aggregation.
MFS-8452	Memory leak in loopback NFS.
MFS-8459	Fixed volume access problems for volumes that reused the volume ID of deleted volumes.
MFS-10328	maprlogin renew (ticket renewal) fails to refresh group memberships.
MFS-10743	Node Manager fails to report container failure and loops between slave CLDBs, without contacting the new master CLDB.
MFS-10825	Cluster is unable to self-heal from the VOLUME_ALARM_DEGRADED_EC_STRIPES (Warm-Tier Data Node Down) alarm, and rebuild does not occur.

last-updated: Apr 11, 2022

MFS-10845	Volume creation fails to honor the credentials of the impersonated user while creating the parent directory.
MFS-11002	fsck crashes due to an inode reservation issue.
MFS-11109	Filesystem crashes due to a leak in orphanage reservation.
MFS-11171	Restrict the tenant ticket so that it cannot mount non-tenant volumes in POSIX.
MFS-11221	Drill query crashes in MapR client due to an unexpected exception during fragment initialization.
MFS-11243	Memory leak in FUSE. Added a FUSE tunable (`fuse.max.cache.pages`) to limit the amount of memory that each FUSE process can use when working with a large number of open files.
MFS-11295	Offloading fails when `mastgateway` is stuck in compaction state.
MFS-11442	FUSE client does not honour the location of the cluster configuration file as defined by the parameter `fuse.cluster.conf.location`.
MFS-11609	Hadoop `distcp` jobs fail when using CLDB hostname and port.
MFS-11647	SlowOPs trace function does not work for NFSv3.
MFS-11674	Enable `gfsck` to perform CRC checks without blocking the operations on the EC frontend volume. See gfsck on page 2102 for the new `-D|--crc` option.
MFS-11682	CLDB volume dump fails with an RPC error due to an unknown session key.
MFS-11729	`mrconfig info` threads crash the filesystem when hardlinks are not enabled.
MFS-11731	Suppress redundant incorrect build version alarms.
MFS-11740	Filesystem crashes when compacting memory.
MFS-11779	MFS dumps core due to stack overflow.
MFS-11823	Service ticket renewal does not honour duration.
MFS-11838	Jobs fail with the "Too many open files" error.

Hadoop

MAPRHADOOP-61	Kerberos fails for services when a custom ticket location is set in the `env.sh` file.
MAPRHADOOP-83	Upgrade Tomcat servers to their latest version or remove them if they are not needed.

last-updated: Apr 11, 2022

MAPRHADOOP-102	Error occurs in ACEs when the Hive resource downloader internally copies files from the MapR filesystem to the local filesystem.
MAPRHADOOP-131	Update Jersey to its latest 1.X version.

MapR-DB

MAPRDB-1236	The Tiny Bucket Flush alarm is raised even when the node has sufficient memory.
MAPRDB-1589	Incorrect key sorting when using the `orderby` clause with conditions.
MAPRDB-1719	DB server crashes when columnset is used without initialization.
MAPRDB-1732	Inserting data into MapR-DB fails intermittently with an *Invalid Argument* error.
MAPRDB-1889	The Java API findById() intermittently fails to retrieve complete projection details from JSON documents.
MAPRDB-1985	The `mapr dbshell find` command crashes when run on a table with a huge number of tablets.
MAPRDB-1995	MapR-DB raises intermittent false VOLUME_ALARM_TABLE_REPL_LAG_HIGH alarms for replicated streams.
MAPRDB-2062	Failed to scan table on a remote secure cluster using the `mapr dbshell` utility because of a wrong ticket that was sent to ZooKeeper.
MAPRDB-2072	Data Access Gateway (DAG) fails to fetch indexes as it queries indexes as the `mapr` user instead of the impersonated user.
MAPRDB-2091	MapR-DB hangs due to inodes being recycled even when they are in use.
MAPRDB-2092	In MapR-DB, adding a table index or replicating a table fails if the cluster administrator (MAPR_USER) does not have write access to the parent volume of the table.
MAPRDB-2098	MapR DB crashes when multiple threads modify the *size_* variable while calculating the serialised JSON document size.
MAPRDB-2103	PUT operations on binary tables fail when the values of the `wireSecurityEnabled` field vary between the FileClient and the FileServer.
MAPRDB-2120	Drill query on MapR DB intermittently fails with a DB Scan exception.
MAPRDB-2125	OJAI APIs fail to connect to ZooKeeper.
MAPRDB-2159	DB Autosetup, Indexing, and Replication fail as MFS is unable to reach the local Gateway.

last-updated: Apr 11, 2022

MAPRDB-2201	Memory leak in *BaseJsonTable* caused by a dangling reference of *MetaTable*.
MAPRDB-2254	AppendStream fails when Gateway closes the inactive stream, and raises the replication lag alarm.
MAPRDB-2267	DB crashes during heap memory allocation.
MAPRDB-2303	The Replication Lag alarm does not display the actual lag value.
MAPRDB-2315	MapR-DB scan fails on large tables.
MAPRDB-2323	The Table Replication (**VOLUME_ALARM_TABLE_REPL_ERROR**) alarms are missing information such as the actual bucket FID that produced the alarm, if applicable, and the error code and description of the replication error.

Performance

MS-560	MapR cluster nodes experience high network traffic from mapr-stream clients.
MAPRDB-1727	Delay in data retrieval on MFS nodes with a large number of outstanding active buckets and high usage of DB memory.
MAPRDB-2113	MapR-DB needs to select the most appropriate index in cases where more than one index has been defined over the same field of a MapR-DB table.
MAPRDB-2156	When running queries with a set timeout, the number of threads on the MapR client increases up to 500, exhausting the Thread Pool, and causing the client to stop responding completely, even after all queries time out.
MAPRDB-2250	Too many *BatchGet* operations in parallel when secondary indexes are present on a table, causes MapR DB to crash. Added a parameter `mfs.db.max.concurrent.internal.ops` to regulate the number of parallel *BatchGet* operations.
MAPRMR-8	Reduce the number of input splits that are generated when a job is processed through the `CombineFileInputFormat()` function. Added the parameter mapreduce.input.fileinputformat.split.maxblocknum that determines the number of blocks that can be added to one split.
MFS-2078	Speed up FUSE path lookups. Added the `fuse.negative.timeout` parameter to cache negative lookup results.
MFS-2082	Optimize directory lookup and traversal to avoid overwhelming MFS with RPCs.
MFS-2324	Optimize disk space reserved for tiering operations.
MFS-2608	Priority of child threads do not change when the priority of their parent process is changed.

last-updated: Apr 11, 2022

MFS-2638	Avoid re-sorting results in CLDB for the default output of the maprcli alarm list -sortby alarmtype command.
MFS-2691	Optimize fetching of Muted and Raised Alarms.
MFS-2711	Optimize removal of expired snapshots to free up CLDB CPU from background activity.
MFS-2749	The Alarm History feature needs to be disabled on large clusters as it can degrade performance. Added a parameter cldb.disable.alarm.history to disable alarm history.
MFS-3291	File client does not honor the number of flusher threads set in the `coresite.xml` file. See the fs.mapr.threads parameter.
MFS-4532	Jobs fail with I/O error or are very slow to complete.
MFS-4670	CLDB process consumes 78.8 GB approximately every 6-7 hours and triggers CLDB failover very often.
MFS-4687	Memory leak in CLDB.
MFS-4750	Disk Balancer is unable to move containers from full Storage Pools as they fail the Volume underweight check. Added a tunable - prevent.volume.skew.by.diskbalancer to let the Disk Balancer allow or prevent volume skew.
MFS-4805	Fixed memory leak in NFS Server version 3 that occurs when profiling memory. Added the following entities in `/opt/mapr/conf/nfsserver.conf`: • `MemDebugEnable` - Set to true to enable Memory Debugging. • `HighMemLimitMB` - Sets the maximum amount of memory that the NFS Server can use.
MFS-5227	NFS Server hangs or is very slow and causes replication and resync failures.
MFS-5700	CLDB master failover time is very high.
MFS-6539	NFS Version 3 Server on edge nodes does not have the ulimit setting, as warden is not available on these nodes.
MFS-5724	The MFS configuration parameter mfs.max.restore.count is not being honored causing mirror resync operations to be delayed due to the lack of sufficient restore slots.
MFS-6547	Massive delay in mounting the configured mount points after starting the NFS service.
MFS-6666	NFS server should throttle RPCs to avoid overwhelming CLDB.

last-updated: Apr 11, 2022

MFS-6785	The response from the mrconfig info containers rw command is slow on a cluster with large number of volumes.
MFS-6869	Fuse client should limit the number of RPCs to prevent overwhelming CLDB.
MFS-7181	FileClient defaults to 8KB reads instead of 512KB.
MFS-8475	The `createsystemvolumes.sh` script took hours to complete when adding a new node to a cluster with a large number of volumes.
MFS-11111	Queuing and CLI RPC processing are slow in CLDB.

Security

COMSECURE-331	Security vulnerability in the JNDI-bindable DataSources library.
COMSECURE-334	Security vulnerability in the DOM4j XML framework.
COMSECURE-335	Security vulnerability in the Jasper library.
CORE-290	The `/opt/mapr` directory contains files and directories with insecure permissions.
CORE-293	After upgrading system security packages, `mapr-zookeeper` and `mapr-warden` are not properly started with `systemd`. The `ps` command reports them as started, while `systemd` reports errors when trying to start these services.
CORE-384	Remote Code Execution vulnerability in the ZooKeeper Java JMX server. Added a parameter `JMXDISABLE` to enable or disable loading ZooKeeper JMX parameters.
MAPRDB-2251	Standardize JMX handling for Java processes to prevent vulnerabilities.
MAPRDB-2255	Stream ACE `u:mapr |` has the potential to lock out the administration of the stream.
MAPRHADOOP-63	Security vulnerability in `jackson-databind`.
CORE-562, MAPRHADOOP-123	Security vulnerabilities in MapR 6.x JAR files.
MAPRHADOOP-58, MAPRHADOOP-64, MAPRHADOOP-136, MAPRHADOOP-137	Multiple security vulnerabilities in Hadoop.
MAPRYARN-241	Remote Code Execution vulnerability in the YARN Java JMX Server.
MFS-2336	File Client impersonation does not honour the permissions of the actual user.
MFS-2493	The `/tmp/cldbinfo/unreachableCldbs` is created with insecure permissions.
MFS-2551	Local Privilege Escalation vulnerability in the `maprexecute` command.

last-updated: Apr 11, 2022

MFS-2645	Fixed a buffer overflow in NFS Server version 3.
MFS-2661	Snapshot creation fails due to a permission error.
MFS-2685	The `maprcli` commands use the wrong ticket to communicate with ZooKeeper in secure, cross cluster environments.
MFS-2700	FUSE kernel sends the wrong user credentials to the MapR FUSE Process.
MFS-2708	Disk failure related log files have insecure permissions.
MFS-3310	Need an alert to warn about expiring SSL certificates. Added the Security Certificate Expiry Alarm.
MFS-5229	Remote Code Execution vulnerability in the MAST Gateway JMX Server.
MFS-5234	Remote Code Execution vulnerability in the CLDB JMX Server.
MFS-5235	Remote Code Execution vulnerability in the Gateway JMX Server.
MFS-5236	Remote Code Execution vulnerability in the Warden JMX Server. Added a new parameter warden.enable.jmxremote that must be explicitly set to `true` to enable the Warden JMX Server.

MapR Streams

MS-762	Customer Streams face cursor commit failures.
MS-557	Commits fail for MapR Streams on volumes that were previously mirror volumes but are now standard volumes.

Upgrade

MS-925	After upgrade to EEP 6.2 (Spark 2.4.0), Kafka/ MapR Streams cannot be consumed.
MFS-2079	After upgrading to MapR version 6.1.0, the volume Name Container hangs when assigning volume names for volumes created with MapR version < 4.0.1.
MFS-2469	After upgrading MapR from version 5.2.2 to 6.1.0, slave CLDB nodes are stuck during initialization.
MFS-2553	Ecosystem jobs using ZooKeeper fail after upgrading to the MapR 6.1 EBF patch.
MFS-2560	CLDB on a MapR 5.2.2 cluster gets overwhelmed with RPC calls when mirroring from a MapR 6.1.x cluster.
MFS-2561	The VOLUME_ALARM_DATA_UNDER_REPLICATED alarm is generated frequently after upgrading MapR from version 3.0.1 to version 6.1.0.
MFS-2675	Certify MapR version 6.1.0 on RHEL 7.7

last-updated: Apr 11, 2022

MON-3922	When upgrading to MapR 6.x, ensure that volumes prior to MapR version 6.0, which lack volume aces are handled gracefully after upgrade.
MON-4862	After upgrading from MapR 5.2.1 to MapR 6.1, API server fails to start with an M5 license without tables support installed.
MON-4892	Snapshot tab in MCS indicates that license upgrade is needed after upgrading from MapR 5.2.1 to MapR 6.1 with M5 license installed.

YARN

MAPRMR-4	With centralized logging, YARN does not populate *stderr* and *stdout* logs.
MAPRMR-19	Applications fail with the *Jobstatus not available* exception. The ApplicationMaster has already finished processing each job but the Job History Server has not yet updated job statuses. This causes the failures. Two options have been added to YARN to retry fetching job statuses. • `yarn.app.mapreduce.job.update-status-max-retries` - The number of times to retry. • `yarn.app.mapreduce.job.update-status-retry-interval` - The interval to wait before each retry attempt.
MAPRYARN-127	Resource Manager fails with a Concurrent Modification Exception.
MAPRYARN-155	Containers fail to launch if property names contain a dash (-) in the `launch_container.sh` script.
MAPRYARN-161	Deletion of History Server logs is stopped when an invalid application directory is found within the log aggregation directory.
MAPRYARN-171	YARN preemption does not occur with Fair and DRF scheduling policies.
MAPRYARN-191	YARN API requests via CLI do not return any result when cluster has Label-Based-Scheduling enabled.
MAPRYARN-192	MapReduce jobs fail if their labels contain the logical operand character (*&&*).
MAPRYARN-193	Resource Manager crashes when sorting Collections using the FairShare comparator.
MAPRYARN-195	Resource Manager exits with a FATAL error.
MAPRYARN-203	Resource preemption fails and returns a Null Pointer Exception.
MAPRYARN-210	Use per-node local volumes for YARN log aggregation instead of a single volume. Added the Local Log Aggregation Feature.

last-updated: Apr 11, 2022

MAPRYARN-221	Containers hang in LOCALIZING state. Added two options: • `yarn.nodemanager.timeout-localizing-container` - The maximum time to wait to localize resources for containers. • `yarn.nodemanager.check-interval-localizing-container.ms` - The frequency at which the ApplicationMaster checks the running time of the localizing container.
MAPRYARN-223	Maximum idle time of the Jetty connection should be configurable.
MAPRYARN-244	Resource Manager hangs when trying to shut down after CLDB failover.
MAPRYARN-246	Resource Manager hangs when there is a space in the name of the queue in the *fair-scheduler.xml* file.
MAPRYARN-249	Resources needed to preempt should not have negative vcore values.
MAPRYARN-250	Job History Server (JHS) hangs under heavy load when scanning MFS for job history files. Added a parameter, mapreduce.jobhistory.intermediate-done-scan-timeout to set the timeout in milliseconds for rescanning the `done_intermediate` user directory.
MAPRYARN-258	Publish system metrics in batches so as to avoid overloading the Application Timeline Server (ATS).
MAPRYARN-261	Administrator users who are not part of the `mapr` group are not able to view the logs of the running jobs of another user.
MAPRYARN-276	Resource Manager crashes with a Null Pointer Exception.
MAPRYARN-284	YARN kills container but the task process is not killed.
MAPRYARN-287	When the *ClientRMService* processes an application kill request, the application diagnostics should report the user and the host that issued the kill request.
MAPRYARN-291	On RHEL 8.2, Warden must run Node Manager with umask 022 on MapR Core 6.1.0.

Packages and Dependencies for MapR 6.1.1 Software

Package and dependency details for MapR 6.1.1 platform and ecosystem components.

For downloadable packages, see these links:

- MapR Core Packages
- EEP Packages
- MapR Installer Packages

For MapR Installer package dependencies, see:

- MapR Installer Prerequisites and Guidelines

This table shows the MapR 6.1 package dependencies for Red Hat / CentOS and SLES. A dash (—) indicates no dependency.

	Depends on . . .	
This package . . .	**These MapR packages**	**And these non-MapR packages**
`mapr-apiserver`	`mapr-core`	—
`mapr-cldb`	`mapr-core` `mapr-fileserver`	—

`mapr-client`	`mapr-librdkafka`	`glibc`
		`libgcc`
		`libstdc++`
		`syslinux`

`mapr-compat-suse`	—	`/etc/default/useradd`
		`/sbin/rpcinfo`
		`/usr/sbin/acpidump`
		`libffi4`
		`libgcc_s1`
		`libsnappy1`
		`libstdc++6`
		`openssl`

`mapr-core`	`mapr-core-internal`
	`mapr-hadoop-core`
	`mapr-librdkafka` (version 0.11.3 or later)
	`mapr-mapreduce2`
		—

`mapr-core-internal`	—	`dmidecode`
		`glibc`
		`hdparm`
		`initscripts`
		`irqbalance`
		`libgcc`
		`libstdc++`
		`nss` (version 3.19 or later)
		`redhat-lsb-core`
		`sdparm`
		`shadow-utils`
		`syslinux`

`mapr-fileserver`	`mapr-core`	—
`mapr-hadoop-core`	`mapr-core-internal`	—

last-updated: Apr 11, 2022

This package . . .	Depends on . . .	
	These MapR packages	**And these non-MapR packages**
mapr-historyserver	mapr-mapreduce2	—
mapr-loopbacknfs	—	iputils nfs-utils redhat-lsb-core rpcbind
mapr-mapreduce2	mapr-hadoop-core	—
mapr-nfs	mapr-core	/usr/sbin/rpcinfo iputils nfs-utils
mapr-nodemanager	mapr-mapreduce2	—
mapr-posix-client	mapr-client	—
mapr-resourcemanager	mapr-mapreduce2	—
mapr-single-node	mapr-cldb mapr-fileserver mapr-nfs mapr-webserver mapr-zookeeper	—
mapr-upgrade	mapr-core	rpmrebuild
mapr-webserver	mapr-apiserver	—
mapr-zk-internal	—	—
mapr-zookeeper	mapr-core mapr-zk-internal	—

These MapR 5.x packages were removed for MapR 6.x:

- mapr-jobtracker
- mapr-mapreduce1
- mapr-metrics
- mapr-tasktracker

This table shows the MapR 6.1 package dependencies for Ubuntu:

This package . . .	Depends on . . .	
	These MapR packages	**And these non-MapR packages**

`mapr-client`	`mapr-librdkafka` (version 0.11.3 or later)	`awk`
		`bash` (version 2.05a-11 or later)
		`coreutils` (version 5.0-5 or later)
		`grep` (version 2.4.2-3 or later)
		`libc6`
		`libgcc1`
		`libstdc++6`
		`perl`
		`procps` (version 1:2.0.7-8 or later)
		`sed` (version 3.02-8 or later)
		`syslinux`

`mapr-core-internal`	—	`adduser` (version 3.11 or later)
		`awk`
		`bash` (version 2.05a-11 or later)
		`coreutils` (version 5.0-5 or later)
		`dmidecode`
		`grep` (version 2.4.2-3 or later)
		`hdparm`
		`libc6`
		`libgcc1`
		`libstdc++6`
		`lsb-base`
		`irqbalance`
		`perl`
		`procps` (version 1:2.0.7-8 or later)
		`sdparm`
		`sed` (version 3.02-8 or later)
		`syslinux`
		`sysvinit-utils`

`mapr-loopbacknfs`	—	`iputils-arping`
		`lsb-base`
		`rpcbind`

last-updated: Apr 11, 2022

This package . . .	Depends on . . .	
	These MapR packages	**And these non-MapR packages**

`mapr-nfs`	`mapr-core`	`awk`
		`bash` (version 2.05a-11 or later)
		`coreutils` (version 5.0-5 or later)
		`grep` (version 2.4.2-3 or later)
		`iputils-arping`
		`nfs-common`
		`perl`
		`procps` (version 1:2.0.7-8 or later)
		`sed` (version 3.02-8 or later)

Patches and Documentation for MapR 6.1.1

Describes important considerations for patches and patch documentation for MapR 6.1.1.

Whenever possible, keep your software up to date by applying the latest patches available on the MapR Support Portal. This practice can help you to resolve issues and minimize downtime.

Some patches enable new features or behaviors that are described in the documentation. However, the MapR documentation does not typically include patch numbers or identify the features or behaviors that are delivered by specific patches. If you see a fix or feature in the documentation that is not available on your platform, you might need to apply a patch in order to use the fix or feature.

To understand which patches apply to your platform, contact your MapR support representative, or visit the Support notices of known issues. For information about applying a patch, see Applying a Patch on page 437.

MapR Container for Developers

The MapR Container for Developers is a Docker container that enables you to create a single-node MapR cluster. The container is lightweight and designed to run on your laptop. It requires no additional configuration for you to connect your clients, also running on your laptop, to the cluster.

Important: The Development Environment for MapR Data Platform is currently unavailable for use with release 7.0.0. The development environment script will be updated and documented for release 7.0.0 as soon as the updated Docker image is available.

The MapR cluster created by the Docker image includes the following components:

- MapR Core 6.1
  - MapR XD Distributed File and Object Store on page 451
  - MapR Database on page 496
  - MapR Event Store For Apache Kafka on page 627
  - MapR Control System on page 682
  - MapR NFS
- Apache Drill 1.14.0-1808
- Apache Spark 2.3.1-1808

Examples in this section show Mac OS X and Linux support for the container, but you can run the container on any operating system that supports Docker containers.

After you deploy the container, the environment inside the container runs Ubuntu16 and JDK 8. By default, the MapR cluster is configured as non-secure.

The MapR Container for Developers is provided *as is* for development purposes. HPE technical support is not available for this product. However, users may post questions or comments on the Ezmeral Data Fabric Community.

Prerequisites to Running the MapR Container for Developers

To run the MapR Container for Developers, you must first install the MapR client and Docker software.

The instructions in this topic are specific to Mac OS X. The container is supported on all operating systems that support Docker containers.

1. Install a MapR client on your laptop:
   a) Download the MapR 6.1 client from https://package.mapr.hpe.com/releases/v6.1.0/mac/ or for one of the following Linux distributions:
      - Oracle Enterprise Linux: https://package.mapr.hpe.com/releases/v6.2.0/oel/
      - RHEL: https://package.mapr.hpe.com/releases/v6.2.0/redhat/
      - Ubuntu: https://package.mapr.hpe.com/releases/v6.2.0/ubuntu/

   b) On a Mac, extract the client software to the `/opt` directory by running the following commands:

```
sudo mv ~/Downloads/
mapr-client-6.1.0.20180926230239.GA-1.x86_64.tar.gz /opt
cd /opt
sudo tar xvfz mapr-client-6.1.0.20180926230239.GA-1.x86_64.tar.gz
```

   c) On Linux, use the appropriate package manager to install the client. For example:
      - RPM-based Systems:

```
rpm -ivh mapr-client-6.2.0.0.20200915234957.GA-1.x86_64.rpm
```

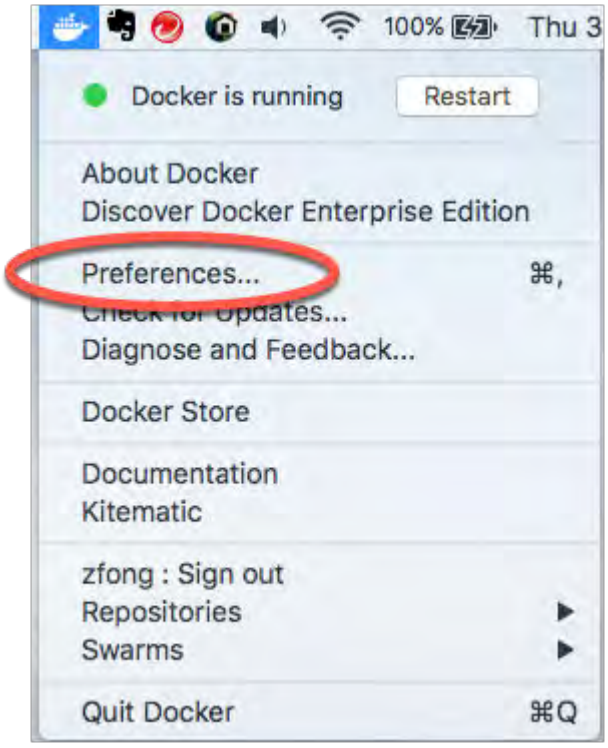
      - On Ubuntu:

```
dpkg -i mapr-client-6.2.0.0.20200915234957.GA-1.amd64.deb
```

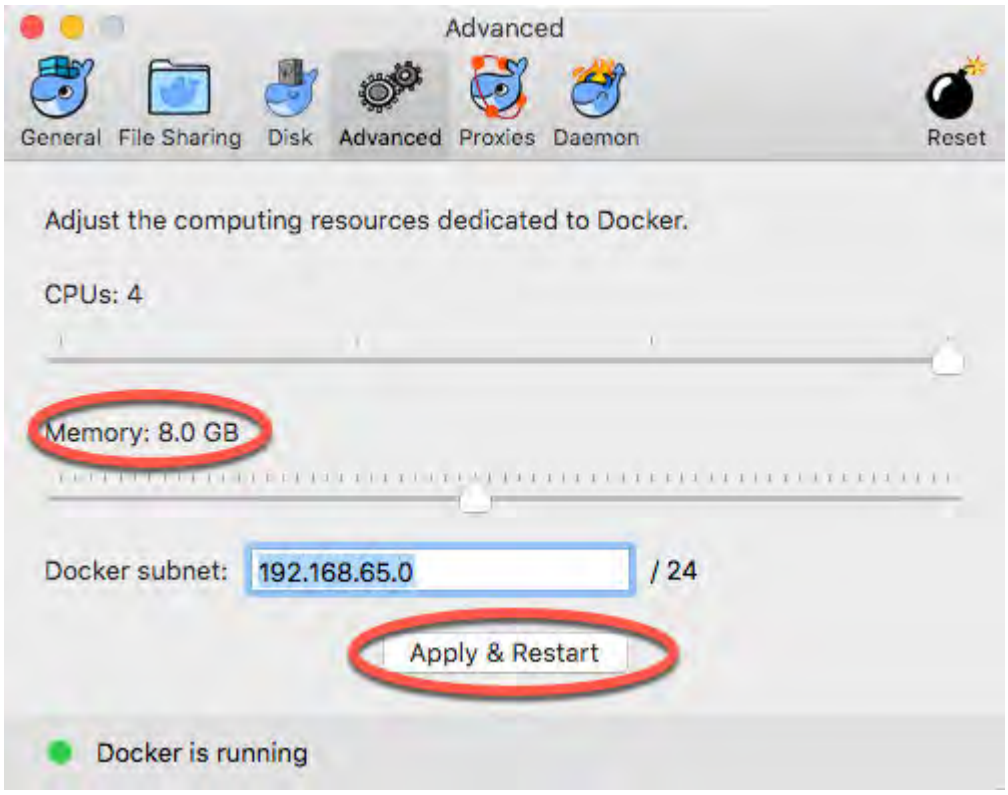
2. Install Docker on your laptop:
   a) Download the software for Mac from https://www.docker.com/docker-mac, or follow the instructions for the appropriate Linux distribution.

   b) Install the software.

   c) On a Mac, verify that Docker is running with at least 8 GB of memory by clicking **Docker->Preferences->Advanced**:



3. Modify the memory settings, if needed, and restart Docker:



Running the MapR Container for Developers Script

The MapR Container for Developers script, `mapr_devsandbox_container_setup.sh`, downloads the Docker image associated with the container and launches the container image that starts the MapR cluster.

It also performs the configuration steps needed to connect local MapR clients to the MapR cluster running in the container.

Important: The Development Environment for MapR Data Platform is currently unavailable for use with release 7.0.0. The development environment script will be updated and documented for release 7.0.0 as soon as the updated Docker image is available.

1. Download `mapr_devsandbox_container_setup.sh` from GitHub.
2. Modify the script so it is executable:

   ```
   chmod +x mapr_devsandbox_container_setup.sh
   ```

3. Run the script:

   ```
   ./mapr_devsandbox_container_setup.sh
   ```

   By default, the script runs the latest version of the container, `maprtech/dev-sandbox-container:latest`,. To run an earlier version, replace `latest` with the tag corresponding to the version you want to use, and pass that as an argument to the script. The following example runs the 6.0.1 version:

   ```
   ./mapr_devsandbox_container_setup.sh maprtech/
   dev-sandbox-container:6.0.1_5.0.0_ubuntu16
   ```

   For a list of available tags, see https://hub.docker.com/r/maprtech/dev-sandbox-container/tags/.

   Note: The script can take 5-10 minutes to run the first time you run it. It requires downloading the Docker image from the Docker repository.

4. When the Docker image is running, you see the following output:

   ```
   Docker Container is coming up....
   Client has been configured with the docker container.

   Please login to the container using (root password mapr): ssh
   root@localhost -p 2222
   Login to the Control System at https://localhost:8443
   ```

5. Log in to the Docker container:

   ```
   ssh root@localhost -p 2222
   ```

6. Wait for the `AdminApplication` java process to start by viewing the output from `jps`:

   ```
   root@maprdemo:~# jps
   3472 WardenMain
   28369 Jps
   5105 CLDB
   13810 RunJar
   28259 FsShell
   13235 AdminApplication
   3232 QuorumPeerManager
   12280 Drillbit
   14122 RunJar
   ```


last-updated: Apr 11, 2022

7. When `AdminApplication` is running, you can access the MapR Control System in your browser using the following URL:

   ```
   https://localhost:8443
   ```

8. After all MapR services are running, you can access the filesystem by using POSIX commands, with `/mapr` as your mount point. The following steps show you how to determine that all services are running:
   a) Determine the id of your Docker container by examining the output from the following command:

      ```
      docker ps
      ```

   b) Examine the contents of the Docker logs using the container id from Step 8a:

      ```
      docker logs ca2c94d9e822
      ```

   c) It can take a few minutes for all services to initialize, depending on the load in your environment. Output similar to the following in your log output indicates that all services are running:

      ```
      This container IP : 172.17.0.2
      ```

   d) Log in to the container using the command from Step 5.

   e) Run the following command to access the MapR filesystem using `ls`:

      ```
      root@maprdemo:~# ls /mapr
      ```

Note: Whenever you change your network environment, you must reconfigure your container. Rerun the `mapr_devsandbox_container_setup.sh` script, and select option 2 when the script shows the following prompt:

```
MapR sandbox container is already running.
1. Kill the earlier run and start a fresh instance
2. Reconfigure the client and the running container for any network
changes
Please enter choice 1 or 2 :
```

Connecting Clients to the MapR Container for Developers

You can access the MapR cluster running in the MapR Container for Developers from your laptop. Simply issue client commands from your laptop.

Setting up New Users

The MapR Container for Developers is setup with only users `mapr` and `root`. If you want to connect clients as some other user, you must add your user name and group to the container.

For example, if running the `id` command on your laptop returns the following:

```
uid=5001(mapruser) gid=5000(maprgroup)
```

Then, run the following commands to add your user name and group to the container:

```
ssh root@localhost -p 2222
groupadd -g 5000 maprgroup
useradd -m -u 5001 -gmaprgroup mapruser
```

Accessing the File System

The following command lists the files in MapR filesystem on the cluster:

```
/opt/mapr/bin/hadoop fs -ls /
```

Accessing MapR Database

To access MapR Database, you can use MapR Database shell:

```
/opt/mapr/bin/mapr dbshell
```

In MapR Database shell, you can create a table, insert into the table, and read from the table:

```
create /tmp/t1
insert /tmp/t1 --v '{"a":"ABC"}' --id "ID1"
find /tmp/t1
```

Accessing Drill

The MapR client that you downloaded in the Prerequisites to Running the MapR Container for Developers on page 97 topic includes a minimum set of clients. To run other clients, you must first copy the client software to your laptop.

The following example shows how to do this for Apache Drill 1.14.0:

1. Determine your Docker `<container-id>` by examining the output of the `docker ps` command
2. Copy Drill from your container to your laptop specifying the `<container-id>`:

   ```
   docker cp <container-id>:/opt/mapr/drill /opt/mapr/drill
   ```

3. Connect to Drill as user `mapr` through JDBC by running `sqlline`:

   ```
   /opt/mapr/drill/drill-1.14.0/bin/sqlline -u
   "jdbc:drill:drillbit=localhost" -n mapr
   ```

   Note: If you are using a different version of Drill, replace the version string with your version.

4. Run a SQL query in `sqlline`:

   ```
   select * from cp.'employee.json' limit 10;
   ```

Demo Applications

Sample applications are available at https://github.com/mapr-demos/mapr-db-60-getting-started. The applications show you how to access a MapR Database JSON table using the following programming interfaces:

- Drill JDBC

last-updated: Apr 11, 2022

- OJAI
- Understanding the MapR Database OJAI Connector for Spark on page 4050

See the README file in the GitHub repository for detailed steps on how to create the data used in the applications and how to run the applications.

Troubleshooting the MapR Container for Developers

This section describes problems you might encounter when deploying, running, and accessing the MapR Container for Developers. It also includes steps to troubleshoot and resolve the problems.

Docker Login Problems

Problem

Attempting to log in to your Docker container returns the following error:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST
IDENTIFICATION HAS CHANGED!    @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@
```

Possible Cause

You have an old ssh key in your `.ssh/known_hosts` file

Solution

Replace the old ssh key with the correct key:

```
ssh-keygen -R [localhost]:2222
```

Docker Failures

Problem

Docker fails to run the container

Possible Cause

Docker encounters problems starting ZooKeeper or Warden

Solution

1. Determine your Docker `<container-id>` by examining the output of the `docker ps` command
2. Examine the Docker log files by running:

   ```
   docker logs <container-id>
   ```

3. Examine the MapR log files specified in the output for further diagnostics. You need to log in to the container to see those files.

Problem

Docker completes its startup as shown by the following output from Docker logs:

```
This container IP : 172.17.0.2
```

But Docker is killed before the MapR processes are running.

Possible Cause	You have not allocated enough memory to Docker
Solution	Make sure you have configured Docker with at least 6 GB of memory as described at Step 2c at Prerequisites to Running the MapR Container for Developers.

MapR Connection Problems

Problem	Unable to connect to Control System in your browser

Possible Cause	Solution
The `AdminApplication` process is not running yet	Run `jps` and wait for `AdminApplication` to appear in the list of running java processes
You are accessing an older, cached copy of the Control System URL	Clear your browser cache and retry connecting to the URL

Unable to Access MapR Database Table

Problem	You cannot access a MapR Database table
Possible Cause	You do not have permissions on the volume where the table is stored
Solution	When creating a volume, make sure you set up the user access controls appropriately. See Creating a Volume on page 864 for details.

Unable to run OJAI Queries Due to Query Service Errors

Problem	When running an OJAI query, you encounter an error indicating that the Query Service is not enabled.
Possible Cause	The MapR Container for Developers is setup with only users `mapr` and `root`. You are running as some other user and your query requires the OJAI Distributed Query Service on page 505.
Solution	Add your user name and group to the container by following the instructions at Setting up New Users on page 100.

MapR Sandbox for Hadoop

The MapR Sandbox for Hadoop is a fully-functional single-node cluster that gently introduces business analysts, current and aspiring Hadoop developers, and administrators (database, system, and Hadoop) to the big data capabilities of Hadoop and its ecosystem.

Use the sandbox to experiment with Hadoop technologies using the Control System and Hue.

Hadoop administrators can launch the Control System and use tutorials to configure, monitor, and manage a cluster.

Hadoop developers or analysts who need to gain an understanding of Hadoop and MapR can:

- Launch the Hue interface.
- Use the tutorials to perform tasks with the provided Hue applications.
- Explore solutions to use cases.

last-updated: Apr 11, 2022

- Run jobs on data in the MapR Sandbox for Hadoop.

To use the MapR Sandbox for Hadoop:

- Verify that the host system meets the prerequisites listed below.
- Download, import, and run the MapR Sandbox for Hadoop, as described in Installing the Sandbox on VMware Player or VMware Fusion or in Installing the Sandbox on VirtualBox.

Prerequisites

Before you install the MapR Sandbox for Hadoop, verify that the host system meets the following prerequisites:

- A supported virtual machine player is installed. The MapR Sandbox for Hadoop runs on the following virtual machine players installed on Windows, Mac, or Linux PCs:
  - VMware Workstation Player (version 12)
  - VMware Fusion (version 8)
  - VirtualBox (version 5.1.22 and later)
  - **Note:** Use of the virtual machine players is subject to the end-user license terms for each player. By downloading and using the virtual machine players, you agree to the terms and conditions.
- At least 20 GB free hard disk space, at least 4 physical cores, and 8 GB of RAM is available. Performance increases with more RAM and free hard disk space.
- Uses one of the following 64-bit x86 architectures:
  - A 1.3 GHz or faster AMD CPU with segment-limit support in long mode
  - A 1.3 GHz or faster Intel CPU with VT-x support
- If you have an Intel CPU with VT-x support, verify that VT-x support is enabled in the host system BIOS. The BIOS settings that must be enabled for VT-x support vary depending on the system vendor. To determine if VT-x support is enabled, see the VMware knowledge base article at http://kb.vmware.com/kb/1003944.

Installing the Sandbox on VMware Player or VMware Fusion

Download the MapR Sandbox for Hadoop, and import the virtual machine into the VMware Player. Configure the network setting, and start the MapR Sandbox for Hadoop to access the MapR Control System and Hue interfaces.

Complete the following steps to install and run the MapR Sandbox for Hadoop:

1. Download the MapR Sandbox for Hadoop file to a directory on your machine. To access the file, go to https://package.mapr.hpe.com/releases/, and select the directory for the latest MapR release. The file is located in the `sandbox/` directory for the release, as shown:

   ```
   https://package.mapr.com/releases/v6.1.0/sandbox/
   MapR-Sandbox-For-Hadoop-6.1.0-vmware.ova
   ```

2. Open the VMware Player, and select the **Open a Virtual Machine** option. If you are running VMware Fusion, select **Import**.

3. Go to the directory where you downloaded the MapR Sandbox for Hadoop file, and select `MapR-Sandbox-For-Hadoop-<version>-vmware.ova`. The *Import Virtual Machine* window appears.
4. In the *Import Virtual Machine* window, click **Import**. The virtual machine player imports the sandbox.
5. Select **Edit virtual machine settings** and then select the **Network Adapter** option.

   Note: The correct setting depends on your network connectivity when you run the sandbox. In general, if you are going to use a wired Ethernet connection, select **NAT**. If you use ODBC or JDBC on a remote host, select **Bridged Adapter**. If you are going to use a wireless network, select **Host-only** or **Bridged Adapter**.
6. Click **OK** to save the settings.
7. Select the `MapR-Sandbox-For-Hadoop-<version>_VM`, and click **Play virtual** machine.
8. When you see the `maprdemo login:_` prompt, enter `mapr` and then enter `mapr` as the password.

   The message, *Welcome to your MapR Demo virtual machine,* appears.
9. Verify that a DNS entry was created on the host machine for the virtual machine. If not, create the entry.
   - For Linux and Mac, create the entry in `/etc/hosts`.
   - For Windows, create the entry in the `%WINDIR%\system32\drivers\etc\hosts` file. To edit and save the hosts file on Windows, you may need to log on as administrator and run an editor, such as Notepad.

   Example: `127.0.0.1 maprdemo`
10. Enter the following URLs in your browser address bar to access the Hue and MapR Control System interfaces, and enter `mapr` as the username and password:
    - **MapR Control System** - Enter https://localhost:8443 or https://127.0.0.1:8443 in the address bar.

      Note: If you encounter a warning message stating that your connection is not private, select the option to accept and continue to the MapR Control System.
    - **Hue** - Enter http://localhost:8888 or http://127.0.0.1:8888 in the address bar.

      Note: If you used a bridged adapter, the network generates and assigns an IP to the sandbox that you must use in place of 127.0.0.1 or localhost in the URL. To get the IP address, log in to the sandbox, using Alt+F2 on Windows or Option+F5 on Mac or Linux. Enter `root` as the username and `mapr` as the password. Run the `ifconfig` command to get the sandbox IP address. The IP address is located in the inet address field. You can access the Control System and Hue interfaces using the IP address, as shown in the following examples:
      - **Control System** - https://192.168.1.104:8443
      - **Hue** - http://192.168.1.104:8888

    Alternatively, you can access the host via SSH using `ssh mapr@localhost -p 2222`.

last-updated: Apr 11, 2022

Installing the Sandbox on VirtualBox

Download the MapR Sandbox for Hadoop, and import the virtual machine into VirtualBox. Configure the network setting, and start the MapR Sandbox for Hadoop to access the MapR Control System and Hue interfaces.

Note: The MapR Sandbox for Hadoop on VirtualBox comes with network address translation (NAT) ports forwarded to `localhost`.

Complete the following steps to install and run the MapR Sandbox for Hadoop:

1. Download the MapR Sandbox for Hadoop file to a directory on your machine. To access the file, go to https://package.mapr.hpe.com/releases/, and select the directory for the latest MapR release. The file is located in the `sandbox/` directory for the release, as shown:

   ```
   https://package.mapr.com/releases/v6.1.0/sandbox/
   MapR-Sandbox-For-Hadoop-6.1.0.ova
   ```

2. Open VirtualBox, and select **File** > **Import Appliance**. The *Import Appliance* window appears.
3. Go to the directory where you downloaded the MapR Sandbox for Hadoop file, select the `MapR-Sandbox-For-Hadoop-<version>.ova` file, and click **Open**. The *Import Virtual Appliance* window appears.
4. Click **Next**.
5. In the *Import Virtual Appliance* window, select the option to **Reinitialize the MAC address of all network cards** and then click **Import**. The Import Appliance imports the sandbox.
6. When the import completes, select the MapR Sandbox for Hadoop and then click **Settings**. The *Settings* window appears.
7. In the *Settings* window, select **Network**. The correct setting depends on your network connectivity when you run the sandbox. In general, if you are going to use a wired Ethernet connection, select **NAT Network**. If you use ODBC or JDBC on a remote host, select **Bridged Adapter**. If you are going to use a wireless network, select **Host-only Adapter** or **Bridged Adapter** with the **Intel(R) Dual Band Wireless-AC 8260** option.
8. Click **OK** to continue.
9. Select the `MapR-Sandbox-For-Hadoop-<version>` in VirtualBox, and click **Start**.
10. When you see `maprdemo login:_` prompt, enter `mapr` and then enter `mapr` as the password.

    The message, *Welcome to your MapR Demo virtual machine,* appears.
11. Verify that a DNS entry was created on the host machine for the virtual machine. If not, create the entry.
    - For Linux and Mac, create the entry in `/etc/hosts`.
    - For Windows, create the entry in the `%WINDIR%\system32\drivers\etc\hosts` file. You may need to run an editor, such as Notepad, as the administrator to edit and save the hosts file on Windows.

    Example: `127.0.0.1 maprdemo`
12. Enter the following URLs in your browser's address bar to access the Hue and MapR Control System interfaces, and enter `mapr` as the username and password:

last-updated: Apr 11, 2022

- **MapR Control System** - Enter https://localhost:8443 or https://127.0.0.1:8443 in the address bar.

  Note: If you encounter a warning message stating that your connection is not private, select the option to accept and continue to the MapR Control System.

- **Hue** - Enter http://localhost:8888 or http://127.0.0.1:8888 in the address bar.

  Note: If you used a bridged adapter, the network generates and assigns an IP to the sandbox that you must use in place of 127.0.0.1 or localhost. To get the IP address, log in to the sandbox, using Alt+F2 on Windows or Option+F5 on Mac or Linux. Enter `root` as the username and `mapr` as the password. Enter `ifconfig` to get the sandbox IP address. The IP address is located in the inet address field. You can access the Control System and Hue interfaces using the IP address, as shown in the following examples:

  - **Control System** - https://192.168.1.104:8443
  - **Hue** - http://192.168.1.104:8888

  If you are unable to access these URLs, upgrade Virtual Box and restart the virtual machine.

Alternatively, you can access the host via SSH using `ssh mapr@localhost -p 2222`.

6.1 Installation

This section contains information about installing and upgrading MapR software. It also contains information about how to migrate data and applications from an Apache Hadoop cluster to a MapR cluster.

The topics in this section assume that you are planning, installing, or upgrading a single cluster. If your environment requires multiple clusters, you must repeat each documented procedure for each cluster.

Planning the Cluster

Describes information and factors used in planning your cluster.

A MapR installation is usually a large-scale set of individual servers, called *nodes*, collectively called a *cluster*. In a typical cluster, most nodes are dedicated to data processing and storage, and a smaller number of nodes run other services that provide cluster coordination and management.

The first step in deploying MapR is planning the servers that will form the cluster, and selecting the services that will run on each node. To determine whether a server is capable of contributing to the cluster, it may be necessary to check the requirements in Preparing Each Node. Each node in the cluster must be carefully checked against these requirements; unsuitability of a node is one of the most common reasons for installation failure.

For an excellent introduction to planning a MapR cluster, see this tech talk.

The objective of a cluster plan is to detail each node's set of services.

Select Services

This section describes some of the services that can be run on a node.

Every installation requires services to manage jobs and applications. **ResourceManager** and **NodeManager** manage MapReduce version 2 and other applications that can run on YARN. In addition, MapR requires the **ZooKeeper** service to coordinate the cluster, and at least one node must run the **CLDB** service. The **WebServer** service is required if the browser-based Control System will be used.

last-updated: Apr 11, 2022

After you install MapR core, you can install ecosystem components that belong to a MapR Ecosystem Pack (EEP). A EEP provides a set of ecosystem components that work together. When a newer version or a revision to a component becomes available, the EEP version is updated to reflect the fact that an update was made. For details on the ecosystem components available in each EEP and the list of EEPs supported by your MapR cluster version, see MapR Ecosystem Packs (EEPs).

The following table shows some of the services that can be run on a node:

Service Category	Service	Description
Management	**Warden**	Warden runs on every node, coordinating the node's contribution to the cluster. Warden is also responsible for managing the service state and its resource allocations on that node.
YARN	**NodeManager**	Hadoop YARN NodeManager service. The NodeManager manages node resources and monitors the health of the node. It works with the ResourceManager to manage YARN containers that run on the node.
MapR Core	**FileServer**	FileServer is the MapR service that manages disk storage for MapR File System and MapR Database on each node.
MapR Core	**CLDB**	Maintains the *container location database (CLDB)* (CLDB) service. The CLDB service coordinates data storage services among MapR File System file server nodes, and access across MapR NFS gateways, and MapR clients.
MapR Core	**NFS**	Provides read-write MapR Direct Access NFS™ access to the cluster, with full support for concurrent read and write access.
Storage	**MapR HBase Client**	Provides access to MapR Database binary tables via HBase APIs. Required on all nodes that will access table data in MapR File System, typically all edge nodes for accessing table data. HBase API can also be accessed through the HBase Thrift and Rest Gateways.
YARN	**ResourceManager**	Hadoop YARN ResourceManager service. The ResourceManager manages cluster resources, and tracks resource usage and node health.
Management	**ZooKeeper**	Internal service. Enables high availability (HA) and fault tolerance for MapR clusters by providing coordination.
YARN	**HistoryServer**	Archives MapReduce application metrics and metadata.

last-updated: Apr 11, 2022

Service Category	Service	Description
Management	**Web Server**	Contains static Control System user interface pages.
Management	**Apiserver**	Allows you to perform cluster administration programmatically, and supports the Control System (see Setting Up the Control System on page 423).
OJAI Distributed Query Service	**Drill**	Provides the distributed query service powered by Apache Drill for MapR Database JSON. Supports the following functionality: • Advanced secondary index selection • Sorts on large data sets • Parallel query execution See OJAI Distributed Query Service on page 505 for more details about the service.
Application	**Hue**	Hue is the Hadoop User Interface that interacts with Apache Hadoop and its ecosystem components, such as Hive, Pig, and Oozie. It also provides interactive notebook access to Spark through Livy.
Application	**Pig**	Pig is a high-level data-flow language and execution framework.
Application	**Hive**	Hive is a data warehouse engine that supports SQL-like adhoc querying and data summarization.
Application	**Flume**	Flume is a service for piping and aggregating large amounts of log data
Application	**Oozie**	Oozie is a workflow scheduler system for managing Hadoop jobs.
Application	**HCatalog**	HCatalog provides applications with a table view of the MapR File System layer of the cluster, expanding your options from read/write data streams to add-[Hive]-table operations such as get row and store row.
Application	**Cascading**	Cascading on page 3185 is an application framework for analyzing and managing big data.

Service Category	Service	Description
Application	**Myriad**	Myriad is an application framework that enables YARN applications and Mesos frameworks to run side-by-side while dynamically sharing cluster resources. When using Myriad, the **ResourceManager** is deployed using Marathon, and **NodeManager** is run as a Mesos task.
Application	**Spark**	Spark is a processing engine for large datasets. While it can be deployed locally or standalone, the recommended deployment is on YARN. The application timeline server component provides a historical view of query details.
Application	**Sqoop**	Sqoop is a library for transferring bulk data between Hadoop and relational databases.

Cluster Design Objectives

This section describes some of the work that your cluster performs, and identifies key design considerations.

Begin by understanding the work that the cluster performs. Establish metrics for data storage capacity, throughput, and characterize the data processing that will typically be performed.

Data Workload

While MapR is relatively easy to install and administer, designing and tuning a large production MapReduce cluster is a complex task that begins with understanding your data needs. Consider the kind of data processing that will occur and estimate the storage capacity and throughput speed required. Data movement, independent of MapReduce operations, is also a consideration. Plan for how data will arrive at the cluster, and how it will be made useful elsewhere.

Network bandwidth and disk I/O speeds are related; either can become a bottleneck. CPU-intensive workloads reduce the relative importance of disk or network speed. If the cluster will be performing a large number of big reduces, network bandwidth is important, suggesting that the hardware plan include multiple NICs per node. The MapR Core can natively take advantage of multiple NICs and distribute workload across them. In general, the more network bandwidth, the faster things will run.

Running NFS on multiple data nodes can improve data transfer performance and make direct loading and unloading of data possible, but multiple NFS instances requires an Converged Enterprise Edition, Hadoop module license. For more information about NFS, see Managing the MapR NFS Service on page 1176.

Plan which nodes will provide NFS access according to your anticipated traffic. For instance, if you need 5Gb/s of write throughput and 5Gb/s of read throughput, the following node configurations would be suitable:

- 12 NFS nodes with a single 1GbE connection each
- 6 NFS nodes with dual 1GbE connections each
- 4 NFS nodes with quadruple 1GbE connections each

When you set up NFS on all of the file server nodes, you enable a self-mounted NFS point for each node. A cluster made up of nodes with self-mounted NFS points enable you to run native applications as

last-updated: Apr 11, 2022

tasks. You can use round-robin DNS or a hardware load balancer to mount NFS on one or more dedicated gateways outside the cluster to allow controlled access.

High Availability

A properly licensed and configured MapR cluster provides automatic failover for continuity throughout the MapR Core stack. Configuring a cluster for HA involves redundant instances of specific services, as well as a correct configuration of the MapR NFS service. HA features are not available with the Converged Community Edition.

The following table describes redundant services used for HA:

Service	Strategy	Min. instances
CLDB	Primary/secondary--two instances in case one fails.	2
ZooKeeper	A majority of ZK nodes (a *quorum*) must be up.	3
ResourceManager	One active and one or more standby instances. If the active one fails, one standby instance takes over. This is configured automatically using Zero Configuration.	2
NFS	The more redundant NFS services, the better.	2
OpenTSDB	At least one instance should be up.	3
Elasticsearch	At least two instances should be up.	3

Note: You should use an odd number of ZooKeeper instances. Setting up more than 5 ZooKeeper instances is not usually needed.

For a high availability cluster, use 5 ZooKeepers, so that the cluster can tolerate 2 ZooKeeper nodes failing and still maintain a quorum.

On a large cluster, you may choose to have extra nodes available in preparation for failover events. In this case, you keep spare, unused nodes ready to replace nodes running control services, such as CLDB or ZooKeeper in case of a hardware failure.

Virtual IP Addresses

You can use virtual IP addresses (VIPs) for load balancing or failover with the Converged Enterprise Edition, Hadoop module. VIPs provide multiple addresses that can be leveraged for round-robin DNS, allowing client connections to be distributed among a pool of NFS nodes. VIPs also enable high availability (HA) NFS. In a HA NFS system, when an NFS node fails, data requests are satisfied by other NFS nodes in the pool. Use a minimum of one VIP per NFS node per NIC that clients will use to connect to the NFS server. If you have four nodes with four NICs each, with each NIC connected to an individual IP subnet, use a minimum of 16 VIPs and direct clients to the VIPs in round-robin fashion. The VIPs should be in the same IP subnet as the interfaces to which they will be assigned. See Managing VIPs for NFS on page 1176 for NFS for details on enabling VIPs for your cluster.

If you plan to use VIPs on your cluster's NFS nodes, consider the following tips:

- Set up NFS on at least three nodes if possible.
- All NFS nodes must be accessible over the network from the machines where you want to mount them.

last-updated: Apr 11, 2022

- To serve a large number of clients, set up dedicated NFS nodes and load-balance between them. If the cluster is behind a firewall, you can provide access through the firewall through a load balancer instead of direct access to each NFS node. You can run NFS on all nodes in the cluster, if needed.
- To provide maximum bandwidth to a specific client, install the NFS service directly on the client machine. The NFS gateway on the client manages how data is sent in or read back from the cluster, using all its network interfaces (that are on the same subnet as the cluster nodes) to transfer data via MapR APIs, balancing operations among nodes as needed.
- Use VIPs to provide High Availability (HA) and failover.

Minimum Cluster Size

All MapR production clusters must have a minimum of four data nodes except for MapR Edge on page 6579.

More Nodes Are Better

In general, it is better to have more nodes. Larger clusters recover faster from disk failures because more nodes are available to contribute. To maximize fault tolerance in the design of your cluster, see Example Cluster Designs on page 118.

A data node is defined as a node running a FileServer process that is responsible for storing data on behalf of the entire cluster. Having additional nodes deployed with control-only services such as CLDB and ZooKeeper is recommended, but they do not count toward the minimum node total because they do not contribute to the overall availability of data.

To understand how to size a MapR cluster, see this video.

Considerations for Clusters Smaller Than 10 Nodes

Note these special considerations for clusters of 10 nodes or fewer:

- Erasure coding and rolling updates are not supported for clusters of four nodes or fewer.
- Erasure coding is not recommended for five- and six-node clusters. See the *Important* note in Erasure Coding Scheme for Data Protection and Recovery on page 926.
- Dedicated control nodes are not needed on clusters with fewer than 10 data nodes.
- As the cluster size is reduced, each individual node has a larger proportional impact on cluster performance. As cluster size drops below 10 nodes, especially during times of failure recovery, clusters can begin to exhibit variable performance depending on the workload, network and storage I/O speed, and the amount of data being re-replicated.
- For information about fault tolerance, see Priority 1 - Maximize Fault Tolerance on page 119 and Cluster Design Objectives on page 110.

For hardware and configuration best practices, see Cluster Hardware on page 112.

Cluster Hardware

Describes important hardware architecture considerations for your cluster.

When planning the hardware architecture for the cluster, make sure all hardware meets the node requirements listed in Preparing Each Node.

The architecture of the cluster hardware is an important consideration when planning a deployment. Among the considerations are anticipated data storage and network bandwidth needs, including intermediate data generated when jobs and applications are executed. The type of workload also is important. Consider whether the planned cluster usage will be CPU-intensive, I/O-intensive, or

last-updated: Apr 11, 2022

memory-intensive. Think about how data will be loaded into and out of the cluster, and how much data is likely to be transmitted over the network.

Planning a cluster often involves tuning key ratios, such as:

- Disk I/O speed to CPU processing power
- Storage capacity to network speed
- Number of nodes to network speed

Typically, the CPU is less of a bottleneck than network bandwidth and disk I/O. To the extent possible, balance network and disk transfer rates to meet the anticipated data rates using multiple NICs per node. It is not necessary to bond or trunk the NICs together. The MapR Data Platform can take advantage of multiple NICs transparently. Each node should provide raw disks to MapR, with no RAID or logical volume manager, as MapR takes care of formatting and data protection.

The following example architecture provides specifications for a recommended standard MapR Hadoop compute/storage node for general purposes. This configuration is highly scalable in a typical data center environment. MapR can make effective use of more drives per node than standard Hadoop, so each node should present enough faceplate area to allow a large number of drives.

Standard Compute/Storage Node

- Dual CPU socket system board
- 2x8 core CPU, 32 cores with HT enabled
- 8x8GB DIMMs, 64GB RAM (DIMM count must be multiple of CPU memory channels)
- 12x2TB SATA drives
- 10GbE network interface
- OS using entire single drive, not shared as data drive

Best Practices

Hardware recommendations and cluster configuration vary by use case. For example, is the application an MapR Database application? Is the application latency-sensitive?

The following recommendations apply in most cases:

Disk Drives	• Drives should be JBOD, using single-drive RAID0 volumes to take advantage of the controller cache. • SSDs are recommended when using MapR Database JSON with secondary indexes. HDDs can be used with secondary indexes only if the performance requirements are thoroughly understood. Performance can be substantially impaired on HDDs because of high levels of disordered I/O requests. SSDs are not needed for using MapR Event Store For Apache Kafka. • SAS drives can provide better I/O latency; SSDs provide even lower latency. • Match aggregate drive throughput to network throughput. 20GbE ~= 16-18 HDDs or 5-6 SSDs or 1 NVMe drive.

last-updated: Apr 11, 2022

Cluster Size	• In general, it is better to have more nodes. Larger clusters recover faster from disk failures because more nodes are available to contribute. For information about fault tolerance, see Example Cluster Designs on page 118. • For smaller clusters, all nodes are likely to fit on a single non-blocking switch. Larger clusters require a well-designed Spine/Leaf fabric that can scale.
Operating System and Server Configuration	• Red Hat Enterprise Linux, SUSE Linux Enterprise Server, Ubuntu, CentOS, and Oracle Enterprise Linux are supported as described in Operating System Support Matrix on page 5522. • Install the minimal server configuration. Use a product like Cobbler to PXE boot and install a consistent OS image. • Install the full JDK (1.8). • For best performance, avoid deploying a MapR cluster on virtual machines. However, VMs are supported for use as clients or edge nodes.
Memory, CPUs, Number of Cores	• Make sure the DIMM count is an exact multiple of the number of memory channels the selected CPU provides. • Use CPUs with as many cores as you can. Having more cores is more important than having a slightly higher clock speed. • MapR Database benefits from lots of RAM: 256GB per node or more. • Filesystem-only nodes can have fewer, faster cores: 6 cores for the first 10GbE of network bandwidth, and an additional 2 cores for each additional 10GbE. For example, dual 25GbE (50GbE) filesystem-only nodes perform best with at least 6+(4*2)=14 cores. • Filesystem-only nodes should have hyperthreading disabled.

Service Layout in a Cluster

Provides an overview of segregating services on different nodes.

How you assign services to nodes depends on the scale of your cluster and the MapR license level. For a single-node cluster – which must not be used in a production environment (see Minimum Cluster Size on page 112) – no decisions are involved. All of the services you are using run on the single node.

On medium clusters, the performance demands of the CLDB and ZooKeeper services require them to be assigned to separate nodes to optimize performance. On large clusters, good cluster performance requires that these services run on separate nodes.

The cluster is flexible and elastic. Nodes play different roles over the lifecycle of a cluster. The basic requirements of a node are not different for management or for data nodes.

As the cluster grows, it becomes advantageous to locate control services (such as ZooKeeper and CLDB) on nodes that do not run compute services. The MapR Converged Community Edition does not include HA

last-updated: Apr 11, 2022

capabilities, which restricts the number of instances that certain services can run. The number of nodes and the services they run evolve over the life cycle of the cluster.

To provide a high-availability, high-performance cluster, the MapR software architecture allows virtually any service to run on any node, or nodes. The following guidelines help you to plan your cluster service layout.

Note: It is possible to install MapR software on a one- or two-node demo cluster. Production clusters can harness hundreds of nodes, but five- or ten-node production clusters are appropriate for some applications.

Node Types

Depending on the size of your cluster, nodes may or may not perform specialized work.

In a production MapR cluster, some nodes are typically dedicated to cluster coordination and management, and other nodes are tasked with data storage and processing duties. An edge node provides user access to the cluster, concentrating open user privileges on a single host. In smaller clusters, the work is not so specialized, and a single node may perform data processing as well as management.

Nodes Running ZooKeeper and CLDB

High latency on a ZooKeeper node can lead to an increased incidence of ZooKeeper quorum failures. A ZooKeeper quorum failure occurs when the cluster finds too few copies of the ZooKeeper service running. If the ZooKeeper node is running other services, competition for computing resources can lead to increased latency for that node. If your cluster experiences issues relating to ZooKeeper quorum failures, consider reducing or eliminating the number of other services running on the ZooKeeper node.

Nodes for Data Storage and Processing

Most nodes in a production cluster are data nodes. FileServer and NodeManager run on data nodes. Data nodes can be added or removed from the cluster as requirements change over time.

Edge Nodes

So-called Edge nodes provide a common user access point for the MapR webserver and other client tools. Edge nodes may or may not be part of the cluster, as long as the edge node can reach cluster nodes. Nodes on the same network can run client services and other services, but edge nodes and client nodes may not host MapR monitoring components.

Related concepts

MapR Monitoring Architecture on page 1331
MapR Monitoring integrates with open-source components to collect, aggregate, store, and visualize metrics and logs.

Service Layout Guidelines for Large Clusters

Describes how to install and segregate services on large clusters.

General Guidelines

The following are guidelines for installing services on large clusters:

- **ResourceManager**: Run the ResourceManager services on dedicated nodes for clusters with over 250 nodes.
- **Elasticsearch:** Elasticsearch consumes significant CPU, disk, and memory resources. Review the following guidelines:
  - Whenever possible, Elasticsearch should have a dedicated disk for its index directory.
  - Depending on the number of indexed logs, you may want to run the Elasticsearch service on five or more dedicated nodes.

last-updated: Apr 11, 2022

  - On production clusters, consider increasing Elasticsearch's memory allocation. After you install MapR Monitoring, see Configure the Elasticsearch Service Heap Size on page 1394.
  - On clusters with high-density racks, run one or more Elasticsearch services on each rack. Also, configure Fluentd to write logs to Elasticsearch services that reside on the same rack as the Fluentd services. After you install MapR Monitoring, see Configure Fluentd Services to Write to Elasticsearch Nodes on the Same Rack on page 1395.
- **OpenTSDB:** Run the OpenTSDB service on five or more nodes for clusters over 100 nodes.

Services to Separate on Large Clusters

The following are guidelines about which services to separate on large clusters:

- **ResourceManager on ZooKeeper nodes:** Avoid running the ResourceManager service on nodes that are running the ZooKeeper service. On large clusters, the ResourceManager service can consume significant resources.
- **Monitoring Services on CLDB Nodes**: Avoid running the OpenTSDB, Elasticsearch, Kibana, or Grafana services on nodes that are running the CLDB service.

Service Layout Guidelines for Replication

Based on the use case, replicating MapR Database tables and MapR Event Store For Apache Kafka may require the installation of MapR Gateways and the HBase client on one or more nodes.

Guidelines for Installing Gateways

When you configure replication for MapR Database tables or MapR Event Store For Apache Kafka, MapR gateways provide one-way communication between a source MapR cluster and a destination cluster. It is recommended to install at least three gateways on the destination cluster. Installing two or more gateways on a destination cluster allows for replication failover in the event that one gateway is unavailable. Installing three gateways on a large cluster enables better throughput for data replication. Installing more than three gateways can improve availability but is not likely to improve replication performance.

Guidelines for Installing HBase Client

When you configure replication for MapR Database tables, the HBase client is not required by default. However, you must install the HBase client to replicate MapR Database tables in the following situations:

- You plan to perform autosetup table replication using the MapR Database C API. In this case, you must install the HBase Client on the node where the C application will run.
- You plan to perform autosetup table replication using the `maprcli table replica autosetup` command without direct copy. In this case, you must install the HBase Client on the node where you submit the `maprcli table replica autosetup` command. For more information about autosetup table replication, see Replica Autosetup for MapR Database Tables on page 624.

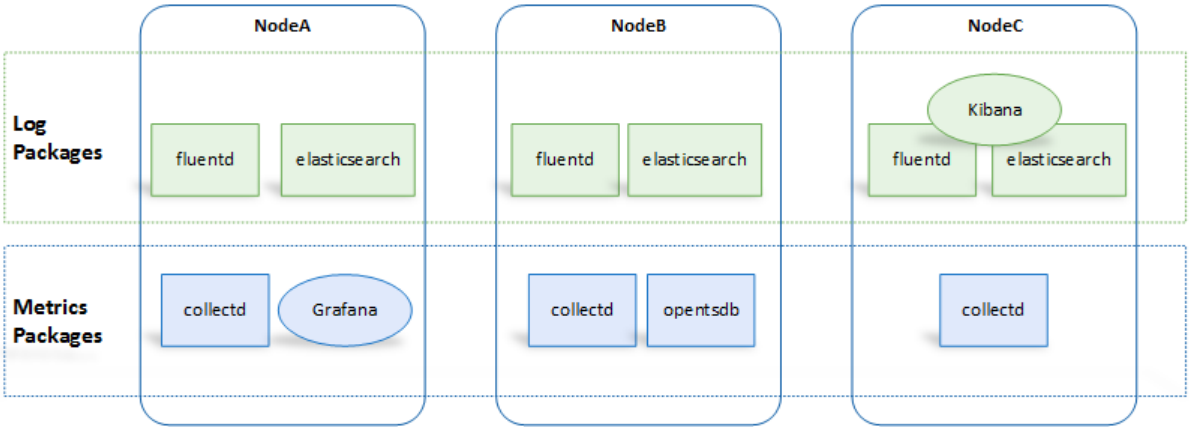
MapR Monitoring Storage Options

Describes various storage options for MapR Monitoring. The Control System relies on MapR monitoring components to display metrics, but can function without the monitoring components. Using MapR monitoring to store logs is optional.

The following installation options are available for metric storage with OpenTSDB and log storage with Elasticsearch. You can store logs and metrics on a non-MapR cluster but this scenario is not supported by MapR.

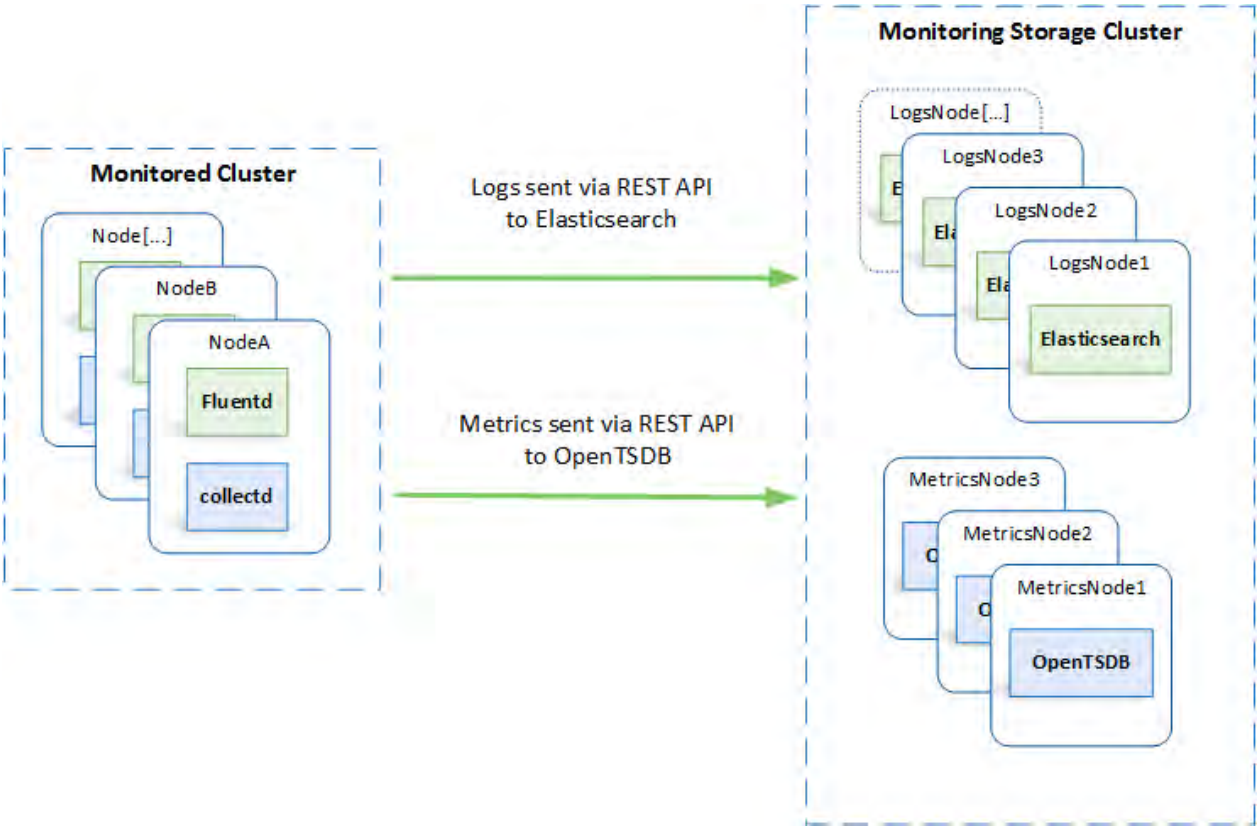
Store Metrics and Logs on the Monitored Cluster

You can store metrics and logs on the nodes in the same MapR cluster that you want to monitor. Note that installing Grafana is optional.



Store Metrics and Logs on a Storage Cluster

You can store metrics and logs for the MapR cluster that you want to monitor, on nodes in a different MapR cluster.

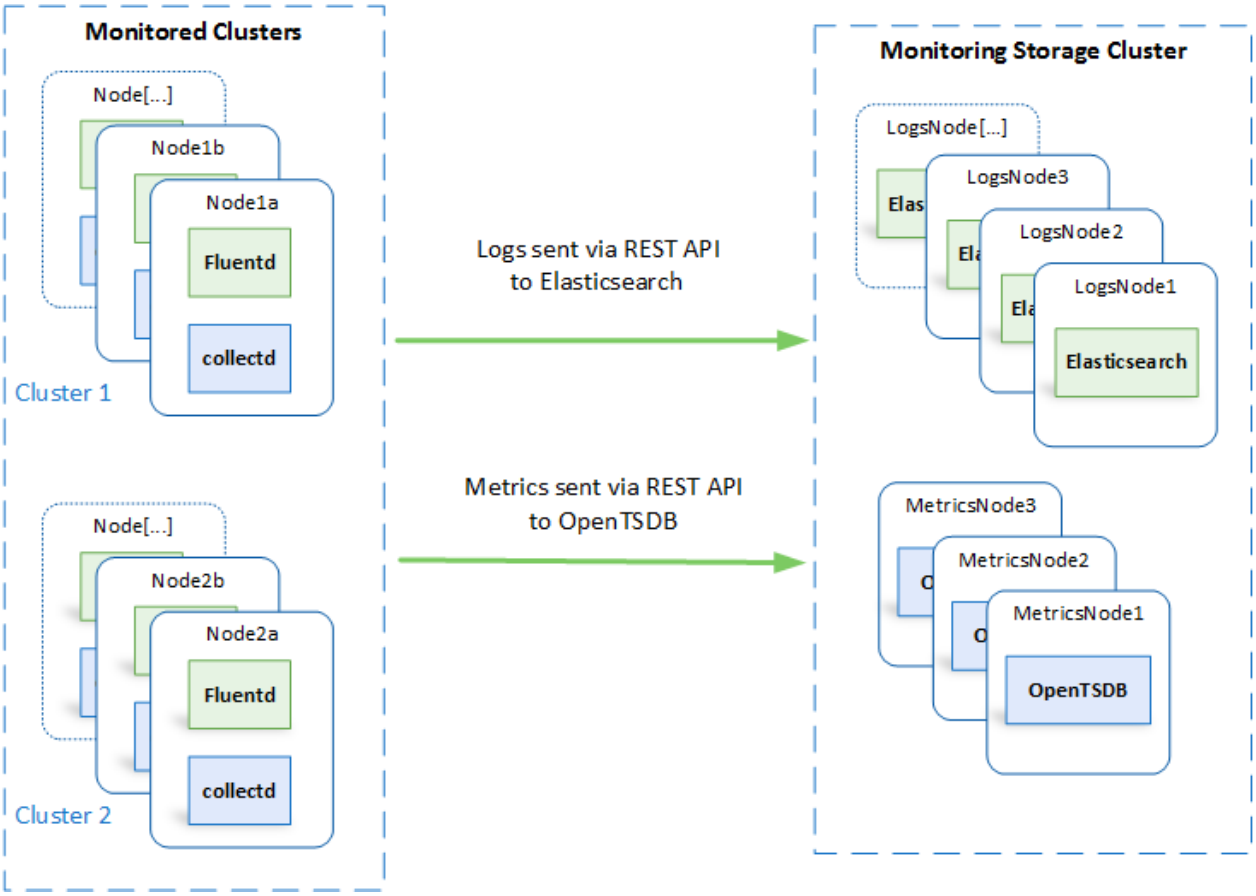


In this case, Kibana and Grafana can be installed on either cluster.

Use a Single Cluster to Store Monitoring Data for Multiple Clusters

You can store metrics and logs for more than one MapR cluster on a shared set of nodes. With this option, a single dashboard can monitor multiple clusters.

last-updated: Apr 11, 2022



In this case, Kibana and Grafana can be installed on any of these clusters.

Example Cluster Designs
Describes how to design a MapR cluster for maximum availability, fault-tolerance, and performance.

The topic includes example cluster designs for 6-node, 12-node, and 50-node clusters:

- Example 1: 6-Node Cluster on page 121
- Example 2: 12-Node Cluster on page 122
- Example 3: 50-Node Cluster on page 123

Design Priorities

Building a cluster requires you to make decisions – and sometimes tradeoffs – that take into account cluster attributes such as:

- Performance
- Fault-tolerance
- Cost
- Ease of use
- Supportability
- Reliability

The following priorities and related best practices can help you plan a durable cluster that includes all or most of these cluster attributes. The priorities are listed in order of importance:

- Priority 1 - Maximize Fault Tolerance on page 119
- Priority 2 - Minimize Resource Contention on page 120
- Priority 3 - Promote High Availability on page 120
- Priority 4 - Use Dedicated Nodes for Key Services for Large Clusters (50-100 Nodes) on page 121

Priority 1 - Maximize Fault Tolerance

Follow these best practices to ensure that your MapR cluster can tolerate failures:

- Ensure an odd number of ZooKeeper services. ZooKeeper fault tolerance depends on a quorum of ZooKeeper services being available. At least three ZooKeeper services are recommended. For a higher level of fault tolerance, use five ZooKeeper services. With five ZooKeepers, two can be lost and quorum maintained.
- For other services, it makes sense for them to be at least as reliable as ZooKeeper. Generally, this means at least two instances of the service for three ZooKeepers and three instances for five ZooKeepers.
- Include enough CLDBs to be as reliable as ZooKeeper. As CLDBs use a primary-secondary configuration, a MapR cluster can function with an odd or even number of CLDBs. The recommended minimum number of active CLDBs is two. To tolerate failures, more CLDBs are needed:
  - If you have three ZooKeepers, configure at least three CLDBs.
  - If you have five ZooKeepers, configure at least four CLDBs. With four CLDBs, the cluster can tolerate two CLDB failures and still provide optimal performance. Adding a fifth CLDB does not increase failure tolerance in this configuration.
- Include enough Resource Manager processes to be as reliable as ZooKeeper. Only one Resource Manager is active at a time:
  - If you have three ZooKeepers, you need at least two Resource Managers.
  - If you have five ZooKeepers, you need at least three Resource Managers. Three Resource Managers can survive the loss of two ZooKeepers.
- For most MapR clusters, the recommended configuration is:
  - Three ZooKeepers
  - Three CLDBs
  - Two or three Resource Managers
  - For larger clusters, increase the number of CLDBs or ZooKeepers for better performance or higher reliability. Table 1 shows the number of failures tolerated by various combinations of ZooKeeper, CLDB, and Resource Manager services.

Table

ZooKeepers	CLDBs	Resource Managers	ZK/CLDB/RM Failures Tolerated
3	2[1]	2	1
3	3	2	1

last-updated: Apr 11, 2022

Table (Continued)

ZooKeepers	CLDBs	Resource Managers	ZK/CLDB/RM Failures Tolerated
5	4	3	2
5	5[2]	3	2

[1]For optimal failure handling, the minimum number of CLDBs is three; hence, three or more CLDBs are recommended. With two CLDBs, the failure of one does not result in an outage, but recovery can take longer than with three.

[2]Using five CLDBs does not improve fault-tolerance significantly when compared with four CLDBs. However, it can be convenient to have the same number of CLDBs as ZooKeepers.

Priority 2 - Minimize Resource Contention

Every service on a node represents a tax on the resources provided by that node. Spreading services evenly across nodes maximizes performance and helps to keep failures isolated to failure domains. Due to power and networking considerations, a rack is usually the most common failure domain.

Follow these best practices to avoid performance bottlenecks:

- Spread like services across racks as much as possible. While not necessary, it is also convenient to put them in the same position, if possible.
- To maximize availability, use three or more racks even for small clusters. Using two racks is not recommended. If a cluster has three ZooKeepers, using two racks means one of the racks will host two ZooKeepers. In this scenario, a loss of a rack having two ZooKeepers can jeopardize the cluster.
  - For services that are replicated, make sure the replicas are in different racks.
  - Put the Resource Manager and CLDB services on separate nodes, if possible.
  - Put the ZooKeeper and CLDB services on separate nodes, if possible.
- Some administrators find it convenient to put web-oriented services together on nodes with lower IP addresses in a rack. This is not required.
- Avoid putting multiple resource-heavy services on the same node.
- Spread the following resources across all data nodes:
  - Clients
  - Drill
  - NFS

Priority 3 - Promote High Availability

Whenever possible, configure high availability (HA) for all services, not just for services that provide HA by default. CLDB, ZooKeeper, Resource Manager, and Drill provide HA by default. Some services are inherently stateless. If possible, configure multiple instances of these services:

- Kafka REST
- HBase Thrift
- HBase REST
- HTTPFS

- HiveServer 2 (HS2)
- Hue
- Kafka Connect
- MapR Data Access Gateway
- MapR Gateway
- Oozie
- OpenTSDB
- WebHCat
- WebServer

Priority 4 - Use Dedicated Nodes for Key Services for Large Clusters (50-100 Nodes)

Large clusters increase CLDB and Resource Manager workloads significantly. In clusters of 50 or more nodes:

- Use dedicated nodes for CLDB, ZooKeeper, and Resource Manager.

  Note: Dedicated nodes have the benefit of supporting fast fail-over for file-server operations.

- If fast fail-over is not critical and you need to minimize hardware costs, you may combine the CLDB and ZooKeeper nodes. For example, a large cluster might include 3 to 9 such combined nodes.
- If necessary, review and adjust the hardware composition of CLDB, ZooKeeper, and Resource Manager nodes. Once you have chosen to use dedicated nodes for these services, you might determine that they do not need to be identical to other cluster nodes. For example, dedicated CLDB and ZooKeeper nodes probably do not need as much storage as other cluster nodes.
- Avoid configuring Drill on CLDB or ZooKeeper nodes.

Example Clusters

The following examples are reasonable implementations of the design priorities introduced earlier in this section. Other designs are possible and may satisfy your unique environment and workloads.

- Example 1: 6-Node Cluster on page 121
- Example 2: 12-Node Cluster on page 122
- Example 3: 50-Node Cluster on page 123

Example 1: 6-Node Cluster

Example 1 shows a 6-node cluster contained in a single rack. When only a single rack is available, this example can work for small clusters. However, the recommended best practice for all clusters, regardless of size, is to use three or more racks, if possible.

Example 1a. Core and Hadoop for 6-Node Cluster

Physical Rack/ Failure Domain	Recommended Topology	Hostname	Core										Hadoop		
			Core (6)	Fileserver (6)	Client Components (6)	Zookeeper (3)	CLDB (3)	NFS (6)	Adminserver (2)	Gateway (2)	Metrics TSDB (2)	Metrics ES (2)	RM (2)	History (2)	Node Mgr (6)
Rack1	/data/rack1	h1	✓	✓	✓		✓	✓	✓					✓	✓
Rack1	/data/rack1	h2	✓	✓	✓		✓	✓	✓					✓	✓
Rack1	/data/rack1	h3	✓	✓	✓		✓	✓		✓		✓			✓
Rack1	/data/rack1	h4	✓	✓	✓	✓		✓		✓		✓			✓
Rack1	/data/rack1	h5	✓	✓	✓	✓		✓			✓		✓		✓
Rack1	/data/rack1	h6	✓	✓	✓	✓		✓			✓		✓		✓

Example 1b. Ecosystem Components for 6-Node Cluster

Physical Rack/ Failure Domain	Recommended Topology	Hostname	Ecosystem														
			HBase REST* (2)	HBase Thrift* (2)	Drill (6)	Spark (6)	Streams REST* (2)	HTTPFS* (2)	HS2 (2)	HS2 Metaserver (2)	Webhcat(2)	Spark HS(2)	Oozie (2)	Sqoop (2)	Hue (2)	MySQL (1)	Total[3]
Rack1	/data/rack1	h1	✓	✓	✓	✓						✓		✓			14
Rack1	/data/rack1	h2	✓	✓	✓	✓						✓		✓			14
Rack1	/data/rack1	h3			✓	✓	✓	✓			✓		✓				14
Rack1	/data/rack1	h4			✓	✓	✓	✓			✓		✓				14
Rack1	/data/rack1	h5			✓	✓			✓	✓					✓		13
Rack1	/data/rack1	h6			✓	✓			✓	✓					✓	✓	14

[3]*Total* cells show the total number of Core, Hadoop, and Ecosystem components installed on each host node for the example cluster.

*Denotes a service that is lightweight and stateless. For greater performance, consider running these services on all nodes and adding a load balancer to distribute network traffic.

Example 2: 12-Node Cluster

Example 2 shows a 12-node cluster contained in three racks:

Example 2a. Core and Hadoop for 12-Node Cluster

Physical Rack/ Failure Domain	Recommended Topology	Hostname	Core										Hadoop		
			Core (12)	Fileserver (12)	Client Components (12)	Zookeeper (3)	CLDB (3)	NFS (12)	Adminserver (3)	Gateway (3)	Metrics TSDB (3)	Metrics ES (3)	RM (3)	History (3)	Node Mgr (12)
Rack1	/data/rack1	h1	✓	✓	✓			✓	✓					✓	✓
Rack1	/data/rack1	h2	✓	✓	✓			✓		✓					✓
Rack1	/data/rack1	h3	✓	✓	✓		✓	✓							✓
Rack1	/data/rack1	h4	✓	✓	✓	✓		✓			✓	✓	✓		✓
Rack2	/data/rack2	h5	✓	✓	✓			✓	✓					✓	✓
Rack2	/data/rack2	h6	✓	✓	✓			✓		✓					✓
Rack2	/data/rack2	h7	✓	✓	✓		✓	✓							✓
Rack2	/data/rack2	h8	✓	✓	✓	✓		✓			✓	✓	✓		✓
Rack3	/dev/rack3	h9	✓	✓	✓			✓	✓					✓	✓
Rack3	/dev/rack3	h10	✓	✓	✓			✓		✓					✓
Rack3	/dev/rack3	h11	✓	✓	✓		✓	✓							✓
Rack3	/dev/rack3	h12	✓	✓	✓	✓		✓			✓	✓	✓		✓

Example 2b. Ecosystem Components for 12-Node Cluster

Physical Rack/ Failure Domain	Recommended Topology	Hostname	Ecosystem													
			HBase REST* (3)	HBase Thrift* (3)	Drill (12)	Spark (12)	Streams REST* (3)	HTTPFS* (3)	HS2 (3)	HS2 metaserver (3)	Webhcat(3)	Spark HS (3)	Oozie (3)	Sqoop (3)	Hue (3)	MySQL (3)	Total[3]
Rack1	/data/rack1	h1			✓	✓			✓			✓			✓		12
Rack1	/data/rack1	h2	✓	✓	✓	✓	✓	✓									12
Rack1	/data/rack1	h3			✓	✓							✓	✓		✓	11
Rack1	/data/rack1	h4			✓	✓				✓	✓						13
Rack2	/data/rack2	h5			✓	✓			✓			✓			✓		12
Rack2	/data/rack2	h6	✓	✓	✓	✓	✓	✓									12
Rack2	/data/rack2	h7			✓	✓							✓	✓		✓	11
Rack2	/data/rack2	h8			✓	✓				✓	✓						13
Rack3	/dev/rack3	h9			✓	✓			✓			✓			✓		12
Rack3	/dev/rack3	h10	✓	✓	✓	✓	✓	✓									12
Rack3	/dev/rack3	h11			✓	✓							✓	✓		✓	11
Rack3	/dev/rack3	h12			✓	✓				✓	✓						13

[3]*Total* cells show the total number of Core, Hadoop, and Ecosystem components installed on each host node for the example cluster.

*Denotes a service that is lightweight and stateless. For greater performance, consider running these services on all nodes and adding a load balancer to distribute network traffic.

Example 3: 50-Node Cluster

Examples 3 shows a 50-node cluster contained in five racks:

Example 3a. Core and Hadoop for 50-Node Cluster (Racks 1-3)

last-updated: Apr 11, 2022

Physical Rack/ Failure Domain	Recommended Topology	Hostname	Core										Hadoop		
			Core/Fluentd/Collectd (30)	Fileserver (30)	Client Components (27)	Zookeeper (3)	CLDB (3)	NFS (27)	Adminserver/Kibana/Grafana (3)	Gateway (3)	Metrics TSDB (3)	Metrics ES (3)	RM (3)	History (3)	Node Mgr (27)
Rack1	/data/rack1	h1	✓	✓	✓			✓	✓					✓	✓
Rack1	/data/rack1	h2	✓	✓	✓			✓							✓
Rack1	/data/rack1	h3	✓	✓	✓			✓					✓		✓
Rack1	/data/rack1	h4	✓	✓	✓			✓							✓
Rack1	/data/rack1	h5	✓	✓	✓			✓							✓
Rack1	/data/rack1	h6	✓	✓	✓			✓							✓
Rack1	/data/rack1	h7	✓	✓	✓			✓				✓			✓
Rack1	/data/rack1	h8	✓	✓	✓			✓			✓				✓
Rack1	/data/rack1	h9	✓	✓	✓			✓		✓					✓
Rack1	/data/rack1	h10	✓	✓		✓	✓								
Rack2	/data/rack2	h11	✓	✓	✓			✓	✓					✓	✓
Rack2	/data/rack2	h12	✓	✓	✓			✓							✓
Rack2	/data/rack2	h13	✓	✓	✓			✓					✓		✓
Rack2	/data/rack2	h14	✓	✓	✓			✓							✓
Rack2	/data/rack2	h15	✓	✓	✓			✓							✓
Rack2	/data/rack2	h16	✓	✓	✓			✓							✓
Rack2	/data/rack2	h17	✓	✓	✓			✓				✓			✓
Rack2	/data/rack2	h18	✓	✓	✓			✓			✓				✓
Rack2	/data/rack2	h19	✓	✓	✓			✓		✓					✓
Rack2	/data/rack2	h20	✓	✓		✓	✓								
Rack3	/data/rack3	h21	✓	✓	✓			✓	✓					✓	✓
Rack3	/data/rack3	h22	✓	✓	✓			✓							✓
Rack3	/data/rack3	h23	✓	✓	✓			✓					✓		✓
Rack3	/data/rack3	h24	✓	✓	✓			✓							✓
Rack3	/data/rack3	h25	✓	✓	✓			✓							✓
Rack3	/data/rack3	h26	✓	✓	✓			✓							✓
Rack3	/data/rack3	h27	✓	✓	✓			✓				✓			✓
Rack3	/data/rack3	h28	✓	✓	✓			✓			✓				✓
Rack3	/data/rack3	h29	✓	✓	✓			✓		✓					✓
Rack3	/data/rack3	h30	✓	✓		✓	✓								

Example 3b. Core and Hadoop for 50-Node Cluster (Racks 4-5)

last-updated: Apr 11, 2022

			Core									Hadoop		
Physical Rack/ Failure Domain	Recommended Topology	Hostname	Core/Fluentd/Collectd (20)	Fileserver (20)	Client components (18)	Zookeeper (2)	CLDB (2)	NFS (18)	Adminserver/Kibana/Grafana (0)	Gateway (2)	Metrics TSDB (2)	Metrics ES (2)	RM (0)	History (0)	Node Mgr (18)
Rack4	/data/rack4	h31	✓	✓	✓			✓							✓
Rack4	/data/rack4	h32	✓	✓	✓			✓							✓
Rack4	/data/rack4	h33	✓	✓	✓			✓							✓
Rack4	/data/rack4	h34	✓	✓	✓			✓							✓
Rack4	/data/rack4	h35	✓	✓	✓			✓							✓
Rack4	/data/rack4	h36	✓	✓	✓			✓							✓
Rack4	/data/rack4	h37	✓	✓	✓			✓				✓			✓
Rack4	/data/rack4	h38	✓	✓	✓			✓			✓				✓
Rack4	/data/rack4	h39	✓	✓	✓			✓		✓					✓
Rack4	/data/rack4	h40	✓	✓		✓	✓								
Rack5	/data/rack5	h41	✓	✓	✓			✓							✓
Rack5	/data/rack5	h42	✓	✓	✓			✓							✓
Rack5	/data/rack5	h43	✓	✓	✓			✓							✓
Rack5	/data/rack5	h44	✓	✓	✓			✓							✓
Rack5	/data/rack5	h45	✓	✓	✓			✓							✓
Rack5	/data/rack5	h46	✓	✓	✓			✓							✓
Rack5	/data/rack5	h47	✓	✓	✓			✓				✓			✓
Rack5	/data/rack5	h48	✓	✓	✓			✓			✓				✓
Rack5	/data/rack5	h49	✓	✓	✓			✓		✓					✓
Rack5	/data/rack5	h50	✓	✓		✓	✓								

Example 3c. Ecosystem Components for 50-Node Cluster (Racks 1-3)

last-updated: Apr 11, 2022

Physical Rack/ Failure Domain	Recommended Topology	Hostname	Ecosystem														
			HBase REST* (3)	HBase Thrift* (3)	Drill (27)	Spark (27)	Streams REST* (3)	HTTPFS* (3)	HS2 (3)	HS2 metaserver (0)	Webhcat(0)	Spark HS/App Timeline UI (3)	Oozie (3)	Sqoop (3)	Hue (3)	MySQL (0)	Total[3]
Rack1	/data/rack1	h1			✓	✓									✓		10
Rack1	/data/rack1	h2			✓	✓			✓			✓					9
Rack1	/data/rack1	h3			✓	✓											8
Rack1	/data/rack1	h4			✓	✓							✓	✓			9
Rack1	/data/rack1	h5		✓	✓	✓											8
Rack1	/data/rack1	h6			✓	✓	✓										8
Rack1	/data/rack1	h7			✓	✓											8
Rack1	/data/rack1	h8	✓		✓	✓											9
Rack1	/data/rack1	h9			✓	✓		✓									9
Rack1	/data/rack1	h10															4
Rack2	/data/rack2	h11			✓	✓									✓		10
Rack2	/data/rack2	h12			✓	✓			✓			✓					9
Rack2	/data/rack2	h13			✓	✓											8
Rack2	/data/rack2	h14			✓	✓							✓	✓			9
Rack2	/data/rack2	h15		✓	✓	✓											8
Rack2	/data/rack2	h16			✓	✓	✓										8
Rack2	/data/rack2	h17			✓	✓											8
Rack2	/data/rack2	h18	✓		✓	✓											9
Rack2	/data/rack2	h19			✓	✓		✓									9
Rack2	/data/rack2	h20															4
Rack3	/data/rack3	h21			✓	✓									✓		10
Rack3	/data/rack3	h22			✓	✓			✓			✓					9
Rack3	/data/rack3	h23			✓	✓											8
Rack3	/data/rack3	h24			✓	✓							✓	✓			9
Rack3	/data/rack3	h25		✓	✓	✓											8
Rack3	/data/rack3	h26			✓	✓	✓										8
Rack3	/data/rack3	h27			✓	✓											8
Rack3	/data/rack3	h28	✓		✓	✓											9
Rack3	/data/rack3	h29			✓	✓		✓									9
Rack3	/data/rack3	h30															4

[3]*Total* cells show the total number of Core, Hadoop, and Ecosystem components installed on each host node for the example cluster.

*Denotes a service that is lightweight and stateless. For greater performance, consider running these services on all nodes and adding a load balancer to distribute network traffic.

Example 3d. Ecosystem Components for 50-Node Cluster (Racks 4-5)

Physical Rack/ Failure Domain	Recommended Topology	Hostname	Ecosystem														
			HBase REST* (2)	HBase Thrift* (2)	Drill (18)	Spark (18)	Streams REST* (2)	HTTPFS* (2)	HS2 (2)	HS2 Metaserver * (2)	Webhcat (2)	Spark HS/App Timeline UI *(0)	Oozie (0)	Sqoop (0)	Hue (0)	MySQL *(2)	Total[3]
Rack4	/data/rack4	h31			✓	✓											7
Rack4	/data/rack4	h32			✓	✓			✓								8
Rack4	/data/rack4	h33			✓	✓										✓	8
Rack4	/data/rack4	h34			✓	✓				✓							8
Rack4	/data/rack4	h35		✓	✓	✓					✓						9
Rack4	/data/rack4	h36			✓	✓	✓										8
Rack4	/data/rack4	h37			✓	✓											8
Rack4	/data/rack4	h38	✓		✓	✓											9
Rack4	/data/rack4	h39			✓	✓		✓									9
Rack4	/data/rack4	h40															4
Rack5	/data/rack5	h41			✓	✓											7
Rack5	/data/rack5	h42			✓	✓			✓								8
Rack5	/data/rack5	h43			✓	✓										✓	8
Rack5	/data/rack5	h44			✓	✓				✓							8
Rack5	/data/rack5	h45		✓	✓	✓					✓						9
Rack5	/data/rack5	h46			✓	✓	✓										8
Rack5	/data/rack5	h47			✓	✓											8
Rack5	/data/rack5	h48	✓		✓	✓											9
Rack5	/data/rack5	h49			✓	✓		✓									9
Rack5	/data/rack5	h50															4

[3]*Total* cells show the total number of Core, Hadoop, and Ecosystem components installed on each host node for the example cluster.

*Denotes a service that is lightweight and stateless. For greater performance, consider running these services on all nodes and adding a load balancer to distribute network traffic.

Plan Initial Volumes

Describes why it is important to define volumes.

MapR manages the data in a cluster in a set of *volumes*. Volumes can be mounted in the Linux filesystem in a hierarchical directory structure, but volumes do not contain other volumes. Each volume has its own policies and other settings, so it is important to define a number of volumes in order to segregate and classify your data.

Plan to define volumes for each user, for each project, and so on. For streaming data, you might plan to create a new volume to store new data every day or week or month. The more volume granularity, the easier it is to specify backup or other policies for subsets of the data. For more information on volumes, see Managing Data with Volumes.

Security Considerations

Planning for security will help you identify security shortcomings and address them before you go into production.

MapR Data Platform releases provide security by default. If your cluster is not already secure, the MapR Converged Data Platform supports many different levels of security. For more information, see Getting Started with MapR Security on page 1405.

Before installing MapR software using the published packages, make sure that you have reviewed the list of known vulnerabilities in Security Vulnerabilities on page 6569. If a vulnerability applies to your release, contact your MapR support representative for a fix, and apply the fix immediately, if applicable.

last-updated: Apr 11, 2022

If the cluster you are planning to install must communicate with other clusters, the clusters should have similar security attributes. Mixing secure and nonsecure clusters is not recommended. For more information, see Setting Up Cross-Cluster Security on page 1482.

User Accounts

This section identifies how to organize authorized users of the cluster.

Part of the cluster plan is a list of authorized users of the cluster. It is preferable to give each user an account, because account-sharing makes administration less secure. Any user of the cluster must be established with the **same Linux UID and GID on every node in the cluster**. Central directory services, such as LDAP, AD, and IPA are often used to simplify user maintenance.

Next Step

After you have a complete cluster plan, you are ready to prepare each node.

It is important to begin installation with a complete Cluster Plan, but plans should not be immutable. Cluster services often change over time, particularly as clusters scale up by adding nodes. Balancing resources to maximize utilization is the goal, and it will require flexibility.

The next step is to prepare each node. Most installation difficulties are traced back to nodes that are not qualified to contribute to the cluster, or which have not been properly prepared. For large clusters, it can save time and trouble to use a configuration management tool such as Puppet or Chef.

Proceed to Preparing Each Node and assess each node.

Installing MapR and MapR Ecosystem Components

Describes how to install MapR software and ecosystem components with or without the MapR Installer.

This *Installation Guide* has been designed as a set of sequential steps. The following topics are listed in the order which you would complete them. You must complete each step before proceeding to the next step.

This section assumes that you have already reviewed Planning the Cluster on page 107.

Within the Data Fabric (v6) Admin Series of training courses, HPE offers an installation training course. For more information, see the nearby Recommended Resources.

You can either use the installer script, or perform a manual installation. The installer script takes care of using the right repositories for your Linux version, and there is nothing that you have to worry about. HPE recommends that you use the installer script. The technically inclined can perform a manual installation.

MapR Repositories and Packages

Describes the repositories for MapR software and the ecosystem components.

Repositories for MapR Core Software

HPE hosts `rpm` and `deb` repositories for installing the MapR core software using Linux package-management tools. For every release of the core software, a repository is created for each supported platform.

Platform-specific installation repositories are hosted at: https://package.mapr.hpe.com/releases/v6.x.x/<platform>.

Repositories for MapR Ecosystem Packs

A MapR Ecosystem Pack (EEP) provides a set of ecosystem components that work together. HPE hosts `rpm` and `deb` repositories for easily installing the ecosystem components.

These platform-specific repositories are hosted at the following location:

- https://package.mapr.hpe.com/releases/MEP/MEP-<version>/<platform>

For more information about the MapR Ecosystem Packs (EEPs), see MapR Ecosystem Packs on page 3174.

GitHub Repositories for Source Code

HPE releases the source code for ecosystem components to GitHub, including all patches that HPE has applied to the components. Source code for all releases since March 2013 are available at https://github.com/mapr.

Maven Repositories for Application Developers

HPE hosts a Maven repository where application developers can download dependencies on MapR software or Hadoop ecosystem components. Maven artifacts for all releases since March 2013 are available at Maven Artifacts for MapR on page 4155.

Other Scripts and Tools

Other MapR scripts and tools can be found in the following locations:

- https://package.mapr.hpe.com/scripts/
- https://package.mapr.hpe.com/tools/

Preparing Each Node

Defines minimum requirements for each node in your cluster.

Every node contributes to the cluster, so each node must be able to run MapR and Hadoop software. Nodes must meet minimum requirements for operating system, memory and disk resources and installed software, such as Java. *Including unsuitable nodes in a cluster is a major source of installation difficulty.*

Table

Component	Requirements
CPU	64-bit x86
OS	RedHat, Oracle Linux, CentOS, SLES, or Ubuntu
Memory	16 GB minimum, more in production
Disk	Raw, unformatted drives and no partitions
DNS	Hostname, reaches all other nodes
Users	Common users across all nodes; passwordless ssh (optional)
Java	Must run Java 1.8
Other	NTP, Syslog, PAM

> **Tip:** For enhanced node performance and reliability, always set the MAPR_SUBNETS enivronment variable.

Use the subsequent sections as a checklist to make each candidate node suitable for its assigned roles. Install MapR software on each node that you identify as meeting the minimum requirements.

CPU and Operating System

Describes how to check whether your processor and operating system are supported by MapR software.

last-updated: Apr 11, 2022

Processor is 64-bit

To determine the processor type, run

```
$ uname -m
x86_64
```

If the output includes "x86_64," the processor is 64-bit. If it includes "i386," "i486," "i586," or "i686," it is a 32-bit processor, which is not supported by MapR software.

If the results are "unknown," or none of the above, use one of the following commands.

```
$ uname -a
Linux mach-name 2.6.35-22-server #33-Ubuntu SMP Sun Sep 19 20:48:58 UTC
2012 x86_64 GNU/Linux
```

In the `cpuinfo` file, the flag 'lm' (for "long-mode") indicates a 64-bit processor.

```
$ grep flags /proc/cpuinfo
flags               : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge
mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss syscall nx rdtscp
lm constant_tsc up arch_perfmon pebs bts rep_good xtopology tsc_reliable
nonstop_tsc aperfmperf pni pclmulqdq ssse3 cx16 sse4_1 sse4_2 popcnt aes
hypervisor lahf_lm ida arat
```

Supported Operating Systems

For the supported operating systems, see Operating System Support Matrix on page 5522.

To determine the name and version of the installed operating system, run the `lsb_release -a` command.

There is no problem if the `lsb_release` command reports "No LSB modules are available."

```
$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 10.10
Release:        10.10
Codename:       maverick
```

If the `lsb_release` command is not found, try one of the following alternatives:

```
$ cat /proc/version
Linux version 2.6.35-22-server (build@allspice) (gcc version 4.4.5 (Ubuntu/
Linaro 4.4.4-14ubuntu4) ) #33-Ubuntu SMP Sun Sep 19 20:48:58 UTC 2012
```

```
$ cat /etc/*-release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=10.10
DISTRIB_CODENAME=maverick
DISTRIB_DESCRIPTION="Ubuntu 10.10"
```

If you determine that the node is running an older version of a supported OS, upgrade to at least a supported version, and test the upgrade before proceeding. If you find a different Linux distribution, such as Fedora or Gentoo, you must reformat and install a supported distribution on the node.

Memory and Disk Space

Describes required and recommended memory, storage, and disk capacities for each node.

Minimum Memory

A minimum of 16 GB memory is required on each node. MapR recommends at least 16 GB for a production environment. Typical MapR production nodes have 32 GB or more.

Run `free -g` to display total and available memory in gigabytes.

```
$ free -g
              total        used        free      shared      buffers
cached
Mem:              3           2           1           0
0           1
-/+ buffers/cache:            0           2
Swap:             2           0           2
```

If the `free` command is not found, you can use other options such as `grep MemTotal: /proc/meminfo`, `vmstat -s -SM`, `top`, or various GUI system-information tools.

MapR does not recommend using the `numad` service, since it has not been tested and validated with MapR software. Using the `numad` service can cause artificial memory constraints to be set, which can lead to performance degradation under load. To stop and disable the `numad` service:

1. Stop the service: `systemctl stop numad`.
2. Set the numad service *not* to start on reboot: `systemctl disable numad`

MapR does not recommend using *always overcommit* as it can lead to the kernel memory manager stopping processes to free memory, resulting in stopped MapR processes and system instability. Leave `vm.overcommit_memory` at its default value of 0, do not change the value to 1 or 2.

You can explore MapR on non-production equipment, but under the demands of a production environment, memory needs to be balanced against disks, network, and CPU.

Storage

For data disks, MapR Installer versions 1.12.0.0 and later require a minimum disk size that is equal to the physical memory on the node. If a data disk does not meet the minimum disk size requirement, a verification error is generated.

To display the currently available disks, use a command such as the following:

```
ls -l /dev/sd*
brw-rw---- 1 root 1000 8,  0 Sep 14 23:49 /dev/sda
brw-rw---- 1 root disk 8,  1 Sep 14 23:49 /dev/sda1
brw-rw---- 1 root disk 8,  2 Sep 14 23:49 /dev/sda2
brw-rw---- 1 root mapr 8, 16 Sep 20 11:44 /dev/sdb
brw-rw---- 1 root mapr 8, 32 Sep 20 11:44 /dev/sdc
brw-rw---- 1 root mapr 8, 48 Sep 20 11:44 /dev/sdd
```

To check the available disk space:

```
df /dev/sda
Filesystem     1K-blocks  Used Available Use% Mounted on
devtmpfs        12225720     0  12225720   0% /dev
```

MapR software works with raw unformatted devices and partitions. For optimized performance and high reliability, MapR recommends using raw unformatted devices. For data nodes, allocate at least three unmounted physical drives or partitions for MapR storage. MapR software uses disk spindles in parallel for faster read/write bandwidth and therefore groups disks into sets of three.

Minimum Disk Allocation: MapR software requires a minimum of one disk or partition for MapR data. However, file contention for a shared disk decreases performance. In a typical production environment,

multiple physical disks on each node are dedicated to the distributed MapR File System, which results in much better performance.

Maximum Disk Allocation: If you are planning to install multiple instances of MapR File System, the number of disks supported on a node can vary based on the number of instances you plan to install. For example, a single node with four instances of the MapR FileServer can support up to 360 disks.

Drive Configuration

Do not use RAID or Logical Volume Management with disks that are added to a MapR node. While MapR supports these technologies, using them incurs additional setup overhead and can affect your cluster's performance. Due to the possible formatting requirements that are associated with changes to the drive settings, configure the drive settings prior to installing MapR.

If you have a RAID controller, disable it, and let the system run in Host Bus Adapter (HBA) mode. For systems that do not support HBA, and have LSI MegaRAID controllers, configure the following drive-group settings for optimal performance:

Property (The actual name depends on the version)	Recommended Setting
RAID Level	RAID0
Stripe Size	>=256K
Cache Policy or I/O Policy	Cached IO or Cached
Read Policy	Always Read Ahead or Read Ahead
Write Policy	Write-Through
Disk Cache Policy or Drive Cache	Disabled

Enabling the Disk Cache policy can improve performance. However, enabling the Disk Cache policy is not recommended because it increases the risk of data loss if the node loses power before the disk cache is committed to disk.

Attention: Disable write caching on all MapR disks if the disks are not battery backed.

Minimum Disk Space

OS Partition. Provide at least 10 GB of free disk space on the operating system partition.

MapR File System. Provide the higher of 8 GB of free disk space or the memory allocated to the MapR MapR File System. Note that the disk space should be greater than the memory allocated to the MapR MapR File System.

Disk. Provide 10 GB of free disk space in the `/tmp` directory and 128 GB of free disk space in the `/opt` directory. Services, such as ResourceManager and NodeManager, use the `/tmp` directory. Files, such as logs and cores, use the `/opt` directory.

Swap space. For production systems, provide at least 4 GB of swap space. If you believe more swap space is needed, consult the swap-space recommendation of your OS vendor. The amount of swap space that a production system needs can vary greatly depending on the application, workload, and amount of RAM in the system. Note that the MapR Installer generates a warning if your swap space is either less than 10% of main memory, or less than 2 GB.

ZooKeeper. On ZooKeeper nodes, dedicate a partition, if practicable, for the `/opt/mapr/zkdata` directory to avoid other processes filling that partition with writes and to reduce the possibility of errors due to a full `/opt/mapr/zkdata` directory. This directory is used to store snapshots that are up to 64 MB. Since the four most recent snapshots are retained, reserve at least 500 MB for this partition. Do not share the physical disk where `/opt/mapr/zkdata` resides with any MapR MapR File System data partitions to avoid I/O conflicts that might lead to ZooKeeper service failures.

last-updated: Apr 11, 2022

Virtual Memory (swappiness)

Swappiness is a setting that controls how often the kernel copies the contents of RAM to swap. By setting `vm.swappiness` to the right value, you can prevent the system from swapping processes too frequently, but still allow for emergency swapping (instead of killing processes). For all Linux distributions, the MapR recommendation is to set `vm.swappiness` to 1.

To check the current value for `vm.swappiness` run:

```
cat /proc/sys/vm/swappiness
```

To change the value, run:

```
sudo sysctl vm.swappiness=1
```

The value of `vm.swappiness` can revert to a system default setting if you reboot the node. To make this setting permanent, enter `vm.swappiness=1` in `/etc/sysctl.conf` and save it.

Connectivity

This section describes and helps you troubleshoot connectivity requirements.

Fully Qualified Domain Names (FQDNs)

When you install a MapR cluster and you specify the host names using the MapR installer or the `configure.sh` script, use fully qualified domain names (FQDNs). **Do not use an alias or IP address to specify the host names.** Using an IP address can prevent services such as the timeline service from verifying security certificates. In addition, monitoring services can fail after installation because of connection requests that are rejected. These issues can be difficult to troubleshoot and can be prevented by using FQDNs.

It is important to use FQDNs when configuring a secure cluster. However, the practice also applies to non-secure clusters that might later be upgraded to be secure. The same connectivity issues can be encountered when a non-secure cluster is upgraded to a secure cluster. If your cluster is non-secure and will not be secured, or if you are not concerned about connection issues for the monitoring services, you may use IP addresses to specify the host names.

Unique Hostnames

Each node in the cluster must be accessible via DNS. More specifically, each node in the cluster must have a unique hostname, resolvable forward and backward with every other node with both normal and reverse DNS name lookup.

Run `hostname -f` to check the node's hostname. For example:

```
$ hostname -f
node125.corp.example.com
```

If `hostname -f` returns a name, run `getent hosts 'hostname'` to return the node's IP address and fully-qualified domain name (FQDN).

```
$ getent hosts 'hostname'
10.250.1.53     node125.corp.example.com
```

To troubleshoot hostname problems, edit the `/etc/hosts` file as `root`. A simple `/etc/hosts` might contain:

```
127.0.0.1     localhost
10.10.5.10    mapr-hadoopn.maprtech.prv mapr-hadoopn
```


last-updated: Apr 11, 2022

A common problem is an incorrect loopback entry (127.0.x.x) that prevents the IP address from being assigned to the hostname. For example, on Ubuntu, the default `/etc/hosts` file might contain:

```
127.0.0.1     localhost
127.0.1.1     node125.corp.example.com
```

A loopback (`127.0.x.x`) entry with the node's hostname will confuse the installer and other programs. Edit the `/etc/hosts` file and delete any entries that associate the hostname with a loopback IP. Only associate the hostname with the actual IP address.

Note: For more information about Ubuntu's default `/etc/hosts` file, see https://bugs.launchpad.net/ubuntu/+source/cloud-init/+bug/871966.

Use the `ping` command to verify that each node can reach the others using each node's hostname. For more information, see the hosts(5) man page.

Common Users

A user that accesses the cluster must have the same credentials and user ID (uid) on each node in the cluster. Every person or department that runs MapR jobs must have an account and must also belong to a common group ID (gid). The uid for each user, and the gid for each group, must be consistent across all nodes.

A `mapr` user must exist, and have the same UID across all the cluster nodes. The `mapr` user has full privileges to administer the cluster. If you create the `mapr` user before you install MapR , you can test for connectivity issues. If you do not create the `mapr` user, installing MapR automatically creates the user for you. The `mapr` user ID is automatically created on each node if you do not use a directory service, such as LDAP.

To create a group, add a user to the group, or create the `mapr` user, run the following command as root substituting a uid for *m* and a gid for *n*. (The error "cannot lock /etc/passwd" suggests that the command was not run as root.)

```
$ useradd mapr --gid n --uid m
```

Example: `$ groupadd -g 5000 mapr` `$ useradd -g 5000 -u 5000 mapr`

To verify that the users or groups were created, run `su mapr`. Verify that a home directory was created (usually `/home/mapr`) and that the users or groups have read-write access to it. The users or groups must have write access to the `/tmp` directory, or Warden will fail to start services.

Optional: Passwordless ssh

Setting up passwordless ssh is straightforward. On each webserver node, generate a key pair and append the key to an authorization file. Then copy this authorization file to each node, so that every node is available from the webserver node.

`su mapr` (if you are not already logged in as mapr) `ssh-keygen -t rsa -P '' -f ~/` *`filename`*

The `ssh-keygen` command creates *`filename`* , containing the private key, and *`filename`* `.pub`, containing the public key. For convenience, you may want to name the file for the hostname of the node. For example, on the node with hostname "node10.10.1.1,"

```
ssh-keygen -t rsa -P '' -f ~/node10.10.1.1
```

In this example, append the file `/home/mapr/node10.10.1.1.pub` to the `authorized_keys` file.

Append each webserver node's public key to a single file, using a command such as `cat` *`filename`* `.pub >> authorized_keys`. (The key file is simple text, so you can append the file in several ways, including a text editor.) When every webserver node's empty passphrase public key has been generated, and the

public key file has been appended to the primary "authorized_keys" file, copy this primary keys file to each node as `~/.ssh/authorized_keys`, where ~ refers to the mapr user's home directory (typically `/home/mapr`).

Recommended: Setting the MAPR_SUBNETS Variable
For enhanced performance and reliability, always set the MAPR_SUBNETS enivronment variable.

Java
To run MapR software and Hadoop, you must install a supported Java Development Kit (JDK) on your node.

Java

MapR Data Platform requires the Java Development Kit (JDK). Installing only the Java runtime environment (JRE) is not sufficient. Verify that one of the following JDK versions is installed on the node:

Java Requirements for MapR Core	• Sun Java JDK 1.8 • OpenJDK 1.8 **Note:** Make sure you have the development kit installed. Some JRE packages include jdk in the name, but do not provide the required JDK software.
Special Requirements for Using OpenJDK	If you use OpenJDK: • RedHat/CentOS must have `java-<version>-openjdk-devel` installed. • SLES nodes must have `java-<version>-openjdk-devel` installed. • Ubuntu nodes must have `openjdk-<version>-jdk` installed.

Note: The `openjdk-devel` and `openjdk-<version>-jdk` packages include the `jps` command that lists running Java processes and can show whether the CLDB has started. This command is not supported in the Sun Java JRE.

To install the Oracle/Sun Java JDK:	Obtain the Oracle/Sun Java Development Kit (JDK), available at Oracle's Java SE website. Find Java SE 8 in the archive of previous versions.

To install or update OpenJDK	Use a package manager, such as **yum** (RedHat, Oracle Linux, or CentOS), **apt-get** (Ubuntu) or **zypper** (SLES).
	• On RedHat, Oracle Linux, or CentOS:
	`# yum install java-1.8.0-openjdk-devel`
	• On Ubuntu:
	`# apt-get install openjdk-8-jdk`

last-updated: Apr 11, 2022

- SLES

```
# zypper install
java-1_8_0-openjdk-devel
```

Related reference

JDK Support Matrix on page 5596
This matrix shows the Java Development Kit versions supported by different MapR Data Platform releases.

Infrastructure
Identifies certain software and settings that contribute to your node's infrastructure.

Network Time

To keep all cluster nodes time-synchronized, MapR requires software such as a Network Time Protocol (NTP) server (or chrony for RHEL 7) to be configured and running on every node. If server clocks in the cluster drift out of sync, serious problems will occur with certain MapR services. MapR raises a Time Skew alarm on any out-of-sync nodes. For more information about obtaining and installing NTP, see http://www.ntp.org/.

Advanced: It is recommended to install an internal time server with which the cluster nodes can sync directly. If internet connectivity is lost, the time on the cluster nodes stays in sync. For more details, refer to the preceding documentation link for NTP

System Locale

Ensure that your system locale is set to `en_us`. For more information about setting the system locale, see this website.

Syslog

`Syslog` should be enabled on each node to preserve logs for killed processes or failed jobs. Modern versions such as `syslog-ng` and `rsyslog` are possible, making it more difficult to be sure that a `syslog` daemon is present. One of the following commands should suffice:

```
syslogd -v
service syslog status

rsyslogd -v
service rsyslog status
```

Default umask

To prevent significant installation problems, ensure that the default umask for the root user is set to 0022 on all MapR nodes in the cluster. You can change the umask setting in the `/etc/profile` file, or in the `.cshrc` or `.login` file. The `root` user must have a 0022 umask because the MapR `admin` user requires access to all files and directories under the `/opt/mapr` directory, even those initially created by root services.

ulimit

`ulimit` is a command that sets limits on a user's access to system-wide resources. Specifically, it provides control over the resources available to the shell and to processes started by it.

The `mapr-warden` script uses the `ulimit` command to set the maximum number of file descriptors (`nofile`) and processes (`nproc`) to 64000. Higher values are unlikely to result in an appreciable performance gain. Lower values, such as the default value of 1024, are likely to result in task failures.

last-updated: Apr 11, 2022

Warning: The MapR recommended value is set automatically every time Warden is started.

Depending on your environment, you might want to set limits manually for service accounts used to run I/O-heavy operations rather than relying on Warden to set them automatically using `ulimit`.

PAM

Nodes that run the **Control System** can take advantage of Pluggable Authentication Modules (PAM) if found. Configuration files in the `/etc/pam.d/` directory are typically provided for each standard Linux command. MapR can use, but does not require, its own profile.

Security - SELinux, AppArmor

See SELinux Support on page 139.

TCP Retries

On each node, set TCP retries for `net.ipv4.tcp_retries2` to 5 so that MapR can detect unreachable nodes with less latency.

Note: The installation automatically sets TCP retries for `net.ipv4.tcp_syn_retries` to 4 on each node.

1. Edit the file `/etc/sysctl.conf` and add the following line:

   ```
   net.ipv4.tcp_retries2=5
   ```

2. Save the file and run:

   ```
   sysctl -p
   ```

NFS

Disable the stock Linux NFS server on nodes that will run the MapR NFS server.

iptables/firewalld

Enabling `iptables` on a node can close ports that are used by MapR. If you enable `iptables`, make sure that required ports remain open. Check your current `iptables` rules by using the following command:

```
$ service iptables status
```

In CentOS 7, `firewalld` replaces `iptables`. To check your current `iptables` rules, use this command:

```
systemctl status firewalld
```

To ensure that the required ports are available, disable `firewalld` by using this command:

```
systemctl disable firewalld
```

Transparent Huge Pages (THP)

For data-intensive workloads, MapR recommends disabling the Transparent Huge Pages (THP) feature in the Linux kernel.

RHEL Example

```
$ echo never > /sys/kernel/mm/transparent_hugepage/enabled
```


last-updated: Apr 11, 2022

CentOS 7 Example

```
echo never > /sys/kernel/mm/transparent_hugepage/enabled
```

Ubuntu Example

```
$ echo never > /sys/kernel/mm/transparent_hugepage/defrag
```

Automated Configuration

Some users find tools such as `Ansible`, `Puppet`, or `Chef` useful to configure each node in a cluster. Make sure, however, that any configuration tool does not reset changes made when MapR packages are later installed. Specifically, do not let automated configuration tools overwrite changes to the following files:

- `/etc/sudoers`
- `/etc/sysctl.conf`
- `/etc/sysctl.d/60-mapr_elasticsearch.conf`
- `/etc/sysctl.d/60-mapr_fluentd.conf` on page 1395
- `/etc/security/limits.conf`
- `/etc/udev/rules.d/99-mapr-disk.rules`

Setting Resource Limits on CentOS/RedHat/Oracle Linux
While you can use Warden to automatically set resource limits, you may want to set limits manually.

Rather than relying on Warden to set resource file-access limits automatically using `ulimit`, you can use the following procedure to set the limits manually.

1. Edit `/etc/security/limits.conf` and add a line to set the resource limits. For example, set the resource limits to 65536.

   ```
   <MAPR_USER> - nofile 65536
   ```

2. Edit `/etc/security/limits.d/90-nproc.conf` to add a similar line.

   ```
   <MAPR_USER> - nproc 64000
   ```

3. Check that the `/etc/pam.d/su` file contains the following settings:

   ```
   #%PAM-1.0
   auth            sufficient      pam_rootok.so
   # Uncomment the following line to implicitly trust users in the "wheel"
   group.
   #auth           sufficient      pam_wheel.so trust use_uid
   # Uncomment the following line to require a user to be in the "wheel"
   group.
   #auth           required        pam_wheel.so use_uid
   auth            include         system-auth
   account         sufficient      pam_succeed_if.so uid = 0 use_uid quiet
   account         include         system-auth
   password        include         system-auth
   session         include         system-auth
   session         required        pam_limits.so
   session         optional        pam_xauth.so
   ```


last-updated: Apr 11, 2022

4. Use `ulimit` to verify settings.
5. Reboot the system.
6. Run the following command as the `mapr` user (not root) at a command line: `ulimit -n`

Setting Resource Limits on Ubuntu

While you can use Warden to automatically set resource limits, you may want to set limits manually.

Rather than relying on Warden to set resource limits automatically using `ulimit`, you can use the following procedure to set the limits manually.

1. Edit `/etc/security/limits.conf` and add a line to set the resource limits. For example, set the resource limits:

```
<MAPR_USER> - nofile 65536
<MAPR_USER> - nproc 64000
```

2. Edit `/etc/pam.d/su` and uncomment the following line.

```
session required pam_limits.so
```

3. Edit the `/etc/pam.d/common-session*` files to make sure the following entry is present:

```
# end of pam-auth-update config
session       required       pam_limits.so
```

4. Use `ulimit` to verify settings.
5. Reboot the system.
6. Run the following command as the `mapr` user (not root) at a command line: `ulimit -n`

Setting Resource Limits on SLES

While you can use Warden to automatically set resource limits, you may want to set limits manually.

Rather than relying on Warden to set resource limits automatically using `ulimit`, you can use the following procedure to set the limits manually.

1. Edit the `/etc/pam.d/common-session*` files to make sure the following entry is present:

```
# end of pam-auth-update config
session       required       pam_limits.so
```

2. Use `ulimit` to verify settings.
3. Reboot the system.
4. Run the following command as the `mapr` user (not root) at a command line: `ulimit -n`

SELinux Support

MapR Data Platform supports SELinux for cluster administrators who observe specific installation and administrative procedures.

Before using the MapR Data Platform with SELinux, note the following considerations and best practices:

last-updated: Apr 11, 2022

- **Installation:** Hewlett Packard Enterprise recommends disabling SELinux before installing data-fabric software. If you install the cluster by using the Installer, the Installer disables SELinux automatically. If you require the extra security provided by SELinux, you can enable SELinux and place it in enforcing mode after installation. Also, rules can be defined by observing regular operations while the cluster is running.
- **Known Issues:** For a list of known issues that you should be aware of when using SELinux with the MapR Data Platform, see Known issues: Running HPE Ezmeral Data Fabric on nodes with SELinux in enforcing mode.
- **Warnings in the Audit Log:** While using the HPE Ezmeral Data Fabric, if you see warnings in the SELinux audit log (`/var/log/audit/`) related to data-fabric services, the cluster admin can fix them by using `chcon` or similar tools.
- **Cluster-Admin Use of systemctl:** The data-fabric cluster admin (typically the `mapr` user) must be allowed to use `systemctl`. Without access to `systemctl`, Warden can fail to start cluster services.
- **System Administration:** SELinux introduces significant complexity and should be managed by an experienced system administrator. Managing SELinux is outside the scope of data-fabric cluster-administration activities.
- **Utilities and Services That Must Not Be Blocked** The following inexhaustive list of utilities and services must remain unblocked at all times for the MapR Data Platform to run successfully in an SELinux environment:
  - `bash`
  - `dmidecode`
  - `glibc`
  - `hdparm`
  - `initscripts`
  - `iputils`
  - `irqbalance`
  - `libgcc`
  - `libstdc++`
  - `lsof`
  - `net-tools`
  - `nfs-utils`
  - `nss`
  - `perl`
  - `python`
  - `redhat-lsb-core`
  - `rpcbind`

- `shadow-utils`
- `syslinux`
- `userspace-rcu`

Installing with the MapR Installer

The MapR Installer automates the process of installing MapR software and offers you a variety of options to complete the installation.

Use this option . . .	When	See for more information
MapR Installer web interface	You need a wizard-like tool to install MapR software, and you want visual feedback about the installation process.	MapR Installer on page 5395
MapR Installer Stanzas	You need a script-based tool to install MapR software, and you do not want to click through the menus and options provided by the web-based MapR Installer.	MapR Installer Stanzas on page 5503
MapR Installer Containers	You want to use either the web-based MapR Installer or Stanzas from a Docker container.	MapR Installer Containers on page 5497
MapR Installer in the Cloud	You want to deploy or modify a MapR cluster in Amazon AWS or Microsoft Azure.	Installing MapR in the Cloud on page 249

Note: If you do not want to use one of the MapR Installer options, you still have the option to install the software manually. See Installing without the MapR Installer on page 141.

Installing without the MapR Installer

Describes how to install MapR software and ecosystem components manually.

These steps describe how to install a secure MapR cluster. If your cluster does not need to be secure, you can still use this procedure and skip over the steps indicated for security.

After you have planned the cluster and prepared each node, you can install the MapR distribution from the MapR repository or package files. Installing the software requires that you perform certain steps on each node. You can install data-fabric ecosystem components, such as Hive, after you bring up the cluster.

Warning: Before you install, make sure that all nodes meet the requirements for installation. See Preparing Each Node on page 129 for more information. Failure to prepare nodes is the primary cause of installation problems. **You must also make sure that the package dependencies are installed. See MapR Installer Prerequisites and Guidelines on page 5396. These packages are downloaded for you when you use the MapR Installer, but must be installed manually before you install without using the MapR Installer.**

You must also have the following information from your cluster plan when you install:

- A list of the hostnames and IP addresses for all nodes and the services that you want to run on each node.
- A list of all disks and/or partitions to use on each node.

Note: For information about repositories and packages for MapR software and Hadoop Ecosystem tools, see MapR Repositories and Packages on page 128.

To successfully install MapR software, complete each step described in subsequent sections. To learn how HPE uses, shares, transfers, and manages personal information, see the HPE Privacy Statement.

last-updated: Apr 11, 2022

Step 1: Install the Package Key
Before you install MapR packages, you must install the package key.

MapR packages are cryptographically signed. Before you can install the packages, you must install the package key: `maprgpg.key`. For SLES only, you do not have to install the key because `zypper` allows package installation with or without the key.

- To install the package key, issue the command appropriate for your Linux distribution:
  - CentOS/RedHat/Oracle Linux

    ```
    rpm --import https://package.mapr.hpe.com/releases/pub/maprgpg.key
    ```

  - Ubuntu

    ```
    wget -O - https://package.mapr.hpe.com/releases/pub/maprgpg.key | sudo
    apt-key add -
    ```

Step 2: Prepare Packages and Repositories
To install services correctly, each node must have access to the package files.

The MapR software distribution is separated into two repositories that contain the package files:

- **MapR packages.** These provide core functionality for MapR clusters, such as the MapR filesystem.
- **Ecosystem packages.** These packages are not specific to MapR. Examples include the packages for Hive and Spark.

Some MapR services have internal dependencies that require additional packages. For example, when you install the CLDB service on a node, the node must also have `mapr-core` and `mapr-fileserver` installed. You can install dependencies on each node before beginning the MapR installation process, or you can configure repositories and allow the package manager on each node to resolve the dependencies. For a list of package dependencies, see Packages and Dependencies for MapR Software.

You can make packages available to each node, as described in subsequent sections, using the MapR Internet repository, a local repository, or a local path with rpm or deb package files. For information about packages and repositories for MapR software and Hadoop Ecosystem tools, see MapR Repositories and Packages on page 128.

Using the MapR Internet Repository
This section describes how to make packages available through the MapR Internet repository.

The MapR repository on the Internet provides all of the packages required to install a MapR cluster using native tools such as:

- `yum` on RedHat, Oracle Linux, or CentOS
- `zypper` on SLES
- `apt-get` on Ubuntu

Installing from the Internet repository is generally the easiest installation method, but requires the greatest amount of bandwidth. With this method, each node is connected to the Internet to download the required packages.

Set up repositories by completing the steps for your RedHat/Oracle Linux/CentOS, SLES, or Ubuntu distribution.

Adding the MapR Repository on RHEL, CentOS, or Oracle Linux
This section describes how to install the MapR Internet repository.

1. Change to the `root` user or use `sudo`.
2. Create a text file called `maprtech.repo` in the `/etc/yum.repos.d/` directory with the following content, replacing `<version>` with the version of MapR that you want to install: (For the correct paths for all past releases, see the MapR Repositories and Packages on page 128.)

```
[maprtech]
name=Your Company Name
baseurl=https://package.mapr.com/releases/v<version>/redhat/
enabled=1
gpgcheck=1
protect=1

[maprecosystem]
name=Your Company Name
baseurl=https://package.mapr.com/releases/MEP/MEP-<version>/redhat
enabled=1
gpgcheck=1
protect=1
```

3. If your connection to the Internet is through a proxy server, you must set the `http_proxy` environment variable before installation: You should also set the value for the `http_proxy` environment variable by adding the following section to the `/etc/yum.conf` file:

```
http_proxy=http://<host>:<port>
export http_proxy
```

```
proxy=http://<host>:<port>
proxy_username=<username>
proxy_password=<password>
```

4. If you are installing release 6.1.0 on RHEL or CentOS 8.x, enable the EPEL repository as described in Enable the EPEL Repository on CentOS 8.x, RHEL 8.x, or Oracle Linux 8.x on page 144. Starting in RHEL 8.x, a `mapr-core-internal` package dependency (`sdparm`) is deprecated and moved to EPEL, and installation cannot complete without enabling it.

Enable the EPEL Repository on CentOS 6.x, RHEL 6.x, or Oracle Linux 6.4 or higher
This section describes how to download and install the EPEL repository.

1. Download the EPEL repository:

```
wget https://archives.fedoraproject.org/pub/archive/epel/6/x86_64/
epel-release-6-8.noarch.rpm
```

2. Install the EPEL repository:

```
rpm -Uvh epel-release-6*.rpm
```

Enable the EPEL Repository on CentOS 7.x, RHEL 7.x, or Oracle Linux 7.0/7.1

This section describes how to download and install the EPEL repository.

1. Download the EPEL repository:

```
wget http://dl.fedoraproject.org/pub/epel/
epel-release-latest-7.noarch.rpm
```

2. Install the EPEL repository:

```
rpm -Uvh epel-release-latest-7*.rpm
```

Enable the EPEL Repository on CentOS 8.x, RHEL 8.x, or Oracle Linux 8.x

This section describes how to download and install the EPEL repository.

1. Download the EPEL repository:

```
wget http://dl.fedoraproject.org/pub/epel/
epel-release-latest-8.noarch.rpm
```

2. Install the EPEL repository:

```
rpm -Uvh epel-release-latest-8*.rpm
```

Adding the MapR Repository on SUSE

This section describes how to install the MapR Internet repository.

To verify that a SUSE release is supported by the MapR Data Platform, see Operating System Support Matrix on page 5522.

1. Change to the `root` user or use `sudo`.
2. Use the following command to add the repository for MapR packages, replacing `<version>` with the version of MapR that you want to install:

```
zypper ar https://package.mapr.hpe.com/releases/v6.x.x/suse/ maprtech
```

3. Use the following command to add the repository for ecosystem packages: (For the correct paths for all past releases, see the MapR Repositories and Packages on page 128.)

```
zypper ar https://package.mapr.hpe.com/releases/MEP/MEP-<version>/suse/
maprecosystem
```

4. If your connection to the Internet is through a proxy server, you must set the `http_proxy` environment variable before installation:

```
http_proxy=http://<host>:<port>
             export http_proxy
```

5. Update the system package index by running the following command:

```
zypper refresh
```

6. MapR packages require a compatibility package in order to install and run on SUSE. Execute the following command to install the SUSE compatibility package:

```
zypper install mapr-compat-suse
```

Installing sshpass

Before installing a cluster on a SUSE image, you must run the following command to install sshpass:

```
zypper --non-interactive -q --no-gpg-checks -p http://download.opensuse.org/
distribution/leap/42.3/repo/oss/ install sshpass
```

Adding the MapR Repository on Ubuntu
This section describes how to install the MapR Internet repository.

1. Change to the `root` user or use `sudo`.
2. Add the following lines to `/etc/apt/sources.list`, replacing `<version>` with the version of MapR that you want to install. See the MapR Repositories and Packages on page 128 for the correct paths for all past releases.

   Release 5.2.1 through 6.2.0

```
deb https://package.mapr.hpe.com/releases/v<version>/ubuntu/ binary
trusty
deb https://package.mapr.hpe.com/releases/MEP/MEP-<version>/ubuntu/
binary trusty
```

3. Update the package indexes:

```
apt-get update
```

4. If your connection to the Internet is through a proxy server, add the following lines to `/etc/apt/apt.conf`:

```
Acquire
{
  Retries "0";
  HTTP
  {
     Proxy "http://<user>:<password>@<host>:<port>";
  };
};
```

Using a Local Repository
This section describes how to make packages available through a local repository.

You can set up a local repository on each node to provide access to installation packages. With this method, nodes do not require internet connectivity. The package manager on each node installs from packages in the local repository. To set up a local repository, nodes need access to a running web server to download the packages.

Subsequent sections describe how to create a single repository that includes both MapR components and the Hadoop ecosystem components.

Creating a Local Repository on RHEL, CentOS, or Oracle Linux
This section describes how to create and use a local repository.

1. Log in as `root` on the node or use `sudo`.

2. Create the following directory if it does not exist: `/var/www/html/yum/base`
3. On a computer that is connected to the internet, download the following files, substituting the appropriate `<version>` number and `<datestamp>`: (See MapR Repositories and Packages on page 128 for the correct paths for all past releases.)

```
https://package.mapr.hpe.com/releases/v6.x.x/redhat/
mapr-<version>GA.rpm.tgz
https://package.mapr.hpe.com/releases/MEP/MEP-<version>/redhat/
mapr-mep-<version>-<datestamp>.rpm.tgz
```

4. Copy the files to `/var/www/html/yum/base` on the node, and extract them there.

```
tar -xvzf mapr-v<version>GA.rpm.tgz
tar -xvzf mapr-mep-<version>-<datestamp>.rpm.tgz
```

5. Create the base repository headers: When finished, verify the content of the new `/var/www/html/yum/base/repodata` directory: `filelists.xml.gz, other.xml.gz, primary.xml.gz, repomd.xml`

```
createrepo /var/www/html/yum/base
```

Add the repository on each node

Each node must contain your local repository.

- Create a text file called `maprtech.repo` in the `/etc/yum.repos.d` directory with the following content:

```
[maprtech]
name=MapR Technologies, Inc.
baseurl=http://<host>/yum/base
enabled=1
gpgcheck=0
```

Warning: The EPEL (Extra Packages for Enterprise Linux) repository contains dependencies for the `mapr-metrics` package on RedHat/CentOS/Oracle Linux. If your RedHat/CentOS/Oracle Linux cluster does not use the `mapr-metrics` service, you can skip EPEL configuration.

Enable the EPEL repository on CentOS 6.x, RHEL 6.x, or Oracle Linux 6.4 or higher

This section describes how to download and install the EPEL repository.

1. On a computer that is connected to the internet, download the EPEL repository:

```
wget https://archives.fedoraproject.org/pub/archive/epel/6/x86_64/
epel-release-6-8.noarch.rpm
```

2. Install the EPEL repository:

```
rpm -Uvh epel-release-6*.rpm
```

Enable the EPEL repository on CentOS 7.x, RHEL 7.x, or Oracle Linux 7.0/7.1

This section describes how to download and install the EPEL repository.

1. Download the EPEL repository:

```
wget http://dl.fedoraproject.org/pub/epel/
epel-release-latest-7.noarch.rpm
```

2. Install the EPEL repository:

```
rpm -Uvh epel-release-7*.rpm
```

Enable the EPEL repository on CentOS 8.x, RHEL 8.x, or Oracle Linux 8.x

This section describes how to download and install the EPEL repository.

1. Download the EPEL repository:

```
wget http://dl.fedoraproject.org/pub/epel/
epel-release-latest-8.noarch.rpm
```

2. Install the EPEL repository:

```
rpm -Uvh epel-release-8*.rpm
```

Creating a Local Repository on SUSE

This section describes how to create and use a local repository.

1. Login as `root` on the node or use `sudo`.
2. Create the following directory if it does not exist: `/var/www/html/zypper/base`
3. On a computer that is connected to the Internet, download the following files, substituting the appropriate `<version>` and `<datestamp>`: (See MapR Repositories and Packages on page 128 for the correct paths for all past releases.)

```
https://package.mapr.hpe.com/releases/v<version>/suse/
mapr-<version>GA.rpm.tgz
https://package.mapr.hpe.com/releases/MEP/MEP-<version>/suse/
mapr-mep-<version>-<datestamp>.rpm.tgz
```

4. Copy the files to `/var/www/html/zypper/base` on the node, and extract them there.

```
tar -xvzf mapr-<version>GA.rpm.tgz
tar -xvzf mapr-mep-<version>-<datestamp>.rpm.tgz
```

5. Create the base repository headers: When finished, verify the content of the new `/var/www/html/zypper/base/repodata` directory: `filelists.xml.gz, other.xml.gz, primary.xml.gz, repomd.xml`

```
createrepo /var/www/html/zypper/base
```

Add the repository on each node

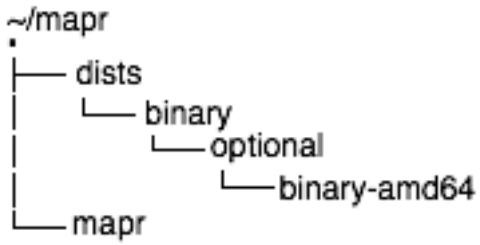
Each node must contain your local repository.

- Issue the following command to add the repository for MapR packages and the MapR ecosystem packages, substituting the appropriate `<host>`:

```
zypper ar http://<host>/zypper/base/ maprtech
```

Creating a Local Repository on Ubuntu
This section describes how to create and use a local repository.

1. Login as `root` on the machine where you will set up the repository.
2. Change to the directory `/root`, and create the following directories within it:



3. On a computer that is connected to the Internet, download the following files, substituting the appropriate `<version>` and `<datestamp>`: (See Data Fabric Repositories and Package Archives for the correct paths for all past releases.)

```
https://package.mapr.hpe.com/releases/v7.x.x/ubuntu/
mapr-<version>GA.deb.tgz
https://package.mapr.hpe.com/releases/MEP/MEP-<version>/ubuntu/
mapr-mep-<version>-<datestamp>.deb.tgz
```

4. Copy the files to `/root/mapr/mapr` on the node, and extract them there:

```
tar -xvzf mapr-<version>GA.deb.tgz
tar -xvzf mapr-mep-<version>-<datestamp>.deb.tgz
```

5. Navigate to the `/root/mapr/` directory.
6. Use `dpkg-scanpackages` to create `Packages.gz` in the `binary-amd64` directory:

```
dpkg-scanpackages . /dev/null | gzip -9c > ./dists/binary/optional/
binary-amd64/Packages.gz
```

7. Move the entire `/root/mapr/mapr` directory to the default directory served by the HTTP server (for example, `/var/www`), and make sure the HTTP server is running.

Add the repository on each node
Each node must contain your local repository.

1. On each node, use *one* of the following methods to add the repository:
   - If you have installed the `software-properties-common` package, use the `add-apt-repository` utility to add the repository:

     Release 7.0.0 and later

EEP	`add-apt-repository 'deb https://package.mapr.hpe.com/releases/MEP/ MEP-<version>/ubuntu binary bionic'`

Core	`add-apt-repository 'deb https://package.mapr.hpe.com/releases/v<version>/ ubuntu binary bionic'`

Releases 5.2.1 through 6.2.0

EEP	`add-apt-repository 'deb https://package.mapr.hpe.com/releases/MEP/ MEP-<version>/ubuntu binary trusty'`
Core	`add-apt-repository 'deb https://package.mapr.hpe.com/releases/v<version>/ ubuntu binary trusty'`

- If the `software-properties-common` package is not installed, create a file in `/etc/apt/sources.list.d` whose content is a single line as follows:

  Release 7.0.0 and later

EEP	`deb https://package.mapr.hpe.com/releases/MEP/MEP-<version>/ubuntu binary bionic`
Core	`deb https://package.mapr.hpe.com/releases/v<version>/ubuntu binary bionic`

  Releases 5.2.1 through 6.2.0

EEP	`deb https://package.mapr.hpe.com/releases/MEP/MEP-<version>/ubuntu binary trusty`
Core	`deb https://package.mapr.hpe.com/releases/v<version>/ubuntu binary trusty`

  Note: File names must end with `.list` and may only contain letters (a-z and A-Z), digits (0-9), underscore (_), hyphen (-), and period (.) characters.

2. On each node, update the package indexes (as `root` or with `sudo`). After performing these steps, you can use `apt-get` to install MapR software and Hadoop ecosystem components on each node from the local repository:

```
apt-get update
```

Using a Local Path with rpm or deb Package Files

This section describes how to make packages available through a local path.

You can download package files, store them locally, and then install MapR from the files. This option is useful for clusters that are not connected to the Internet.

Warning: In order for the installation to succeed, this method requires that you pre-install the MapR package dependencies on each node.

For a list of the dependency packages required for the MapR services that you are installing, see Packages and Dependencies for MapR Software on page 68. Manually download the packages and install them.

To install MapR from downloaded package files, complete the following steps:

1. Using a machine connected to the internet, download the tarball for the core components and the ecosystem components, substituting the appropriate `<platform>`, `<version>`, and `<datestamp>`:
   - `https://package.mapr.hpe.com/releases/v6.x.x/<platform>/mapr-v<version>GA.rpm.tgz` (or `.deb.tgz`)

last-updated: Apr 11, 2022

- `https://package.mapr.hpe.com/releases/MEP/MEP-<version>/<platform>/mapr-mep-<version>-<datestamp>.rpm.tgz` (or `.deb.tgz`)

  For the correct paths for all past releases, see MapR Repositories and Packages on page 128.

2. Extract the tarball to a local directory, either on each node or on a local network accessible by all nodes:

```
tar -xvzf mapr-<version>GA.rpm.tgz
tar -xvzf mapr-mep-<version>-<datestamp>.rpm.tgz
```

Step 3: Install Cluster Service Packages
The installation process varies based on the location of your packages and the configuration of your cluster.

Install services based on your cluster plan and service layout.

Before Installing Packages

Note these considerations:

- **Review security vulnerabilities:** Make sure that you have reviewed the list of known vulnerabilities in Security Vulnerabilities on page 6569. If a vulnerability applies to your release, contact your support representative for a fix. Apply the fix immediately, if applicable.
- **Collectd Known Issue (MFS-10783):** After a manual installation of release 6.1.0 or 6.1.1 on RHEL or CentOS 8.x, Collectd can fail to start because of a known issue. Error messages indicate that `libcrypto.so.10` cannot open a shared object file. If you are installing release 6.1.0 or 6.1.1 on RHEL or CentOS 8.x, install the `compat-openssl10` package on all cluster nodes before installing the `mapr-*` packages:

```
yum install compat-openssl10
```

List of Packages by Node

The following table lists the core packages to install on cluster nodes:

On These Nodes	Install These Packages
On all [compute] cluster nodes	`mapr-fileserver`

On designated cluster nodes	`mapr-cldb`
	`mapr-zookeeper`
	`mapr-mastgateway`
	`mapr-nfs` or `mapr-loopbacknfs`[1]
	`mapr-webserver`[2]
	`mapr-apiserver`[2]
	`mapr-gateway`
	`mapr-resourcemanager`
	`mapr-nodemanager`
	`mapr-historyserver`
	`mapr-timelineserver`

On client machines that run Hadoop commands that are not already part of the cluster	`mapr-client`

[1]See NFS Considerations on page 151.

[2]For special considerations related to the installation of the `mapr-webserver` and `mapr-apiserver` packages, see API Server and Web Server Packages for EEP 8.1.0 on page 6507.

Warning: This table is a rough guide and does not include the additional non-MapR packages required for internal Packages and Dependencies for MapR Software or Hadoop ecosystem components.

Install the packages based on a thorough plan. For example cluster designs, see Example Cluster Designs on page 118.

To install MapR, select one of the installation methods in the subsequent topics, depending on your operating system.

NFS Considerations

When you install `mapr-nfs`, NFSv3 is installed. To install NFSv4, you must use the `mapr-nfs4server` package. NFS is not secure by default. If you wish to configure NFSv4 server to work with Kerberos servers, you must first install Active Directory and Kerberos servers. For more information, see Installing MapR NFS on page 386 and Configuring NFSv4 Server for Kerberos on page 1209.

Consider installing `mapr-loopbacknfs` if you need a secure POSIX client. Note that the Installer installs `mapr-loopbacknfs` on all nodes in the cluster when **Enable NFS** is not specified. For more information about `mapr-loopbacknfs`, see MapR POSIX Clients on page 399.

Installing from a Repository

Before installing from the repository, change to the `root` user or use `sudo`.

- On RedHat, CentOS, or Oracle Linux, use the `yum` command to install the services that you want to run on the node.

  Syntax and Example

  ```
  yum install <package_name> <package_name> <package_name>
  ```

  ```
  yum install mapr-fileserver mapr-webserver
  ```

- On SLES, use the `zypper` command to install the services that you want to run on the node.

  Syntax and Example

  ```
  zypper install <package_name> <package_name> <package_name>
  ```

  ```
  zypper install mapr-fileserver mapr-webserver
  ```

- On Ubuntu, use the `apt-get` commands to update the Ubuntu package cache and install the services that you want to run on the node.

  1. Update the Ubuntu package cache:

     ```
     apt-get update
     ```

2. Install the services:

   Syntax and Example

```
apt-get install <package_name> <package_name> <package_name>
```

```
apt-get install mapr-fileserver mapr-webserver
```

Installing from a Local Repository

Before installing from the repository, change to the `root` user or use `sudo`.

- On RedHat, CentOS, Oracle Linux, or SLES, use `rpm` command to install the appropriate packages for the node:

  1. Change the working directory to the location where the `rpm` package files are located.
  2. Install the services:

     Syntax and Example

```
yum install <package_file> <package_file> <package_file>
```

```
yum install /path/to/mapr-core-<version>.x86_64.rpm
mapr-cldb-<version>.x86_64.rpm \
             mapr-resourcemanager-<version>.x86_64.rpm
mapr-webserver-<version>.x86_64.rpm \
```

     Note: Replace `<version>` with the exact version string found in the package filename.

- On Ubuntu, use the `dpkg` command to install the appropriate packages for the node.

  1. Change the working directory to the location where the `deb` package files are located.
  2. Install the services:

     Syntax and Example

```
dpkg -i <package_file> <package_file> <package_file>
```

```
dpkg -i mapr-core-<version>.x86_64.rpm mapr-cldb-<version>.x86_64.rpm \
             mapr-resourcemanager-<version>.x86_64.rpm
mapr-webserver-<version>.x86_64.rpm \
```

     Note: Replace `<version>` with the exact version string found in the package filename.

Installing from Package Files

When you install from package files, you must manually pre-install any dependency packages in order for the installation to succeed. Most MapR packages depend on the package `mapr-core`. Similarly, many Hadoop ecosystem components have internal dependencies. See Packages and Dependencies for MapR Software for details.

Step 4: Verify Installation Success
To confirm success, check each node.

To verify that the software was installed successfully, check the `/opt/mapr/roles` directory on each node. The software is installed in the `/opt/mapr` directory and a file is created in `/opt/mapr/roles` for every service that installs successfully. The following example shows the `/roles` directory with services that installed successfully:

Example

```
# ls -l /opt/mapr/roles
total 0
-rw-r--r-- 1 root root 0 Jul 11 07:29 cldb
-rw-r--r-- 1 root root 0 Jul 11 07:29 fileserver
-rw-r--r-- 1 root root 0 Jul 11 07:01 nodemanager
-rw-r--r-- 1 root root 0 Jul 11 07:01 resourcemanager
-rw-r--r-- 1 root root 0 Jul 11 07:29 apiserver
-rw-r--r-- 1 root root 0 Jul 11 07:29 zookeeper
```

Step 5: Set Environment Variables
Before starting ZooKeeper or Warden, you must complete this step.

Set Environment Variables on page 2289 for the cluster. The `/opt/mapr/conf/env.sh` script looks for the directory where Java is installed and sets `JAVA_HOME` automatically. However, if you need to specify a different location for `JAVA_HOME`, edit `/opt/mapr/conf/env_override.sh`. This variable *must* be set before starting ZooKeeper or Warden. For more information, see About env_override.sh on page 2290.

Step 6: Configure Nodes
Connect nodes to the cluster, configure security, and arrange node storage.

You run the `configure.sh` script on a node to enable the node to communicate with the cluster. You must configure each node that is part of the cluster and each node that connects to the cluster as a client.

Perform the following operations to configure a node:

Operation	Description
Prepare to run configure.sh	This topic describes some information you will need to gather before running the `configure.sh` script.
Enabling Security on page 154	These steps configure your cluster for security, enabling security for authentication, authorization, and data. Data-at-rest (DARE) encryption can also be enabled during this task. If you do not want your cluster to be secure, do not perform the steps in this procedure. Go to Configuring Nodes without Security on page 157.
Configure nodes without security	These steps configure each node to be a part of the cluster but do not configure security. If you want your cluster to be secure, do not perform this procedure. Go to Enabling Security on page 154.
Configure storage	To configure storage on a node, you can manually run `disksetup`. You need to perform this step for nodes in the cluster that are installed with the `mapr-fileserver`. The steps you use to configure storage vary depending on whether or not DARE is enabled or disabled.

Preparing to Run configure.sh
Before you run `configure.sh`, collect the information that you need to run the script based on your requirements.

The `configure.sh` script can configure a node for the first time or update existing node configurations. Therefore, it has many configuration options that you can use.

- Note the hostnames of the CLDB and ZooKeeper nodes. Optionally, you can specify the ports for the CLDB and ZooKeeper nodes as well. The default CLDB port is 7222. The default ZooKeeper port is 5181.

last-updated: Apr 11, 2022

- If a node in the cluster runs the HistoryServer, note the hostname for the HistoryServer. The HistoryServer node must be specified by using the `-HS` parameter.
- If one or more nodes in the cluster runs the ResourceManager, note the hostname or IP address for each ResourceManager node. Based on the version you install and your ResourceManager high availability requirements, you may need to specify the ResourceManager nodes using the `-RM` parameter. High availability for the ResourceManager is configured by default and does not need to be specified.
- If `mapr-fileserver` is installed on a node, you can use `configure.sh` with the `-F` option to format the disks and set up partitions. The `-F` option allows you to create a text file that lists the disks and partitions for use by the filesystem on the node. `configure.sh` passes the file to the `disksetup` utility. Each line lists either a single disk or all applicable partitions on a single disk. When listing multiple partitions on a line, separate each partition with a space. For example:

  ```
  /dev/sdb
  /dev/sdc1 /dev/sdc2 /dev/sdc4
  /dev/sdd
  ```

  Or you can manually run `disksetup` after you run `configure.sh`. See Configuring Storage on page 158.
- For a cluster node that is on a VM, use the `--isvm` parameter when you run `configure.sh`, so that the script uses less memory.

Enabling Security
Describes how to enable security for the cluster, platform, ecosystem components, and network-based connections.

The following steps enable:

- Security for the cluster nodes
- Wire-level encryption for the platform and ecosystem components
- Authentication for all network-based connections
- (Optional) Data-at-rest encryption on the cluster

When you set up a cluster, you must run the `configure.sh on page 2053` script on each node that you want to add to the cluster. After you enable security, review the System Behavior Changes After Enabling Security on page 1410.

Basic Procedure

To enable security for the cluster, follow these steps in order:

1. If the cluster is running, shut it down.

2. Delete the `ssl_keystore`, `ssl_keystore.p12`, `ssl_keystore.pem`, `ssl_truststore`, `ssl_truststore.p12`, and `ssl_truststore.pem` files on a machine where wire-level security is not enabled, as the `configure.sh` script fails if you already have these files in the directory.

   Note: The `ssl_keystore`, `ssl_keystore.p12`, `ssl_keystore.pem`, `ssl_truststore`, `ssl_truststore.p12`, and `ssl_truststore.pem` files are generated during installation of the Web server even if you did not enable security.

   For example, to delete the files, run the following commands:

   ```
   cd /opt/mapr/conf
   /bin/rm -f ssl_keystore ssl_keystore.p12 ssl_keystore.pem ssl_truststore
   ssl_truststore.p12 ssl_truststore.pem
   ```

   If you are re-running the script due to an invocation error from a previous run, the `cldb.key` and `maprserverticket` files may have already been created. Delete these files, as the script fails if you already have these files in the directory. For example, run the following command to delete these files:

   ```
   cd /opt/mapr/conf
   /bin/rm -f cldb.key maprserverticket ssl_keystore ssl_keystore.p12
   ssl_keystore.pem ssl_truststore ssl_truststore.p12 ssl_truststore.pem
   ```

3. Run the `configure.sh` script with the `-secure -genkeys -dare` options on the first CLDB node in your cluster:

   ```
   /opt/mapr/server/configure.sh -secure -dare -genkeys -Z
   <Zookeeper_node_list> -C <CLDB_node_list> -N <cluster_name>
   ```

   where both `<Zookeeper_node_list>` and `<CLDB_node_list>` have the form `hostname[:port_no][,hostname[:port_no]...]` and `-N <cluster_name>` specifies the cluster name. For the hostname, specify a fully-qualified domain name (FQDN) as described in Connectivity on page 133. Do not specify an alias or IP address. The `-dare` option is required only if you wish to enable data at rest encryption at the cluster-level.

   Important: You must run `configure.sh -dare -genkeys` only *once* on one CLDB node, since the resulting files must be copied to other nodes.

   This command generates the following files in the `/opt/mapr/conf` directory:

   - `cldb.key`
   - `dare.master.key`
   - `maprserverticket`
   - `ssl_keystore`
   - `ssl_keystore.p12`
   - `ssl_keystore.pem`
   - `ssl_truststore`
   - `ssl_truststore.p12`
   - `ssl_truststore.pem`

last-updated: Apr 11, 2022

Note: The `dare.master.key` file is generated only if data at rest encryption is enabled on the cluster.

Tip: A comprehensive listing of the Trust and Key Store files is at: Understanding the Key Store and Trust Store Files on page 1408.

4. Copy the `cldb.key` to any node with the CLDB or Zookeeper service installed, and copy the `dare.master.key` to any node with the CLDB service installed.

   Note: Copy the `dare.master.key` file only if you are enabling data at rest encryption on the cluster.

5. Verify that the files `ssl_keystore`, `ssl_keystore.p12`, `ssl_keystore.pem`, `ssl_truststore`, `ssl_truststore.p12`, and `ssl_truststore.pem` are owned by the user that runs cluster services. This user is `mapr` by default. Also, the `maprserverticket`, `ssl_keystore`, `ssl_keystore.p12`, and `ssl_keystore.pem` files must have their UNIX permission-mode bits set to `400`, and the `ssl_truststore`, `ssl_truststore.p12`, and `ssl_truststore.pem` files must be readable to all users.

6. Copy the `maprserverticket`, `ssl_keystore`, `ssl_keystore.p12`, `ssl_keystore.pem`, `ssl_truststore`, `ssl_truststore.p12`, and `ssl_truststore.pem` files to the `/opt/mapr/conf` directory of *every* node in the cluster.

7. Run `configure.sh` on each existing node in the cluster using the same arguments as in Step 3 but without the `-genkeys` option.

   ```
   /opt/mapr/server/configure.sh -secure -dare -Z <Zookeeper_node_list> -C
   <CLDB_node_list> -N <cluster_name>
   ```

   The `-secure` option indicates that security must be enabled on the node where the command is run and the `-dare` option indicates that data at rest encryption must be enabled on the node and must be specified only if it was specified in Step 3.

   Important: You must also do this on any nodes that you add to the cluster in the future.

8. Copy the `ssl_truststore`, `ssl_truststore.p12`,and `ssl_truststore.pem` files to any client nodes outside the cluster.

   Important: If you run `configure.sh -secure` on a node *before* you copy the necessary files to that node, the command fails.

9. Optionally, enable encrypted quorum ZooKeeper communication. See zoo.cfg on page 2220 for more information.

10. Log in as the `mapr` superuser using the maprlogin command: `maprlogin password` (in this command, `password` is literal text).

11. If clients will connect to multiple secure clusters, merge the `ssl_truststore` files with the `/opt/mapr/server/manageSSLKeys.sh` tool. See Setting Up the Client for more information on MapR clients.

Advanced Procedure

In certain situations, you may opt for variant of the basic procedure. Such situations include, but are not limited to the following:

- You are running the script on a host that is configured without a domain name.
- You have a cluster where all the machines do not have the same domain name.
- You wish to import your own custom certificates instead of the self-signed certificates generated by the `configure.sh` utility.

Running on Hosts with no Configured Domain Name

When used without the `-certdomain` argument, the `configure.sh` script discovers the domain name of the node on which it is being executed using the `hostname -d` command and then creates a 100-year self-signed certificate using the PKCS#1 v1.5 with SHA-512 hash function (SHA512withRSA) with a wildcard certificate with the common name `(CN) *.<domain>`. For example, if `hostname -d` returns the domain name `mycompany.com`, then the CN of the certificate is `*.mycompany.com`. This certificate works for all machines within the `mycompany.com` domain and can therefore be copied to all cluster nodes as specified in Step 5 in the Basic Procedure.

Certificate generation fails if the host that you are running the script from is configured without a domain name. To fix this, modify your machine configuration so that `hostname -d` returns a non-empty string and then run the `configure.sh` script.

Alternatively, re-run the script with the `-certdomain` option as shown in Step 3 of the Basic Procedure:

```
/opt/mapr/server/configure.sh -secure -genkeys -certdomain <domain_name> -Z
<Zookeeper_node_list> -C <CLDB_node_list> [ -N <cluster_name> ]
```

Securing Clusters with Multiple Domain Names

Generally, all machines within a cluster should belong to the same domain. In the unusual case where you have a cluster with different machines belonging to different domains, applications that perform hostname verification can fail if you run the `configure.sh` script (as described in Step 3 of the Basic Procedure on page 1406) to generate a single-domain wildcard certificate. In this case, you must provide your own multi-domain wildcard certificate and import the custom certificate into the keystore as described in Importing a Certificate Authority Signed (CA Signed) SSL Certificate Into a MapR Cluster.

Using Custom Certificates

To import your own custom certificates into the keystore instead of using the self-signed certificates, see Importing a Certificate Authority Signed (CA Signed) SSL Certificate Into a MapR Cluster.

Related reference

zoo.cfg on page 2220
Lists the ZooKeeper configuration file.

Configuring Nodes without Security
Describes how to configure all nodes without security during installation without the MapR Installer.

Note: This step does not configure your cluster for enhanced security. If you want your cluster to be secure, do not perform this step. Go to Enabling Security on page 154.

Run the `configure.sh` script with the following options on all nodes in the cluster:

```
/opt/mapr/server/configure.sh -Z <Zookeeper_node_list> -C <CLDB_node_list>
[-RM <host>] [-HS <host>] [-L <logfile>] [-N <cluster name>] [-F <disk
list>]
```

where:

- Both `<Zookeeper_node_list>` and `<CLDB_node_list>` have the form `hostname[:port_no][,hostname[:port_no]...]`.

last-updated: Apr 11, 2022

- For the hostname, specify a fully-qualified domain name (FQDN) as described in Connectivity on page 133. Do not specify an alias or IP address.
- Specify `-N <cluster_name>` if the cluster is not yet named.

Configuring Storage
This section describes how to format disks for cluster storage manually by using `disksetup`.

The disksetup utility formats disks for use by the MapR cluster. `disksetup` removes all data from the specified disks. Make sure you specify the disks correctly, and back up any data that you want to save. If you are re-using a node that was used previously in another cluster, it is important to format the disks to remove all data from the old cluster. For more information about the utility, see disksetup.

Note: The `disksetup` script assumes that you have free, unmounted physical partitions or hard disks for use by MapR software. To determine if a disk or partition is ready for use by MapR software, see Setting Up Disks for MapR.

disksetup and DARE

If data-at-rest-encryption (DARE) is enabled, you must use a different set of steps for configuring storage using `disksetup`. See the appropriate subtopic in this section.

Configuring Storage with DARE Disabled
This section describes how to format disks for cluster storage manually using `disksetup` with data-at-rest encryption (DARE) disabled.

Manually Running disksetup

You can create a text file that lists the disks and partitions for use by MapR software on a node. Each line should list either a single disk or all applicable partitions on a single disk. When listing multiple partitions on a line, separate each partition with a space. For example:

```
/dev/sdb
/dev/sdc1 /dev/sdc2 /dev/sdc4
/dev/sdd
```

In the following example, `/tmp/disklist` is a text file that lists the disks and partitions:

Example

```
/opt/mapr/server/disksetup /tmp/disklist
```

Configuring Storage with DARE Enabled
This section describes how to format disks for cluster storage manually using `disksetup` with data-at-rest encryption (DARE) enabled.

Manually Running disksetup

You can create a text file that lists the disks and partitions for use by MapR software on a node. Each line should list either a single disk or all applicable partitions on a single disk. When listing multiple partitions on a line, separate each partition with a space. For example:

```
/dev/sdb
/dev/sdc1 /dev/sdc2 /dev/sdc4
/dev/sdd
```

In the following example, `/tmp/disklist` is a text file that lists the disks and partitions:

Example

```
/opt/mapr/server/disksetup /tmp/disklist
```

Using disksetup with DARE-Enabled Nodes

In a DARE-enabled cluster, you must use a different set of steps to run `disksetup`. `disksetup` will fail on some nodes if the DARE master key is not available. CLDB nodes have a local copy of the DARE master key, so `disksetup` works on CLDB nodes. Other nodes require a connection with a running CLDB node in order to run `disksetup`.

Use these steps to run `disksetup` on the CLDB nodes and then start the CLDB nodes so that you can then run `disksetup` on the remaining nodes and start those nodes:

1. Format the disks on the CLDB nodes (the nodes that contain the `dare.master.key`):

   ```
   /opt/mapr/server/disksetup /tmp/disklist
   ```

2. Start ZooKeeper and Warden so that other nodes can access the DARE master key on the CLDB nodes:
   a. Start ZooKeeper on all the ZooKeeper nodes:

      ```
      service mapr-zookeeper start
      ```

   b. Start Warden on the CLDB nodes:

      ```
      service mapr-warden start
      ```

3. Format the remaining node disks:

   ```
   /opt/mapr/server/disksetup /tmp/disklist
   ```

4. Start Warden on the remaining nodes:

   ```
   service mapr-warden start
   ```

Step 7: Bring up the Cluster
Before you can install monitoring or ecosystem components, you must enable the cluster by starting ZooKeeper and Warden and verifying the cluster installation status.

Bringing up the cluster involves starting ZooKeeper and Warden, installing a MapR license, and viewing the cluster installation status. Once these initial steps are done, the cluster is functional, and you can use the Control System or the Command Line Interface (CLI) to examine nodes and activity on the cluster.

Starting ZooKeeper and Warden
Starting ZooKeeper and Warden brings up the cluster.

Depending on the options that you specified when running configure.sh on page 2053, Zookeeper and Warden might already be started.

Note: For a DARE-enabled cluster, you can skip this step because you already started the cluster in order to configure disk storage.

To check that Zookeeper is started, use this command on Zookeeper nodes:

```
systemctl status mapr-zookeeper
```

To check that Warden is started:

```
systemctl status mapr-warden
```

To start the cluster if Zookeeper and Warden are not started:

1. Start ZooKeeper on all nodes where it is installed, by issuing the following command:

   ```
   systemctl start mapr-zookeeper
   ```

2. Start Warden on all nodes:

   ```
   systemctl start mapr-warden
   ```

For clusters, ensure that Zookeeper has established a quorum. Use the nc command to check.

To install nc, use one of the following commands:

```
On RHEL: dnf install nmap-ncat
On SLES: zypper install netcat-openbsd
On Ubuntu: apt-get install netcat
```

To check for a quorum, run:

```
echo srvr | nc localhost 5181 | grep Mode
```

This command returns *leader* or *follower* if Zookeeper has established a quorum.

Note: For a single Zookeeper node (is not part of a cluster), the nc command always returns *standalone*.

Troubleshooting Installation
If you are having difficulty bringing up the cluster, you have a number of options.

Difficulty bringing up the cluster seems daunting, but most cluster problems are easily resolved. For the latest support tips, visit the Ezmeral Data Fabric Community.

- Can each node connect with the others? For a list of ports that must be open, see Ports Used by MapR Software on page 2290.
- Is the Warden service running on each node? On the node, run the following command as root:

  ```
  service mapr-warden status
     WARDEN running as process 18732
  ```

  If the Warden service is not running, check the Warden log file, `/opt/mapr/logs/warden.log`, for clues. To restart the Warden service:

  ```
  service mapr-warden start
  ```

- The ZooKeeper service is not running on one or more nodes:
  - Check the Warden log file for errors related to resources, such as low memory

    - Check the Warden log file for errors related to user permissions
    - Check for DNS and other connectivity issues between ZooKeeper nodes
- The `maprcli` program `/opt/mapr/bin/maprcli` won't run
    - Did you configure this node?
- Instance Mismatch Node Alarm is raised
    - Restart Warden to ensure that the number of MapR File System instances is as configured.
- Permission errors appear in the log
    - Check that MapR changes to the following files have not been overwritten by automated configuration management tools:

`/etc/sudoers`	Allows the `mapr` user to invoke commands as root
`/etc/security/limits.conf`	Allows MapR services to increase limits on resources such as memory, file handles, threads and processes, and maximum priority level
`/etc/udev/rules.d/99-mapr-disk.rules`	Covers permissions and ownership of raw disk devices

Before contacting your HPE support representative, collect your cluster logs by using the mapr-support-collect script.

Installing the Cluster License
You must have a valid license to unlock the enterprise features of the MapR Data Platform. You can obtain a trial license by contacting your HPE support representative.

For details, see Adding a License on page 777.

Verifying the Cluster Installation Status
You can use the command line or the Control System to verify the status of your installation.

Using the CLI to Check the Cluster Installation Status

1. Log in to a cluster node.

2. Use the following command to list the MapR services:

```
maprcli service list
  logpath                                displayname
name             state
  /opt/mapr/logs/mfs.log                 FileServer
fileserver       0
  /opt/mapr/hadoop/hadoop-2.7.0/logs     ResourceManager
resourcemanager  0
  /opt/mapr/hadoop/hadoop-2.7.0/logs     JobHistoryServer
historyserver    0
  /opt/mapr/logs/cldb.log                CLDB
cldb             0
  /opt/mapr/logs/nfsserver.log           NFS Gateway
nfs              0
  /opt/mapr/hadoop/hadoop-2.7.0/logs     NodeManager
nodemanager      0
  /opt/mapr/logs/gateway.log             GatewayService
gateway          0
  /opt/mapr/logs/hoststats.log           HostStats
hoststats        0
  /opt/mapr/apiserver/logs/apiserver.log APIServer
apiserver        0

maprcli license list
maprcli disk list -host <name or IP address>
```

3. Restart Warden on all remaining nodes using the following command:

```
service mapr-warden restart
```

Warden is then responsible for starting the rest of the services configured on each node.

Using the Control System to Check the Cluster Installation Status

1. Log in to the Control System using the host name of the node where you installed the `mapr-webserver`. For more information, see Setting Up the Control System on page 423.

   Note: Because MapR monitoring has not been installed yet and the Control System relies on MapR monitoring for metrics collection, some Control System functions will not be available. You can still use the Control System to check the cluster services.

2. Click the **Nodes** tab to verify that all nodes are present and healthy (no alarms are present).

3. Click the **Services** tab to check for any stopped or failed services.

Step 8: Install Metrics Monitoring

To enable metering and the display of certain metrics by the Control System, you must install collectd and OpenTSDB components. Metrics monitoring is part of MapR monitoring, which also includes log monitoring. MapR Monitoring components are available as part of the MapR Ecosystem Pack (EEP) that you selected for the cluster.

Complete these steps to install metrics monitoring as the `root` user or using `sudo`. Installing metrics monitoring components on a client node or edge node is not supported.

1. For metrics monitoring, install the following packages:

Component	Requirements
collectd	Install the `mapr-collectd` package on each node in the MapR cluster.

Component	Requirements
OpenTSDB and AsyncHBase	Install the `mapr-opentsdb` on one or more nodes. To allow failover of metrics storage when one OpenTSDB node is unavailable, install OpenTSDB on at least three nodes in the cluster. **Note:** `mapr-opentsdb` depends on `mapr-asynchbase`, and `mapr-asynchbase` is automatically installed on the node where you install `mapr-opentsdb`.
Grafana	Optional: Install the `mapr-grafana` package on at least one node in the MapR cluster. Grafana does not need to be installed for metering and is optional for metrics monitoring in general.

On a three-node cluster, you could run the following commands to install metrics packages:

- For CentOS/RedHat:
  - Node A: `yum install mapr-collectd mapr-grafana`
  - Node B: `yum install mapr-collectd mapr-opentsdb`
  - Node C: `yum install mapr-collectd`
- For Ubuntu:
  - Node A: `apt-get install mapr-collectd mapr-grafana`
  - Node B: `apt-get install mapr-collectd mapr-opentsdb`
  - Node C: `apt-get install mapr-collectd`
- For SLES:
  - Node A: `zypper install mapr-collectd mapr-grafana`
  - Node B: `zypper install mapr-collectd mapr-opentsdb`
  - Node C: `zypper install mapr-collectd`

2. **Release 6.0.1 and later:** Configure a password for Grafana:
   - For a **secured cluster**, ensure that the `/opt/mapr/conf/ssl_truststore.pem` file is present in `/opt/mapr/conf` on the Grafana nodes. If the `/opt/mapr/conf/ssl_truststore.pem` file is not present, you must copy it from the CLDB primary node to `/opt/mapr/conf` on the Grafana nodes.

     Note: In a secure cluster, Grafana uses PAM to authenticate using the cluster administrator login ID (typically the `mapr` user ID) and password, so no additional information is needed.

   - For a **nonsecured cluster**, you must configure a password for the Grafana `admin` user ID at the time of running `configure.sh`. If no password is provided, the password reverts to `admin`, which is the default pre-MEP 5.0.0 password. Use *one* of the following methods to pass the password to Grafana:

- On the nodes where Grafana is installed, export the password as an environment variable before calling `configure.sh`:

  ```
  export GRAFANA_ADMIN_PASSWORD="<newGrafanaPassword>"
  ```

  Then run `configure.sh` as you normally would run it (go to step 3).

- Add the following options to the `configure.sh` command in step 3. This method explicitly passes the password on the `configure.sh` command line:

  ```
  -EPgrafana '-password <newGrafanaPassword>'
  ```

- Add the following options to the `configure.sh` command in step 3. This method explicitly passes the password on the `configure.sh` command line by specifying a file:

  ```
  -EPgrafana '-password <name of local file containing new password>'
  ```

3. On **every cluster node**, run `configure.sh` with the `-R` and `-OT` parameters, and other parameters, as needed, for the Grafana password. A Warden service must be running when you use `configure.sh -R -OT`.

   ```
   /opt/mapr/server/configure.sh -R -OT <comma-separate list of OpenTSDB
   nodes>
   ```

Parameter	Description
`-R`	After initial node configuration, specifies that `configure.sh` should use the previously configured ZooKeeper and CLDB nodes.
`-OT`	Specifies a comma-separated list of host names or IP addresses that identify the OpenTSDB nodes. The OpenTSDB nodes can be part of the current MapR cluster or part of a different MapR cluster. The list is in the following format: • `hostname/IP address[:port_no] [,hostname/IP address[:port_no]...]` **Note:** The default OpenTSDB port is 4242. If you want to use a different port, specify the port number when you list the OpenTSDB nodes.

For example, to configure monitoring components you can run one of the following commands:

- In this example, default ports are used for the OpenTSDB nodes.

  ```
  /opt/mapr/server/configure.sh -R -OT NodeB
  ```

- In this example, non-default ports are specified for the OpenTSDB nodes:

  ```
  /opt/mapr/server/configure.sh -R -OT NodeB:4040
  ```

- If you are installing metrics monitoring only for use with the Metering utility, you can use the `-EPcollectd` option to collect only metering metrics and omit database metrics. See Installing Metering Using Manual Steps on page 284.

After you run `configure.sh -R`, if errors are displayed see Troubleshoot MapR Monitoring Installation Errors on page 172.

4. To start collecting metrics for the NodeManager and ResourceManager services, restart these services on each node where they are installed.

```
maprcli node services -name nodemanager -nodes <space separated list of
hostname/IPaddresses> -action restart
```

```
maprcli node services -name resourcemanager -nodes <space separated list
of hostname/IPaddresses> -action restart
```

Step 9: Install Log Monitoring

Installing the MapR monitoring logging components is optional. The logging components enable the collection, storage, and visualization of MapR core logs, system logs, and ecosystem component logs. MapR Monitoring components are available as part of the MapR Ecosystem Pack (EEP) that you selected for the cluster.

Note: As of release 6.0, if you install the logging components on a secure cluster, you must generate the Elasticsearch keys, certifications, and trust stores on an Elasticsearch node (designated as the primary) and then distribute them to the other Elasticsearch nodes in the cluster, as shown below in step 3.

Complete the steps to install the logging components as the `root` user or using `sudo`. Installing logging components on a client node or edge node is not supported.

1. For log monitoring, install the following packages:

Component	Requirements
fluentd	Install the `mapr-fluentd` package on each node in the cluster.
Elasticsearch	Install the `mapr-elasticsearch` package on at least three nodes in the cluster to allow failover of log storage if one Elasticsearch node is unavailable.
Kibana	Install the `mapr-kibana` package on at least one node in the cluster.

Note: On secure Ubuntu 14.04 or 16.04 clusters, Elasticsearch can fail to generate a keystore password if the `uuid-runtime` package is not installed. `uuid-runtime` is one of the package dependencies required for installing without the MapR Installer. See the entry for SPYG-934 and `uuid-runtime` in the Known Issues at Release (MapR 6.1.0) on page 47.

For example, on a three-node MapR cluster, you can run the following commands to install log packages:

- For CentOS/RedHat:
  - Node A: `yum install mapr-fluentd mapr-elasticsearch`
  - Node B: `yum install mapr-fluentd mapr-elasticsearch`
  - Node C: `yum install mapr-fluentd mapr-elasticsearch mapr-kibana`
- For Ubuntu:
  - Node A: `apt-get install mapr-fluentd mapr-elasticsearch`

last-updated: Apr 11, 2022

- Node B: `apt-get install mapr-fluentd mapr-elasticsearch`
- Node C: `apt-get install mapr-fluentd mapr-elasticsearch mapr-kibana`

- For SLES:
  - Node A: `zypper install mapr-fluentd mapr-elasticsearch`
  - Node B: `zypper install mapr-fluentd mapr-elasticsearch`
  - Node C: `zypper install mapr-fluentd mapr-elasticsearch mapr-kibana`

2. *For secure MapR clusters*, run `maprlogin print` to verify that you have a user ticket for the MapR user and the `root` user. These user tickets are required for a successful installation. If you need to generate a MapR user ticket, run `maprlogin password`. For more information, see Generating a MapR User Ticket on page 1426.

3. *For secure MapR clusters*, use the following steps to generate the Elasticsearch keys, certificates, and trust stores on the master Elasticsearch node, and then distribute them to other nodes:

   a. Select one Elasticsearch node as the master, and run `configure.sh -R` on that node.

   b. Run the following command on the master Elasticsearch node:

   ```
   /opt/mapr/server/configure.sh -OT <comma-separated list of
   OpenTSDB nodes> -ES <comma-separated list of Elasticsearch
   nodes> -R -EPelasticsearch -genESKeys
   ```

   For descriptions of the parameters used in this command, see the table in step 6.

   Important: If DNS resolution on the nodes is not completely configured, running the `configure.sh -genESKeys` step will fail. For example, key creation will fail if reverse lookups do not work. Suppose that the DNS lookup for a node with `hostname node1.qa.lab` returns 192.100.1.1. If the DNS lookup for 192.100.1.1 returns nothing (instead of `hostname node1.qa.lab`), key creation will fail.

   c. For Elasticsearch, copy the specified files to the locations indicated in the following table.

File	Permissions	Copy from location (on master)	Copy to location (all other ES nodes)
`es-root-ca.pem`	0600	`/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/ca/es-root-ca.pem`	`/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/ca`
`admin-usr-private-key.pem`	0600	`/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/admin-usr-private-key.pem`	`/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/certs`
`admin-usr-signed.pem`	0600	`/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/admin-usr-signed.pem`	`/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/certs`

last-updated: Apr 11, 2022

File	Permissions	Copy from location (on master)	Copy to location (all other ES nodes)
admin-usr-clientCombo.pem	0600	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/admin-usr-clientCombo.pem	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/certs
truststore.jks	0640	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/truststore.jks	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/keystores
admin-usr-keystore.jks	0640	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/admin-usr-keystore.jks	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/keystores/admin-usr-keystore.jks
<node-fqdn>-srvr-keystore.jks	0640	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/<node-fqdn>-srvr-keystore.jks	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/keystores **Note:** Only include the file that matches the nodename.
sg2.yml	0600	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/sg2.yml	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/
sg_http_<node-fqdn>.yml	0600	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/sg_http_<node-fqdn>.yml	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch **Note:** Only include the file that matches the nodename.
sg_ssl_<node-fqdn>.yml	0600	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/sg_ssl_<node-fqdn>.yml	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch **Note:** Only include the file that matches the nodename.
.keystore_password	0600	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/.keystore_password	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/.keystore_password

last-updated: Apr 11, 2022

Note: In the preceding table and the tables that follow, `<ver>` in a directory path represents the Elasticsearch, Kibana, or Fluentd version. These versions depend on the currently installed EEP. This table shows the versions for various EEPs:

EEP	Elasticsearch Version	Kibana Version	Fluentd Version
6.3.0	6.5.3	6.5.3	1.4.0
6.2.0	6.5.3	6.5.3	1.4.0
6.1.1	6.5.3	6.5.3	1.3.2.1
6.1.0	6.5.3	6.5.3	1.3.2.0
6.0.x	6.2.3	6.2.3	1.1.2
4.0.0 through 5.0.2	5.4.1	5.4.1	0.14.20

d. For Kibana, copy the specified files to the locations indicated in the following table:

Note: If the `/ca` and `/certs` copy-to directories do not exist for Kibana, you must create them. In addition, the directories and all files within them must be owned by `mapr:mapr`, or the services will not start.

File	Permissions	Copy from location (on master)	Copy to location (all other Kibana nodes)
`es-root-ca.pem`	0600	`/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/ca/es-root-ca.pem`	`/opt/mapr/kibana/kibana-<ver>/config/ca`
`kibanaserver-usr-clientCombo.pem`	0600	`/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/kibanaserver-usr-clientCombo.pem`	`/opt/mapr/kibana/kibana-<ver>/config/certs`
`kibanaserver-usr-signed.pem`	0600	`/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/kibanaserver-usr-signed.pem`	`/opt/mapr/kibana/kibana-<ver>/config/certs`
`kibanaserver-usr-private-key.pem`	0600	`/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/kibanaserver-usr-private-key.pem`	`/opt/mapr/kibana/kibana-<ver>/config/certs`

e. For Fluentd, copy the specified files to the locations indicated in the following table:

Note: If the `/ca` and `/certs` copy-to directories do not exist for Fluentd, you must create them. In addition, the directories and all files within them must be owned by `mapr:mapr`, or the services will not start.

File	Permissions	Copy from location (on master)	Copy to location (all other Fluentd nodes)
es-root-ca.pem	0600	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/ca/es-root-ca.pem	/opt/mapr/fluentd/fluentd-<ver>/etc/fluentd/ca/
fluentd-usr-signed.pem	0600	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/fluentd-usr-signed.pem	/opt/mapr/fluentd/fluentd-<ver>/etc/fluentd/certs
fluentd-usr-private-key.pem	0600	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/fluentd-usr-private-key.pem	/opt/mapr/fluentd/fluentd-<ver>/etc/fluentd/certs
fluentd-usr-clientCombo.pem	0600	/opt/mapr/elasticsearch/elasticsearch-<ver>/etc/elasticsearch/sg/fluentd-usr-clientCombo.pem	/opt/mapr/fluentd/fluentd-<ver>/etc/fluentd/certs

4. **MapR 6.0.1 and later:** *For secure MapR clusters,* copy the `$ES_HOME/etc/elasticsearch/.keystore_password` file from the Elasticsearch master node to `$KIBANA_HOME/config/.keystore_password` on the nodes where you are running Kibana. The `.keystore_password` file contains a system-generated password that is used to authenticate Kibanaserver-to-Elasticsearch-server communication.

5. **Release 6.0.1 and later:** *For secure MapR clusters,* configure a password for the Elasticsearch `admin` user to enable authentication for the end user using Kibana to search the Elasticsearch log index. This password needs to be provided at the time of running `configure.sh`. If no password is specified, you will default to the pre-mep-5.0.0, default password of `admin`. Use *one* of the following methods to pass the password to Elasticsearch/Kibana:

   - On the nodes where Fluentd/Elasticsearch/Kibana is installed, export the password as an environment variable before calling `configure.sh`:

     ```
     export ES_ADMIN_PASSWORD="<newElasticsearchPassword>"
     ```

     Then run `configure.sh` as you normally would run it (go to step 6).

   - Add the following options to the `configure.sh` command in step 6. This method explicitly passes the password on the `configure.sh` command line:

     ```
     -EPelasticsearch '-password <newElasticsearchPassword>' -EPkibana
     '-password <newElasticsearchPassword>' -EPfluentd '-password
     <newElasticsearchPassword>'
     ```

     Example

     ```
     /opt/mapr/server/configure.sh -R -v -ES mfs74.qa.lab -ESDB /opt/
     mapr/es_db -OT mfs74.qa.lab -C mfs74.qa.lab -Z
     mfs74.qa.lab -EPelasticsearch '-password helloMapR' -EPkibana
     '-password helloMapR' -EPfluentd '-password helloMapR'
     ```

- Add the following options to the `configure.sh` command in step 6. This method explicitly passes the password on the `configure.sh` command line by specifying a file:

```
-EPelasticsearch '-password <name of local file containing new
password>'  -EPkibana '-password <name of local file containing new
password>' -EPfluentd '-password <name of local file containing new
password>'
```

Example

```
/opt/mapr/server/configure.sh -R -v -ES mfs74.qa.lab -ESDB /opt/
mapr/es_db -OT mfs74.qa.lab -C mfs74.qa.lab -Z
mfs74.qa.lab -EPelasticsearch '-password /tmp/es_password' -EPkibana
'-password /tmp/es_password' -EPfluentd '-password /tmp/es_password'
```

6. Run `configure.sh` on each node in the MapR cluster with the `-R` and `-ES` parameters, adding parameters to configure the Fluentd/Elasticsearch/Kibana password as needed. Optionally, you can include the `-ESDB` parameter to specify the location for writing index data. A Warden service must be running when you use `configure.sh -R`.

```
/opt/mapr/server/configure.sh -R -ES <comma-separate list of
Elasticsearch nodes> [-ESDB <filepath>]
```

Parameter	Description
`-ES`	Specifies a comma-separated list of host names or IP addresses that identify the Elasticsearch nodes. The Elasticsearch nodes can be part of the current MapR cluster or part of a different MapR cluster. The list is in the following format: • `hostname/IPaddress[:port_no] [,hostname/IPaddress[:port_no]...]` **Note:** The default Elasticsearch port is 9200. If you want to use a different port, specify the port number when you list the Elasticsearch nodes.
`-ESDB`	Specifies a non-default location for writing index data on Elasticsearch nodes. In order to configure an index location, you only need to include this parameter on Elasticsearch nodes. By default, the Elasticsearch index is written to `/opt/mapr/elasticsearch/elasticsearch-<version>/var/lib/MaprMonitoring/`. **Note:** Elasticsearch requires a lot of disk space. Therefore, a separate filesystem for the index is strongly recommended. It is not recommended to store index data under the `/` or the `/var` file system. Upgrading to a new version of MapR monitoring removes the `/opt/mapr/elasticsearch/elasticsearch-<version>/var/lib/MaprMonitoring/` directory. If you want to retain Elasticsearch index data through an upgrade, you must use the `-ESDB` parameter to specify a separate filesystem or back up the default directory before upgrading. The Pre-Upgrade Steps for MapR Monitoring on page 342 include this step.

last-updated: Apr 11, 2022

Parameter	Description
`-genESKeys`	Generates needed keys and certificates for the master Elasticsearch node in a secure cluster.
`-OT`	Specifies a comma-separated list of host names or IP addresses that identify the OpenTSDB nodes. The OpenTSDB nodes can be part of the current MapR cluster or part of a different MapR cluster. Do not use this option when you configure a node for the first time. Use this option along with the `-R` parameter. A Warden service must be running when you use `configure.sh -R -OT`. The hostname list should use the following format: `hostname/IP address[:port_no] [,hostname/IP address[:port_no]...]` **Note:** The default OpenTSDB port is 4242. If you want to use a different port, specify the port number when you list the OpenTSDB nodes.
`-R`	After initial node configuration, specifies that `configure.sh` should use the previously configured ZooKeeper and CLDB nodes.

For example, to configure MapR monitoring components you can run one of the following commands:

- In this example, a location is specified for the Elasticsearch index directory, and default ports are used for Elasticsearch nodes:

```
/opt/mapr/server/configure.sh -R -ES NodeA,NodeB,NodeC -ESDB /opt/mapr/myindexlocation
```

- In this example, non-default ports are specified for Elasticsearch, and the default location is used for the Elasticsearch index directory:

```
/opt/mapr/server/configure.sh -R -ES NodeA:9595,NodeB:9595,NodeC:9595
```

After you run `configure.sh -R`, if errors are displayed see Troubleshoot MapR Monitoring Installation Errors on page 172.

7. If you installed Kibana, perform the following steps:
   a) Use one of the following methods to load the Kibana URL:
      - From the Control System, select the **Kibana** view. After you select the **Kibana** view, you may also need to select the **Pop-out page into a tab** option.
      - From a web browser, launch the following URL: `https://<IPaddressOfKibanaNode>:5601`
   b) When the Kibana page loads, it displays a `Configure an index pattern` screen. Provide the following values:

      Note: The **Index contains time-based events** option is selected by default and should remain selected.

last-updated: Apr 11, 2022

Field	Value
Index name or pattern	`mapr_monitoring-*`
Time-field	`@timestamp`

c) Click **Create**.

Troubleshoot MapR Monitoring Installation Errors

Review the following solutions to errors that you may encounter when you run `configure.sh` to configure MapR monitoring.

Elasticsearch Errors

could not determine matching interface

Cause: The DNS is probably not setup correctly on this node.

Solution: Contact your DNS administrator or verify that `/etc/hosts` and `/etc/nssswitch.conf` are configured correctly. The `etc/hosts` file should list the host names and the hosts parameter in `/etc/nssswitch.conf` should be set to files dns.

Failed to create <esdb directory name>

Cause: There is not enough disk space, `configure.sh` was not run as the `root` user, or the index directory path is not valid.

Solution: Complete the following steps and then re-run `configure.sh`:

- Verify that there is enough disk space.
- Verify that the file system where the index directory will be created is not read-only. See Log Aggregation and Storage on page 1391 for more information on the index directory location.
- If you included the `-ESDB` parameter, verify that the index directory path uses a valid format: `/<existing_directory1>/<existing_directory2>/<new index directory>`.

Failed to resolve hostname

Cause: The DNS is probably not set up correctly on this node.

Solution: Contact your DNS administrator or verify that `/etc/hosts` and `/etc/nssswitch.conf` are configured correctly. `/etc/hosts` should list the host names if the DNS database is not updated, and the hosts parameter in `/etc/nssswitch.conf` should be set to `files dns`

OpenTSDB Errors

Incompatible asynchbase jar found

Cause: The version of asynchbase installed on this node is not compatible with OpenTSDB.

Solution: Install the correct asynchbase package on each OpenTSDB node. See the EEP release notes to determine the compatibility between package versions.

Failed to install asynchbase Jar file

Cause: There is not enough disk space, the file system is read-only, or `configure.sh` was not run as the root user.

last-updated: Apr 11, 2022

Solution: Complete the following steps and then re-run `configure.sh`:

- Verify that you are logged in as the root user or using sudo.
- Verify that there is enough disk space.
- Verify that the file system containing the `/opt/mapr/opentsdb` directory is mounted as read/write.

Failed to create TSDB tables - need to rerun `configure.sh -R` or run `create_table.sh` as `$MAPR_USER`

Cause: On a secure cluster, this issue usually occurs when you run `configure.sh` to configure the nodes to use MapR monitoring without first creating user tickets for the `root` user and the `$MAPR_USER`.

Solution: Complete one or all of the following steps:

- Verify that you have a user ticket for both root and the $MAPR_USER before running `configure.sh` or `create_table.sh`. Run `maprlogin print` to verify that you have a user ticket for the MapR user and the `root` user. If you need to generate a MapR user ticket, run `maprlogin password`.
- Re-run `configure.sh`. For example, `configure.sh -R -OT <comma-separate list of OpenTSDB nodes> -ES <comma-separate list of Elasticsearch nodes>`
- Run `${OTSDB_HOME}/share/opentsdb/tools/create_table.sh` to create OpenTSDB tables in the `mapr.monitoring` volume.

Fluentd Errors

fluentd service not enabled - missing clusterid

Cause: The MapR cluster was not up and running before `configure.sh` was run with the options to configure MapR monitoring.

Solution: Complete the following steps and then re-run `configure.sh`:

- Verify that the CLDB services is running. If not, start Warden with the following command: `service mapr-warden start`.
- Verify that the `$MAPR_HOME/conf/clusterid` file exists.

Collectd Errors

collectd service not enabled - missing clusterid

Cause: The cluster was not up and running before `configure.sh` was run with the options to configure MapR monitoring.

Solution: Complete one or all of the following steps and then re-run `configure.sh`:

	• Verify that the CLDB services is running. If not, start Warden with the following command: `service mapr-warden start`. • Verify that the `$MAPR_HOME/conf/clusterid` file exists.

Grafana Errors

Failed to pick default data source host	**Cause:** The OpenTSDB nodes list defined by the `-OT` parameter was incorrect. **Solution:** Check the syntax and the validity of those nodes.
Failed to create scratch config file	**Cause:** There is not enough disk space, `configure.sh` was not run as the `root` user, or the file system was mounted as read-only. **Solution:** Complete one or all of the following steps and then re-run `configure.sh`: • Verify that you are logged in as the `root` user or using sudo. • Verify that there is enough disk space. • Verify that the file system containing `/opt/mapr/grafana` is mounted as read/write.
Failed to change the port	**Cause:** The `sed` utility failed to edit a port value in the temporary configuration file. This may occur due to issues with file system permissions, disk space, or file corruption. **Solution:** Complete the following steps and then re-run `configure.sh`: • Verify that the file system containing `/opt/mapr/grafana` is mounted as read/write. • Verify that there is enough disk space. • Re-install the MapR monitoring packages.
Failed to configure ssl for grafana	**Cause:** The `sed` utility failed to edit a port value in the temporary configuration file. This can occur due to issues with file system permissions, disk space, or file corruption. **Solution:** Complete the following steps and then re-run `configure.sh`: • Verify that the file system containing `/opt/mapr/grafana` is mounted as read/write. • Verify that there is enough disk space. • Re-install the MapR monitoring packages.
ERROR: Failed to install grafana warden config file	**Cause:** There is not enough disk space, `configure.sh` was not run as the root user, or the file system containing `/opt/mapr/grafana` is mounted as read-only.

last-updated: Apr 11, 2022

Solution: Complete the following steps and then re-run `configure.sh`:

- Verify that you are logged in as the `root` user or using sudo.
- Verify that there is enough disk space.
- Verify that the file system containing `/opt/mapr/grafana` is mounted as read/write.

Kibana Errors

Failed to configure elasticsearch server URL

Cause: There is not enough disk space, `configure.sh` was not run as the root user, or the file system containing `/opt/mapr/kibana` is mounted read-only.

Solution: Complete the following steps and then re-run `configure.sh`:

- Verify that you are logged in as the `root` user or using sudo.
- Verify that there is enough disk space.
- Verify that the file system containing `/opt/mapr/kibana` is mounted as read/write.

Failed to configure ssl for Kibana

Cause: There is not enough disk space or the root user does not have the required directory permissions to create the file in the `/opt/mapr/kibana/kibana-<version>/config` directory.

Solution: Complete the following steps and then re-run `configure.sh`:

- Verify that there is enough disk space.
- Verify that the file system containing `/opt/mapr/kibana` is mounted as read/write.

Kibana logon unsuccessful because Searchguard "Service Unavailable"

Cause: On a slower server, Kibana logons sometimes do not succeed because the Searchguard configuration containing the user names and passwords has not finished loading. You might notice an error like the following in the Elasticsearch `MapRMonitoring.log`:

```
[2018-06-30T07:47:15,062][ERROR]
[c.f.s.a.BackendRegistry  ] Not
yet initialized (you may
need to run sgadmin)
```

Solution: Try using the following command to restart Elasticsearch:

```
maprcli node services -name
elasticsearch -action restart -nodes $
(hostname -f)
```

Step 10: Install Ecosystem Components Manually

You can install one or more ecosystem components from any MapR Ecosystem Pack (EEP) that is supported by the MapR cluster version. An EEP consists of a group of ecosystem components that work together.

Prerequisite: Set up the EEP Repository

Complete the following steps on each node in the cluster:

1. Verify that each node can access the ecosystem packages associated with the EEP version that you want to install. For information about how to set up the ecosystem repositories or to manually download each package, see Step 2: Prepare Packages and Repositories on page 142.
2. Update the repository cache to get the latest list of available packages:
   - On RedHat/CentOS:

     ```
     yum clean all
     ```

   - On SLES:

     ```
     zypper refresh
     ```

   - On Ubuntu:

     ```
     apt-get update
     ```

Manually Install Ecosystem Components

Review the MapR Ecosystem Pack Release Notes to determine the list of ecosystem components available in the EEP that you have selected. Then, complete the installation steps for each component that you want to install.

Note:

- If you want to use the optional OJAI Distributed Query Service on page 505, you must install Drill. See Installing Drill on page 177.
- For special considerations related to the installation of the `mapr-apiserver` and `mapr-webserver` packages (for the control system) in releases 6.2.0 and later, see Setting Up the Control System on page 423.

Installing AsyncHBase Libraries

This topic includes instructions for using package managers to download and install AsyncHBase from the EEP repository.

For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

Execute the following commands as `root` or using `sudo`.

1. Based on your operating system, run one of the following commands to install the package:

On Red Hat /Centos	`yum install mapr-asynchbase`

On SLES	`zypper install mapr-asynchbase`
On Ubuntu	`apt-get install mapr-asynchbase`

2. Run the `configure.sh` script with the following command to configure the AsyncHBase role for the node:

```
/opt/mapr/server/configure.sh -R
```

After installing the `mapr-asynchbase` package, the AsyncHBase JAR file `asynchbase-<version>-mapr.jar` is in the `/opt/mapr/asynchbase/asynchbase-<version>` directory.

Installing Drill

This topic provides instructions for using package managers to download and install Drill.

You can install and run Drill on any number of nodes in your MapR cluster. You can install Drill to run under the Warden service or under YARN. Starting in MapR 6.0 and Drill 1.11, Drill is secured by default when you install Drill on a secure MapR 6.x cluster.

Note: See Component Versions for Released EEPs for version support in each EEP release.

MapR Default Security Configuration

Starting in Drill 1.11, Drill is automatically secured when you install Drill in a 6.x MapR cluster that was installed with the default MapR security configuration. The default security configuration uses MapR-SASL (mapr tickets) to provide authentication, authorization, and encryption for cluster security.

Note: The default security configuration does not support Drill-on-YARN.

The default MapR security configuration is not required. You can install Drill and configure custom security, or turn security off after installing with the default security configuration. See the *Drill Installation Security Scenarios* section below for more information. See Securing Drill.

Installing Drill Under Warden or YARN

You can install and run Drill under Warden or you can install and run Drill under YARN. If you are currently running Drill under Warden, you can upgrade Drill and continue to run Drill under Warden, or you can migrate Drill to run under YARN. See Migrate Drill to Run Under YARN for instructions.

When Warden manages the Drill cluster, you can use the MapR Control System for monitoring. YARN (Yet Another Resource Negotiator) is a cluster management tool that automates the resource sharing process in a cluster. When you launch Drill under YARN, YARN deploys (localizes) Drill onto each node. You can monitor the Drill cluster using the Drill-on-YARN Application Master web UI.

Drill Packages

You can use package managers to manually install the appropriate Drill package. The Drill packages provide the software needed to run Drill. MapR provides mapr-drill package and also a mapr-drill-yarn package.

Drill includes the Drill daemon, the core Drillbit service that runs on a node. Each node running the Drillbit service can receive, plan, and execute queries sent from a client. The software also includes the drill-shell command line interface, a pure-Java console-based utility, for connecting to a Drill cluster and executing SQL commands.

The following sections list the Drill packages and their descriptions:

last-updated: Apr 11, 2022

mapr-drill	The mapr-drill package is required to run Drill under the MapR Warden service. This package installs or upgrades the Drill software in `/opt/mapr/drill` and integrates Drill with the MapR Warden service. You install this package on all nodes designated to run Drill. **Note:** Verify that you get both the mapr-drill and mapr-drill-internal packages, especially if you install or upgrade through the URL or download through the MapR repositories. Also, verify that the packages have the same version. For example, if you install Drill 1.14, both packages should be version 1.14.
mapr-drill-yarn	The mapr-drill-yarn package is required to run Drill under YARN. This package installs the Drill software in `/opt/mapr/drill`. You install this package on the node that you designate as the Drill-on-YARN client. See Install Drill to Run Under YARN for details. YARN deploys Drill to every node included in the Drill cluster. Installing this package on every node is not required. **Note:** If any users need to access SQLLine, you must install the mapr-drill-yarn package on every node where users expect access to SQLLine.

Drill and Query Services

To use the optional OJAI Distributed Query Service on page 505, you must install Drill and configure and register the service. See Configure the OJAI Distributed Query Service on page 182.

Drill Installation Security Scenarios

The following sections describe some manual installation scenarios for Drill with information about security configuration:

Installing or Upgrading Drill

You can install Drill on a MapR cluster with or without default security. After you install the Drill package, you must run the configuration script, configure.sh -R, to configure the Drill service on the nodes. When you run the configuration script, the script recognizes whether your MapR cluster is using the default security or not, and configures Drill accordingly.

In a secure cluster, an internal Drill configuration script automatically adds the security configuration to the `drill-distrib.conf` and `distrib-env.sh` files. See Securing Drill.

Note: You can override these default security settings in the `drill-override.conf` file, but doing so is not recommended or supported.

If your cluster is not using the default security, the internal Drill configuration script does not configure any security for Drill. Instead, it copies warden.drill-bits.conf to the conf.d directory.

Installing Drill with MapR and Configuring Custom Security

If you install MapR and Drill, and you want to manually secure the cluster and Drill instead of using the default security option, you must add a .customSecure file to the /opt/mapr/conf/ directory before you run configure.sh, as shown:

1. Run `/usr/bin/touch /opt/mapr/conf/.customSecure` to add the .customSecure file.
2. Run `configure.sh -R`.

last-updated: Apr 11, 2022

The configuration script recognizes the .customSecure file which indicates not to configure the default security settings. At this point, you can manually configure security in drill-override.conf.

Turning Default Security On|Off After You Install Drill

If you installed Drill on a MapR cluster with the default security option enabled, you can disable it. Likewise, if you did not enable the default security option, you can enable it. When you enable or disable the default security option, security is applied or removed across the cluster.

For example, if your cluster is not secure and you enable the default security option, the entire MapR cluster is secured, as well as all supported ecosystem components, including Drill.

In such a scenario, you must run the configure.sh script with the `-secure` or `-unsecure` and `-forceSecurityDefaults` flags to update the configuration of the nodes and apply or remove security across the cluster, as shown:

```
/opt/mapr/server/configure.sh -forceSecurityDefaults [ -unsecure | -secure ]
 -C <CLDB_node> -Z <ZK_node>
```

The configuration script communicates with internal ecosystem scripts to automatically configure or remove security across the cluster.

Component and System Compatibility Matrix

See the Interoperability Matrix pages for information about the compatibility of Drill with operating systems and ecosystem projects.

Drill Storage and Format Plug-in Support Matrix

See the Drill Storage and Format Plugin Support Matrix page for a list of supported and unsupported data sources and formats in Drill on MapR.

Install Drill to Run Under Warden

Verify that your system meets the prerequisites listed below and then follow the instructions listed in Installing Drill to Run Under Warden to install the mapr-drill package on all nodes designated to run Drill under the MapR Warden service.

Note: See Component Versions for Released EEPs for version support in each EEP release.

Prerequisites

Before you install Drill, read Installing Drill, and verify that the system meets the following prerequisites:

- The MapR cluster is installed and running. Installing Drill first can result in configuration issues
- The EEP repository is configured. For instructions, see Step 9: Install Ecosystem Components Manually.

Refer to the Apache Drill Release Notes and Drill Release Notes for a list of known issues.

Hive and HBase Support

Installation of a supported version of Hive is optional. Support differs, depending on the MapR Ecosystem Pack version that you install. See Component Versions for Released EEPs for version support in each EEP release.

Note: As of MapR 6.0, HBase is not supported.

Installing Drill to Run Under Warden

last-updated: Apr 11, 2022

Explains how to manually install the latest version of Drill to run under the MapR Warden service on the MapR Converged Data Platform.

Note: Starting in Drill 1.11, Drill is automatically secured when installed on a 6.x MapR cluster with the default MapR security configuration. The default security configuration uses MapR security (mapr tickets) to provide authentication, authorization, and encryption for cluster security. See Securing Drill amd Component Versions for Released EEPs for more information.

Complete the following steps as `root` or using `sudo` to install Drill on a client or server node:

1. To install Drill, issue the command appropriate for your system:

RedHat/CentOS	`yum install mapr-drill`
Ubuntu	`apt-get install mapr-drill`
SLES	`zypper install mapr-drill`

   Note: SLES is supported as of Drill 1.9.0-1703 and Drill 1.10.0-1703.

2. Run the configuration script to update the node configuration, as shown:

```
/opt/mapr/server/configure.sh -R
```

   Note: See configure.sh for more information about the script.

3. Verify that Drill is configured and running on the node. You can use one of the following methods to verify that the Drillbit service is running on the node:
   - Issue the following command to verify the status of the Drillbit service from the command line:

```
jps
```

   - Log in to the Control System at `https://<host_name>:8443` to verify the status of the Drillbit service.

     Note: You should see the Drillbit listed as a service running on the node.

4. Optionally, modify the Drill configuration. For example, you can change the log file directory, increase heap space and direct memory, or configure the MapR File System as the persistent configuration storage. See Configuring Drill on page 3237.

   Note: You must restart the drillbit for the new configuration to take effect.

5. Repeat steps 1 through 3 on any other nodes designated to run Drill.

   Note:

   You can start|stop|restart the Drillbit service on one or more nodes using the Control System or the following command:

   ```
   $ maprcli node services -name drill-bits -action start|restart|
   stop -nodes <node host names separated by a space>
   ```

   Use the host name if possible. Using host names instead of IP addresses is a best practice.

   You can access the Drill log files in `/opt/mapr/drill/drill-<version>/logs/drillbit.log`.

Install Drill to Run Under YARN

You can install and configure Drill to run under YARN. See Drill-on-YARN Overview. If you are currently running Drill under Warden, back up the directory from your previous Drill installation, and migrate Drill to run under YARN. See Migrate Drill to Run Under YARN.

Note: Drill-on-YARN is an advanced feature used to manage a production Drill cluster. Only skilled Drill and MapR administrators, familiar with YARN, should configure Drill to run under YARN. If you are new to Drill, consider using Drill under Warden until you are familiar with Drill and the Drill cluster.

Note: The MapR default security feature introduced in MapR 6.0 is not supported with Drill-on-YARN.

Verify that your system meets the prerequisites below and then follow the instructions in Installing Drill to Run Under YARN to install the mapr-drill-yarn package on the node designated as the Drill-on-YARN client to run Drill under YARN.

Prerequisites

Verify that your system meets the following prerequisites before you install Drill to run under YARN:

- The MapR cluster is installed and running. Installing Drill first can result in configuration issues
- You have planned the YARN cluster. See YARN on page 738, Planning the Cluster on page 107, and Example Cluster Designs on page 118.
- - ResourceManager is installed on one node in the YARN cluster, and you have calculated disk requirements for the YARN ResourceManager.
  - NodeManager is installed on all nodes in the YARN cluster.
- You have designated one node to act as the Drill-on-YARN client. This is the node on which you install the mapr-drill-yarn package. The Drill-on-YARN client is a command-line program that starts, stops, and monitors the Drill cluster. The client provides the information that YARN needs to start the Application Master.
- Cluster resources can accommodate the Drill memory, CPU, and disk requirements.
- The EEP repository is configured. For instructions, see Step 8: Install Ecosystem Components Manually.

Hive and HBase Support

Installation of a supported version of Hive and HBase is optional. Support differs based on the EEP version that you install. See Component Versions for Released EEPs for version support in each EEP release.

last-updated: Apr 11, 2022

Installing Drill to Run Under YARN

This topic includes instructions for using package managers to download and install Drill.

Verify that the system meets the prerequisites. You must install the mapr-drill-yarn package on a node designated to run as the Drill-on-YARN client.

Note: The MapR default security feature introduced in MapR 6.0 is not supported with Drill-on-YARN.

Complete the following steps as `mapr` using `sudo` to install Drill on the node:

Note: If users need to access SQLLine, you must install the mapr-drill-yarn package on every node used to access SQLLine.

1. To install Drill, issue the command appropriate for your system:

RedHat/CentOS	`yum install mapr-drill-yarn`
Ubuntu	`apt-get install mapr-drill-yarn`
SLES	`zypper install mapr-drill-yarn`

   Note: SLES is supported as of Drill 1.9.0-1703 and Drill 1.10.0-1703.

   Note: Drill does not automatically start after you install the packages.

2. Configure Drill to run under YARN. See Configuring Drill to Run Under YARN.

Configure the OJAI Distributed Query Service

Use these steps to install and configure the OJAI Distributed Query Service on page 505:

1. Using the EEP 4.0 or later repository, install Drill to run under Warden on all data nodes. See Install Drill to Run Under Warden on page 179.
2. Configure the query service by running the following configure.sh script command on the Drill nodes:

```
/opt/mapr/server/configure.sh -R -QS
```

3. Register the query service:

   Note: In the following command `<clustername>` is the name of the cluster as specified in `/opt/mapr/conf/mapr-clusters.conf`.

```
maprcli cluster queryservice setconfig -enabled true -clusterid
<clustername>-drillbits -storageplugin dfs -znode /drill
```

Installing Flume

This topic includes instructions for using package managers to download and install Flume from the EEP repository.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

last-updated: Apr 11, 2022

For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176

Execute the following commands as `root` or using `sudo`.

1. On each planned Flume node, install `mapr-flume`:
   - On Ubuntu: `apt-get install mapr-flume`
   - On CentOS/Redhat: `yum install mapr-flume`
   - On SLES: `zypper install mapr-flume`
2. If you want to integrate Flume with MapR Event Store For Apache Kafka, install the Streams Client on each Flume node.
   - On Ubuntu:

     ```
     apt-get install mapr-kafka
     ```

   - On RedHat/CentOS:

     ```
     yum install mapr-kafka
     ```

   - On SLES:

     ```
     zypper install mapr-kafka
     ```

Installing HBase

This topic includes instructions for using package managers to download and install HBase from the EEP repository.

Before installing HBase, you should plan which cluster nodes should run the HBase Master service, and which nodes should run the HBase RegionServer. At least one node (generally three nodes) should run the HBase Master. For example, install HBase Master on the ZooKeeper nodes. Only a few of the remaining nodes or all of the remaining nodes can run the HBase RegionServer.

The following procedures use the operating-system package managers to download and install from the EEP repository.

Install HBase on a Cluster Node

The following instructions use the package manager to download and install HBase from the EEP repository to a cluster node.

For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

Run the following commands as `root` or using `sudo`.

1. On each planned HBase Master node, install `mapr-hbase-master`:
   - Ubuntu:

     ```
     apt-get install mapr-hbase-master
     ```

   - RedHat/CentOS:

     ```
     yum install mapr-hbase-master
     ```

- SLES:

```
zypper install mapr-hbase-master
```

2. On each planned HBase RegionServer node, install `mapr-hbase-regionserver`:
   - Ubuntu:

```
apt-get install mapr-hbase-regionserver
```

   - RedHat/CentOS:

```
yum install mapr-hbase-regionserver
```

   - SLES:

```
zypper install mapr-hbase-regionserver
```

3. On all HBase nodes, run `configure.sh -R`:

```
/opt/mapr/server/configure.sh -R
```

Installing HBase Client and Tools

MapR 6.0.x does not support HBase as an ecosystem component. Beginning with EEP 6.3.0, MapR 6.1 reintroduced HBase as an ecosystem component. With EEP 6.3.0 and later you can install the HBase Client and tools even if you decide not to install HBase as an ecosystem component. This topic describes the HBase Client and other tools that are available for use with the MapR Database.

Service	Description
HBase Client	After installing the HBase Client, you can use the HBase Shell to manipulate MapR Database tables. MapR Database also supports a number of HBase APIs for use with MapR Database binary tables. For information on installation and configuration, see Installing HBase on a Client Node on page 185. For more information about MapR Database tables and HBase APIs, see Developing Applications for Binary Tables on page 2452.
HBase Thrift Gateway	HBase Thrift Gateway includes an API and a service that accepts Thrift requests to connect to MapR Database tables. For information on installation and configuration, see Installing the HBase Thrift Gateway.
HBase REST Gateway	HBase REST Gateway includes an API and a service that accepts REST requests to connect to MapR Database tables. For information on installation and configuration, see Installing the HBase REST Gateway.
AsyncHBase Libraries	AsyncHBase library provides asynchronous Java APIs to access MapR Database tables. For information on installation and configuration, see Installing AsyncHBase Libraries on page 176.

last-updated: Apr 11, 2022

Installing HBase on a Client Node
The following instructions use the package manager to download and install HBase from the EEP repository to a client node. When you install HBase on a client node, you can use the HBase shell from a machine outside the cluster.

MapR 6.0.x does not support HBase as an ecosystem component. Beginning with EEP 6.3.0, MapR 6.1 reintroduced HBase as an ecosystem component. With EEP 6.3.0 and later you can install the HBase Client and tools even if you decide not to install HBase as an ecosystem component.

Before you being, verify the following prerequisites:

- The EEP repository is set up. For the steps to set up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176
- The MapR client must be installed on the node where you install the HBase client.

  For MapR client setup instructions, see Setting Up the Client.
- You must know the IP addresses or hostnames of the ZooKeeper nodes on the cluster.

Run the following commands as `root` or using `sudo`.

1. On the client computer, install `mapr-hbase`:

CentOS or Red Hat	`yum install mapr-hbase`
Ubuntu	`apt-get install mapr-hbase`
SLES	`zypper install mapr-hbase`

2. On all HBase nodes, run configure.sh with a list of the CLDB nodes and ZooKeeper nodes in the cluster.

Configuring HBase on a Client Node
You can use a script to configure client nodes for use with HBase 1.1.13 or later on a secure or nonsecure MapR cluster.

You configure client nodes as part of a new installation or when you need to upgrade `mapr-hbase` on the client node from a previous HBase version (for example, HBase 1.1.8) to 1.1.13 or later.

Configuration Using the configure_client.sh Script

The `configure.sh` utility does not support the configuration of client nodes for HBase. However, you can use the following script to configure a client by specifying the ZooKeeper host name and port. The script supports secure (MapR-SASL) and nonsecure clusters, but does not support Kerberos (see Manual Configuration on page 185):

```
/opt/mapr/hbase/hbase-1.1.13/bin/configure_client.sh -zkServer <host>:<port>
```

Manual Configuration

To configure a client node manually for use with HBase 1.1.13 or later on a secure or nonsecure MapR cluster, do the following:

last-updated: Apr 11, 2022

Desired Security	Do this . . .	Example

Nonsecure	**1.** Modify the `hbase.zookeeper.quorum` property to change the hostname and add the ZooKeeper port.	`<property>`
		`  <name>hbase.zookeeper.quorum</name>`
		`  <value><hostname>:5181</value>`
		`</property>`

Secure (MapR-SASL)	**1.** Modify the `hbase.zookeeper.quorum` property as shown in the nonsecure example.
	2. Add the properties shown in this example.
		`<property>`
		`  <name>hbase.security.authentication</name>`
		`  <value>maprsasl</value>`
		`</property>`
		`<property>`
		`  <name>hbase.rpc.protection</name>`
		`  <value>privacy</value>`
		`</property>`

Secure (Kerberos)	**1.** Modify the `hbase.zookeeper.quorum` property as shown in the nonsecure example.
	2. Add the properties shown in this example.
		`<property>`
		`  <name>hbase.security.authentication</name>`
		`  <value>kerberos</value>`
		`  </property>`
		`  <property>`
		`  <name>hbase.rpc.protection</name>`
		`  <value>privacy</value>`
		`  </property>`
		`<property>`
		`  <name>hbase.master.kerberos.principal</name>`
		`  <value><username>/<fqdn>@<realm></value>`
		`</property>`
		`<property>`
		`  <name>hbase.regionserver.kerberos.principal</name>`
		`  <value><username>/<fqdn>@<realm></value>`
		`</property>`

Installing the HBase Thrift Gateway

The HBase Thrift Gateway can be installed on any node where the `mapr-client` package or the `mapr-core` package is installed.

Complete the following steps to install the HBase Thrift Gateway:

1. Run the following command to install the HBase Thrift package:

On CentOS / Red Hat	`yum install mapr-hbasethrift`
Ubuntu	`apt-get install mapr-hbasethrift`
SLES	`zypper install mapr-hbasethrift`

2. Run configure.sh on the node where you installed the HBase Thrift package:

```
/opt/mapr/server/configure.sh -R
```

After you install the HBase Thrift package and run `configure.sh -R`, Warden starts and monitors the service. Warden also displays the status of the HBase Thrift service on the MapR Control System user interface.

Installing the HBase REST Gateway

The HBase REST Gateway can be installed on any node where the `mapr-client` package or the `mapr-core` package is installed.

Complete the following steps to install the HBase REST Gateway:

1. Run the following command to install the HBase REST Gateway package:

On CentOS / Red Hat	`yum install mapr-hbase-rest`
Ubuntu	`apt-get install mapr-hbase-rest`
SLES	`zypper install mapr-hbase-rest`

2. Run configure.sh on the node where you installed the HBase REST Gateway package:

```
/opt/mapr/server/configure.sh -R
```

After you install the HBase REST package and run `configure.sh -R`, Warden starts and monitors the service. Warden also displays the status of the HBase REST service on the MapR Control System user interface.

Installing Hive

This topic includes instructions for using package managers to download and install Hive from the EEP repository.

For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176

You can install Hive on a node in the MapR cluster or on a MapR client node. Installation of HiveServer2 on a client node is not supported by MapR. If you wish to install HS2 on a client node, note that one or more required JAR files may **not** be installed during the installation of `mapr-client`. Copy the following JAR file from a resource manager node to the MapR client node:

```
/opt/mapr/hadoop/hadoop-<X.X.X>/share/hadoop/yarn/
hadoop-yarn-server-resourcemanager-<X.X.X>-mapr-<YYYY>.jar
```

Here:

X.X.X	Refers to the version (for example, hadoop-2.7.0)
YYYY	Refers to the release tag of ecosystem component (for example, 1602)

Note: Copying the JAR file may allow you to work with Hive in non-secure mode.

See the Hive Release Notes for a list of fixes and new features.

Hive is distributed as the following packages:

last-updated: Apr 11, 2022

Package	Description
`mapr-hive`	The core Hive package.
`mapr-hiveserver2`	The Hive package that enables HiveServer2 to be managed by the warden, allowing you to start and stop HiveServer2 using maprcli or the MapR Control System. The `mapr-hive` package is a dependency and will be installed if you install `mapr-hiveserver2`. At installation time, Hiveserver2 is started automatically.
`mapr-hivemetastore`	The Hive package that enables the Hive Metastore to be managed by the warden, allowing you to start and stop Hive Metastore using maprcli or the MapR Control System. The `mapr-hive` package is a dependency and will be installed if you install `mapr-hivemetastore`. At installation time, the Hive Metastore is started automatically.
`mapr-hivewebhcat`	The Hive package that enables WebHCat to be managed by the warden, allowing you to start and stop WebHCat using maprcli or the MapR Control System. The `mapr-hive` package is a dependency and will be installed if you install `mapr-hivewebhcat.` At installation time, the WebHCat is started automatically.

Make sure the environment variable `JAVA_HOME` is set correctly. For example:

```
# export JAVA_HOME=/usr/lib/jvm/java-7-sun
```

You can set these system variables by using the shell command line or by updating files such as `/etc/profile` or `~/.bash_profile`. See the Linux documentation for more details about setting system environment variables.

Note: The MapR cluster must be up and running before installing Hive.

On Ubuntu, while configuring the new version of Hive, you could have an issue caused by an incomplete removal of previously installed Hive packages. To avoid this issue, use the `purge` command for complete removal of all previously installed Hive packages.

Execute the following commands as `root` or using `sudo`.

1. On each planned Hive node, install Hive packages.
   - To install Hive:

On CentOS / RedHat	`yum install mapr-hive`
On SLES	`zypper install mapr-hive`
On Ubuntu	`apt-get install mapr-hive`

   - To install Hive and HiveServer2:

On CentOS / RedHat	`yum install mapr-hive mapr-hiveserver2`
On SLES	`zypper install mapr-hive mapr-hiveserver2`
On Ubuntu	`apt-get install mapr-hive mapr-hiveserver2`

last-updated: Apr 11, 2022

- To install Hive, HiveServer2, and HiveMetastore:

On CentOS / RedHat	`yum install mapr-hive mapr-hiveserver2 mapr-hivemetastore`
On SLES	`zypper install mapr-hive mapr-hiveserver2 mapr-hivemetastore`
On Ubuntu	`apt-get install mapr-hive mapr-hiveserver2 mapr-hivemetastore`

- To install Hive, HiveServer2, HiveMetastore and WebHCat:

On CentOS / RedHat	`yum install mapr-hive mapr-hiveserver2 mapr-hivemetastore mapr-hivewebhcat`
On SLES	`zypper install mapr-hive mapr-hiveserver2 mapr-hivemetastore mapr-hivewebhcat`
On Ubuntu	`apt-get install mapr-hive mapr-hiveserver2 mapr-hivemetastore mapr-hivewebhcat`

Note: Starting from EEP-5.0.2 and EEP-6.0.1+, you can use Apache Derby as the underlying database (only for test purposes). To configure Hive on Derby DB, install all Hive packages (`mapr-hive`, `mapr-hiveserver2` `mapr-hivemetastore`, and `mapr-hivewebhcat`), and run the `configure.sh` command, as described in Step 3 later in this procedure.

CAUTION: Do not use `datanucleus.schema.autoCreateAll` for populating underlying databases. For more details, see prohibited usage of `datanucleus.schema.autoCreateAll` property .

2. Configure the database for Hive Metastore.
   See Configuring Database for Hive Metastore on page 3411 for more information.
3. Run `configure.sh on page 2053` with the `-R` option.

```
/opt/mapr/server/configure.sh -R
```

After Hive is installed, the executable is located at: `/opt/mapr/hive/hive-<version>/bin/hive`.

See Hive User Impersonation for the steps to configure user impersonation for Hive and the MapR cluster.

To configure Hive on Tez, see Configuring Hive and Tez on page 3502.

Installing HttpFS
This topic includes instructions for using package managers to download and install HttpFS from the EEP repository.

For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176

Execute the following commands as `root` or using `sudo`.

1. Install the `mapr-httpfs` package:
   - Ubuntu
     ```
     # apt-get install mapr-httpfs
     ```
   - RedHat/CentOS
     ```
     # yum install mapr-httpfs
     ```
   - SLES
     ```
     # zypper install mapr-httpfs
     ```
2. Run `configure.sh` on page 2053 with the `-R` option.
   ```
   /opt/mapr/server/configure.sh -R
   ```

Installing Hue

This topic includes instructions for using package managers to download and install Hue from the EEP repository.

For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

Note: The Hue package, `mapr-hue`, can be installed on either a MapR cluster node (recommended) or a MapR client node. If you choose to install on a client node, keep in mind that Hue directories are owned by the user who installed Hue.

Execute the following commands as `root` or using `sudo`.

1. Install the Hue packages.

   On Ubuntu:
   ```
   apt-get install mapr-hue
   ```
   On RedHat/ CentOS:
   ```
   yum install mapr-hue
   ```
   On SLES:
   ```
   zypper install mapr-hue
   ```
2. If the node is a MapR Client node, follow the additional instructions:

   a) To determine who the <INSTALL_USER> is, enter:
   ```
   logname
   ```

b) Set the following properties in `hue.ini` to that user.

```
server_user=<INSTALL_USER>
server_group=<INSTALL_USER>
default_user=<INSTALL_USER>
```

c) Change the `default_hdfs_superuser` property to the owner of `/var` on the cluster.

Warning: The `<INSTALL_USER>` must exist on the cluster on *all* nodes. It must also be set as the proxy user in *all* configuration files listed in Configure Hue, depending on the Hue version you are installing.

3. Run `configure.sh -R`:

```
/opt/mapr/server/configure.sh -R
```

Installing Other Components

Based on your requirements, you may also want to install the following components on at least one node in the cluster.

Component	Description
MapR Database	Required for access to MapR Database tables
HttpFS	Required for viewing files in MapR File System through Hue file browser
Sqoop2	Required to create and submit jobs to transfer bulk data between structured datastores and the MapR File System.
Spark	Required to process data using the Notebook UI.
Hive	Required to run queries with HiveServer2.
Oozie	Required for Oozie workflows

When you finish installing Hue, go to Configure Hue to learn how to configure Hue.

Installing Impala

This section describes how to use package managers to download and install Impala from the MEP repository.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Impala is comprised of a set of components that you install and run on a single node or on multiple nodes in a cluster. To run Impala in your cluster, you install the required Impala packages on designated nodes. The Impala packages contain the following Impala components:

- Impala daemon
- Impala statestore
- Impala catalog
- Impala binaries

The following table lists the Impala packages and their descriptions:

last-updated: Apr 11, 2022

Package	Description
`mapr-impala`	A package that contains all of the Impala binaries, including the Impala server, impala-shell, statestore, and catalog.
`mapr-impala-server`	The role package that installs the Impala daemon role on the node. This package enables Warden to manage the service. It is recommended (not required) that you install the Impala daemon on a node with the MapR fileserver installed.
`mapr-impala-statestore`	The role package that installs the Impala statestore role on the node. This package enables Warden to manage the service.
`mapr-impala-catalog`	The role package that installs the Impala catalog role on the node. This package enables Warden to manage the service.

Verify that your system meets the prerequisites and then install Impala.

Impala Prerequisites
This section describes software prerequisites for Impala.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

To successfully install and run Impala, verify that the system meets the following hardware and software requirements. Note that Impala must be installed on a node that contains MapR Core. Because of tight coupling with Core, it is not possible to run Impala on a client node.

Prerequisite	Requirement
Operating System	The MapR repository provides packages for the 64-bit operating systems listed in Operating System Support Matrix on page 5522. For restrictions associated with a specific operating system or Impala version, see the support table for your EEP under EEP Components and OS Support on page 5536.
MapR Software	See Component Versions for Released EEPs on page 5586. Also see EEP Support and Lifecycle Status on page 5531.
EEP repository	Set up the EEP repository, if you have not already done so. For instructions on setting up the EEP repository, see Step 2: Prepare Packages and Repositories on page 142.

Hive Metastore	You must install and configure a Hive metastore. Configure the Hive metastore service and connect to a MySQL database through the service. For more information, see Installing Hive.
	Note: Verify that `hive-site.xml` contains the `hive.metastore.uris` setting, and substitute the appropriate host name for `metastore_server_host` on every Impala server node.
	Example
	`<property>`
	`  <name>hive.metastore.uris</name>`
	`  <value>thrift://<metastore_server_host>:9083</value>`
	`</property>`

Hive	Hive 2.3 is required to run Impala 2.12.x. Impala must have access to the same metastore database that Hive uses. For Hive configuration information, see the Hive documentation.
MapR Database	MapR Database binary tables (HBase 1.1.13 API) are required to run Impala. Install `mapr-hbase-1.1.13.0` from the EEP repository.
Java	JDK 1.8 is required. See JDK Support Matrix on page 5596.

OpenSSL	Version 1.0.2 (or newer)

Impala Installation Steps
This topic provides instructions for using package managers to download and install Impala from the EEP repository.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Install the Impala package on nodes in the cluster that you have designated to run Impala. Install the Impala server on every node designated to run impalad. Install the statestore and catalog packages on only one node in the cluster.

Note: It is recommended that you install statestore and catalog together on a separate machine from the Impala server.

Complete the following steps to install Impala, impala-server, statestore, and catalog:

1. Install the `mapr-impala` package on all the nodes designated to run Impala. To install the package, issue the following command:

   ```
   $ sudo yum install mapr-impala
   ```

2. In `/opt/mapr/impala/impala-<version>/conf/env.sh`, complete the following steps:
   a) Verify that the statestore address is set to the address where you plan to run the statestore service.

      ```
      IMPALA_STATE_STORE_HOST=<IP address hosting statestore>
      ```

   b) Change the catalog service address to the address where you plan to run the catalog service.

      ```
      CATALOG_SERVICE_HOST=<IP address hosting catalog service>
      ```

   c) Add the `mem_limit` and `num_threads_per_disk` parameters to `IMPALA_SERVER_ARGS` to allocate a specific amount of memory to Impala, and limit the number of threads that each disk processes per impala server daemon. The default Impala memory setting is high, which can result in conflict between Impala and other frameworks running in the cluster. Adding these parameters can alleviate any potential resource conflicts.

      ```
      export IMPALA_SERVER_ARGS=${IMPALA_SERVER_ARGS:- \
          -log_dir=${IMPALA_LOG_DIR} \
          -state_store_port=${IMPALA_STATE_STORE_PORT} \
          -use_statestore -state_store_host=${IMPALA_STATE_STORE_HOST} \
          -catalog_service_host=${CATALOG_SERVICE_HOST} \
          -be_port=${IMPALA_BACKEND_PORT} \
          -mem_limit=<absolute notation or percentage of physical memory> \
          -num_threads_per_disk=<n>}
      ```

   See Additional Impala Configuration Options for more information about these options and other options that you can modify in `env.sh`.

Warning:

The default maximum heap space allocated to the MapR File System file server should provide enough memory for the MapR File System file server to run concurrently with Impala, however you can modify it if needed. To modify the maximum heap space, navigate to `/opt/mapr/conf/warden.conf`, and change the `service.command.mfs.heapsize.maxpercent` parameter. Issue the following command to restart Warden after you modify the parameter:

```
service mapr-warden restart
```

Refer to warden.conf for more Warden configuration information.

3. Verify that the following property is configured in `hive-site.xml` on all the nodes:

```
<property>
        <name>hive.metastore.uris</name>
        <value>thrift://<metastore_server_host>:9083</value>
</property>
```

4. Install the Impala components.
   a) To install the statestore service, issue the following command:

```
$ sudo yum install mapr-impala-statestore
```

   b) To install the catalog service, issue the following command:

```
$ sudo yum install mapr-impala-catalog
```

   c) To install the Impala server, issue the following command:

```
$ sudo yum install mapr-impala-server
```

5. Run `configure.sh` to refresh the node configuration.

```
/opt/mapr/server/configure.sh -R
```

6. If the Hive metastore has MapR-SASL enabled, copy `$HIVE_HOME/conf/hive-site.xml` to `$IMPALA_HOME/conf/`. Repeat this step any time `hive-site.xml` is modified.

At this point, the Impala servers, catalog, and statestore should be running. For instructions on how to run a simple Impala query and how to query MapR Database tables, refer to Example: Running an Impala SQL Query and Query MapR Database and HBase Tables with Impala.

Installing Kafka Schema Registry (Developer Preview)
This topic includes instructions for using package managers to download and install the Kafka Schema Registry from a MapR repository.

Note: This feature is presented as a *developer preview*. Developer previews are not tested for production environments, and should be used with caution.

Because it is a *developer preview* feature, the Kafka Schema Registry is not provided in the EEP 6.1 repository. You must download the packages from https://package.mapr.com/labs/mapr-schema-registry. Using the MapR Installer to install the Kafka Schema Registry is not currently supported.

The Kafka Schema Registry is a service that provides a RESTful interface for storing and retrieving schemas. The Kafka Schema Registry can be installed on one or several nodes. To install the `schema registry` package on a node, run the following commands as `root` or using `sudo`:

1. Install the schema registry package:

   On Ubuntu:

   ```
   apt-get install mapr-schema-registry
   ```

   On RedHat/ CentOS:

   ```
   yum install mapr-schema-registry
   ```

   On SLES:

   ```
   zypper install mapr-schema-registry
   ```

2. Run `configure.sh -R`:

   ```
   /opt/mapr/server/configure.sh -R
   ```

   Note: Because the Kafka Schema Registry is managed by Warden, you don't have to restart Warden after installing the registry. Warden brings up the service after a few minutes.

To manage and administer the Kafka Schema Registry, see Kafka Schema Registry on page 3941.

Installing KSQL
This topic describes how to use package managers to download and install KSQL from the EEP repository.

Note: You cannot upgrade from KSQL 4.1.1. You must uninstall version 4.1.1 and then install the newer version of KSQL.

Preparing for Installation

KSQL is included in EEP repositories beginning with EEP 6.0.0. For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

The default KSQL configuration parameters are stored in `/opt/mapr/ksql/ksql-<version>/etc/ksql`.

KSQL Operational Modes

To install KSQL, you can use the MapR Installer or the manual steps on this page. KSQL can be used in one of two modes:

Mode	Description
Interactive Mode	This mode is non-secure and allows developers to write KSQL queries interactively using the KSQL CLI.
Non-interactive Mode	This mode is more secure than the Interactive mode and is designed for KSQL query production deployment. Since the queries are known ahead of time, you can run non-interactive queries with more restrictive permissions.

Installation steps are the same for both modes. Run the following commands as `root` or using `sudo`.

last-updated: Apr 11, 2022

Install KSQL in Interactive or Non-interactive Mode

You can install the `mapr-ksql` package on as many or as few nodes as you want. Installing on multiple nodes can increase availability of the service.

On Ubuntu:

```
apt-get install mapr-ksql
```

On RedHat/ CentOS:

```
yum install mapr-ksql
```

On SLES:

```
zypper install mapr-ksql
```

Verify KSQL Installation

To confirm successful installation, you can:

- Check for the presence of the KSQL home folder at `/opt/mapr/ksql/ksql-<version>`.
- Perform a test run:
  1. Start the KSQL server:

     ```
     /opt/mapr/ksql/ksql-[version]/bin/ksql-server-start /opt/mapr/ksql/
     ksql-<version>/etc/ksql/ksql-server.properties
     ```

  2. Verify that KSQL is running by making a call to `http://localhost:8084/info`. For example:

     ```
     curl http://localhost:8084/info
     ```

     The expected response is:

     ```
     {"KsqlServerInfo":{"version":"(version)"}}
     ```

Configure KSQL

To configure KSQL, see KSQL Configuration on page 3846.

Installing Kafka Streams

Kafka Streams is a Java library and is part of the `mapr-kafka` package. Kafka Streams does not require special installation steps; however, you must install the `mapr-core` and `mapr-kafka` packages to use Kafka Streams.

Maven Dependency

To compile a Kafka Streams application, you must add the appropriate Maven dependency. Add a mapr maven repository and the Kafka Streams dependency to your `pom.xml` file to pull in the Maven artifacts.

- For Maven repository information and Kafka Streams dependency versions, see Maven Artifacts for MapR on page 4155
- For more information about Maven artifacts and running a Kafka Streams Java application, see Running a Kafka Streams Java App on page 3856.

Example

The following `pom.xml` example may not correlate with the product versions you are installing.

```
<repository>
    <id>mapr-releases</id>
    <url>https://repository.mapr.com/maven/</url>
</repository>
<dependency>
    <groupId>org.apache.kafka</groupId>
    <artifactId>kafka-streams</artifactId>
    <version>2.1.1.200-mapr-710</version>
</dependency>
<dependency>
    <groupId>org.apache.kafka</groupId>
    <artifactId>kafka-clients</artifactId>
    <version>2.1.1.200-mapr-710</version>
</dependency>
```

Configure Kafka Streams

To configure Kafka Streams, see Kafka Streams Configuration on page 3855.

Installing MapR Event Store For Apache Kafka Clients
This topic includes instructions for using package managers to download and install MapR Event Store For Apache Kafka Clients from the EEP repository.

Note: The MapR Event Store For Apache Kafka Java client is installed with the MapR Client on page 388.

Installing MapR Event Store For Apache Kafka C Client
The MapR Event Store For Apache Kafka C Client is a distribution of librdkafka that works with MapR Event Store For Apache Kafka.

- For instructions on installing the MapR Client, see Installing the MapR Client on page 389
- For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

Installation

As of MapR 6.0.1, the MapR C client is installed as part of the MapR Core installation and the mapr-client package installation. The MapR C client is available on Linux, Mac, and Windows operating systems.

Note: Specific installation is *not* required as of MapR 6.0.1!

For MapR 5.2.1 through MapR 6.0.0, the MapR C client must be installed. The MapR C client is available on Linux and Mac operating systems. As `root` or using `sudo`, install the `mapr-librdkafka` package on nodes where you want to run or build applications.

- On Ubuntu:

  ```
  apt-get install mapr-librdkafka
  ```

- On RedHat/CentOS:

  ```
  yum install mapr-librdkafka
  ```

- On SLES:

```
zypper install mapr-librdkafka
```

- On Mac OS:

1. Download the following TAR file: `https://package.mapr.hpe.com/releases/MEP/<MEP version>/<operating system>/<package>.tar.gz`
2. Extract the TAR file under /opt/mapr:

```
tar -C /opt/mapr/ -zxf <librdkafka_tarFile_location>
```

Note: The `mapr-librdkafka` package pulls in the mapr-client as a dependency if the node does not have the `mapr-client` or `mapr-core` package installed.

Configuration

For MapR 6.0.1 and higher, use the following configuration instructions.

Linux	For Linux installations, add `/opt/mapr/lib` to the end of LD_LIBRARY_PATH. `export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/mapr/lib`
Mac	For Mac installations, add `/opt/mapr/lib` to the end of DYLD_LIBRARY_PATH. `export DYLD_LIBRARY_PATH=$DYLD_LIBRARY_PATH:/opt/mapr/lib`
Windows	For Windows installations, no additional configuration is required. Link your application and run your programs against the MapR Data Platform Client dynampic link libraries (dll) located at: `C:\opt\mapr\lib`. The corresponding librdkafka header is `C:\opt\mapr\include\librdkafka`.

Attention: For MapR 6.0.0 and earlier, see Configuring the MapR Event Store For Apache Kafka C Client on page 2796 for instructions on configuring the client.

Installing MapR Event Store For Apache Kafka Python Client
The MapR Event Store For Apache Kafka Python Client is a binding for librdkafka that is dependent on the MapR Event Store For Apache Kafka C client (MapR Event Store For Apache Kafka C Client is a distribution of librdkafka that works with MapR Event Store For Apache Kafka).

Prerequisites

Note: As of MapR 5.2.1, you can create Python client applications for MapR Event Store For Apache Kafka.

Verify that the following components are installed on the node:

- MapR Event Store For Apache Kafka C Client (mapr-librdkafka)

last-updated: Apr 11, 2022

- GNU Compiler Collection (GCC) is installed on the node.
- Python version 2.7.1 and above, up to Python version 3.6.x.
- Python pip
- python-devel (This is required for nodes with the Linux operating system.)

Note: For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

Important: Because the MapR Event Store For Apache Kafka Python Client is dependent on the MapR Event Store For Apache Kafka C Client, you must configure the MapR Event Store For Apache Kafka C Client before using the MapR Event Store For Apache Kafka Python Client. See Configuring the MapR Event Store For Apache Kafka C Client on page 2796.

Installation

Note: The Python client is available for Linux or Mac operating systems.

To install the MapR Event Store For Apache Kafka Python Client using the Python Software Foundation, run the following command as `root` or using `sudo`:

- On Linux:

```
pip
install --global-option=build_ext --global-option="--library-dirs=/opt/
mapr/lib" --global-option="--include-dirs=/opt/mapr/include/"
mapr-streams-python
```

- On Mac:

```
pip
install --user --global-option=build_ext --global-option="--library-dirs=/
opt/mapr/lib" --global-option="--include-dirs=/opt/mapr/include/"
mapr-streams-python
```

Note: The referenced package works on nodes with the Linux or the Mac operating system. The Python Client for MapR Event Store For Apache Kafka is *not* supported on Windows.

Alternatively, you can install the MapR Event Store For Apache Kafka Python Client via the MapR package repository:

```
https://package.mapr.hpe.com/releases/MEP/<MEP version>/mac/
mapr-streams-python-<version>.tar.gz
```

Troubleshooting Mac OS Installation

If you install the MapR Event Store For Apache Kafka Python Client on a Mac without the `--user` flag, you may encounter a "Not Permitted" error, signifying that you don't have permission to create a LICENSE file. The error will look similar to the following:

```
Copying LICENSE -> /System/Library/Frameworks/Python.framework/Versions/2.7/
                error: [Error 1] Operation not permitted:
                '/System/Library/Frameworks/Python.framework/Versions/2.7/
LICENSE'
```

To fix this issue, execute the following steps. These steps apply to users with a new Mac OS Sierra version 10.12.5.

1. Reboot your Mac while simultaneously holding the **Command** and **R** keys to go into Mac OS X Recovery mode.
2. Select **Utilities**, and then **Terminal**.
3. Type `csrutil disable`. A message will pop up, informing you that your System Integrity Protection (SIP) has been sucessfully disabled.
4. Reboot your Mac again.
5. Go into the Terminal and execute the following command as the `root` user:

```
pip
install --global-option=build_ext  --global-option="--library-dirs=/opt/
mapr/lib" --global-option="--include-dirs=/opt/mapr/include/"
mapr-streams-python
```

6. Once the script runs successfully, reboot your Mac while holding the **Command** and **R** keys. You will go into Mac OS X Recovery mode.
7. Select **Utilities**, and then **Terminal**.
8. Type `csruntil enable` to enable your SIP.

Note: To avoid this error, consider using a virtual Python environment or the `--user` flag.

Installing MapR Event Store For Apache Kafka C#/.NET Client
The MapR Event Store For Apache Kafka C#/.NET client is a binding for librdkafka that is dependent on the MapR Event Store For Apache Kafka C client (MapR Event Store For Apache Kafka C Client is a distribution of librdkafka that works with MapR Event Store For Apache Kafka).

Note: As of MapR 6.0.1/ EEP 5.0, you can create C#/.NET client applications for MapR Event Store For Apache Kafka.

Requirements

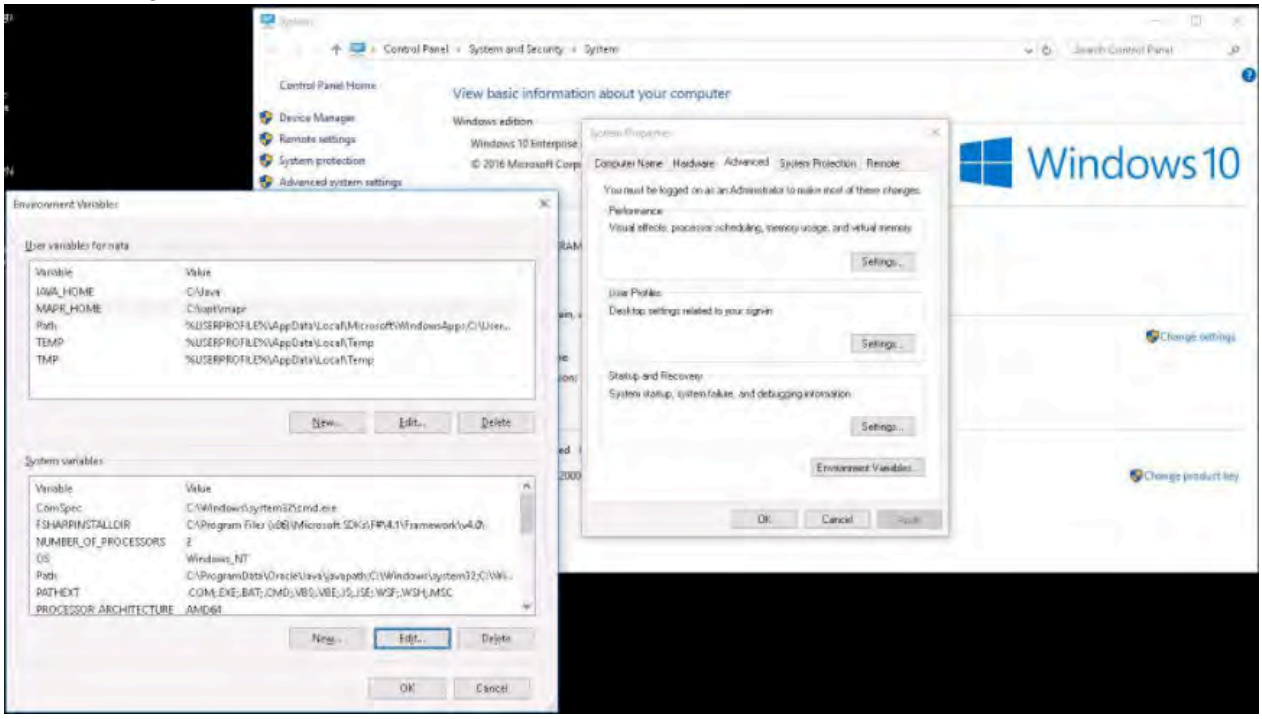
Verify that the following components are installed on the node:

- MapR Client on Windows 7 (or higher) x64 operating systems
- MapR cluster version 6.0.1 or greater
- Java 8 SDK and set Java HOME
- MapR Event Store For Apache Kafka C Client (mapr-librdkafka 0.11.3)
- MapR Event Store For Apache Kafka C#/.NET Client (mapr-streams-dotnet)
- .NET SDK 4.5.x or 4.6.x or .NET Core SDK 1.1
- nuget.exe

For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

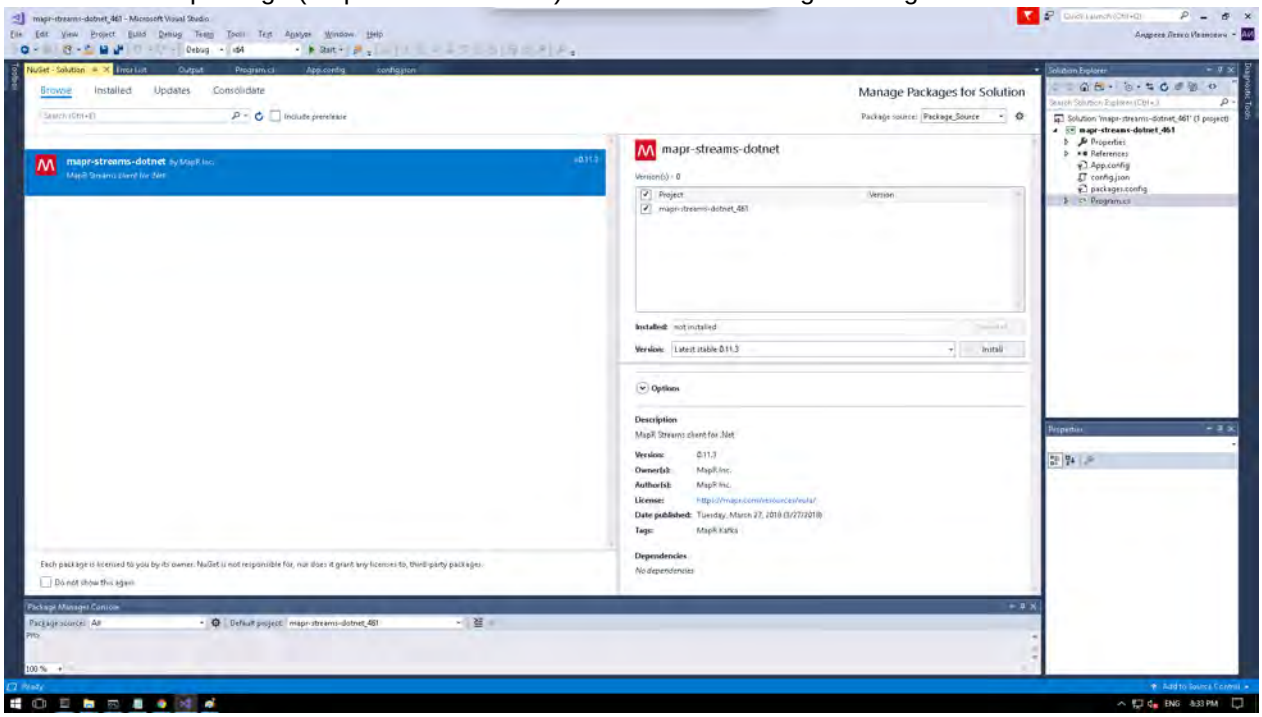
Important: Because the MapR Event Store For Apache Kafka C#/.NET Client is dependent on the MapR Event Store For Apache Kafka C Client, you must configure the MapR Event Store For Apache Kafka C Client before using the MapR Event Store For Apache Kafka C#/.NET Client. See Configuring the MapR Event Store For Apache Kafka C Client on page 2796.

The following screenshot shows the Environment Variables on Windows:



Installing on Windows

To install from the Visual Studio, search for the MapR Event Store For Apache Kafka C#/.NET package (mapr-streams-dotnet) in the NuGet Package Manager UI.



To install from PowerShell:

1. Run the following command in the Package Manager Console:

```
Install-Package mapr-streams-dotnet -<version>
```

To add the package initial in .NET Core:

1. Create the application, for example: `dotnet new console`
2. Add the C#/.NET Client package, for example: `dotnet add package mapr-streams-dotnet`
3. Add a dependency in your **.csproj** file:

```
<ItemGroup>
<PackageReference Include="mapr-streams-dotnet" Version="<version
number>" />
</ItemGroup>
```

Installing MapR Event Store For Apache Kafka Tools
This topic includes instructions for using package managers to download and install MapR Event Store For Apache Kafka Tools from the EEP repository.

Note: For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176

To install manually, first run the following commands as `root` or using `sudo`.

- On RedHat/Centos:

```
yum install <package_name>
```

- On Ubuntu:

```
apt-get install <package_name>
```

- On SLES:

```
zypper install <package_name>
```

After you install the Kafka tools, configure the Kafka components by running `configure.sh -R` on each node where you installed the Kafka tools:

```
/opt/mapr/server/configure.sh -R
```

Table

Package	Description
mapr-kafka	The core Kafka package. Required for MapR Event Store For Apache Kafka and other Kafka components for MapR Event Store For Apache Kafka.
mapr-ksql	The KSQL for MapR Event Store For Apache Kafka package.
mapr-schema-registry	The Schema Registry for MapR Event Store For Apache Kafka package.

last-updated: Apr 11, 2022

Table (Continued)

Package	Description
mapr-kafka-rest	The Kafka REST Proxy for MapR Event Store For Apache Kafka package.
mapr-kafka-connect-jdbc	The JDBC connect package for Kafka Connect for MapR Event Store For Apache Kafka.
mapr-kafka-connect-hdfs	The HDFS connect package for Kafka Connect for MapR Event Store For Apache Kafka.

Kafka REST Proxy for MapR Event Store For Apache Kafka

The following packages are required for Kafka REST Proxy for MapR Event Store For Apache Kafka:

- mapr-kafka
- mapr-kafka-rest
- mapr-client

Note: Before manually installing, verify that the **/opt/mapr/conf/daemon.conf** file exists and contains the mapr user and group.

Note: After installation, the Warden process automatically detects the configuration and starts the Kafka REST Proxy for MapR Event Store For Apache Kafka service on port 8082. This service is viewable on the Control System.

Note: The Kafka REST Proxy for MapR Event Store For Apache Kafka service can be run on multiple cluster nodes simultaneously.

Kafka Connect for MapR Event Store For Apache Kafka

The following packages are required for Kafka Connect for MapR Event Store For Apache Kafka:

- mapr-kafka
- mapr-kafka-connect-jdbc
- mapr-kafka-connect-hdfs

Note: The Kafka Connect for MapR Event Store For Apache Kafka service can be run on multiple cluster nodes simultaneously.

Installing MapR Data Access Gateway

This topic includes instructions for using package managers to download and install the MapR Data Access Gateway from the EEP repository.

For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

The MapR Data Access Gateway is included in EEP repositories beginning with EEP 5.0.0. The MapR Data Access Gateway is a service that acts as a proxy and gateway for translating requests between lightweight client applications and the MapR cluster. For EEP 6.0.0, the MapR Database JSON REST API and Python OJAI client use this service.

In the MapR Installer, the gateway is not visible as a unique service but is installed when the MapR DataBase service is selected and can also be installed manually using this procedure. The gateway should be installed on at least two nodes, if possible, but not on CLDB or Zookeeper nodes. It is recommended to install the service on the same node as the gateway server.

last-updated: Apr 11, 2022

Run the following commands as `root` or using `sudo`.

1. Install the MapR Data Access Gateway package:

   On Ubuntu:

   ```
   apt-get install mapr-data-access-gateway
   ```

   On RedHat/ CentOS:

   ```
   yum install mapr-data-access-gateway
   ```

   On SLES:

   ```
   zypper install mapr-data-access-gateway
   ```

2. Run `configure.sh -R`:

   ```
   /opt/mapr/server/configure.sh -R
   ```

   Note: As the MapR Data Access Gateway is managed by Warden, you don't have to restart Warden after installing the gateway. Warden brings up the service after a few minutes.

To manage and administer the gateway, see Administering the MapR Data Access Gateway on page 1492.

To learn more about the gateway, see Understanding the MapR Data Access Gateway on page 750.

Installing Livy

This topic includes instructions for using package managers to download and install Livy from the EEP repository.

For instructions to set up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

Run the following commands as `root` or using `sudo`:

1. Install the mapr-livy package:

   RedHat/CentOS

   ```
   # yum install mapr-livy
   ```

   SLES

   ```
   # zypper install mapr-livy
   ```

   Ubuntu

   ```
   # apt-get install mapr-livy
   ```

2. Run the configure.sh script with the following command to configure the Livy role for the node:

   ```
   /opt/mapr/server/configure.sh -R
   ```

Installing S3 Gateway

The S3 Gateway is an object storage server that can be used with the Amazon S3 cloud storage service.

Notice: The S3 gateway is included in EEP 6.0.0 - EEP 8.0.0 repositories.

This topic provides instructions for manually installing the S3 gateway version 1.0.x on an Edge node or on a server node (for example, a MapR cluster node) that is running MapR software.

The S3 gateway has two packages:

Package Name	Description
`mapr-objectstore-client`	The binary package.
`mapr-objectstore-gateway`	The role package that includes Warden configuration information.

Note: You cannot install the S3 gateway with the MapR Installer.

Requirements

The S3 gateway has the following requirements:

- A working MapR cluster. Connecting to a MapR cluster requires either `mapr-core` or `mapr-client`, depending on the node type. A server node must have `mapr-core` installed. An Edge node must have `mapr-client` installed.
- For Ubuntu only, install the `syslinux-utils` package:

```
sudo apt install syslinux-utils
```

- For Zypper only, install the `libcap-progs` package:

```
sudo zypper install libcap-progs
```

- A POSIX client must be installed and configured on each node where the S3 gateway will be used.
- JDK version 8 or higher.

Installing and Configuring a POSIX client for the S3 gateway

Note: The following instructions install the POSIX client (for a basic license) and JDK 8. If JDK 8 is already installed, omit it when you run the installation command. For more information about POSIX, see Installing FUSE-based POSIX Client Packages on page 400.

1. To install the POSIX client and JDK 8, run the command appropriate for your system:
   - Ubuntu

     ```
     sudo apt install openjdk-8-jdk-headless mapr-posix-client-basic
     ```

   - RedHat/CentOS

     ```
     sudo yum install java-1.8.0-openjdk.x86_64 mapr-posix-client-basic
     ```

   - SLES

     ```
     sudo zypper install java-1.8.0-openjdk.x86_64 mapr-posix-client-basic
     ```

2. In the `/opt/mapr/conf/fuse.conf` file, set the mount directory and the path to the user ticket.

last-updated: Apr 11, 2022

- `fuse.mount.point` - Use this parameter to set the mount directory. The default setting is `/mapr`.
- `fuse.ticketfile.location` – Use this parameter to set the path to the user ticket.

3. Create a mount directory and set permissions, as shown:

```
mkdir -m 777 /mapr
```

4. Start the MapR Posix service:

```
sudo systemctl start
mapr-posix-client-basic.service
```

5. Verify that the MapR cluster is mounted to the `/mapr` directory:

```
ls -la /mapr
```

You should see a directory with the MapR cluster name and cluster data inside. By default, the directory is `'my.cluster.com'` unless you define a different name for the cluster, for example:

```
ls -la /mapr/
total 1
drwxr-xr-x 7 mapr mapr 6 Feb 16 18:06 example.mapr.cluster
```

Edge Node Installation
An Edge node must have the `mapr-client` installed before you install the S3 gateway.

Run commands as `root` or using `sudo`.

To install the S3 gateway on an Edge node, complete the following steps:

1. Configure the repositories for MapR core and EEP components:
   - Ubuntu

     ```
     sudo add-apt-repository 'deb
     <link-to-release-repository>
     binary trusty' -y
     sudo add-apt-repository 'deb
     <link-to-mep-repository> binary
     trusty' -y
     ```

   - RedHat/CentOS

     See Adding the MapR Repository on RHEL, CentOS, or Oracle Linux on page 143.

   - SLES

     See Adding the MapR Repository on SUSE on page 144.

2. Update the repositories on the Edge node:
   - Ubuntu

     ```
     sudo apt-get update
     ```

- RedHat/CentOS

```
sudo yum clean all
sudo yum makecache --refresh
```

- SLES

```
sudo zypper ref
```

3. To install the S3 gateway, run the command appropriate for your system:
   - Ubuntu

```
sudo apt install
mapr-objectstore-client
```

   - RedHat/CentOS

```
sudo yum install
mapr-objectstore-client
```

   - SLES

```
sudo zypper install
mapr-objectstore-client
```

4. Run the `/opt/mapr/objectstore-client/objectstore-client-x.x.x/bin/configure.sh` script to configure the S3 gateway:

```
sudo /opt/mapr/objectstore-client/
objectstore-client-<version>/bin/
configure.sh -c -u
<userName> -g <groupName> --path /
path/to/filesystem/

#In this command:
-c is the client node with the
objectstore-client.
-u(--user) is the user
admin for working with the
objectstore-client.
-g(--group) is the group.
-p(--path) is the path for
mounting the filesystem.
```

5. To start the S3 gateway, run the `objectstore.sh` script:

```
sudo /opt/mapr/objectstore-client/
objectstore-client-<version>/bin/
objectstore.sh start
```

6. Verify that the S3 gateway process is running:

```
sudo /opt/mapr/objectstore-client/
objectstore-client-<version>/bin/
objectstore.sh status
```

When the S3 gateway is running, the system returns: `Minio is running`

7. Configure the S3 gateway. See Configuring S3 Gateway on page 3961 for instructions.

To delete the S3 gateway on an Edge node, complete the following steps:

1. Stop the S3 gateway service:

```
sudo /opt/mapr/objectstore-client/
objectstore-client-<version>/bin/
objectstore.sh stop
```

2. Delete the packages from the system:
   - Ubuntu

```
sudo apt remove
mapr-objectstore-client --purge
```

   - RedHat/CentOS

```
sudo yum remove
mapr-objectstore-client
```

   - SLES

```
sudo zypper remove
mapr-objectstore-client
```

Server Node Installation

A server in this context refers to a node in a MapR cluster. A server node must have `mapr-core` installed before you can install the S3 gateway.

Run commands as a user with admin privileges on the cluster.

To install the S3 gateway on a server, complete the following steps:

1. Verify that the following requirements are met:
   - MapR software is installed on the server, and the EEP 6.x repositories are configured. For more information about configuring repositories, see Step 2: Prepare Packages and Repositories on page 142.
   - The MapR POSIX client is installed.
   - For Ubuntu clusters, you must install `syslinux-utils`:

```
sudo apt install syslinux-utils
```

- To support impersonation in SLES clusters, you must install the `libcap-progs` library:

```
sudo zypper install libcap-progs
```

2. Update the repositories:
   - Ubuntu

```
sudo apt-get update
```

   - RedHat/CentOS

```
sudo yum clean all
sudo yum makecache --refresh
```

   - SLES

```
sudo zypper ref
```

3. To install the S3 gateway, run the command appropriate for your system:
   - Ubuntu

```
sudo apt install
mapr-objectstore-gateway
mapr-objectstore-client
```

   - RedHat/CentOS

```
sudo yum install
mapr-objectstore-gateway
mapr-objectstore-client
```

   - SLES

```
sudo zypper install
mapr-objectstore-gateway
mapr-objectstore-client
```

4. To configure Warden, use `configure.sh` on the first run:

```
sudo /opt/mapr/server/
configure.sh -R
```

5. Verify that the S3 gateway process is running:

```
sudo /opt/mapr/objectstore-client/
objectstore-client-<version>/bin/
objectstore.sh status
```

When the S3 gateway is running, the system returns: `Minio is running`

6. Configure the S3 gateway. See Configuring S3 Gateway on page 3961 for instructions.

To delete the S3 gateway on a Server node, complete the following steps:

1. Stop the objectstore service:

```
/opt/mapr/bin/maprcli node
services -name objectstore -nodes
<node_name> -action stop
```

2. Delete the packages from the system:
   - Ubuntu

```
sudo apt remove
mapr-objectstore-gateway --purge

sudo apt remove
mapr-objectstore-client --purge
```

   - RedHat/CentOS

```
sudo yum remove
mapr-objectstore-gateway
sudo yum remove
mapr-objectstore-client
```

   - SLES

```
sudo zypper remove
mapr-objectstore-gateway
sudo zypper remove
mapr-objectstore-client
```

For More Information

To use the S3 gateway, see these topics:

- S3 Gateway on page 3959
- Configuring S3 Gateway on page 3961
- AWS CLI

For more information about S3 commands, see the Amazon S3 Documentation.

Uninstalling the S3 Gateway
Describes how to stop the Object Store service and remove the packages for an edge node or server node.

Uninstalling from an Edge Node

To uninstall the S3 gateway from an edge node, you must stop the service and then delete the package from the node:

1. Stop the S3 gateway service:

```
sudo /opt/mapr/objectstore-client/objectstore-client-<version>/bin/
objectstore.sh stop
```

2. Delete the packages from the system:
   - Ubuntu

```
sudo apt remove mapr-objectstore-client --purge
```

   - RedHat/CentOS

```
sudo yum remove mapr-objectstore-client
```

   - SLES

```
sudo zypper remove mapr-objectstore-client
```

Uninstalling from a Server Node

To uninstall the S3 gateway from a server node, you must stop the service and then delete the packages from the node:

1. Stop the S3 gateway service:

```
/opt/mapr/bin/maprcli node services -name objectstore -nodes
node_name -action stop
```

2. Delete the packages from the system:
   - Ubuntu

```
sudo apt remove mapr-objectstore-gateway --purge
sudo apt remove mapr-objectstore-client --purge
```

   - RedHat/CentOS

```
sudo yum remove mapr-objectstore-gateway
sudo yum remove mapr-objectstore-client
```

   - SLES

```
sudo zypper remove mapr-objectstore-gateway
sudo zypper remove mapr-objectstore-client
```

Installing Myriad

Describes Myriad requirements and installation.

last-updated: Apr 11, 2022

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following table specifies the Myriad installation requirements.

Table

Requirement	Description
Operating Systems	• RedHat/CentOS 6.2, 6.6, and 7.0 • Ubuntu 14.04
Myriad Requirements	• JDK 7 or 8
MapR Minimum Requirements	• MapR Core • MapR Zookeeper • MapR CLDB • MapR FileServer • MapR Webserver

The following table specifies the ports used by Myriad.

Table

Application	Port
Mesos Master UI	http://<hostname>:5050
Marathon UI	http://<hostname>:8080
Myriad UI	http://<hostname>:8192 **Note:** The Myriad UI is available once a Resource Manager is launched.

Note: If your environment has both Marathon and Spark installed on the same node, a conflict occurs because the default port for both is 8080. To resolve this conflict, change the port for one of the applications.

The Myriad installation is a manual installation where Apache Mesos is installed before MapR Myriad.

1. Download and build Mesos and (optionally) Marathon. See http://mesos.apache.org/getting-started/ for Mesos and https://mesosphere.github.io/marathon/docs/ for Marathon.

   Important: Install Apache Mesos on all node where resources will be shared. In addition, at least one (1) node should be running both Primary and Secondary installations and one (1) node running only Secondary installation.

2. Download and deploy MapR Myriad.
   a) Download the Myriad 0.2 tarball. See for more information.

   b) Myriad 0.2-1710 Release Notes on page 6265Extract Myriad from the tarball.

   ```
   tar -zxf mapr-myriad-0.2.tar.gz
   ```


last-updated: Apr 11, 2022

c) Deploy Myriad and configuration files.

```
mkdir -p /opt/mapr/hadoop/hadoop-{HADOOP_VERSION}/share/hadoop/
myriad/lib
ln -sf {PROJECT_HOME}/libs/* {HADOOP_HOME}/share/hadoop/myriad/lib
ln -sf {PROJECT_HOME}/conf/* {HADOOP_HOME}/etc/hadoop/
```

3. On the Mesos primary node, install (as a minimum) Apache Mesos and the following MapR packages:

```
mapr-core
mapr-fileserver
mapr-cldb
mapr-webserver
```

On RedHat / CentOS	`yum -nogpgcheck install <package> <package> ...`
On Ubuntu	`apt-get install <package> <package> ...`

4. Install MapR Zookeeper on its primary node:

   The MapR `mapr-zookeeper` package can be installed on any node that is designated as its primary node. For example, on the Mesos master node:

```
mapr-zookeeper
```

5. (optional) Install Mesos-DNS on its master node:

   Mesos-DNS can be installed on any node that is designated as its master. For example, on the Mesos master node. See https://github.com/mesosphere/mesos-dns/tree/master/docs.

6. On the Mesos secondary nodes, install (as a minimum) Apache Mesos and the following MapR packages.

```
mapr-core
mapr-fileserver
```

Small Cluster Installation

The following example shows an installation on a CentOS6 operating system, a three (3) node cluster where one node is the master node for Mesos, Mesos-DNS, MaprR CLDB, and MapR Zookeeper, and two nodes are Mesos secondary nodes.

Download and build Mesos on all nodes.

Unpack and install Myriad from the tarball.

On the master node, install (as a minimum) Apache Mesos and the following MapR packages:

```
yum -nogpgcheck install
mapr-core mapr-fileserver mapr-zookeeper mapr-cldb mapr-webserver
```


last-updated: Apr 11, 2022

On the two Mesos secondary nodes, install (as a minimum) Apache Mesos and the following MapR packages:

```
yum -nogpgcheck install
mapr-core mapr-fileserver
```

See Configure Myriad.

Installing Oozie
This topic includes instructions for using package managers to download and install Oozie from the EEP repository.

For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

The Oozie client/server architecture requires you to install two packages, `mapr-oozie` and `mapr-oozie-internal`, on the client node and the server node. `mapr-oozie` is dependent on `mapr-oozie-internal. mapr-oozie-internal` is automatically installed by the package manager when you install `mapr-oozie`.

Execute the following commands as `root` or using `sudo` on a MapR cluster:

1. Install `mapr-oozie` and `mapr-oozie-internal`:

   RedHat/CentOS

   ```
   yum install mapr-oozie mapr-oozie-internal
   ```

   SLES

   ```
   zypper install mapr-oozie mapr-oozie-internal
   ```

   Ubuntu

   ```
   apt-get install mapr-oozie mapr-oozie-internal
   ```

2. For non-secure clusters, add the following two properties to `core-site.xml` located in `/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop/core-site.xml`:

   ```
   <property>
     <name>hadoop.proxyuser.mapr.hosts</name>
     <value>*</value>
   </property>
   <property>
     <name>hadoop.proxyuser.mapr.groups</name>
     <value>*</value>
   </property>
   ```

3. Restart the YARN services. For YARN mode, restart NodeManager and ResourceManager:

   ```
   maprcli node services -name nodemanager -action restart -nodes <space
   delimited list of nodes>
   maprcli node services -name resourcemanager -action restart -nodes
   <space delimited list of nodes>
   ```

4. Run `configure.sh -R`:

```
/opt/mapr/server/configure.sh -R
```

5. On client or edge nodes installed with EEP 6.3.1 or EEP 7.0.0 or later, run the following `configure.sh` command:

   For secure clusters, use:

```
/opt/mapr/oozie/oozie-5.1.0/bin/configure.sh -R -c --secure
```

   For non-secure clusters, use:

```
/opt/mapr/oozie/oozie-5.1.0/bin/configure.sh -R -c --unsecure
```

6. Export the Oozie URL to your environment:

   For secure clusters, use the following `export` command, and specify the `oozie_port_number` as `11443`:

```
export OOZIE_URL='<https://<Oozie_node>:<oozie_port_number>/oozie>'
```

   For non-secure clusters, use the following `export` command, and specify the `oozie_port_number` as `11000`:

```
export OOZIE_URL='<http://<Oozie_node>:<oozie_port_number>/oozie>'
```

7. Check the Oozie status with the following command:

```
/opt/mapr/oozie/oozie-<version>/bin/oozie admin -status
```

8. If high availability for the Resource Manager is configured, edit the `job.properties` file for each workflow and insert the following statement

```
JobTracker=maprfs:///
```

9. If high availability for the Resource Manager is not configured, provide the address of the node running the active ResourceManager and the port used for ResourceManager client RPCs (port 8032). For each workflow, edit the `job.properties` file and insert the following statement:

```
JobTracker=<ResourceManager_address>:8032
```

10. Restart Oozie:

```
maprcli node services -name oozie -action restart -nodes <space
delimited list of nodes>
```

Note: If high availability for the Resource Manager is not configured and the ResourceManager fails, you must update the `job.properties` with the active ResourceManager.

Installing Pig

This topic includes instructions for using package managers to download and install Pig from the EEP repository.

last-updated: Apr 11, 2022

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

On a MapR cluster, execute the following commands as `root` or using `sudo`:

1. On each planned Pig node, install `mapr-pig`:

Ubuntu	`apt-get install mapr-pig`
RedHat/ CentOS	`yum install mapr-pig`
SLES	`zypper install mapr-pig`

2. Make sure the `JAVA_HOME` environment variable is set correctly. For example:

```
# export JAVA_HOME=/usr/lib/jvm/java-<version>-sun
```

Installing Sentry

This topic includes instructions for using package managers to download and install Sentry from the EEP repository.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Set up the EEP repository, if you have not done so. For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

Install Sentry as `root` or using `sudo`.

On the nodes designated to run Sentry, install `mapr-sentry`:

- On Ubuntu:

```
apt-get install mapr-sentry
```

- On RedHat/CentOS:

```
yum install mapr-sentry
```

- On SLES:

```
zypper install mapr-sentry
```

Installing Spark Standalone

This topic describes how to use package managers to download and install Spark Standalone from the EEP repository.

To set up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

Spark is distributed as four separate packages:

Package	Description
`mapr-spark`	Install this package on any node where you want to install Spark. This package is dependent on the `mapr-client` package.
`mapr-spark-master`	Install this package on Spark master nodes. Spark master nodes must be able to communicate with Spark worker nodes over SSH without using passwords. This package is dependent on the `mapr-spark` and the `mapr-core` packages.
`mapr-spark-historyserver`	Install this optional package on Spark History Server nodes. This package is dependent on the `mapr-spark` and `mapr-core` packages.
`mapr-spark-thriftserver`	Install this optional package on Spark Thrift Server nodes. This package is available starting in the EEP 4.0 release. It is dependent on the `mapr-spark` and `mapr-core` packages.

Run the following commands as `root` or using `sudo`.

1. Create the `/apps/spark` directory on the cluster filesystem, and set the correct permissions on the directory.

```
hadoop fs -mkdir /apps/spark
hadoop fs -chmod 777 /apps/spark
```

Note: Beginning with EEP 6.2.0, the `configure.sh` script creates the `/apps/spark` directory automatically.

2. Install Spark using the appropriate commands for your operating system:

On CentOS / Red Hat

```
yum install
mapr-spark mapr-spark-master
mapr-spark-historyserver
mapr-spark-thriftserver
```

On Ubuntu

```
apt-get install
mapr-spark mapr-spark-master
mapr-spark-historyserver
mapr-spark-thriftserver
```

On SLES

```
zypper install
mapr-spark mapr-spark-master
mapr-spark-historyserver
mapr-spark-thriftserver
```

Note: The `mapr-spark-historyserver`, `mapr-spark-master`, and `mapr-spark-thriftserver` packages are optional.

Spark is installed into the `/opt/mapr/spark` directory.

last-updated: Apr 11, 2022

3. For Spark 2.x:

   Copy the `/opt/mapr/spark/spark-<version>/conf/slaves.template` into `/opt/mapr/spark/spark-<version>/conf/slaves`, and add the hostnames of the Spark worker nodes. Put one worker node hostname on each line.

   For example:

   ```
   localhost
   worker-node-1
   worker-node-2
   ```

4. Set up passwordless ssh for the `mapr` user such that the Spark master node has access to all secondary nodes defined in the `conf/slaves` file for Spark 2.x .

5. As the `mapr` user, start the worker nodes by running the following command in the master node. Since the Master daemon is managed by the Warden daemon, do not use the `start-all.sh` or `stop-all.sh` command.

   For Spark 2.x:

   ```
   /opt/mapr/spark/spark-<version>/sbin/start-slaves.sh
   ```

6. If you want to integrate Spark with MapR Event Store For Apache Kafka, install the Streams Client on each Spark node:

   - On Ubuntu:

     ```
     apt-get install mapr-kafka
     ```

   - On RedHat/CentOS:

     ```
     yum install mapr-kafka
     ```

7. If you want to use a Streaming Producer, add the `spark-streaming-kafka-producer_2.11.jar` from the MapR Data Platform Maven repository to the Spark classpath (`/opt/mapr/spark/spark-<versions>/jars/`).

8. After installing Spark Standalone but before running your Spark jobs, follow the steps outlined at Configuring Spark Standalone on page 4030.

Installing Spark on YARN

This topic describes how to use package managers to download and install Spark on YARN from the EEP repository.

To set up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

Spark is distributed as three separate packages:

Package	Description
`mapr-spark`	Install this package on each node where you want to install Spark. This package is dependent on the `mapr-client` package.
`mapr-spark-historyserver`	Install this optional package on Spark History Server nodes. This package is dependent on the `mapr-spark` and `mapr-core` packages.

Package	Description
`mapr-spark-thriftserver`	Install this optional package on Spark Thrift Server nodes. This package is available starting in the EEP 4.0 release. It is dependent on the `mapr-spark` and `mapr-core` packages.

To install Spark on YARN (Hadoop 2), execute the following commands as `root` or using `sudo`:

1. Verify that JDK 1.7 or later is installed on the node where you want to install Spark.
2. Create the `/apps/spark` directory on the cluster filesystem, and set the correct permissions on the directory:

   ```
   hadoop fs -mkdir /apps/spark
   hadoop fs -chmod 777 /apps/spark
   ```

   Note: Beginning with EEP 6.2.0, the `configure.sh` script creates the `/apps/spark` directory automatically when using the Installer. However, you must manually create this directory when performing a manual installation.
3. Install the packages:

   On Ubuntu

   ```
   apt-get install
   mapr-spark mapr-spark-historyserver
   mapr-spark-thriftserver
   ```

   On CentOS / Red Hat

   ```
   yum install mapr-spark
   mapr-spark-historyserver
   mapr-spark-thriftserver
   ```

   On SLES

   ```
   zypper install
   mapr-spark mapr-spark-historyserver
   mapr-spark-thriftserver
   ```

   Note: The `mapr-spark-historyserver` and `mapr-spark-thriftserver` packages are optional.
4. If you want to integrate Spark with MapR Event Store For Apache Kafka, install the Streams Client on each Spark node:
   - **On Ubuntu:**

     ```
     apt-get install mapr-kafka
     ```

   - **On CentOS / Red Hat**:

     ```
     yum install mapr-kafka
     ```
5. If you want to use a Streaming Producer, add the `spark-streaming-kafka-producer_2.11.jar` from the MapR Maven repository to the Spark classpath (`/opt/mapr/spark/spark-<versions>/jars/`).

   For repository-specific information, see Maven Artifacts for MapR on page 4155

6. After installing Spark on YARN but before running your Spark jobs, follow the steps outlined at Configuring Spark on YARN on page 4033.

Installing Spark on Mesos
This section includes instructions to download and install Apache Spark on Apache Mesos.

The MapR distribution of Spark on Mesos is only certified on CentOS.

Spark 2.1.0 runs with Apache Mesos 1.0.0 or later. You do not need to apply any special patches of Mesos. If you are already running a Mesos cluster, you can skip this topic.

Install Mesos following the instructions at Getting Started with Mesos.

Note: If you are building Mesos, execute the build steps as user 'mapr'. Also change the owner of the directory where you have unpacked the Mesos archive to user and group 'mapr'.

```
cd /path/to/mesos
sudo chown -R mapr:mapr /path/to/mesos
```

Installing Sqoop1
This topic includes instructions for using package managers to download and install Sqoop1 from the EEP repository.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

Execute the following commands as `root` or using `sudo:`

On each planned Sqoop node, install `mapr-sqoop`:

On Ubuntu	`apt-get install mapr-sqoop`
On RedHat / CentOS	`yum install mapr-sqoop`
On SLES	`zypper install mapr-sqoop`

Installing Sqoop2
This topic includes instructions for using package managers to download and install Sqoop2 from the EEP repository.

For instructions on setting up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

Sqoop2 includes two packages:

- The client package, `mapr-sqoop2-client` (install on each node that will act as a client)
- The server package, `mapr-sqoop2-server` (install on at least one node in the cluster)

The Sqoop2 server also acts as a MapReduce client, so each node where you install the `mapr-sqoop2-server` package must also have Hadoop installed and configured. If you have a lot of client users, you can install multiple servers on multiple nodes, so as not to overload a single server.

Warning: Sqoop2 packages cannot be installed on the same nodes as Sqoop1 packages. However, you can use both versions in the same Hadoop cluster by installing Sqoop1 and Sqoop2 on different nodes.

Execute the following commands as `root` or using `sudo`:

1. On each Sqoop2 server node, install `mapr-sqoop2-server`:

On Ubuntu	`apt-get install mapr-sqoop2-server`
On RedHat and CentOS	`yum install mapr-sqoop2-server`
On SLES	`zypper install mapr-sqoop2-server`

2. On each Sqoop2 client node, install `mapr-sqoop2-client`:

On Ubuntu	`apt-get install mapr-sqoop2-client`
On RedHat and CentOS	`yum install mapr-sqoop2-client`
On SLES	`zypper install mapr-sqoop2-client`

3. Run `configure.sh -R`:

```
/opt/mapr/server/configure.sh -R
```

4. Start each Sqoop server node.

```
maprcli node services -name sqoop2 -action start -nodes <space delimited
list of nodes>
```

Step 11: Run configure.sh
Run `configure.sh` with the `-R` option to complete the configuration of ecosystem components that were added manually.

After installing ecosystem components manually, you must run the `configure.sh` script with the `-R` option on each node in the cluster:

```
/opt/mapr/server/configure.sh -R
```

Configuring the Cluster

Describes post-installation configuration tasks for MapR clusters.

After installing MapR core and any desired ecosystem components, you might need to perform additional tasks to ready the cluster for production. To learn more about configuring clusters, see this course.

Configure the OJAI Distributed Query Service on page 182	If you want to use the optional OJAI Distributed Query Service on page 505, you must install Drill and configure and register the service.

last-updated: Apr 11, 2022

Setting up Topology	The locations of nodes and racks in a cluster determine the location of replicated copies of data. Optimally defined cluster topology results in data being replicated to separate racks, providing continued data availability in the event of rack or node failure.
Setting Up Volumes	Keeping volume hierarchy efficient to maximize data availability. Without a volume structure in place, performance will be negatively affected. Referring to the volume plan created in Planning the Cluster, use the Control System or the `maprcli` command to create and mount distinct volumes to allow more granularity in specifying policy for subsets of data. If you do not set up volumes, and instead store all data in the single volume mounted at /, it creates problems in administering data policy later as data size grows.
Setting Up Central Configuration	MapR services can be configured globally across the cluster, from primary configuration files stored in a MapR filesystem, eliminating the need to edit configuration files on all nodes individually.
Designating NICs for MapR on page 844	If multiple NICs are present on nodes, you can configure MapR to use one or more of them, depending on the cluster's need for bandwidth. See Cluster Design Objectives on page 110 for more information.
Setting up NFS	Access data on a licensed MapR cluster, mount the MapR cluster, and use standard shell scripting to read and write live data through NFS, which can be faster than using `hadoop fs` commands.
Configuring Authentication	If you use Kerberos, LDAP, or another authentication scheme, make sure PAM is configured correctly to grant MapR access.
Configuring Permissions	By default, users are able to log on to the Control System, but do not have permission to perform any actions. You can grant specific permissions to individual users and groups.
Setting Usage Quotas	You can set specific quotas for individual users and groups.
Configuring Alarm Notifications	If an alarm is raised on the cluster, MapR sends an email notification. For example, if a volume goes over its allotted quota, MapR raises an alarm and sends an email to the volume creator.
Setting Up the Client and MapR POSIX Client	You can access the cluster either by logging into a node on the cluster, or by installing MapR client software on a machine with access to the cluster's network.
Working with Mirror Volumes	To access multiple clusters or mirror data between clusters, work with mirror volumes.

Installing the File Migration Service on the Edge Cluster

Describes how to install the File Migration service on CentOS nodes.

To install the package on CentOS, run the following command:

```
yum install mapr-filemigrate
```

Note: The File Migration service is only supported on CentOS. There are no packages for Debian-based OS.

Since the File Migration service provides for high availability, you can install it on as many nodes as you wish. Warden will ensure that it is always active on exactly one node.

After installation, a sample properties file, `FileMigrate.properties.default`, with default values is placed in the `/opt/mapr/filemigrate/filemigrate-1.0.0/conf` directory. To manually edit the properties file, follow the steps for Configuring the File Migration Service Using the Properties File on page 1009. Using the UI to configure the service, automatically creates the necessary file in the `maprfs:///var/mapr/filemigrate` directory. For more information, see Configuring File Migration Service Using the UI on page 1007.

last-updated: Apr 11, 2022

Installing the MapR XD Distributed File and Object Store

Describes how to install MapR XD software with or without the MapR Installer.

The steps for installing the MapR XD Distributed File and Object Store are the same as the steps for installing MapR software, with some exceptions as follows.

For more information about the MapR XD, see MapR XD Distributed File and Object Store on page 451.

Before Installing the MapR XD Using the MapR Installer

Regardless of the method you use to install the MapR XD, before installing, you should review the information in these topics:

- MapR Repositories and Packages on page 128
- Preparing Each Node on page 129

Installing the MapR XD Using the MapR Installer

Use these steps to install the MapR XD using the web-based MapR Installer:

1. Download the MapR Installer. See MapR Installer on page 5395.
2. On the **Version & Services** page of the MapR Installer, specify these values:
   - **MapR Version:** 6.1.0 or later.
   - **Edition:** MapR Data Platform Enterprise Edition.
   - **Select Configuration Options:** Select security options as needed.
   - **License Option:** Apply the MapR XD license after the installation completes.
   - **EEP Version:** The EEP version is pre-selected.
   - **Auto-Provisioning Template:** MapR File System and Object Store (MapR XD). You do not need to select any services.
3. Click **Next** to advance through the menus.
4. On the **Monitoring** page, enable metrics collection with either the full or minimum configuration.
5. When the MapR Installer indicates that the installation is complete, go to After Installing a MapR XD Cluster on page 224.

Installing the MapR XD without Using the MapR Installer

Use the following information to install the MapR XD using manual steps.

You can use the manual installation steps described in Installing without the MapR Installer on page 141 to install the MapR 6.0.0 or later packages. Perform all steps unless otherwise indicated. Note the following considerations for some steps:

Step 1	Follow the documented steps.
Step 2	A repository needs to be configured for the Metrics Monitoring components. These components are available in the EEP repository.
Step 3	Install these MapR 6.0.0 or later packages at a minimum:

last-updated: Apr 11, 2022

	• `mapr-core` • `mapr-fileserver` • `mapr-cldb` • `mapr-nfs`[1] • `mapr-mastgateway` • `mapr-webserver` • `mapr-zookeeper` For information about packages and dependencies, see Packages and Dependencies for MapR Software on page 68.
Step 4	Follow the documented steps.
Step 5	Follow the documented steps.
Step 6	Follow the documented steps.
Step 7	Apply the MapR XD license.
Step 8	Complete the steps to install Metrics Monitoring.
Step 9	Installing Log Monitoring is optional.
Step 10	Skip this step. Except for the Metrics Monitoring components, you do not need to install any ecosystem components.
Step 11	Complete this step. Running `configure.sh` with the `-R` option is required.

[1]When you install `mapr-nfs`, NFSv3 is installed. To install NFSv4, you must use the `mapr-nfs4server` package. Neither NSFv3 nor NSFv4 provides security by default. You can configure NFSv4 server to work with Active Directory and Kerberos servers, but you must first install Active Directory and Kerberos servers. For more information, see Configuring NFSv4 Server for Kerberos on page 1209. NFSv3 does not support security. .

After Installing a MapR XD Cluster

After successfully installing the cluster, configure the cluster and set up clients using this information:

- Configuring the Cluster on page 221
- Setting Up Clients and Services on page 385

Installing MapR Data Fabric for Kubernetes

This section describes how to plan for and install the MapR Container Storage Interface (CSI) Storage Plugin and the MapR Data Fabric for Kubernetes FlexVolume Driver.

Getting Started with the MapR Container Storage Interface (CSI) Storage Plugin

This section describes how to plan for, install, and upgrade the MapR Container Storage Interface (CSI) Storage Plugin.

See MapR Container Storage Interface (CSI) Storage Plugin Overview on page 666 for more information.

Planning for the MapR Container Storage Interface (CSI) Storage Plugin
Includes information you should review before installing or using the MapR Container Storage Interface (CSI) Storage Plugin.

For release notes information, see MapR CSI Storage Plugin Release Notes on page 6550.

Downloads (CSI)
Lists the downloads for the MapR Container Storage Interface (CSI) Storage Plugin.

Downloads for the MapR Container Storage Interface (CSI) Storage Plugin are available at these locations:

Site	URL	Contents
Docker Hub	FUSE: • https://hub.docker.com/r/maprtech/csi-kdfplugin • https://hub.docker.com/r/maprtech/csi-kdfprovisioner • https://hub.docker.com/r/maprtech/csi-kdfdriver Loopback NFS: • https://hub.docker.com/r/maprtech/csi-nfsplugin • https://hub.docker.com/r/maprtech/csi-kdfprovisioner • https://hub.docker.com/r/maprtech/csi-nfsdriver	Docker containers for the MapR installation files
GitHub Repository	https://github.com/mapr/mapr-csi	MapR installation and example `.yaml` files: • A **deploy** folder that contains the latest CSI Plugin deployment `.yaml` files in `deploy/kubernetes/fuse` and `deploy/kubernetes/nfs` • An **examples** folder that contains example `.yaml` files • A **build** folder that contains the custom template to build CSI plugin container image

Prerequisites for Installing the MapR Container Storage Interface (CSI) Storage Plugin
Lists the prerequisites for installing and using the MapR Container Storage Interface (CSI) Storage Plugin.

Hardware and Software Requirements

To install and use the MapR Container Storage Interface (CSI) Storage Plugin, you must have the following:

Component	Supported Versions
MapR XD Distributed File and Object Store	6.1.0 or later. For additional version compatibility information, see CSI Version Compatibility on page 5596.
MapR Ecosystem Pack (EEP)	Any EEP supported by MapR 6.1.0 or later. See EEP Support by MapR Core Version.

last-updated: Apr 11, 2022

Component	Supported Versions
Kubernetes Software	1.17 and later*
OS (Kubernetes nodes)	All nodes in the Kubernetes cluster must use the same Linux OS. Configuration files are available to support: • CentOS • RHEL (use CentOS configuration file) • Ubuntu **Note:** Docker for Mac with Kubernetes is not supported as a development platform for containers that use MapR for Kubernetes.
CSI Driver	FUSE and Loopback NFS drivers (implementing the CSI spec with v1.3.0). The download location shows the latest version of the driver.
Sidecar Containers	The CSI plugin pod uses: • `csi-node-driver-registrar` — v1.3.0 • `livenessprobe` — v2.2.0 The CSI provisioner pod uses: • `csi-attacher` — v2.2.0 • `csi-provisioner` — v1.6.0 • `csi-snapshotter` — v3.0.2 • `snapshot-controller` — v3.0.2 • `livenessprobe` — v2.2.0 • `csi-resizer` — v0.5.0
POSIX License	The Basic POSIX client package is included by default when you install MapR for Kubernetes. The Platinum POSIX client can be enabled by specifying a parameter in the pod specification. To enable the Platinum POSIX client package, see Enabling the Platinum Posix Client for MapR Data Fabric for Kubernetes FlexVolume Driver on page 3166. For a comparison of the Basic and Platinum POSIX client packages, see Preparing for Installation (MapR POSIX Client) on page 400.

*Kubernetes alpha features are not supported.

Before You Install

Before installing the MapR Container Storage Interface (CSI) Storage Plugin, note that the installation procedure assumes that the Kubernetes cluster is already installed and functioning normally. In addition:

1. Ensure that all Kubernetes nodes use the same Linux distribution.

   For example, all nodes can be CentOS nodes, or all nodes can be Ubuntu nodes. A cluster with a mixture of CentOS and Ubuntu nodes is not supported.

2. Configure your Kubernetes cluster to allow privileged pods by running the following commands:

```
$ ./kube-apiserver ...  --allow-privileged=true …
```

```
$ ./kubelet ...  --allow-privileged=true ...
```

3. Enable mount propagation to share volumes mounted by one container with other containers in the same pod and other pods on the same node.

   See Mount Propagation for more information.

4. Apply CRDs to your Kubernetes cluster if they are not already present:

   Kubernetes 1.20 and Later

```
kubectl apply -f https://
raw.githubusercontent.com/kubernetes-csi/external-snapshotter/v4.2.1/
client/config/crd/snapshot.storage.k8s.io_volumesnapshotclasses.yaml
kubectl apply -f https://
raw.githubusercontent.com/kubernetes-csi/external-snapshotter/v4.2.1/
client/config/crd/snapshot.storage.k8s.io_volumesnapshotcontents.yaml
kubectl apply -f
https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/
v4.2.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshots.yaml
```

   Kubernetes 1.19 and Earlier

```
kubectl apply -f https://raw.githubusercontent.com/
kubernetes-csi/external-snapshotter/release-3.0/client/config/crd/
snapshot.storage.k8s.io_volumesnapshotclasses.yaml
kubectl apply -f https://raw.githubusercontent.com/
kubernetes-csi/external-snapshotter/release-3.0/client/config/crd/
snapshot.storage.k8s.io_volumesnapshotcontents.yaml
kubectl apply -f https://raw.githubusercontent.com/
kubernetes-csi/external-snapshotter/release-3.0/client/config/crd/
snapshot.storage.k8s.io_volumesnapshots.yaml
```

   For more information see Snapshot Controller.

5. For OpenShift, install the SecurityContextConstraints by applying `deploy/openshift/csi-scc.yaml` in the `mapr-csi` GitHub repository:

```
oc apply -f deploy/openshift/csi-scc.yaml
```


last-updated: Apr 11, 2022

6. Create the state volume-mount path, and update the CSI driver `yaml`. In prior releases, the state of dynamically provisioned volumes and their snapshots was held in memory. The provisioner would lose this state if the controller pod was restarted or upgraded. After restarts, the provisioner would fail to take snapshots, restore snapshots, resize or clone previously created volumes.

   With the latest version of the CSI driver, the provisioner persists the encrypted state of the dynamically provisioned volumes and their snapshots in a volume on the data-fabric cluster. If the controller pod is restarted, the state is automatically recovered, and operations on previously created volumes work as intended.

   You can change the state volume-mount prefix by updating the `--statevolmountprefix=/path/to/dir` argument in the `mapr-kdfprovisioner` image of the CSI driver `yaml`.

   Note: The directory you specify needs to be read-writable for all users who provision volumes on the data-fabric cluster using CSI drivers:

   ```
   # Create state volume mount path
   hadoop fs -mkdir /apps/k8s
   hadoop fs -chmod 777 /apps/k8s

   # Update csi driver yaml
   --statevolmountprefix=/apps/k8s
   ```

7. Understand the number of volume mounts per node that your application requires. The CSI driver default is 20 volume mounts per node. You can modify the number of volume mounts per node by adjusting the value of the `maxvolumepernode` parameter in the `csi-maprkdf-<version>.yaml` or `csi-maprnfskdf-<version>.yaml` file.

Installing, Uninstalling, and Upgrading the MapR Container Storage Interface (CSI) Storage Plugin
This section describes the steps for installing, uninstalling, and upgrading the MapR Container Storage Interface (CSI) Storage Plugin.

By default, the CSI Driver includes CentOS 8 as the base image. If you want to customize the installation, you can build your own container with a FUSE-based POSIX supported OS. See Building Your Own Container on page 229 for more information.

Installing the CSI Driver

1. Download and install the CSI Driver custom resource definition on the Kubernetes cluster by running the following command:

   ```
   kubectl create -f csi-maprkdf-v<version>.yaml
   ```

   where `<version>` is the driver version being installed.

   FUSE

   ```
   kubectl create -f csi-maprkdf-v<version>.yaml
   ```

   Loopback NFS

   ```
   kubectl create -f csi-maprnfskdf-v<version>.yaml
   ```

   When you run the command to install the CSI Driver, the service accounts, rule-based access controls (RBAC), and the statefulset and daemonset are created on the pods on the Kubernetes cluster.

2. Verify the installation by running the following command.

   ```
   kubectl get pods --all-namespaces -o wide
   ```

After installing, you can use the CSI Driver to statically and dynamically provision and mount a MapR volume. See Using the MapR Container Storage Interface (CSI) Storage Plugin on page 3104 for more information.

Uninstalling the CSI Driver

- To uninstall the CSI driver, run the following command:

  ```
  kubectl delete -f csi-maprkdf-v<version>.yaml
  ```

  where `<version>` is the driver version being installed.

  FUSE

  ```
  kubectl delete -f csi-maprkdf-v<version>.yaml
  ```

  Loopback NFS

  ```
  kubectl delete -f csi-maprnfskdf-v<version>.yaml
  ```

  When you run the command to uninstall, all the pods with the mount provisioned by CSI Driver are removed.

Upgrading the CSI Driver

Online upgrades for the CSI driver are not currently supported. To perform an offline upgrade, use the following steps:

1. Shut down all application pods that have a persistent volume mounted in the HPE Ezmeral Data Fabric.
2. Reapply the new CSI driver `.yaml`, and wait for Kubernetes to restart all the CSI pods:

   ```
   kubectl apply -f csi-maprkdf-<version>.yaml
   ```

3. Restart application pods.

Building Your Own Container

Describes how to build a container using the MapR Container Storage Interface (CSI) Storage Plugin template.

The MapR Container Storage Interface (CSI) Storage Plugin includes a template in the `build` directory to build your own container. The following template shows the MapR Container Storage Interface (CSI) Storage Plugin build for FUSE POSIX with CentOS 8 image. In the example, `<tag>` is the image version (available tags):

```
## FOR FUSE
# Copyright (c) 2009 & onwards. MapR Tech, Inc., All rights reserved
# CentOS Package Build
FROM centos:centos8
LABEL mapr.os=centos8
ENV container docker
# Setup repos and dl prereqs + Mapr Core
COPY mapr.repo /etc/yum.repos.d/
RUN rpm --import http://dl.fedoraproject.org/pub/epel/RPM-GPG-KEY-EPEL-8; \
   rpm --import https://package.mapr.hpe.com/releases/pub/maprgpg.key; \
   rpm --import https://package.mapr.hpe.com/releases/pub/gnugpg.key; \
   yum -y update && yum -y clean all; \
   yum -y install epel-release; \
   sed -i 's/^mirror/#mirror/g' /etc/yum.repos.d/epel.repo; \
   yum install -y mapr-client mapr-posix-client-basic
```

```
mapr-posix-client-platinum && \
   yum -y update && yum clean all && rm -rf /var/cache/yum; \
   mkdir -p /opt/mapr/lib/fusebasic /opt/mapr/lib/fuseplatinum; \
   cp /opt/mapr/lib/libMapRClient_c.so.1 /opt/mapr/lib/fusebasic/
libMapRClient_c.so.0; \
   rm -rf /opt/mapr/lib/libMapRClient_c.so.1
# Add Tini
ENV TINI_VERSION v0.18.0
ADD https://github.com/krallin/tini/releases/download/${TINI_VERSION}/tini /
tini
RUN chmod +x /tini
# Copy utils, driver and set entrypoint
COPY --from=docker.io/maprtech/csi-kdfdriver:<tag> \
    /go/src/plugin/bin/* /opt/mapr/bin/
RUN chmod +x /opt/mapr/bin/csi-kdfplugin; \
   chmod +x /opt/mapr/bin/start-fuse;
WORKDIR /opt/mapr
ENTRYPOINT ["/tini", "--", "bin/csi-kdfplugin"]
```

The template contains the information on the image for setting up the repository, deploying the (Basic, Container, or Platinum) POSIX client, information on the entry point, and Tini for POSIX process management. You can customize the template and build it by running the `docker-custom-build.sh` utility in the `build` directory or by running the `docker build` command with the custom image tag.

The following template shows the MapR Container Storage Interface (CSI) Storage Plugin build for Loopsback NFS with CentOS 8 image. In the example, `<tag>` is the image version (available tags):

```
## FOR NFS

# Copyright (c) 2009 & onwards. MapR Tech, Inc., All rights reserved

# CentOS Package Build
FROM centos:centos8
LABEL mapr.os=centos8
ENV container docker
# Setup repos and dl prereqs + Mapr Core
COPY mapr.repo /etc/yum.repos.d/
RUN rpm --import http://dl.fedoraproject.org/pub/epel/RPM-GPG-KEY-EPEL-8; \
    rpm --import https://package.mapr.hpe.com/releases/pub/maprgpg.key; \
    rpm --import https://package.mapr.hpe.com/releases/pub/gnugpg.key; \
    yum -y update && yum -y clean all; \
    yum -y install epel-release; \
    sed -i 's/^mirror/#mirror/g' /etc/yum.repos.d/epel.repo; \
    sed -i 's/^#base/base/g' /etc/yum.repos.d/epel.repo; \
    yum install -y mapr-loopbacknfs; \
    yum clean all && rm -rf /var/cache/yum

# Add Tini
ENV TINI_VERSION v0.18.0
ADD https://github.com/krallin/tini/releases/download/${TINI_VERSION}/tini /
tini
RUN chmod +x /tini

# Copy utils, driver and set entrypoint
COPY --from=docker.io/maprtech/csi-nfsdriver:<tag> \
     /go/src/plugin/bin/* /opt/mapr/bin/
RUN chmod +x /opt/mapr/bin/csi-nfsplugin; \
    chmod +x /opt/mapr/bin/start-loopbacknfs;
WORKDIR /opt/mapr
ENTRYPOINT ["/tini", "--", "bin/csi-nfsplugin"]
```

Installing the HPE Ezmeral CSI Operator
Describes how to download and install the HPE Ezmeral CSI Operator for Kubernetes for deployment in Kubernetes and OpenShift environments.

Overview

The HPE Ezmeral CSI Operator for Kubernetes packages, deploys, and manages HPE Ezmeral CSI Drivers on Kubernetes and OpenShift. After installing the operator and creating a CSI Driver object, you can enable static and dynamic provisioning of persistent volumes on the MapR Data Platform platform.

Installing the Operator in Kubernetes

To install the operator in a Kubernetes environment:

1. Install the Operator Lifecycle Manager (OLM) tool. The OLM allows you to manage the operators running on your cluster:

   ```
   $ curl -sL https://github.com/operator-framework/
   operator-lifecycle-manager/releases/download/v0.17.0/install.sh |
   bash -s v0.17.0
   ```

2. Install the HPE Ezmeral CSI Operator by running the following command:

   ```
   $ kubectl create -f https://operatorhub.io/install/
   hpe-ezmeral-csi-operator.yaml
   ```

   The operator is installed in the `my-hpe-ezmeral-csi-operator` namespace and is usable only from this namespace.

3. After installation, use the following command to watch the operator come up:

   ```
   $ kubectl get csv -n my-hpe-ezmeral-csi-operator
   ```

4. To instantiate the driver object, create a file named `hpe-csi-operator.yaml`, and populate it according to the CSI Driver that is being deployed.

5. Create a CSI Driver object. The operator supports FUSE and Loopback NFS drivers. In the following examples, `<tag>` is the image tag:

   - **HPE Ezmeral CSI Driver (FUSE)**

     ```
     apiVersion: ezmeral.hpe.com/v1
     kind: HPEEzmeralCSIDriver
     metadata:
       name: hpeezmeralcsidriver
       namespace: my-hpe-ezmeral-csi-operator
     spec:
       controllerImage: maprtech/csi-kdfprovisioner:<tag>
       nodeImage: maprtech/csi-kdfplugin:<tag>
       pullPolicy: IfNotPresent
     ```

- **HPE Ezmeral CSI Driver (Loopback NFS)**

```
apiVersion: ezmeral.hpe.com/v1
kind: HPEEzmeralNFSCSIDriver
metadata:
  name: hpeezmeralnfscsidriver
  namespace: my-hpe-ezmeral-csi-operator
spec:
  controllerImage: maprtech/csi-kdfprovisioner:<tag>
  nodeImage: maprtech/csi-nfsplugin:<tag>
  pullPolicy: IfNotPresent
```

6. Verify that the HPE Ezmeral CSI Operator and CSI Driver pods are running in the namespace:

```
$ oc get pods -n my-hpe-ezmeral-csi-operator
```

The CSI Driver is now ready to use. To use the CSI Driver to statically and dynamically provision and mount a data-fabric volume, see Using the MapR Container Storage Interface (CSI) Storage Plugin on page 3104.

Installing the Operator in OpenShift

You can install the HPE Ezmeral CSI Operator in the OpenShift environment by using the OpenShift CLI or the web console.

Prerequisites for OpenShift Installation

The HPE Ezmeral CSI Driver needs to run in privileged mode and needs access to host ports in the host network and must be able to mount `hostPath` volumes. Hence, before deploying the HPE Ezmeral CSI Operator on OpenShift, you must create a set of security context constraints (SCCs) to allow the CSI Driver to run with these privileges:

```
curl -sL https://raw.githubusercontent.com/mapr/mapr-csi/master/deploy/
openshift/operator-scc.yaml > hpe-ezmeral-csi-scc.yaml
```

1. Change `my-hpe-ezmeral-csi-driver` to the name of the project (for example, `hpe-ezmeral-csi` below) where the CSI Operator is being deployed:

```
oc new-project hpe-ezmeral-csi --display-name="HPE Ezmeral CSI Drivers
for Kubernetes"
sed -i  's/my-hpe-ezmeral-csi-driver/hpe-ezmeral-csi/g'
hpe-ezmeral-csi-scc.yaml
```

2. Deploy the SCC:

```
oc create -f hpe-ezmeral-csi-scc.yaml
securitycontextconstraints.security.openshift.io/hpe-ezmeral-csi-scc
created
```

Installing the Operator Using the OpenShift CLI

The following steps show an example of operator deployment using `oc`. This example assumes that the SCC has been applied to the project and has `kube:admin` privileges. The example deploys to the `hpe-ezmeral-csi` project, as described in the previous steps.

1. Create an operator group:

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: hpe-ezmeral-csi-operator
  namespace: hpe-ezmeral-csi
spec:
  targetNamespaces:
  - hpe-ezmeral-csi
```

2. Create a subscription to the operator:

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: hpe-ezmeral-csi-operator
  namespace: hpe-ezmeral-csi
spec:
  channel: stable
  installPlanApproval: Automatic
  name: hpe-ezmeral-csi-operator
  source: certified-operators
  sourceNamespace: openshift-marketplace
```

The operator is now installed on the OpenShift cluster. Creation of the subscription triggers the creation of the InstallPlan and CSV:

3. Display information about the InstallPlan and CSV:

```
oc get installplan -n hpe-ezmeral-csi
NAME            CSV                                  APPROVAL    APPROVED
install-5lmzg   hpe-ezmeral-csi-operator.v<ver>   Automatic   true

oc get csv -n hpe-ezmeral-csi
NAME
DISPLAY                                                  VERSION
REPLACES   PHASE
hpe-ezmeral-csi-operator.v<ver>   HPE Ezmeral Data Fabric CSI Operator
for Kubernetes   <ver>                Succeeded
```

4. Create a CSI Driver object. The operator supports FUSE and Loopback NFS drivers:

   - **HPE Ezmeral CSI Driver (FUSE)**

```
apiVersion: ezmeral.hpe.com/v1
kind: HPEEzmeralCSIDriver
metadata:
  name: hpeezmeralcsidriver
  namespace: hpe-ezmeral-csi
spec:
  controllerImage: registry.connect.redhat.com/maprtech/
csi-kdfprovisioner:latest
  nodeImage: registry.connect.redhat.com/maprtech/csi-kdfplugin:latest
  pullPolicy: IfNotPresent
```

- **HPE Ezmeral CSI Driver (LoopbackNFS)**

```
apiVersion: ezmeral.hpe.com/v1
kind: HPEEzmeralNFSCSIDriver
metadata:
  name: hpeezmeralnfscsidriver
  namespace: hpe-ezmeral-csi
spec:
  controllerImage: registry.connect.redhat.com/maprtech/
csi-kdfprovisioner:latest
  nodeImage: registry.connect.redhat.com/maprtech/csi-nfsplugin:latest
  pullPolicy: IfNotPresent
```

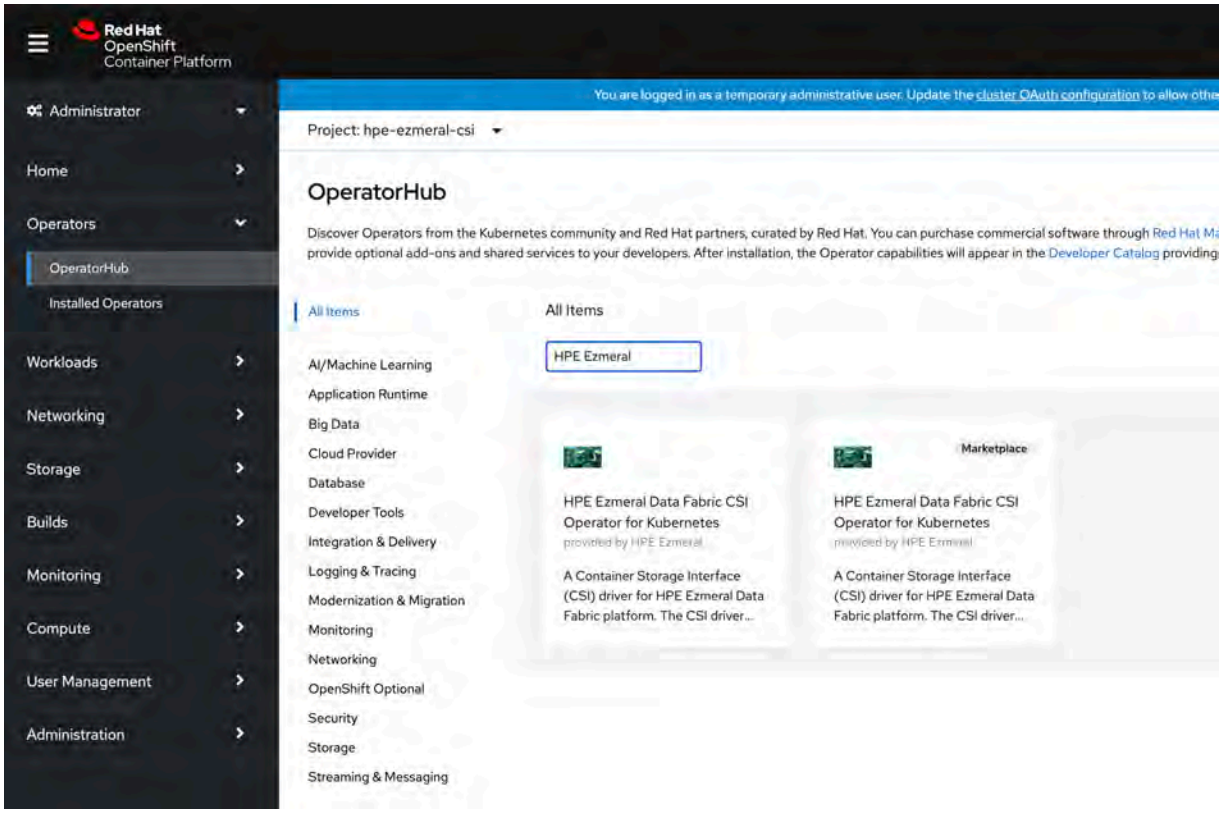
5. Verify that HPE Ezmeral CSI Operator and CSI Driver pods are running in the namespace:

```
# oc get pods -n hpe-ezmeral-csi
NAME                                              READY   STATUS
RESTARTS   AGE
hpe-ezmeral-csi-controller-0                      7/7     Running
0          62s
hpe-ezmeral-csi-driver-operator-9dd887bf7-hdxc9   1/1     Running
0          4m6s
hpe-ezmeral-csi-node-79xw5                        3/3     Running
0          61s
hpe-ezmeral-csi-node-m2gpv                        3/3     Running
0          61s
hpe-ezmeral-csi-node-x25dr                        3/3     Running
0          61s
hpe-ezmeral-nfscsi-controller-0                   7/7     Running
0          29s
hpe-ezmeral-nfscsi-node-hhrhv                     3/3     Running
0          28s
hpe-ezmeral-nfscsi-node-jz5cx                     3/3     Running
0          28s
hpe-ezmeral-nfscsi-node-tvtgm                     3/3     Running
0          28s
```

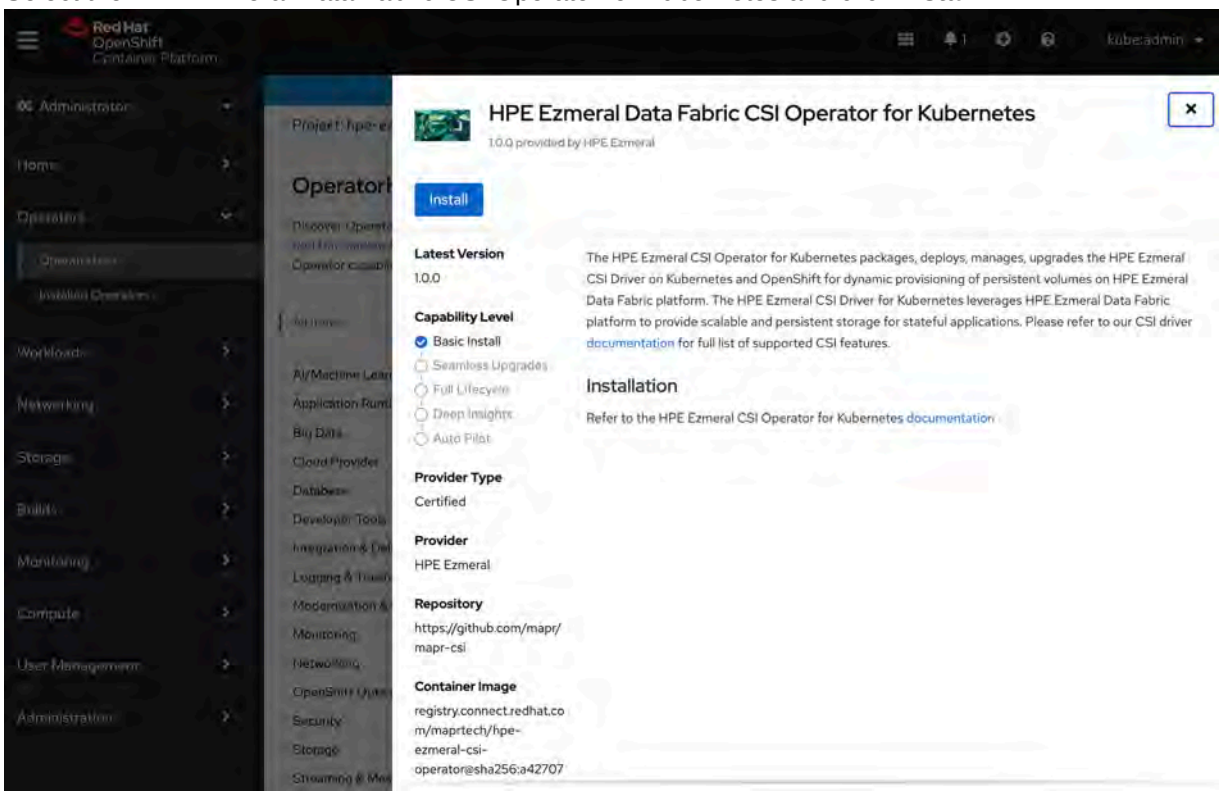
Installing the Operator Using the OpenShift Web Console

Use the following steps to install the operator using the web console:

1. Once the SCC has been applied to the project, log in to the OpenShift web console as `kube:admin`, and navigate to **Operators > OperatorHub**.
2. In the search field, type `HPE Ezmeral`, and press enter:

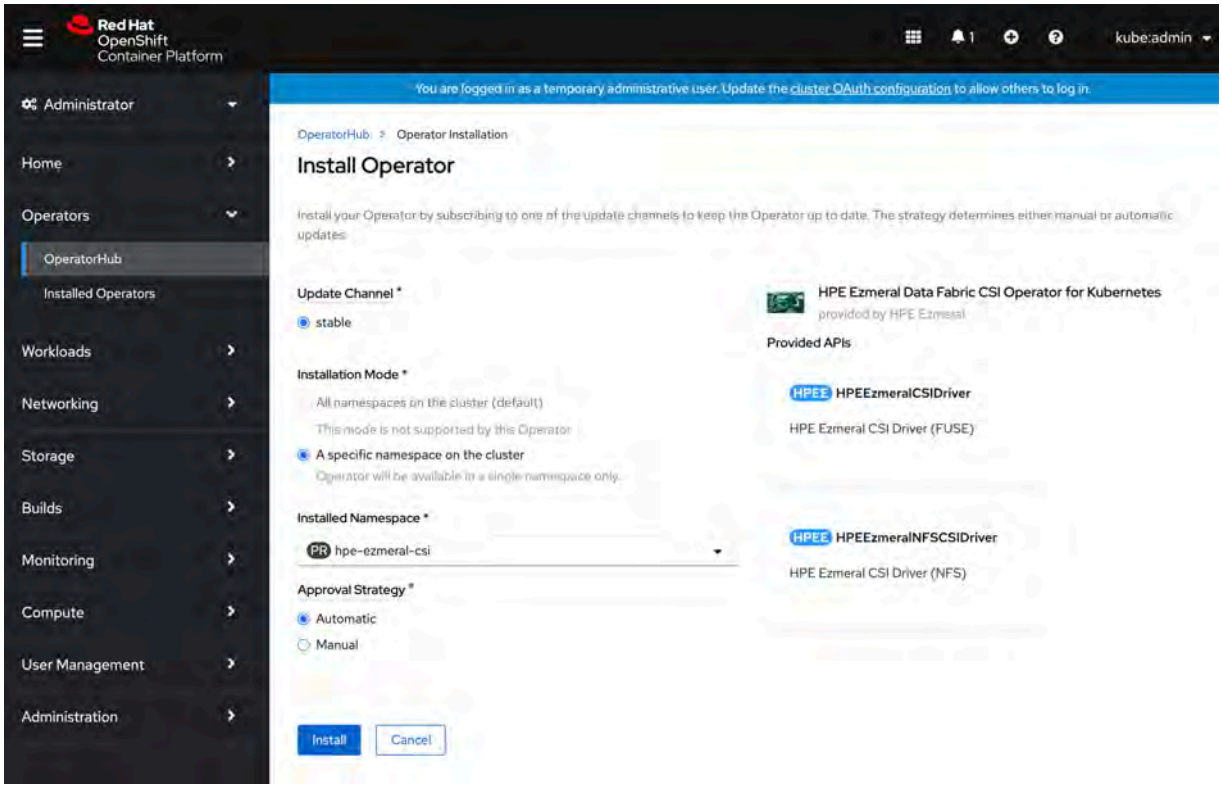


3. Select the HPE Ezmeral Data Fabric CSI Operator for Kubernetes and click **Install**:

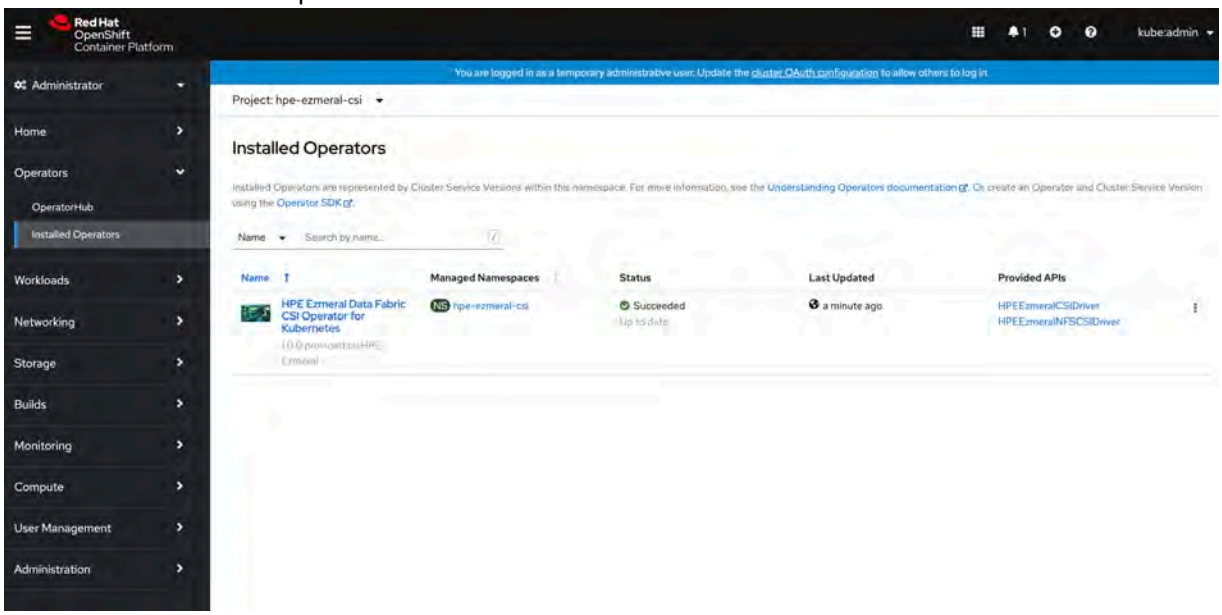


4. In the next pane, click **Install**:

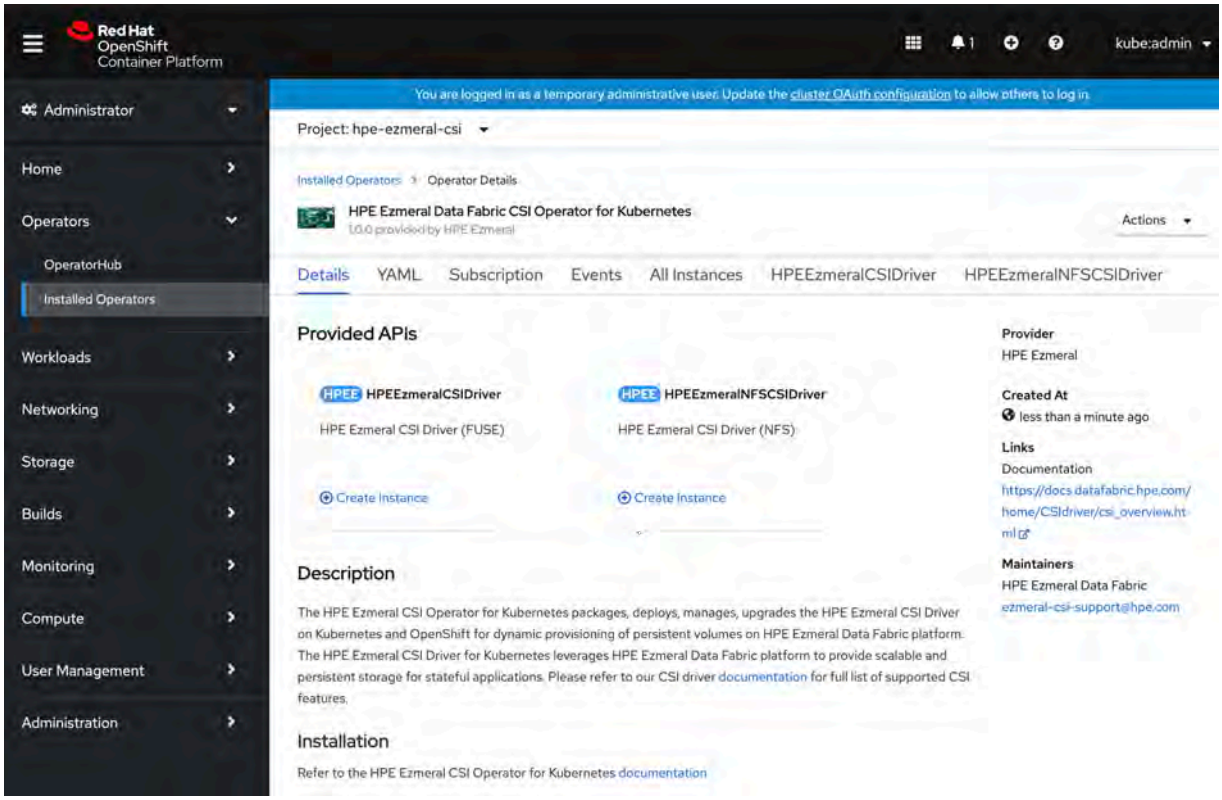
last-updated: Apr 11, 2022



5. The HPE Ezmeral CSI Operator is now installed:

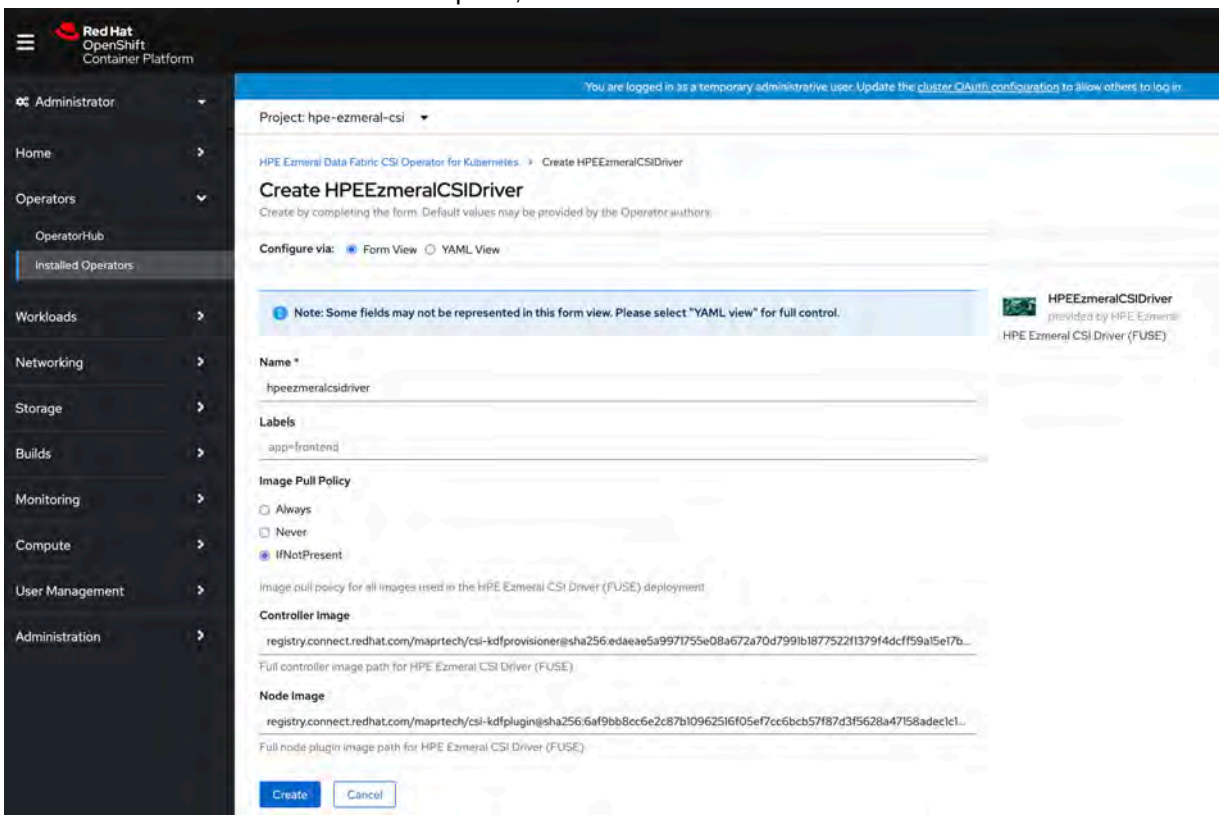


6. Click the HPE Ezmeral Data Fabric CSI Operator for Kubernetes to view the **Operator Details**:



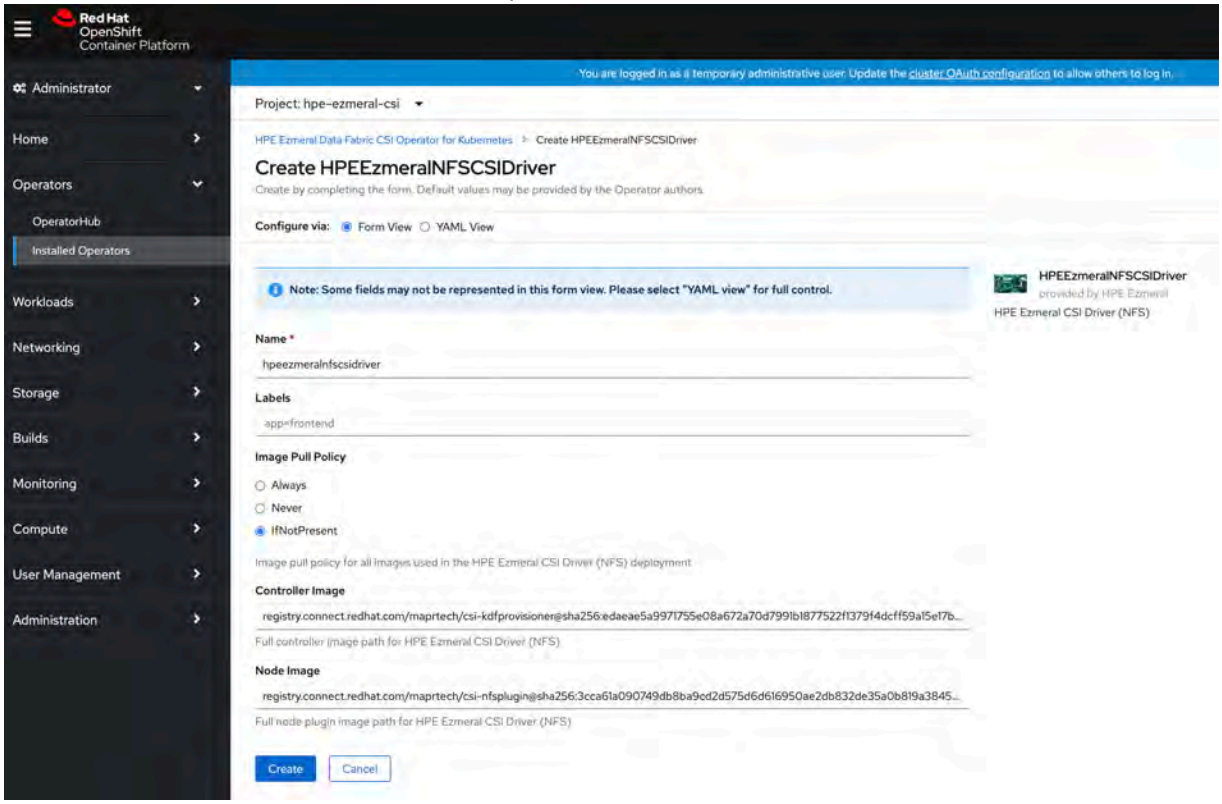
7. To create the HPE Ezmeral CSI Driver (FUSE), click **Create Instance** under **HPEEzmeralCSIDriver**.

8. In the **Create HPEEzmeralCSIDriver** pane, click **Create**:

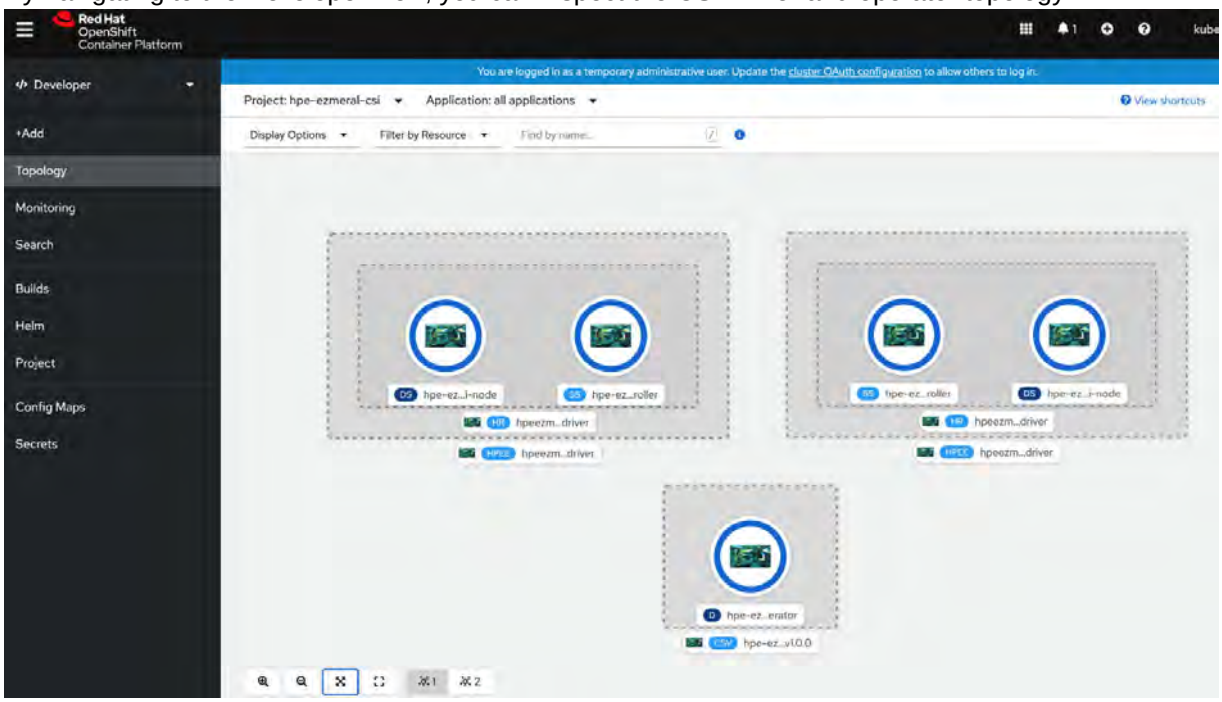


9. To create the HPE Ezmeral CSI Driver (NFS), click **Create Instance** under **HPEEzmeralNFSCSIDriver**.

10. In the **Create HPEEzmeralNFSCSIDriver** pane, click **Create**:



11. By navigating to the Developer view, you can inspect the CSI Driver and operator topology:



last-updated: Apr 11, 2022

The CSI Driver is now ready for use. To use the CSI Driver to statically and dynamically provision and mount a data-fabric volume, see Using the MapR Container Storage Interface (CSI) Storage Plugin on page 3104.

Migrating from MapR Data Fabric for Kubernetes FlexVolume Driver to MapR Container Storage Interface (CSI) Storage Plugin
Describes how to migrate from the MapR Data Fabric for Kubernetes FlexVolume driver to the MapR Container Storage Interface (CSI) Storage Plugin.

Installing both the CSI Driver and FlexVolume Driver on the same Kubernetes cluster can lead to an unstable Kubernetes environment. To migrate from the FlexVolume Driver to CSI Driver:

1. Stop all the container workloads using the FlexVolume Driver and de-provision the FlexVolume Driver.
2. Uninstall the FlexVolume Driver.
3. Install the CSI Driver.
   For more information, see Installing, Uninstalling, and Upgrading the MapR Container Storage Interface (CSI) Storage Plugin on page 228.
4. Modify the existing storage classes, PersistentVolumeClaims, PersistentVolumes, and Pod specifications to refer to the CSI Driver as the default Driver.
5. Resume the workloads you stopped (in step 1 above).

Getting Started with the MapR Data Fabric for Kubernetes FlexVolume Driver

Serves as a pointer on how to plan for, install, and upgrade MapR for Kubernetes.

For more information about the MapR for Kubernetes, see MapR Data Fabric for Kubernetes FlexVolume Driver Overview on page 671.

Planning for the MapR Data Fabric for Kubernetes FlexVolume Driver
Points to information you should review before installing or using the MapR for Kubernetes FlexVolume driver.

For release note information, see MapR Data Fabric for Kubernetes Release Notes.

Prerequisites for Using the MapR Data Fabric for Kubernetes FlexVolume Driver

To use the *MapR Data Fabric for Kubernetes* FlexVolume driver, you must have the following software versions:

Component	Supported Versions
MapR XD Distributed File and Object Store	5.2.2 or later
MapR Ecosystem Pack (EEP)	Any EEP supported by MapR 5.2.2 or later. See EEP Support by MapR Core Version.
Kubernetes Software	1.9*
OS (Kubernetes nodes)	All nodes in the Kubernetes cluster must use the same Linux OS. Configuration files are available to support: • CentOS • Red Hat (use CentOS configuration file) • SLES (use CentOS configuration file) • Ubuntu

Component	Supported Versions
	Note: Docker for Mac with Kubernetes is not supported as a development platform for containers that use the MapR Data Fabric for Kubernetes.
Volume Plug-in	1.0 or later. The download location shows the available versions of the plug-in. Plug-ins are supported for: • CentOS • Ubuntu • Microsoft Azure AKS • Red Hat OpenShift** • Google Kubernetes Engine (GKE) **Note:** Amazon EKS is not currently supported.
Provisioner	1.0 or later. The download location shows the available versions of the provisioner.
POSIX License***	The Basic POSIX client package is included by default when you install the MapR Data Fabric for Kubernetes. The Platinum POSIX client can be enabled by specifying a parameter in the Pod spec. To enable the Platinum POSIX client package, see Enabling the Platinum Posix Client for MapR Data Fabric for Kubernetes FlexVolume Driver on page 3166. For a comparison of the Basic and Platinum POSIX client packages, see Preparing for Installation (MapR POSIX Client) on page 400.

*Kubernetes alpha features are not supported.

**OpenShift Origin is supported because it supports Kubernetes 1.9. The OpenShift Container Platform (formerly known as OpenShift Enterprise) can be used only if it supports Kubernetes 1.9.

***Only the POSIX client is supported. NFSv3 is not supported.

Downloads (FlexVolume)

Downloads for the MapR Data Fabric for Kubernetes FlexVolume driver are available at these locations:

Site	URL	Content
MapR Software Downloads Site	https://package.mapr.hpe.com/tools/KubernetesDataFabric	MapR installation (`.yaml`) files
Docker Hub	https://hub.docker.com/r/maprtech/kdf-provisioner/ https://hub.docker.com/r/maprtech/kdf-plugin/	Docker containers for the MapR installation files
GitHub Repository	https://github.com/mapr/KubernetesDataFabric	Three types of resources: • A **build** folder that contains the Docker images used to build the data fabric • A **deploy** folder that contains the same files provided on the MapR Software Download site

Site	URL	Content
		• An **examples** folder that contains example `.yaml` files

Installing the MapR Data Fabric for Kubernetes FlexVolume Driver

This section describes the steps you must take to prepare for installation and install the configuration files for the MapR Data Fabric for Kubernetes FlexVolume driver.

Installing MapR and Kubernetes Software on Separate Nodes

This section describes how to install the configuration files for the MapR Data Fabric for Kubernetes. In this configuration, MapR and Kubernetes software must be installed on separate nodes.

To install the MapR Data Fabric for Kubernetes, you must download the configuration files and use the Kubernetes `kubectl` interface to install the namespace, RBAC, plug-in, and provisioner `.yaml` files.

Before Installation

Before installing the MapR Data Fabric for Kubernetes, note these preinstallation best practices:

- You must install the configuration files in the order shown in the steps below. Using a different installation order can cause problems.
- Ensure that all Kubernetes nodes use the same Linux distribution. For example, all nodes can be CentOS nodes, or all nodes can be Ubuntu nodes. But a cluster with a mixture of CentOS and Ubuntu nodes is not supported.
- This procedure does not allow you to install the MapR Data Fabric for Kubernetes on a Kubernetes node that is also a node in a MapR cluster. If a Kubernetes node already has MapR software installed, installing the MapR Data Fabric for Kubernetes can cause issues with the running MapR cluster. See Installing MapR and Kubernetes Software on the Same Nodes on page 243.
- Do not install the MapR client on a node where the volume plug-in configuration file is installed. The MapR client can be installed on a node in the Kubernetes cluster, but it must be installed **before** the MapR Data Fabric for Kubernetes is installed on the same Kubernetes cluster.

Installation Steps

Use these steps to install the configuration files:

1. Download the following configuration (`.yaml`) files from `https://package.mapr.hpe.com/tools/KubernetesDataFabric/v<version>/` to a directory on a node in the Kubernetes cluster:

File	Description
`kdf-namespace.yaml`	Configuration file for the `mapr-system` namespace, under which all MapR components are installed.
`kdf-rbac.yaml`	RBAC configuration file. This file enables the provisioner to call the Kubernetes APIs that it needs to function.
• `kdf-plugin-centos.yaml` • `kdf-plugin-ubuntu.yaml` • `kdf-plugin-azure.yaml`[1] • `kdf-plugin-openshift.yaml`[2]	Configuration files used to install the plug-in. Download the plug-in file that matches your environment. You can use the CentOS configuration file for Red Hat, CentOS, or SLES Kubernetes hosts.

File	Description
• `kdf-plugin-gke.yaml`[3]	
`kdf-provisioner.yaml`	Configuration file used to install the provisioner inside the Kubernetes cluster.

[1]Before installing the `kdf-plugin-azure.yaml`, see Azure AKS Considerations on page 245.

[2]To install the `kdf-plugin-openshift.yaml`, see OpenShift Considerations on page 246.

[3]To install the `kdf-plugin-gke.yaml`, see Google Kubernetes Engine (GKE) Considerations on page 247.

2. In Kubernetes, use the `kubectl create` command with the `-f` option to create the namespace for the plug-in and provisioner:

   Note: The examples in this procedure assume that you are running each `kubectl create` command from the directory containing the downloaded configuration files.

   ```
   kubectl create -f kdf-namespace.yaml
   ```

3. In Kubernetes, use the `kubectl create` command with the `-f` option to install the RBAC file:

   Note: Do not apply the RBAC file in OpenShift environments. See OpenShift Considerations on page 246.

   ```
   kubectl create -f kdf-rbac.yaml
   ```

4. In the plug-in configuration file that you downloaded in step 1, set the Kubernetes service location and the FlexVolume plug-in path. To specify the Kubernetes service location, specify the external location and port of your API server. You can find the correct values by doing a `kubectl config view` and looking at the current context and then looking at the cluster selected for that context. This information is used to look up tickets:

   ```
   - name : KUBERNETES_SERVICE_LOCATION
     value: "changeme!:6443"
   ```

   If your Kubernetes environment has a nonstandard location for FlexVolume plug-ins (for example, Azure environments sometimes use a nonstandard location), specify the FLEXVOLUME_PLUGIN_PATH by changing the directory in the `value:` field:

   ```
   - name : FLEXVOLUME_PLUGIN_PATH
     value: "/usr/libexec/kubernetes/kubelet-plugins/volume/exec"
   ```

5. Use the `kubectl create` command with the `-f` option to install the plug-in. The plug-in that you specify in the `create` command depends on your operating system environment:

   Note: When you issue the `kubectl create -f` command, a daemon set copies the plug-in to every node in the Kubernetes cluster.

   ```
   kubectl create -f kdf-plugin-centos.yaml
   ```

   or

   ```
   kubectl create -f kdf-plugin-ubuntu.yaml
   ```

   or

   ```
   kubectl create -f kdf-plugin-azure.yaml
   ```

   or

   ```
   kubectl create -f kdf-plugin-openshift.yaml
   ```

   or

   ```
   kubectl create -f kdf-plugin-gke.yaml
   ```

6. In Kubernetes, use the `kubectl create` command with the `-f` option to install the provisioner on a single node of the Kubernetes cluster. Kubernetes determines the node on which to install the provisioner.

   ```
   kubectl create -f kdf-provisioner.yaml
   ```

7. To begin using the MapR Data Fabric for Kubernetes, see Kubernetes FlexVolume Driver Configuration on page 3150.

Installing MapR and Kubernetes Software on the Same Nodes

Note: This feature is presented as a developer preview. Developer previews are not tested for production environments, and should be used with caution.

This section describes how to install the configuration files for the MapR Data Fabric for Kubernetes. In this configuration, MapR and Kubernetes software can coexist on the same nodes if certain version requirements are met.

Important: Some versions of the MapR Data Fabric for Kubernetes do not support installing MapR and Kubernetes software on the same nodes. To ensure that you are using a version that supports this feature, see the MapR Data Fabric for Kubernetes release notes.

Before Installation

Before installing the MapR Data Fabric for Kubernetes, note these preinstallation requirements:

- This procedure assumes that the Kubernetes cluster is already installed and functioning normally.

last-updated: Apr 11, 2022

- Ensure that all Kubernetes nodes use the same Linux distribution. For example, all nodes can be CentOS nodes, or all nodes can be Ubuntu nodes. But a cluster with a mixture of CentOS and Ubuntu nodes is not supported.
- This procedure requires stopping Warden and Zookeeper on all nodes in the MapR cluster and then restarting Warden and Zookeeper on all nodes. The steps cannot be performed online one node at a time.
- Do not install the MapR client on a node where the volume plug-in configuration file is installed. The MapR client can be installed on a node in the Kubernetes cluster, but it must be installed **before** the MapR Data Fabric for Kubernetes is installed on the same Kubernetes cluster.

CAUTION: Do not try to install the volume plug-in without following the steps below. Doing so can cause MapR libraries to be overwritten.

Install the MapR 6.0.1 or Later Cluster on the Kubernetes Nodes

Use any of the methods described in Installing with the MapR Installer on page 141 to install a MapR 6.0.1 or later cluster on the existing Kubernetes nodes.

Install the MapR Data Fabric for Kubernetes

Use these steps to install the MapR Data Fabric for Kubernetes on the Kubernetes cluster:

1. Stop all running jobs on the MapR cluster.
2. Stop Warden on all MapR cluster nodes by running the following command on each node:

   ```
   service mapr-warden stop
   ```

3. Stop Zookeeper on all MapR Zookeeper nodes by running the following command on each node:

   ```
   service mapr-zookeeper stop
   ```

4. Deploy the MapR Data Fabric for Kubernetes components by using steps 1 through 6 of Installing MapR and Kubernetes Software on Separate Nodes on page 241.
5. Configure the `MAPR_SUBNETS` environment variable to ensure that MapR software does not use the `docker0` network interface on each node. See Designating NICs for MapR on page 844.

   If `MAPR_SUBNETS` is not set, the CLDB uses all NICs present on the node. When Docker is installed on a node, the`docker0` bridge is created as a virtual NIC for use by the Docker containers. You must configure the `MAPR_SUBNET` setting to include the physical NICs that you want the CLDB to use and *exclude* the `docker0` network interface. In this way, you can avoid issues with duplicate or non-routable IP addresses. For more information about `docker0`, see Docker container networking.
6. Start Zookeeper on all MapR Zookeepr nodes by running the following command on each node:

   ```
   service mapr-zookeeper start
   ```

7. Start Warden on all MapR cluster nodes by running the following command on each node:

   ```
   service mapr-warden start
   ```

Pod Security Policies and the MapR Data Fabric for Kubernetes

If your Kubernetes administrator has turned on Pod Security Policies, you must create a PSP for the MapR Data Fabric for Kubernetes. You should use your organization's best practices for writing a PSP, but you must enable several parameters in the PSP for your `maprkdf` service account:

```
volumes:
    - 'hostPath'
    - 'flexVolume'
allowedHostPaths:
    - pathPrefix: "/opt"
    - pathPrefix: "/usr/libexec/kubernetes/kubelet-plugins/volume/exec/"
    - pathPrefix: "/etc/kubernetes"
    - pathPrefix: "/etc/localtime"
 allowedFlexVolumes:
    - driver: mapr.com/maprfs
```

Here is an example of a PSP that would work:

```
# Copyright (c) 2009 & onwards. MapR Tech, Inc., All rights reserved
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: mapr-kdf-psp
spec:
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'projected'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
    - 'hostPath'
    - 'flexVolume'
  allowedHostPaths:
    - pathPrefix: "/opt"
    - pathPrefix: "/usr/libexec/kubernetes/kubelet-plugins/volume/exec/"
    - pathPrefix: "/etc/kubernetes"
    - pathPrefix: "/etc/localtime"
  allowedFlexVolumes:
    - driver: mapr.com/maprfs
  runAsUser:
    rule: 'RunAsAny'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'RunAsAny'
  fsGroup:
    rule: 'RunAsAny'
```

You enable a PSP for a ServiceAccount as part of a ClusterRole that is bound to the ServiceAccount. See Using RBAC Authorization. For example, add the `mapr-kdf-psp` to a ClusterRole like this:

```
- apiGroups: ['extensions']
    resources: ['podsecuritypolicies']
    verbs:     ['use']
    resourceNames:
    - mapr-kdf-psp
```

Azure AKS Considerations

Microsoft Azure turns on PodSecurityPolicies by default. This means you must create RBAC and PodSecurityPolicies for both the plug-in and any containers that call the plug-in.

last-updated: Apr 11, 2022

Here is an example of a PSP. It is recommended that you adapt this PSP to the security best practices of your organization:

```
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: mapr-kdf-psp
spec:
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'projected'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
    - 'hostPath'
    - 'flexVolume'
  allowedHostPaths:
    - pathPrefix: "/opt"
    - pathPrefix: "/usr/libexec/kubernetes/kubelet-plugins/volume/exec/"
    - pathPrefix: "/etc/kubernetes"
    - pathPrefix: "/etc/localtime"
  allowedFlexVolumes:
    - driver: mapr.com/maprfs
  runAsUser:
    rule: 'RunAsAny'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'RunAsAny'
  fsGroup:
    rule: 'RunAsAny'
```

Azure uses a non-standard FlexVolume path: `/etc/kubernetes/volumeplugins`. This path has already been changed in `kdf-plugin-azure.yaml`.

You must set the KUBERNETES_SERVICE_LOCATION for Azure. You can find the correct value by connecting to your Azure cluster using the `kubectl` interface. Use the `kubectl config view` command, and find the server name and port for the current context.

In Azure, the Kubelet process is running inside a *hypercube* container. The MapR plug-in must run inside that container. This means that the plug-in log is somewhat hidden. To view the plug-in log:

```
docker ps <to find the hyperkube container>
docker exec -it <hyperkube container ID> /bin/bash
cd /opt/mapr/logs
cat plugin plugin-k8s.log
```

OpenShift Considerations

For OpenShift environments, the installation steps are the same as described in Installing the MapR Data Fabric for Kubernetes FlexVolume Driver on page 241. However, you must not apply the RBAC file. Instead run the following commands:

```
oc create -f kdf-openshift-sa.yaml
oc create -f kdf-openshift-scc.yaml
oc adm policy add-scc-to-user maprkdf-scc
system:serviceaccount:mapr-system:maprkdf
oc create -f kdf-openshift-cr.yaml
oc adm policy add-cluster-role-to-user mapr:kdf
system:serviceaccount:mapr-system:maprkdf
```

All other installation steps are the same.

Google Kubernetes Engine (GKE) Considerations

To create a Google Kubernetes Engine (GKE) cluster, you must use Ubuntu node images instead of CentOS.

The high-level installation steps are as follows:

1. Create a cluster with Ubuntu nodes.
2. Follow the steps later on this page to create a PodSecurityPolicy (PSP).
3. Install the namespace, as described in Installing the MapR Data Fabric for Kubernetes FlexVolume Driver on page 241.
4. Install the PSP.
5. Install the RBAC file, as described in Installing the MapR Data Fabric for Kubernetes FlexVolume Driver on page 241.
6. Modify the service location in the plug-in, as described later on this page.
7. Install the `kdf-plugin-gke.yaml`, as described in Installing the MapR Data Fabric for Kubernetes FlexVolume Driver on page 241
8. Install the provisioner, as described in Installing the MapR Data Fabric for Kubernetes FlexVolume Driver on page 241

Creating a PSP

GKE turns on PodSecurityPolicies by default. This means that you must create Role-Based Access Control (RBAC) and PodSecurityPolicies for both the plug-in and any containers that call the plug-in. Before you can edit RBAC and PSPs in GKE, you have to give your `kubectl id` sufficient permissions. Assuming you have already logged into Google Cloud and connected your cluster to `kubectl`, you need to execute the following command:

```
gcloud info | grep Account
```

The command returns an email address. Copy the email address into the following command:

```
kubectl create clusterrolebinding
yourname-cluster-admin-binding --clusterrole=cluster-admin --user=myname@exa
mple.org
```

If this command is successful, you will have permissions to create a Pod security policy. Here is an example of a PSP. It is recommended that you adapt this PSP to the security best practices of your organization:

```
apiVersion: extensions/v1beta1
kind: PodSecurityPolicy
metadata:
  name: mapr-kdf-psp
spec:
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'projected'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
```


last-updated: Apr 11, 2022

```
    - 'hostPath'
    - 'flexVolume'
  allowedHostPaths:
    - pathPrefix: "/opt"
    - pathPrefix: "/usr/libexec/kubernetes/kubelet-plugins/volume/exec/"
    - pathPrefix: "/etc/kubernetes"
    - pathPrefix: "/etc/localtime"
  allowedFlexVolumes:
    - driver: mapr.com/maprfs
  runAsUser:
    rule: 'RunAsAny'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'RunAsAny'
  fsGroup:
    rule: 'RunAsAny'
```

Nonstandard FlexVolume Path and Service Location

GKE uses a non-standard FlexVolume path: `/home/kubernetes/flexvolume`. This path has already been changed in `kdf-plugin-gke.yaml`. However, you must set the KUBERNETES_SERVICE_LOCATION for GKE. To do this, you must edit the `kdf-plugin-gke.yaml` file to specify the service location. You can find the correct value by connecting to your GKE cluster using the `kubectl` interface. Use the `kubectl config view` command, and find the server name and port for the current context.

Upgrading the MapR Data Fabric for Kubernetes

This section describes how to upgrade the plug-in and dynamic provisioner, or upgrade Pods with attached volumes.

Upgrading the Plug-in and Provisioner

Before upgrading the plug-in, stop any Pods using the plug-in. You may want to quiesce any traffic hitting the Pod before shutdown. Failure to shut down the Pods before replacing the plug-in can lead to the Pod not being able to access its data until it is restarted.

Removing the plug-in does not kill existing Pods. The Pods should only lose their mounted storage when a new version of the plug-in is installed and the libraries used to communicate with MapR software are deleted.

Upgrading the provisioner does not require stopping Pods, but dynamic provisioning (creating MapR volumes for new PersistentVolumeClaims) will be unavailable during the provisioner upgrade.

Use these steps to upgrade the plug-in:

1. Stop any Pods using the the plug-in to be upgraded. Before shutting down the Pod, you might want to quiesce any traffic hitting the Pod.

   Note: If any Pods that use the MapR Data Fabric for Kubernetes are not shut down during the plug-in upgrade, those Pods will have mount access removed and will need to be deleted and re-created as new Pods. If existing Pods need to be removed or are stuck in the Terminating state, you can delete them forcefully by using the `kubectl delete pod` command:

   ```
   kubectl delete pod <pod-name> -n
   <pod-namespace> --force --grace-period=0
   ```

2. Download the new plug-in. See Downloads (FlexVolume) on page 240.

last-updated: Apr 11, 2022

3. Delete the old plug-in:

```
kubectl delete -f kdf-<old_plugin>.yaml
```

4. Deploy the new plugin:

```
kubectl create -f kdf-<new_plugin>.yaml
```

Upgrading Pods with Attached Volumes

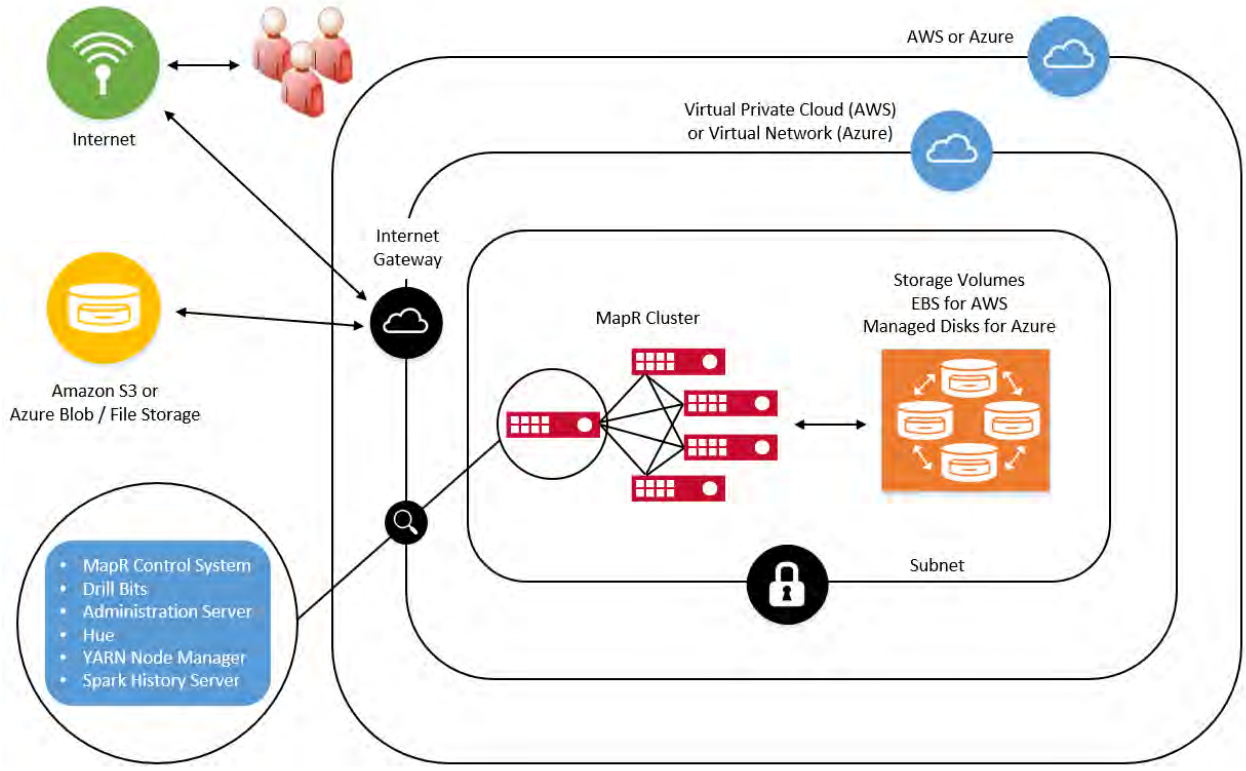
Pods with mounted volumes can be patched in place. See Update API Objects in Place Using kubectl patch. Volumes will disappear only when the Pod is deleted. Patching a Pod does not affect the mount. When a Pod is deleted, a volume disappears. However, if you delete a Pod using a PersistentVolume and you leave the PVC alive, you can remount the PersistentVolumeClaim and its PersistentVolume with a new Pod. In this scenario, there is no disruption or need to recreate the PersistentVolume.

Installing MapR in the Cloud

Provides an overview of MapR cloud installer capabilities.

With the latest MapR Installer, you can deploy a MapR cluster in the cloud quickly and customize your deployment using HPE-provided reference templates. You can provision a cluster in Amazon AWS or in Microsoft Azure and take advantage of the benefits of cloud computing.

The following diagram shows the high-level architecture for a cloud-based MapR cluster:



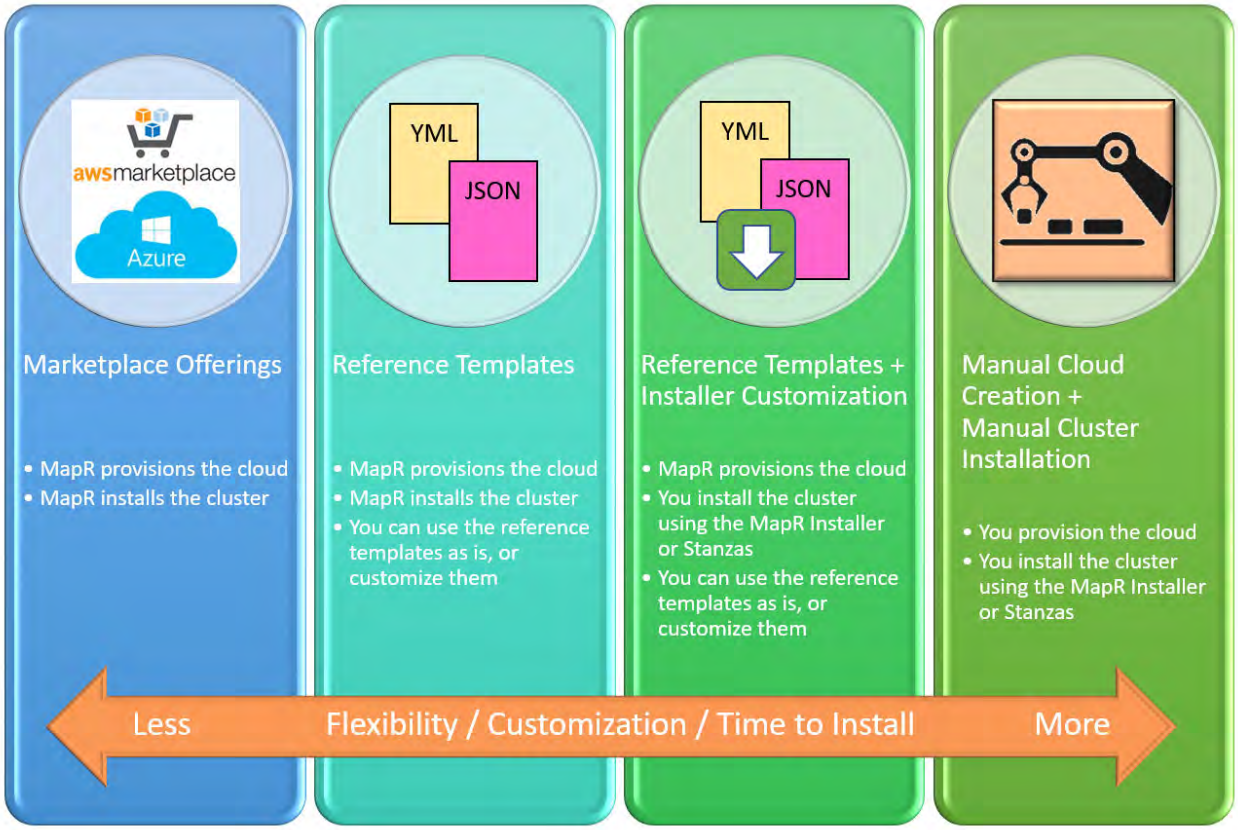
MapR Installer tools and templates provide several options for provisioning the cloud and installing the cluster. These options vary in:

- The time it takes to perform the installation

last-updated: Apr 11, 2022

- The level of customization each option supports
- The complexity of the installation

The following diagram shows the different options for provisioning to the cloud and installing the cluster. These options allow you to customize your MapR deployments to meet your business goals and make it easier for you to pay for only the MapR software that you use.



Marketplace Offerings

MapR offerings for Amazon AWS and Microsoft Azure allow you to deploy MapR clusters quickly to test new features and software versions:

- MapR Offerings in the AWS Marketplace on page 252
- MapR Offerings in the Azure Marketplace on page 267

MapR Reference Templates

MapR reference templates assist in provisioning. You can use the templates as is or modify them to control the way your installation is provisioned. The reference templates both provision your instance in the cloud and install a MapR cluster to your specifications. For more information, see:

- About the MapR Reference Templates for AWS on page 258
- About the MapR Reference Templates for Azure on page 272

MapR Installer Web Interface

The MapR Installer allows you to install, add services, upgrade, scale, and gather information about your MapR on-premise or cloud-based cluster. You can use the Installer after your cluster is deployed, or deploy just the MapR Installer and use the web interface or MapR Installer Stanzas to customize your installation.

MapR Installer Stanzas

MapR Installer Stanzas give you a programmatic interface to perform most of the functions available in the MapR Installer web insterface. See MapR Installer Stanzas on page 5503.

Custom Images

MapR tools allow you to build custom Amazon Machine Images (AMIs) or Azure VM images to meet additional security or software requirements for your environment. See Creating Custom Images on page 278.

Supported Operating Systems

Supported operating systems and MapR software for the cloud Installer are:

Operating systems	RHEL 7.4, Ubuntu 14.04, Ubuntu 16.04, CentOS 7.4, SLES 12 SP 2
MapR Core	5.2.2 or later
MapR Ecosystem Pack (EEP)	3.0.1 or later

Planning for Access to the Cluster

Before installing a MapR cluster in the cloud, be sure to review this information so that you can plan for access to the cluster.

Public and Private IP Addresses

Cloud-based MapR clusters are installed by default in a virtual private cloud (AWS) or virtual network (Azure) using private IP addresses. You must make sure you have a way to connect to the private IP addresses of each node in the cluster. For the Azure marketplace offering and the reference templates (AWS and Azure), OpenVPN can be configured to facilitate secure access.

For the AWS marketplace offering, if you don't use a public IP address, you need to ensure that you have a way to connect to the VPC and the private IP of the nodes.

Security Groups

The MapR reference templates for AWS and Azure support the use of security groups. Security groups allow you to control precisely the traffic that can access your instance. For AWS, you can use existing security groups or create new ones. For more information, see:

- AWS Security Group Information
- Azure Network Security Group Information

OpenVPN

Some MapR reference templates for AWS and Azure provide an option to install OpenVPN. OpenVPN is an open-source software application that creates a virtual private network (VPN) between your workstation and the cloud-based MapR cluster. If you do not want to use your network infrastructure or public IP addresses to connect to the cloud, OpenVPN can provide a secure connection. For more information, see Using OpenVPN on page 279.

last-updated: Apr 11, 2022

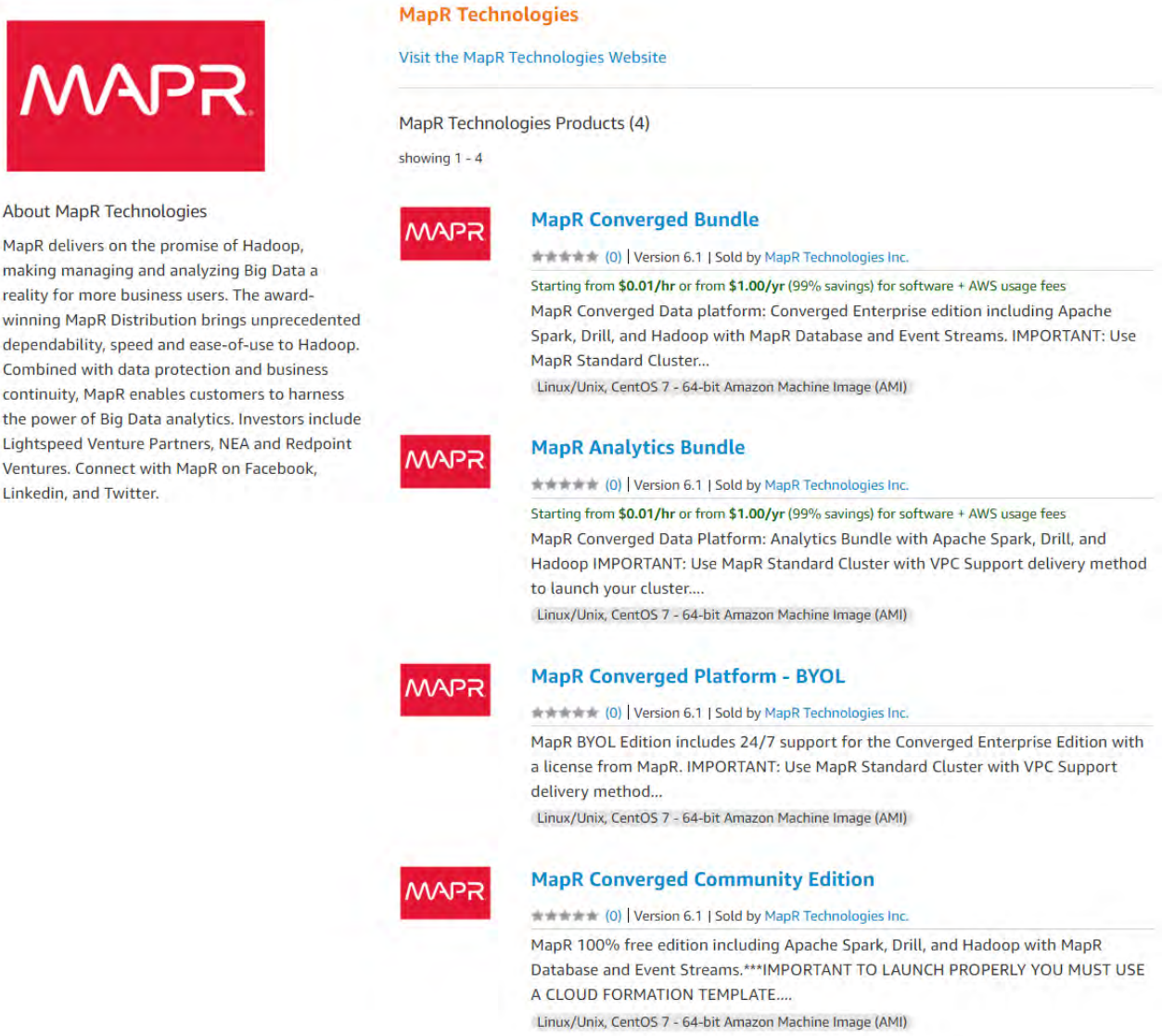
Deploying MapR Clusters on AWS

This section describes cloud-deployment procedures for Amazon AWS.

For similar procedures supporting Microsoft Azure, see Deploying MapR Clusters on Azure on page 267.

MapR Offerings in the AWS Marketplace

To view the MapR offerings in the AWS Marketplace, open a browser, and navigate to the MapR AWS Seller page:



MapR offerings include AMIs with hourly, bring-your-own (BYO), and community licenses:

MapR CDP Offering	Licensed Hourly?	Customizable?	Enterprise Features	MapR Support	Description
Analytics Bundle	Yes	No	Enabled	Included	MapR XD Distributed File and Object Store, Apache Spark, Apache Drill, Apache Hadoop
Converged Bundle	Yes	Yes	Enabled	Included	MapR XD, MapR Database, MapR Event Store For Apache Kafka, Apache Spark, Apache Drill, Apache Hadoop

MapR CDP Offering	Licensed Hourly?	Customizable?	Enterprise Features	MapR Support	Description
Community Edition	No	Yes	Disabled	Not included	MapR XD, MapR Database, MapR Event Store For Apache Kafka, Apache Spark, Apache Drill, Apache Hadoop
BYOL (Bring Your Own License)	No	Yes	Available Separately	Available Separately	MapR XD, MapR Database, MapR Event Store For Apache Kafka, Apache Spark, Apache Drill, Apache Hadoop

Deploying a Cluster in AWS Using a Marketplace Offering
Explains how to deploy a cluster using the AWS Marketplace Offering.
Prerequisites for Using a Marketplace Offering

MapR marketplace offerings leverage predefined provisioning templates that simplify the process of provisioning the instance and installing the cluster. To run the predefined provisioning templates, you need to have the following permissions to create roles and policies in AWS:

- `iam:GetRole`
- `iam:PassRole`
- `iam:PutRolePolicy`
- `iam:CreateRole`
- `iam:AddRoleToInstanceProfile`
- `iam:CreateInstanceProfile`

If you do not have these permissions, consider using the MapR reference templates to deploy your cluster. See About the MapR Reference Templates for AWS on page 258.

Steps for Deploying Using a Marketplace Offering

Use these steps to deploy a cluster using one of the MapR Marketplace offerings for AWS:

1. Navigate to the MapR page in the AWS Marketplace. For more information, see MapR Offerings in the AWS Marketplace on page 252.
2. Click the title of the MapR offering that you want to use. AWS displays product overview and pricing information for you to review.
3. Set the region and EC2 instance type.
4. Click **Continue to Subscribe**. The **Subscribe to this software** page appears.
5. Click **Continue to Configuration**. The **Configure this software** page appears.
6. In the **Fulfillment Option** field, select **MapR Standard Cluster with VPC Support**.
7. Click **Continue to launch**. The **Launch this software** page appears.
8. Under **Choose Action**, select **Launch CloudFormation**.
9. Click **Launch**. The **Create Stack** page is displayed. Note that the **Specify an Amazon S3 Template URL** option is selected, and the URL for the MapR template is prefilled in the **URL** field.
10. Click **Next**. The MapR CloudFormation template is displayed.

11. Fill in the template as follows.

Filling in the Template Values

The Community Edition and bring-your-own-license (BYOL) offerings contain some template parameters that are not present in the Converged and Analytics offerings. These parameters are marked accordingly in the following steps.

1. In the **Specify Details** section, enter a stack name in the **stack name** field. The stack name cannot contain spaces.
2. In the **Parameters** section, enter a cluster name for the **clusterName** field.
3. Enter a password for the **clusterAdminPassword** field. This is the password you will use to log into the Control System or the MapR Installer.

   Note: For cloud deployments, the installer uses the Cluster Admin Password as the MySQL root password when it creates a MySQL database for services that require a database.
4. Enter the MapR Ecosystem Pack version for the **EEP** field. Typically, the EEP value is pre-filled for the MapR release you selected. For more information about supported EEPs, see EEP Support by MapR Core Version.
5. In the **provisioningTemplate** list, select from the list of auto-provisioning templates. For more information about the templates, see Auto-Provisioning Templates on page 5448. This parameter is provided only for the BYOL / Community and Converged offerings.
6. In the **nodeCount** field, specify the number of nodes in the cluster.
7. In the **installerOnitsOwn** field, select **true** to configure the MapR Installer on a node that is not part of the cluster. If **true** is selected, the installer is started on a t2.small instance. Select **false** to configure the installer on a node in the cluster. If you select **true**, the cluster will consist of the **nodeCount** plus another node for the installer. This parameter is provided only for the BYOL / Community offerings.
8. In the **Node Configuration** section, select an instance type from the **InstanceType** list. The MapR-supported instance types in AWS are:
   - m4.xlarge, m4.2xlarge, m4.4xlarge, m4.10xlarge, m4.16xlarge
   - m3.xlarge, m3.2xlarge
   - c4.2xlarge, c4.4xlarge, c4.8xlarge
   - c3.2xlarge, c3.4xlarge, c3.8xlarge
   - r4.large, r4.xlarge, r4.2xlarge, r4.4xlarge, r4.8xlarge
   - r3.large, r3.xlarge, r3.2xlarge, r3.4xlarge, r3.8xlarge
   - p2.xlarge
   - g2.8xlarge
   - Local Storage Instances (Use only local disks, not EBS):
     - i3.large, i3.xlarge, i3.2xlarge, i3.4xlarge, i3.8xlarge
     - d2.xlarge, d2.2xlarge, d2.4xlarge, d2.8xlarge

   For more information about instance types, see Amazon EC2 Instance Types.

last-updated: Apr 11, 2022

9. In the **useinstanceStore** field, set the value to **true** if the machine type supports instance stores (ephemeral disks) and you want to use only those disks. No EBS volumes will be attached. This parameter is provided only for the BYOL / Community offerings.

   Note: This field is enabled for d2.* and i3.* instance types. i3.* uses NVMe disks, which are added by default in AWS. If you use i3.* images, the option is implicitly true.

10. In the **diskCount** field, specify the number of disks per node.

11. In the **diskType** field, specify the disk type. For more information, see Amazon EBS Volume Types.

12. In the **diskSize** field, specify the disk size.

13. In the **AWS Infrastructure Configuration** section, enter a key pair in the **keyName** field. Use the key pair that you configured in "Creating a Key Pair."

14. In the **useExistingSubnet** field, leave this field empty if you would like a new VPC and subnets created. Specify a subnet ID if you want to use existing subnets.

    If you choose to use your own subnet:

    - The subnet must have access to the Internet.
    - The subnet must be able to communicate to all hosts within the subnet on all ports.
    - You must have a mechanism to connect with the hosts on the subnet.

15. In the **securityGroups** field, specify the security group IDs for an existing subnet if you specified a subnet ID for **useSubnet**. Leave this field empty to create a new VPC and subnets.

16. In the **assignPublicIP** field, set the value to **true** if you want to assign a public IP address to each node. Otherwise, set the value to **false**.

    Note: It is recommended that you enable `assignPublicIP` if you are not using your own subnet, and you are letting the template create the VPC and subnet.

17. In the **publicAccessCIDR** field, specify an IP address range if you want to restrict the IP address from which the stack can be accessed. This field applies only when the template creates the VPC and subnets.

18. Click **Next**. The **Options** section is displayed.

19. Click the arrow to expand the **Advanced** subsection. MapR templates do not require any specific values in this section. However, in the **Advanced** subsection, for debugging purposes it can be useful to set **Rollback on failure** to **No** so that resources are not deleted in the event of failure.

20. Click **Next**. The Review page allows you to review your selections. Click **Previous** if you need to change a selection.

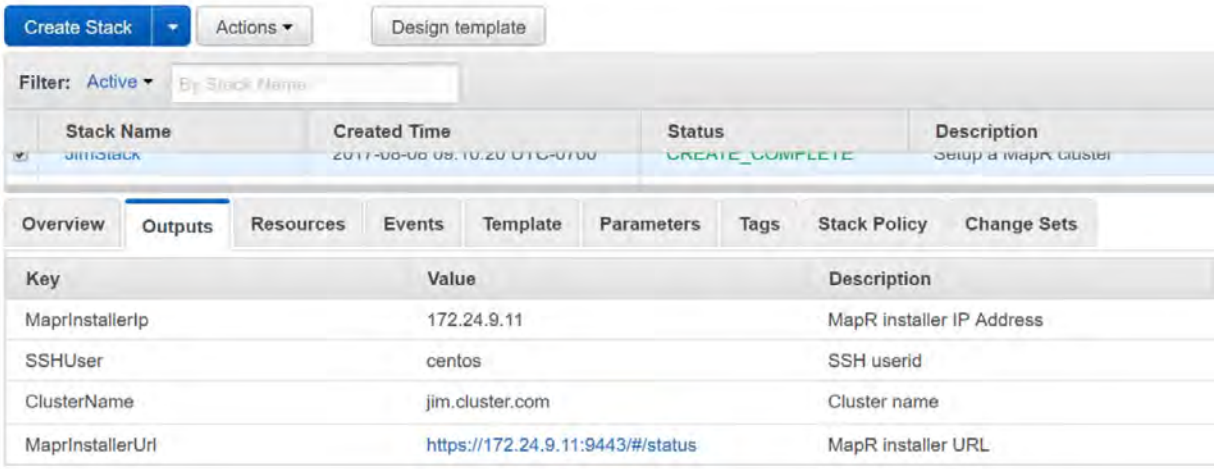
21. Otherwise, select **I acknowledge that AWS CloudFormation might create IAM resources**.

22. Click **Create** to start the process of stack creation. After a moment, the new stack is listed at the top of the CloudFormation console with a status of CREATE_IN_PROGRESS. To view the status of stack creation, select the stack, and click the **Events** tab.

    Note: Stack creation for a three-node cluster can take 15-20 minutes. Depending on the size of the cluster, stack creation can take longer in some cases.

last-updated: Apr 11, 2022

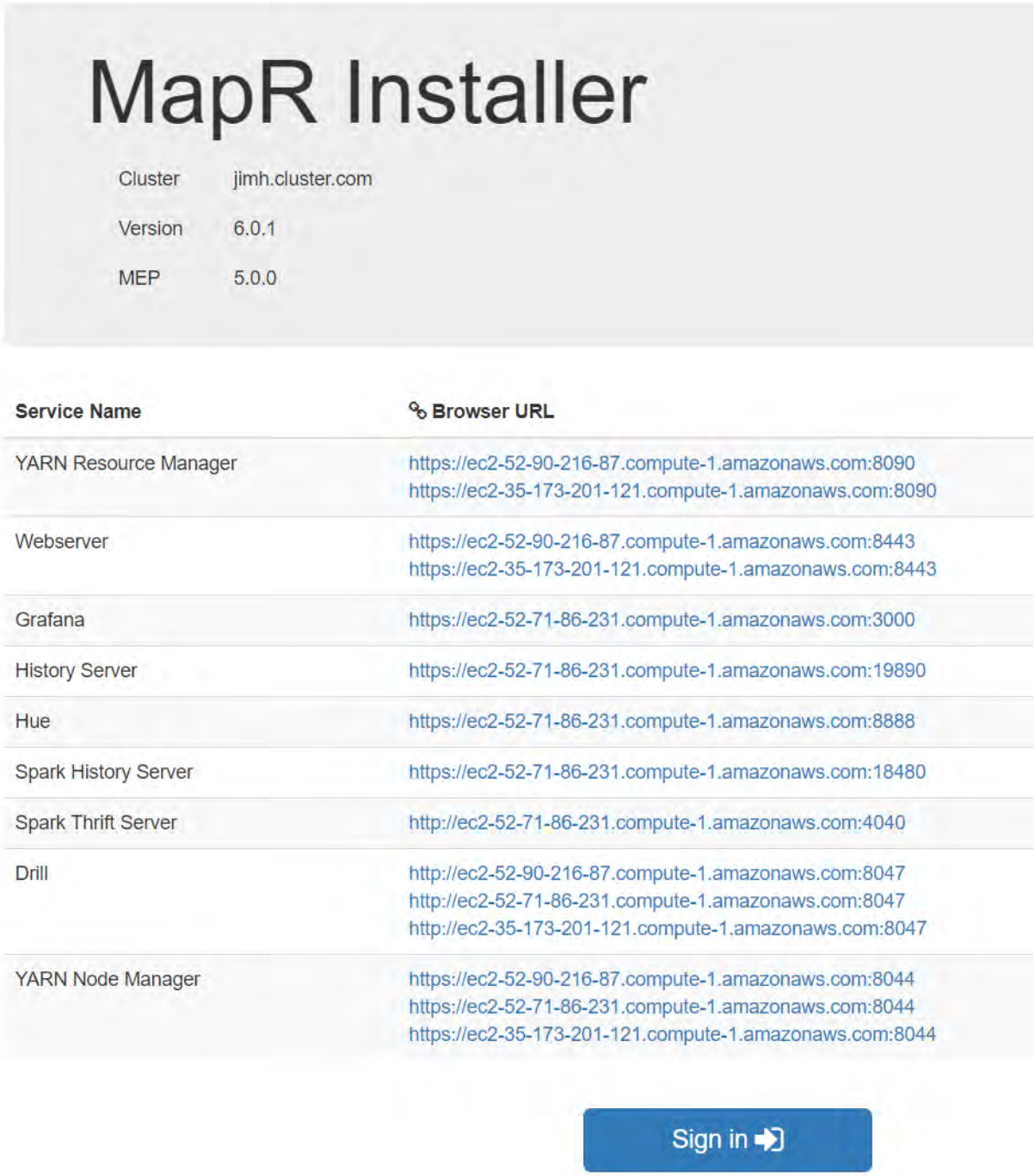
23. When the stack creation status shows CREATE_COMPLETE, the stack is created, and the cluster is installed. Click the **Outputs** tab to view the outputs. For example:



24. If stack creation generates an error, click the **Events** tab to get more information. You can expand the event message to see more information about the event.

25. If you are using an existing subnet, set up a secure connection to the subnet by using VPN or other means.

26. Once you are connected, you can click the **MaprInstallerUrl** link on the **Outputs** tab to start the web-based installer. The installer status page shows links to resources you can use to get more information about the cluster. To log on to the Control System, click one of the **Webserver** links.



Deploying a Cluster in AWS Using the MapR Reference Templates

If you need more flexibility than the marketplace offerings provide, you can use the MapR reference templates. The reference templates for AWS contain the CloudFormation parameters for deploying a MapR cluster. The reference templates allow you to:

- Create a customized CloudFormation template
- Limit the template choices
- Limit the machine types used

last-updated: Apr 11, 2022

- Customize AWS resources
- Use your own base image
- Use your own custom Stanza template

About the MapR Reference Templates for AWS

The MapR reference templates for AWS are sample CloudFormation templates that you can use to set up a MapR cluster. You can use the templates without modification, or you can customize them to suit your environment. Be sure to review the following information before using the reference templates. For more information about CloudFormation, see AWS CloudFormation.

The reference templates can be found on GitHub in folders organized by the MapR Installer version. To determine your MapR Installer version, see Checking the MapR Installer Version on page 5412.

This table describes the AWS reference templates:

Template Type	Template File Name	Function
Primary Template	`aws_cf_maprcluster_with_role.yml`	Creates a MapR cluster along with all the privileges and roles required by the MapR Installer. Use this template if you are privileged to create roles and policies in AWS.
Primary Template	`aws_cf_maprcluster.yml`	Creates a MapR cluster. This template requires the IAM role to be provided as an input. If you do not have the permission to create roles and privileges, ask your administrator to create the role for them. The `aws_cf_maprcluster_role.yml` template can be used for this purpose.
Primary Template	`aws_cf_maprcluster_ami.yml`	Creates a MapR cluster. For this template, you must provide an AMI ID and ssh user with SUDO privilege as inputs. The AMI can be any CentOS, Ubuntu, or SLES image on which the `mapr-setup.sh image prep` has not been run. The `root` disk (where `/opt/mapr` will be installed) must have a minimum of 128 GB, and swap space must be at least 10% of memory or less than 2 GB. Note that the CentOS, Ubuntu, or SLES version must be a version that is supported by MapR 5.2.2.
Primary Template	`aws_cf_maprcluster_with_cred.yml`	Creates a MapR Cluster. This template requires an AWS access key and an AWS secret key to be provided as an input. If you do not know the access key and secret key, contact your AWS administrator to obtain them.
Utility Template*	`aws_cf_maprcluster_role.yml`	Creates a role with the privileges necessary for the MapR Installer to create, modify, and release AWS instances. Once the role is created, you can pass the role name into the `aws_cf_maprcluster.yml` template.
Utility Template*	`aws_cf_maprcluster_nodes.yml`	Creates the launch configuration and auto-scaling group. If you are doing a custom deployment, you can customize this template to your needs.
Utility Template*	`aws_cf_vpc_openvpn.yml`	Deploys a VPC capable of supporting a MapR cluster. It can also, optionally, set up an OpenVPN Access Server. The OpenVPN Access Server supports two client connections for testing purposes. You can buy licenses for the Access Server from the OpenVPN website. Once the VPC is created, you can pass the subnet ID into any of the deployment templates.
Utility Template*	`aws_cf_maprcluster_vpc.yml`	Unused.
Utility Template*	`aws_cf_maprcluster_dsr.yml`	Unused

Template Type	Template File Name	Function
Sample Stanza Input File	`mapr-core.yml`	Provided as a convenience for users who need a script-based tool to install MapR software. See MapR Installer Stanzas on page 5503.

*Utility templates support the primary templates. You can modify them to suit your needs. Updated utility templates must be uploaded to s3 and referenced in your customized primary templates. The primary templates contain fully-qualified references to the utility templates.

Before Using the MapR AWS Templates

Before using the MapR AWS templates, make sure you have configured the following security prerequisites.

Configuring an AWS Account

Users who run MapR cloud formation templates must have certain minimum permissions. Contact your AWS administrator to ensure that the following AWS Policy is attached to your user account:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "autoscaling:*",
                "cloudformation:*",
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::*:role/*"
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::*"
        }
    ]
}
```

To create a more secure user, you can restrict the user to a specific role using `iam:PassRole` when you create the role. For example, if your role name is `maprinstaller`, the resource entry for iam role would be:

```
"Resource": "arn:aws:iam::*:role/maprinstaller"
```

For more information about the role used for the MapR Installer, see Creating a Role for the MapR Installer on page 260.

Creating a Key Pair

Key pairs allow you to prove your identity electronically in AWS. Key pairs are region specific. You need a key pair for every region in which you intend to deploy. To create a key pair:

1. In AWS, navigate to **Services > EC2 > Key Pairs**.
2. Click **Create Key Pair**.

3. Give the key pair a name, and click **Create**.
4. Save the key-file. You will need the key-file if you need to ssh into any of the hosts.

Creating a Role for the MapR Installer

During the provisioning process, you must supply an IAM role name that can be passed on to the MapR Installer. IAM roles allow instances to delegate permissions in the absence of AWS credentials.

To create the role for the MapR Installer:

1. Log on to the AWS Management Console.
2. In the list of **Services**, under **Security, Identity, and Compliance**, click **IAM**.
3. Click the **Policies** link.
4. Click **Create policy**.
5. Give the policy a name and a description.
6. In the **Policy Document** field, paste the following policy statement:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "autoscaling:Describe*",
                "autoscaling:CreateLaunchConfiguration",
                "autoscaling:DeleteLaunchConfiguration",
                "autoscaling:SuspendProcesses",
                "autoscaling:UpdateAutoScalingGroup",
                "cloudformation:DescribeStack*",
                "cloudformation:GetTemplate",
                "cloudformation:UpdateStack",
                "ec2:CreateKeyPair",
                "ec2:DeleteKeyPair",
                "ec2:ImportKeyPair",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceAttribute",
                "ec2:ModifyInstanceAttribute",
                "ec2:DescribeSubnets",
                "ec2:RunInstances",
                "ec2:StartInstances",
                "ec2:StopInstances",
                "ec2:CreateVolume",
                "ec2:AttachVolume",
                "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        }
    ]
}
```

7. Click **Create policy**.
8. Navigate to **Services > IAM > Roles**.
9. Click **Create new role**.

10. For the role type, select the **AWS Service Role > Amazon EC2**.
11. Filter by **Customer Managed**.
12. Select the policy created earlier.
13. Give the role a name and description.
14. Click **Create role**.

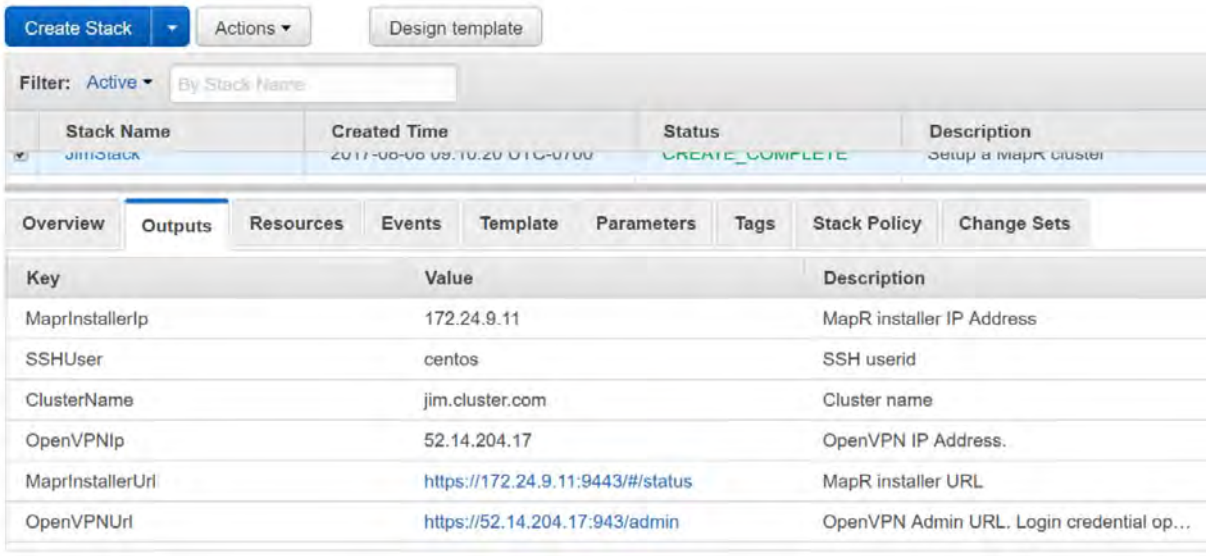
Later, during the provisioning process, you will provide the role name as an input to the reference template.

Running the MapR AWS Templates

You can use this procedure to install a MapR cluster in AWS using one of the MapR reference templates. You can run the templates as is or modify them and provide the modified template as an input to this procedure.

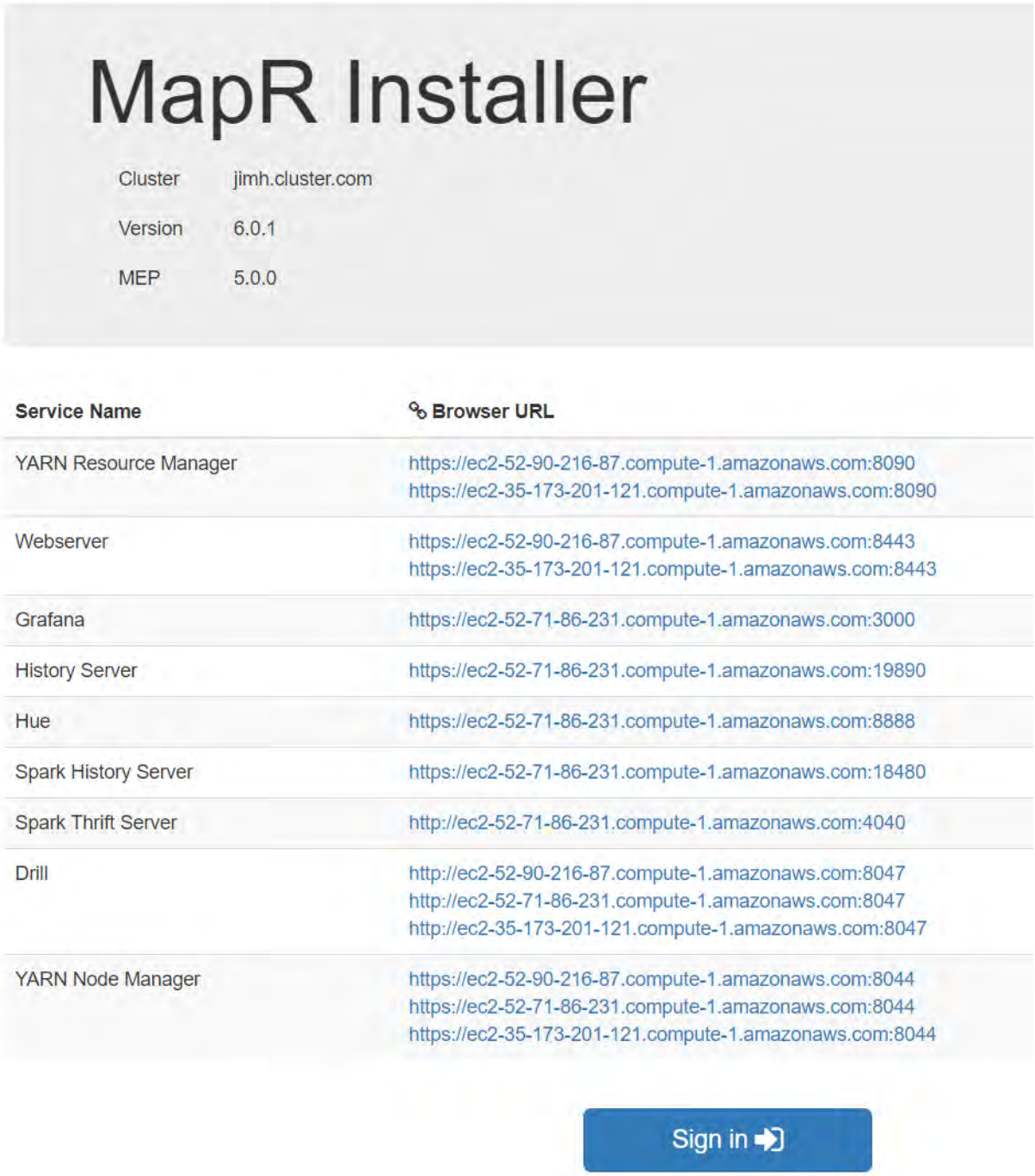
1. Sign in to the AWS Management Console.
2. Navigate to the **CloudFormation** console. The **Stacks** page is displayed.
3. Click **Create Stack**.
4. In the **Choose a template** section, select the option to upload a local template or specify the URL of a template on S3. For the locations of the MapR reference templates, see About the MapR Reference Templates for AWS on page 258.
5. Click **Next**. The MapR CloudFormation template is displayed.
6. Fill in the parameter values. See Parameter Information for AWS Reference Templates on page 263.
7. Click **Next**. The **Options** section is displayed. MapR templates don't require any specific values in this section. However, in the Advanced subsection, for debugging purposes it can be useful to set **Rollback on failure** to **No** so that resources are not deleted in the event of failure.
8. Click **Next**. The Review page allows you to review your selections. Click **Previous** if you need to change a selection.
9. Otherwise, select **I acknowledge that AWS CloudFormation might create IAM resources with custom names**.
10. Click **Create** to start the process of stack creation. The new stack is listed at the top of the CloudFormation console with a status of CREATE_IN_PROGRESS. Stack creation for a three-node cluster can take 15-20 minutes. Depending on the size of the cluster, stack creation can take longer in some cases. To view the status of stack creation, select the stack, and click the **Events** tab. You can expand the event message to see more information about an event.

    If there are errors during CloudFormation, see CloudFormation Troubleshooting on page 265.
11. When the stack creation status shows CREATE_COMPLETE, the stack is created. Click the **Outputs** tab to view the outputs. For example:



12. If you are using an existing subnet, set up a secure connection to the subnet by using OpenVPN or other means. See Using OpenVPN on page 279.

13. Once you are connected, you can click the **MaprInstallerUrl** to start the installer. The installer status page shows links to resources you can use to get more information about the cluster.



Parameter Information for AWS Reference Templates
Describes the parameters in the MapR reference templates for AWS.

Note that different templates can display different parameters and may present the parameters in a different order.

Parameter	Do this . . .
`Stack name`	Specify the stack name. MapR resources in AWS are grouped together as a stack that you create and delete as a single unit. The stack name cannot contain spaces.

last-updated: Apr 11, 2022

Parameter	Do this . . .
`keyName`	Enter the key pair that you configure in Creating a Key Pair on page 259.
`IAMInstanceProfile`	Enter the role you created in Creating a Role for the MapR Installer on page 260.
`clusterName`	Enter a name for the cluster.
`clusterAdminPassword`	Enter the password you will use to log into the Control System or the MapR Installer. This is the password for the UID `mapr`. **Note:** For cloud deployments, the installer uses the **Cluster Admin Password** as the MySQL root password when it creates a MySQL database for certain services that require a database.
`clusterAdminPasswordConfirm`	Enter the `clusterAdminPassword` again for verification.
`MEP`	Select the MapR Ecosystem Pack version.
`provisioningTemplate`	Select from the list of auto-provisioning templates. For more information about the templates, see Auto-Provisioning Templates on page 5448. Some offerings are preconfigured for provisioning and might not display this parameter.
`nodeCount`	Specify the number of nodes in the cluster.
`amiID`	Specify the Amazon Machine Image (AMI) ID. Leave this field empty to use a default AMI.
`sshUser`	Specify the ssh user with sudo privilege for the AMI you specified in `amiID`.
`instanceType`	Select from the list of AWS instance types. MapR CloudFormation templates support the following instance types: • m4.xlarge, m4.2xlarge, m4.4xlarge, m4.10xlarge, m4.16xlarge • m3.xlarge, m3.2xlarge • c4.2xlarge, c4.4xlarge, c4.8xlarge • c3.2xlarge, c3.4xlarge, c3.8xlarge • r4.large, r4.xlarge, r4.2xlarge, r4.4xlarge, r4.8xlarge • r3.large, r3.xlarge, r3.2xlarge, r3.4xlarge, r3.8xlarge • p2.xlarge • g2.8xlarge • Local Storage Instances (Use only local disks, not EBS) • i3.large, i3.xlarge, i3.2xlarge, i3.4xlarge, i3.8xlarge • d2.xlarge, d2.2xlarge, d2.4xlarge, d2.8xlarge For more information about instance types, see Amazon EC2 Instance Types.
`useInstanceStore`	Set this value to **true** if the machine type supports instance stores (ephemeral disks) and you want to use only those disks. No EBS volumes will be attached. **Note:** This field is enabled for d2.* and i3.* instance types. i3.* uses NVMe disks, which are added by default in AWS. If you use i3.* images, the option is implicitly true.
`diskCount`	Specify the disks per node.
`diskType`	Specify the disk type. For more information, see Amazon EBS Volume Types.

last-updated: Apr 11, 2022

Parameter	Do this . . .
`diskSize`	Specify the disk size.
`installerOnItsOwn`	Select **true** to configure the MapR Installer on a node that is not part of the cluster. When you select **true**, the installer is started on a t2.small instance. Select **false** to configure the installer on a node in the cluster. Note that if you select **true**, the cluster will consist of the **nodeCount** plus another node for the installer.
`useExistingSubnet`	Specify a subnet ID if you want to use an existing subnet. Leave this field empty if you would like a new BPC and subnets created. If you choose to use your own subnet: • The subnet must have access to the Internet. Hosts on that subnet must have transparent outbound access to the Internet with no additional configuration required. This means either an automatic VPN or public IP addresses. The system will connect to the Internet to update packages and software. • The subnet must be able to communicate to all hosts within the subnet on all ports. • You must have a mechanism to connect with the hosts on the subnet.
`securityGroups`	Specify the security group IDs for an existing subnet if you specified a subnet ID for **useExistingSubnet**. Leave this field empty to create a new VPC and subnets.
`installOpenVPN`	Select **True** if you want to install OpenVPN. For more information, see Using OpenVPN on page 279.
`openVPNuser`	Specify a user name for OpenVPN if you will use OpenVPN to connect to the cluster. For more information, see Using OpenVPN on page 279.
`openVPNpassword`	Specify a password for OpenVPN if you will use OpenVPN to connect to the cluster. For more information, see Using OpenVPN on page 279
`assignPublicIP`	If you want to assign a public IP address to each node, set this value to **true**. If you set the value to **false**, you must ensure that you have a way to connect to the private IP addresses. **Note:** It is recommended that you enable `assignPublicIP` if you are not using your own subnet, and you are letting the template create the VPC and subnet.
`publicAccess CIDR`	If you want to restrict the IP address from which the stack can be accessed, specify the IP address range in this field. This field applies only when the template creates the VPC and subnets.

CloudFormation Troubleshooting

If there are errors during CloudFormation:

- Check the stack creation events for more information. Click the **Events** tab to get more information. You can expand the event message to see more information about the event.
- Check the status of the auto scaling group. In EC2, click **Auto Scaling Groups**. Select a group. Then select the **Activity History** tab.
- Consult the AWS troubleshooting information.

Modifying the MapR AWS Templates

Suppose you want to customize the installation of a MapR cluster. You might want to predefine variables for your users or restrict the choices for some fields. You might want to define subnet information in a more detailed way. You can do this by modifying the MapR-provided reference templates.

last-updated: Apr 11, 2022

For more information about customizing a template, see About the MapR Reference Templates for AWS on page 258 and Working with AWS CloudFormation Templates

After customizing a template, you can run the template using the steps in Running the MapR AWS Templates on page 261.

Installing the AWS Instance and MapR Cluster Manually

For more information, see these MapR blogs:

- 9 Step to Deploying the MapR Converged Data Platform on AWS
- Manual AWS Deployment

Deleting an AWS Stack

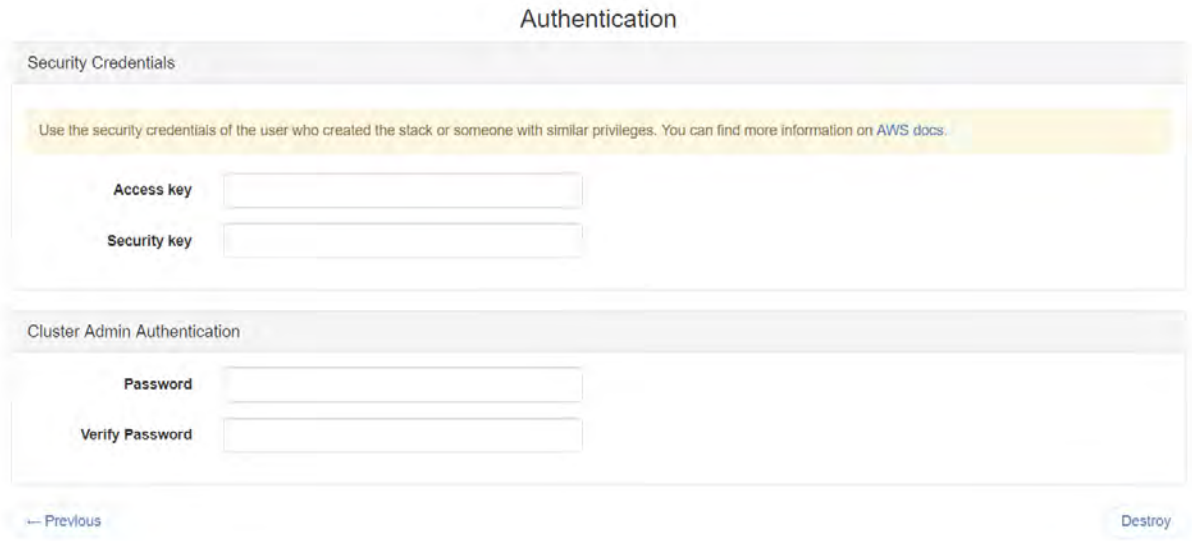
MapR resources in AWS are grouped together, making it easy for you to remove them. You can use the MapR Installer **Destroy** button to delete an AWS stack in one operation. The **Destroy** button appears when the MapR Installer detects cloud installations that were created using MapR scripts.

You can also use AWS menu commands to delete an AWS stack. Because the MapR stack is an Auto Scaling Group, deleting it using AWS menu commands requires some extra steps. First, you must disable certain AWS processes that protect against shutting down the stack. Then you must delete the stack.

Delete an AWS Stack Using the Destroy Button

To remove an AWS stack using the MapR Installer, you can use the **Destroy** button:

1. Log in to the MapR Installer.
2. On the status page, click the **Destroy** button. A confirmation box asks if you want to delete the MapR cluster.
3. Click **OK**. The **Authentication** page appears:



4. Enter your security credentials, and click **Destroy**. The Installer displays:

```
We will start to delete your cluster and all the resources associated
with it. Please check the AWS console to verify that all resources were
deleted successfully.
```

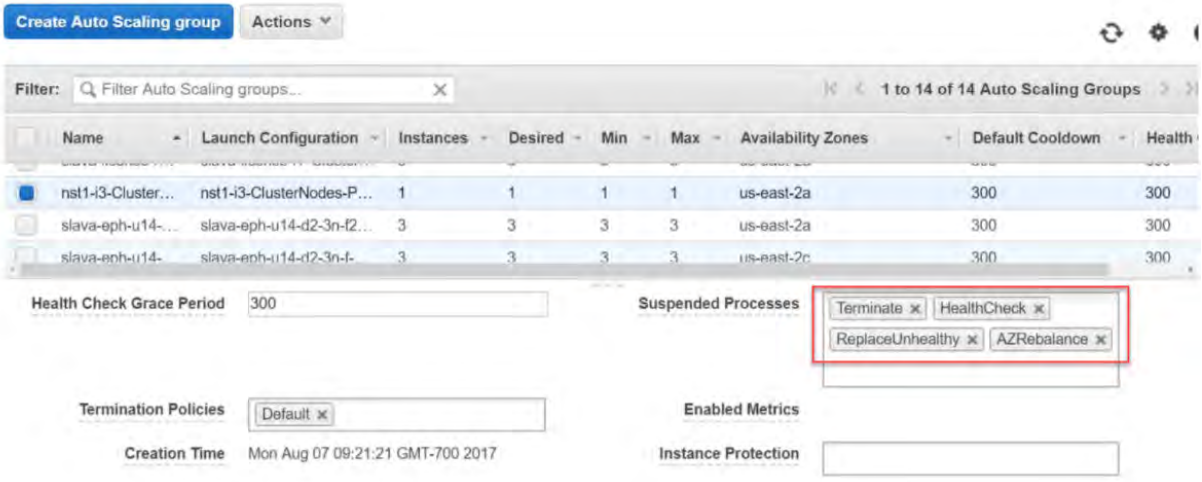
You can monitor the status of stack deletion in the AWS console.

Delete Suspended Processes

Use these steps if you want to delete your MapR stack using AWS menu commands. You do not need to use these steps if you use the MapR Installer **Destroy** button. To remove a MapR stack using AWS menu commands, you must delete suspended processes and then delete the stack.

1. Click **Services > EC2**.
2. In the left navigation pane, under **Auto Scaling**, click the link to **Auto Scaling Groups**.
3. In the list of Auto Scaling Groups, find and select the name of your stack.

   Tip: Type the first few letters of the stack name in the **Filter** field to find the stack.
4. Click the **Details** tab to select it.
5. Click **Edit**.
6. Scroll down to the **Suspended Processes** section:



7. Click the "x" in each of the suspended processes to delete it.
8. Click **Save**.

Delete the Stack

You do not need to use these steps if you use the MapR Installer **Destroy** button. To remove an AWS stack manually, you must delete suspended processes and then delete the stack.

1. Navigate to the CloudFormation section, and select your stack.
2. From the **Actions** menu, select **Delete Stack**.
3. To confirm the delete operation, click **Yes, Delete**.

Deploying MapR Clusters on Azure

This section describes cloud-deployment procedures for Microsoft Azure.

To view the MapR reference architecture for Azure, see the Reference Architecture For Azure Deployments.

For procedures supporting Amazon AWS, see Deploying MapR Clusters on AWS on page 252.

MapR Offerings in the Azure Marketplace

last-updated: Apr 11, 2022

To view the MapR offerings in Azure:

1. In a browser, navigate to the Azure portal.
2. Click **Marketplace**.
3. In the Search field, enter `MapR`, and press **Enter**.

The documentation in this section supports the following offering:

```
MapR Converged Data Platform v6
```

Prerequisites for Deploying a MapR Cluster in Azure

To deploy a MapR cluster in Azure, you must have an Azure subscription and sufficient subscription linits or quotas to deploy a cluster of the specified size. For more information, see:

- Azure Subscription
- Subscription Limits / Quotas

Deploying a Cluster in Azure Using the MapR Marketplace Offering

Explains how to deploy a cluster in Azure using the Marketplace Offering.

Before you can use a MapR marketplace offering to deploy a cluster in Microsoft Azure, you must select and configure certain parameters and gain access to the cluster.

Select the Offering

1. Navigate to the MapR page in the Azure Marketplace. For more information, see MapR Offerings in the Azure Marketplace on page 267.
2. Select **MapR Converged Data Platform v6**.
3. Click **Create**.

Configure Basic Settings

1. In the **Admin username** field, enter the user name for the cluster administrator (for example, `admin_mapr`). This is an operating-system account that will be created on all nodes in the cluster.
2. In the **Authentication type** field, select **Password** or **SSH public key**.
3. In the **Admin password** field, enter the password for the Admin user. The password must be between 12 and 72 characters long and must have three of the following:
   - One lower case character
   - One upper case character
   - One number
   - One special character that is not "\" or "-".

   Note: For cloud deployments, the installer uses the Cluster Admin Password as the MySQL root password when it creates a MySQL database for certain services that require a database.
4. Reenter the **Admin password**.
5. In the **Subscription** field, enter the subscription option that is appropriate for your installation.

6. In the **Resource group** field, either select the option to create a new resource group and give it a name, or select the option to use an existing group. The examples in this procedure use `myRG` as the resource group name.
7. In the **Location** field, select your region.
8. Click **OK**.

Configure the MapR Cluster Configuration

1. In the **MapR cluster name** field, enter a name for the cluster. Give your cluster a DNS-friendly name, such as `my.cluster.com`. The name should be unique across all of your clusters.
2. In the **Cluster admin password** field, enter the password for the UID `mapr`. The password must be between 6 and 30 characters, and must include at least one uppercase letter, one lowercase letter, and one numeric digit.

   Note: The UID `mapr` is not the same account as the Admin user account that you created in step 1 of Configure Basic Settings on page 268. The Admin user account is used for MapR software installation and can be used subsequently for SSH access and server administration. The UID `mapr` is the MapR administrator account, which is an additional OS account that is created on each node in the cluster and acts as the MapR system user.
3. Confirm the password.
4. In the **EEP version** field, ensure that the MapR Ecosystem Pack (EEP) version is selected.
5. In the **License type** field, select either the Enterprise or Community
6. In the **Provisioning template** field, select the auto-provisioning template. For more information, see Auto-Provisioning Templates on page 5448.
7. In the **Node count** field, enter the number of nodes.
8. Click **OK**.

Complete the Node Configuration

1. In the **Virtual Machine Size** menu, select a VM size, first clicking **View all** if you need to see the supported machines for the selected region.
2. In the **Disk count** field, enter the number of disks.
3. In the **Disk type** field, select between SSD and HDD disks.
4. In the **Disk size** field, enter the disk size in gigabytes.
5. Click **OK**.

Configure Global Access and Security

1. In the **Virtual network** field, create a new virtual network or choose an existing one.
2. Complete the **Subnets** values as needed:
   - Private subnet name
   - Private subnet address prefix
   - Public subnet name

last-updated: Apr 11, 2022

- Public subnet address prefix

3. In the **Public internet access CIDR** field, enter the public Internet access CIDR, or use * for all Internet traffic. This field enables you to set or restrict the outside Internet IP addresses that can access the cluster.
4. In the **Install OpenVPN** field, select **True** if you want to install OpenVPN. For more information about using OpenVPN to access this MapR cluster, see Using OpenVPN on page 279.
5. In the **OpenVPN login user** field, specify a user name for OpenVPN.
6. In the **OpenVPN login user password** field, specify a password for OpenVPN. The password must be at least 6 characters, no more than 30 characters, and must include at least one upper case letter, one lower case letter, and one numeric digit.
7. Confirm the password.
8. In the **Disk type** field, select **SSD** or **HDD** as the disk type.
9. Click **OK**.

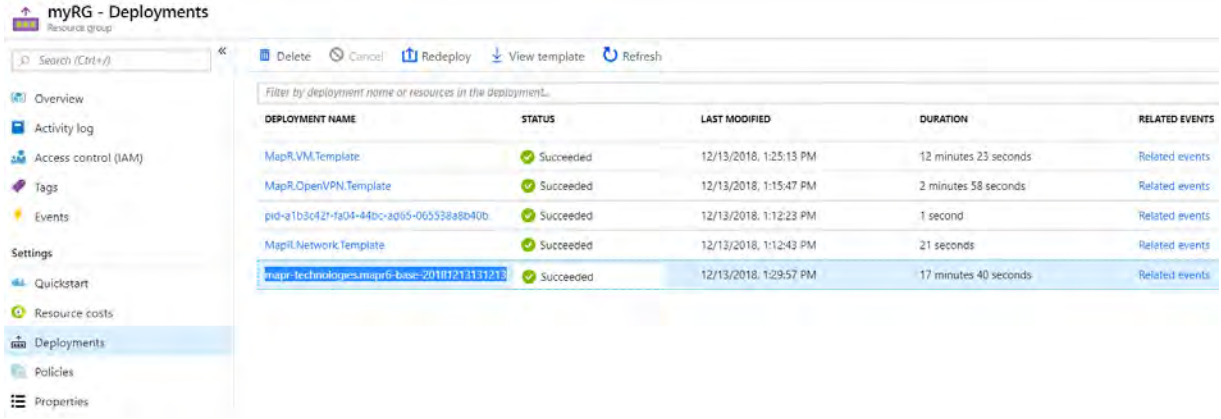
Complete the Summary Information and Purchase

1. Review the summary information and, if necessary, navigate back through the menus to make corrections or updates.
2. Click **OK**.
3. On the **Create** screen, review your contact information and then click **Create**.

Gain Access to the Cluster

1. Navigate to the **Resource groups** menu.
2. Click the link for your resource group (for example, `myRG`) to monitor the creation of resources.
3. Once the cluster is deployed, you can access it through a collection of URLs provided by the MapR Installer. These URLs may require a VPN connection, and you may need to use a tool such as OpenVPN to connect to the virtual machine. For more information, see Using OpenVPN on page 279.

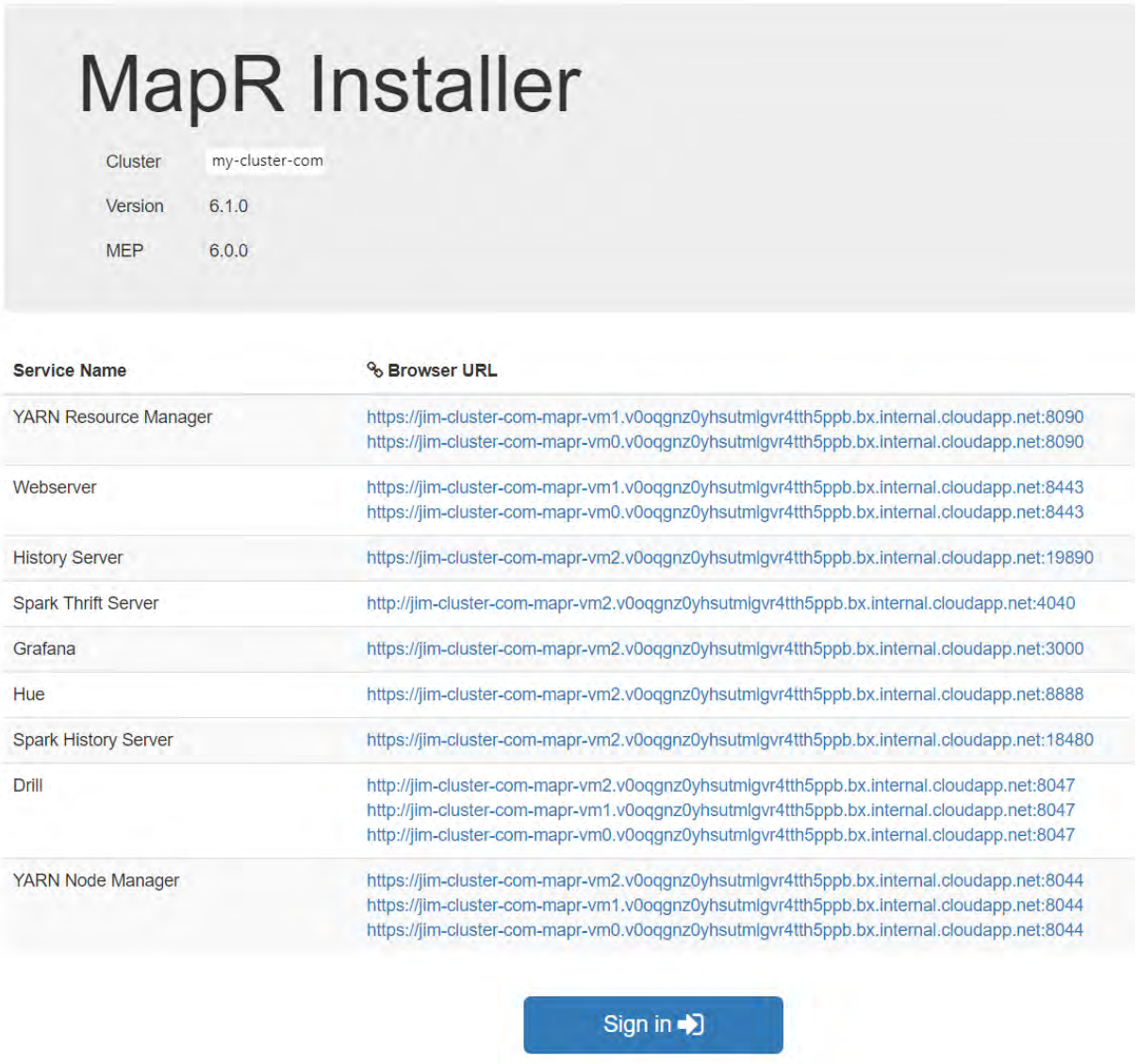
   If you did not choose to install OpenVPN in step 4 of Configure Global Access and Security on page 269, you must use another method or tool to connect to the virtual machine.
4. Once you are connected, use the **MAPRINSTALLERURL** to access the status page of the MapR Installer. To locate the **MAPRINSTALLERURL**, navigate to **Resource Groups > myRG > Deployments**, and click the cluster deployment name. For example:



The **Overview** page appears.

5. Click the **Outputs** tab to display the **MAPRINSTALLERURL** information.

6. Paste the **MAPRINSTALLERURL** into a browser to access the status page of the MapR Installer, which shows links to resources you can use to get more information about the cluster:



7. To log in to the Control System, click one of the **Webserver** links, using the password you specified for the UID `mapr` in Configure the MapR Cluster Configuration on page 269.

last-updated: Apr 11, 2022

About the MapR Reference Templates for Azure

The MapR reference templates for Azure contain the scripts and JSON templates for deploying a MapR cluster. You can use the reference templates without modification, or you can customize them to suit your environment.

The reference templates can be found on GitHub in folders organized by the MapR Installer version. To determine your MapR Installer version, see Checking the MapR Installer Version on page 5412.

This table describes the Azure reference templates:

Note: For a given JSON file name, the deployment name in Azure is similar, but does not necessarily match the file names below.

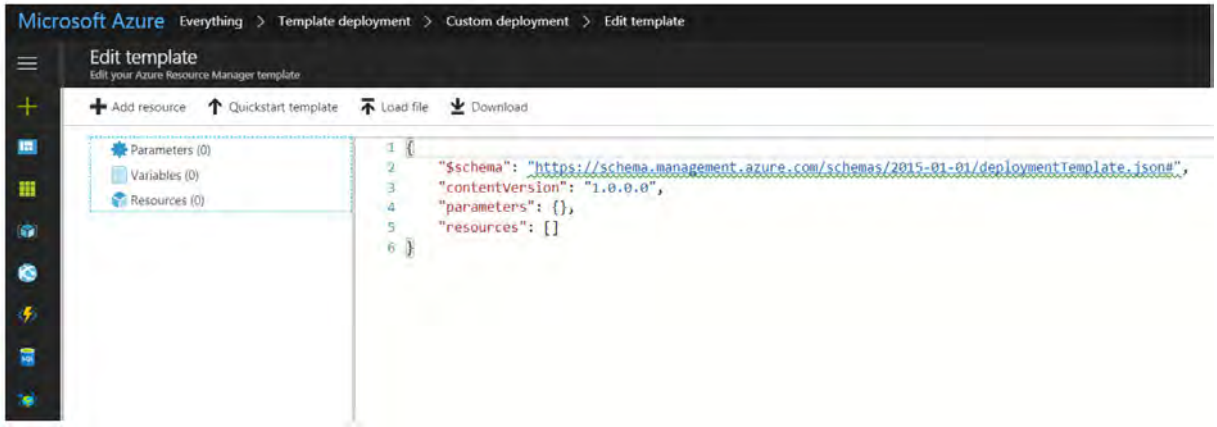
Template Type	Template File Name	Function
Primary Template	`mainTemplate.json`	References the child templates that do segregated work to create Azure resources and deploy MapR software. This is where most of the defaults can be changed to customize deployments.
Child Template	`createUiDefinition.json`	MapR marketplace GUI implementation. This is a standalone file and isn't used to deploy MapR software. This file generates JSON from user input and then sends the JSON to the `mainTemplate.json`.
Child Template	`network.json`	Child template to `mainTemplate.json`. Creates a Virtual Network and Subnet when the option to create a new is supplied.
Child Template	`openvpn.json`	Child template to `mainTemplate.json`. Creates all VM resources for the OpenVPN server then executes the `mapr-vpn.sh` script.
Child Template	`vm.json`	Child template to `mainTemplate.json`. Creates all VM resources in the cluster, including the installer node. For password-enabled and public-key-enabled OS users only.
Script	`install_openvpn_access_server.sh`	Installs OpenVPN core to an Ubuntu 17 image.
Script	`mapr-cloud-configure.sh`	Installs MapR to the newly created VMs. The `vm_xxx.json` files will send parameters to this file to initiate a Stanza MapR installation. This file is run only on the installer node.
Script	`mapr-init.sh`	Initialization script that is run on all nodes in the cluster (not just the install node). For example, this script sets passwords and sudo access.
Script	`mapr-vpn-configure.sh`	Alters the configuration of the core OpenVPN to match inputs from the user.
Script	`mapr-vpn.sh`	Decides if OpenVPN has been installed and configured already. Since the templates might be run several times by expanding the cluster, this file will not attempt to install or configure OpenVPN if it has been done already.
Sample Stanza Input File	`mapr-core.yml`	Basic Stanza file. This file has several placeholders that `mapr-cloud-configure.sh` changes. For example, changes to services that are global across all types of deployments can be changed here. Provided as a convenience for users who need a script-based tool to install MapR software. See MapR Installer Stanzas.

Running the MapR Azure Templates

last-updated: Apr 11, 2022

This procedure describes how to run the MapR reference template for Azure to deploy your MapR cluster.

1. Download the `mainTemplate.json` file from GitHub. For more information about the template, see About the MapR Reference Templates for Azure on page 272.
2. Navigate to the Azure portal, and click **New (+)**.
3. In the search field, type **Template deployment**.
4. Click **Create**.
5. Click **Build your own template in the editor**. In the **Edit template** screen, you can load a file or paste the contents of a template into the work area.



6. Paste the contents of the `mainTemplate.json` file into the **Edit template** working area.
7. Click **Save**. The template fields are displayed.
8. In the Basics section, enter your site-specific values for the following fields:
   - Subscription
   - Resource group (new or existing)
   - Location
9. In the **Settings** section, enter the username for the virtual machine in the **Admin Username** field. This is the username for the virtual machine.
10. In the **Admin Auth Type** field, enter the authentication type (password or SSH public key) for the virtual machine.
11. In the **Admin Password** field, enter the password for the virtual machine.
12. In the **Admin Public Key** field, enter the public key authentication for the virtual machine. If you entered an **Admin Password**, this field can be left blank.
13. In the **Cluster Name** field, enter your cluster name.
14. In the **Cluster Admin Password** field, enter the password corresponding to the MapR UID. For MapR images, the user defaults to `mapr`.

    Note: For cloud deployments, the installer uses the **Cluster Admin Password** as the MySQL root password when it creates a MySQL database for certain services that require a database.
15. In the **EEP Version** field, enter the MapR Ecosystem Pack (EEP) version.

last-updated: Apr 11, 2022

16. In the **Service Template** field, enter the auto-provisioning template corresponding to the services you need. For more information, see Auto-Provisioning Templates on page 5448.
17. In the **Node Count** field, enter the number of nodes.
18. In the **Instance Type** field, enter one of the following instances. MapR Azure templates support the following instance types. For more information about Azure instance types, see Sizes for Linux Virtual Machines in Azure.
    - Standard_D4s_v3
    - Standard_D8s_v3
    - Standard_D16s_v3
    - Standard_DS12_v2
    - Standard_DS13_v2
    - Standard_DS14_v2
    - Standard_DS15_v2
    - Standard_DS3_v2
    - Standard_DS4_v2
    - Standard_DS5_v2
    - Standard_E4s_v3
    - Standard_E8s_v3
    - Standard_E16s_v3
    - Standard_E32s_v3
    - Standard_F8s
    - Standard_F16s
    - Standard_GS2
    - Standard_GS3
    - Standard_GS4
    - Standard_L4s
    - Standard_L8s
    - Standard_L16s
    - Standard_L32s
19. In the **Disk Type** field, enter the disk type to use for managed disks (Premium_LRS, Standard_LRS).
20. In the **License Type** field, enter your license type.
21. In the **Disk Size** field, enter the size in gigabytes (GB) of each disk.

22. In the **Disk Count** field, enter the number of disks.
23. In the **Public Access CIDR** field, enter the public Internet access CIDR. (Use * for all Internet traffic.) This field allows you to set or restrict the outside Internet IP addresses that can access the cluster.
24. In the **Install Open VPN** field, enter `true` if you want to install OpenVPN to enable access to a private cluster. For more information about OpenVPN, see Using OpenVPN on page 279.
25. In the **Open Vpn User** field, enter the login user for OpenVPN.
26. In the **Open Vpn Password** field, enter the password for the OpenVPN login user. Use a strong password to protect your cluster.
27. In the **Vnet Name** field, enter the name of the virtual network.
28. In the **Vnet Resource Group** field, enter the resource group where the virtual network will be created.
29. In the **Vnet Address Prefix** field, enter the address prefix for the virtual network.
30. In the **Vnet New or Existing** field, enter `new` or `existing` to describe the virtual network.
31. In the **Vnet Private Subnet Name** field, enter the subnet name of the virtual network.
32. In the **Vnet Private Subnet Address Prefix** field, enter the private subnet address prefix of the virtual network.
33. In the **Vnet Public Subnet Name** field, enter the subnet name of the virtual network.
34. In the **Vnet Public Subnet Address Prefix** field, enter the public subnet address prefix of the virtual network.
35. In the **Vnet Public Subnet Start Address** field, enter the public subnet start address of the virtual network.
36. In the **Location** field, enter your region
37. Click the option to agree to the terms and conditions.
38. Click **Purchase**.
39. Navigate to the Azure resource groups page to monitor the creation of the cluster.

Note: Cluster creation for a three-node cluster can take 15-20 minutes. Depending on the size of the cluster, cluster creation can take hours in some cases.

Modifying the MapR Azure Templates

Suppose you want to customize the installation of a MapR cluster. You might want to predefine variables for your users or restrict the choices for some fields. You might want to define subnet information in a more detailed way. You can do this by modifying the MapR-provided reference templates.

For more information about customizing a template, see About the MapR Reference Templates for Azure on page 272 and Azure Resource Manager Template Deployment.

After customizing a template, you can run the template using the steps in Running the MapR Azure Templates on page 272.

What Happens During Azure Deployment

last-updated: Apr 11, 2022

The primary reference template for Azure (`mainTemplate.json`) is a starting point for all the child templates that create Azure resources. When the primary template is deployed, some resources are created in parallel. Here is what happens in a typical deployment:

1. The primary template is deployed.
2. If the option to create a new network was selected, the `MapR.Network.Template` is deployed.
3. `MapR.Network.Template` creates the virtual network and subnet(s).
4. `MapR.VM.Template` and, optionally, the `MapR.OpenVPN.Template` are started in parallel.
5. `MapR.VM.Template` creates all the resources needed to install and run MapR software. The number of resources varies depending on the Node Count you selected. This process creates:
   - Virtual machines
   - Network interfaces
   - Managed disks

   After these resources are completely created, the MapR installation starts via Stanza.
6. `MapR.OpenVPN.Template` creates one virtual machine, network interface, and managed disk. This Ubuntu machine has the OpenVPN server installed via script after these resources are completely created.

Azure Deployment Troubleshooting

If Azure reports a problem during deployment, follow these steps to determine where the problem occurred. Typically, the problem is either an Azure resource issue or a MapR installation issue.

1. Use these steps to learn more about the issue that needs troubleshooting:
   a. Log in to the Azure portal.
   b. In the left navigation pane, select **Resource groups**.
   c. Find the resource group to which the MapR installation was deployed.
   d. Click **Deployments**.
   e. Find the deployment that had a failure. Typically, an error on the Microsoft.Template will not be relevant, and there should be at least two deployments with a failure.
   f. If the error does not report an issue with `mapr-installer`, the issue is not MapR-related, and the error reported should indicate the Azure issue that caused the problem.
2. If the issue appears to be MapR-related:
   a. Use `ssh` to connect to the MapR installer node.

      Note: The MapR Installer is always created on the virtual machine that ends in the number 0. For example, `certtest1-cluster-com-mapr-vm0` would have the running installer, while `certtest1-cluster-com-mapr-vm3` (or any other number) would not.

   b. As `root`, go to `/var/lib/waagent/custom-script/download/0/`.
   c. Inspect the `stdout` and `stderr` files. These files contain the messages output from running the scripts that install the MapR software using Stanzas.

last-updated: Apr 11, 2022

d. If the problem cannot be identified, check the MapR Installer logs at `/opt/mapr/installer/logs`. For more information, see Logs for the MapR Installer on page 5453.

Installing the Azure Resource Group and MapR Cluster Manually

For more information, see the MapR blogs.

Deleting a MapR Cluster in Azure

You delete a MapR cluster in Azure by using the MapR Installer Destroy button or by deleting the resource group.

The **Destroy** button is present when the MapR Installer detects cloud installations that were created using MapR scripts. Note that any data contained in the cluster will be lost when you use the **Destroy** button or delete the resource group.

Delete an Azure Cluster Using the Destroy Button

To remove an Azure cluster by using the **Destroy** button:

1. Log in to the MapR Installer.
2. On the status page, click the **Destroy** button. The MapR Installer prompts you for security credentials and the cluster admin password.
3. Enter your security credentials and MapR cluster admin password, and click **OK**.

Deleting an Azure Cluster Using the Azure Console

You can also delete a MapR cluster in Azure by deleting the resource group.

1. In the Azure portal, navigate to the **Resource groups** page.
2. Select the resource group to be deleted.
3. Right-click the resource group, and select **Delete resource group**.
4. Confirm the resource to be deleted by typing the resource group name.
5. Click **Delete**.

Customizing Your Deployment by Using the MapR Installer Web Interface

If you use a MapR offering or reference template, the cluster is installed for you according to the parameters you specified. Another way to customize the resources and services provisioned to a MapR cluster on AWS or Azure is to create a stack or resource group with only the MapR Installer. Once the stack or resource group is created, you can use the MapR Installer web interface to finish configuring the cluster.

This procedure is the same for AWS and Azure:

1. Choose one of these procedures to create an AWS stack or an Azure resource group:
   - Running the MapR AWS Templates on page 261
   - Running the MapR Azure Templates on page 272
2. When you run the template, fill in the MapR template values as described in one of previous topics:

   Important: In the **provisioningTemplate** field (AWS) or **Service Template** field (Azure), select **Custom-Configuration** for the Auto-Provisioning Templates on page 5448. Selecting **Custom-Configuration** instructs the template to install only the MapR Installer and related software so that you can finish the cluster installation on your own.

last-updated: Apr 11, 2022

3. After stack creation (AWS) or resource group creation (Azure), log on to the MapR Installer, and use the web interface and on-screen user assistance to configure the resources and services you need in the cluster.

   Note: For cloud-based installations, the MapR Installer displays **Authentication** screens with cloud-specific parameters that are different from the **Authentication** screens for on-premise clusters. Use the tooltips on these screens to understand the authentication information that you need to provide for your cluster.

Creating Custom Images

The `mapr-setup.sh` script allows you to build custom Amazon Machine Images (AMIs) or Azure VM images. You can use these images to create more instances of the original image. With a couple of exceptions, the steps for creating custom images are the same for AWS and Azure.

To create a custom image:

1. Create the instance from which you want to build the image.
2. Download or copy `mapr-setup.sh` to the instance. For more information, see MapR Installer.
3. Prep the image. This step adds MapR software to the image. For example:

   ```
   ./mapr-setup.sh image prep --version 5.2.2
   ```

4. Respond to the questions in the script.
5. (Optional): Install other software as needed on the image.
6. (Azure only): Finalize the image:

   ```
   ./mapr-setup.sh image finalize azure
   ```

7. Create the image by using the `image create` command:

   Note: (Azure only): The image create command must be run off the VM that you want to make an image out of. The command can be run anywhere the Azure CLI is installed or anywhere the MapR Installer 1.6 setup script is installed.

   - **AWS example**

     ```
     ./mapr-setup.sh image create
     ```

   - **Azure example**

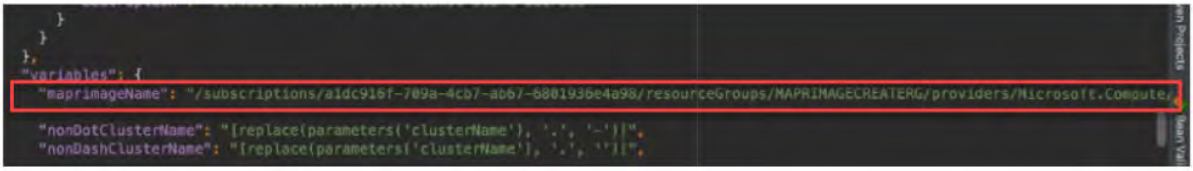
     In this example, the script looks in the `MapRImageCreateRG` resource group for a machine called `MyVirtualMachine` and creates an image called `MapRImage2`:

     ```
     ./mapr-setup.sh image create azure -u user@mapr.com -g
     MapRImageCreateRG -n MyVirtualMachine -i MapRImage2
     ```

8. (AWS only): Once the image is created, you can see the image ID in the EC2 console.
9. (Azure only): Once the image is created, you can see the image in the list of resource groups. You can copy the image ID from the script:

Note: Another way to obtain the Azure image ID is to navigate to the Azure console and check the resource ID for the new image:

10. (AWS only): Update the AMI ID in the `RegionMap` of your CloudFormation template.
11. (Azure only): Update the resource ID in the `maprimageName` field in the ARM template.



12. (AWS only): Run the CloudFormation template using the steps in Running the MapR AWS Templates on page 261.
13. (Azure only): Run the ARM template using the steps in Running the MapR Azure Templates on page 272.

Adding Nodes to a Cloud-Based Cluster

You can increase the capacity of a cloud-based MapR cluster by adding nodes using the MapR Installer or MapR Installer Stanzas. See Extending a Cluster by Adding Nodes.

Shutting Down and Restarting a Cloud-Based MapR Cluster

Shutting down and restarting a cloud-based MapR cluster is different from shutting down and restarting an on-premise cluster. For on-premise clusters, using the MapR Installer **Shutdown** button stops MapR services but leaves the nodes running. For cloud-based clusters, the shutdown operation halts all nodes in the cluster. Shutdown and restart are the same for AWS and Azure clusters.

For shutdown instructions, see one of these topics:

- Shutting Down a Cluster Using the MapR Installer Shutdown Button on page 5446
- Shutting Down a Cluster Using an Installer Stanza Command on page 5514

Restarting a Cloud-Based Cluster

In EC2 or the Azure portal, find the servers that are in the shutdown state, and restart them. Note that it takes a few minutes for the nodes to discover each other.

Using OpenVPN

You can use OpenVPN to create a secure tunnel between your workstation and the private subnet for the MapR cluster.

Using OpenVPN with AWS

To download the OpenVPN desktop client:

1. On the **Outputs** tab, copy the **OpenVPNUrl**. For example:

   ```
   https://52.14.236.170:943/admin
   ```

2. Paste the URL into a browser, but remove the `admin`. For example:

   ```
   https://52.14.236.170:943/
   ```

   OpenVPN displays download links for the OpenVPN Connect app.

3. Click the appropriate link to download the app for your workstation. You need to do this only once.

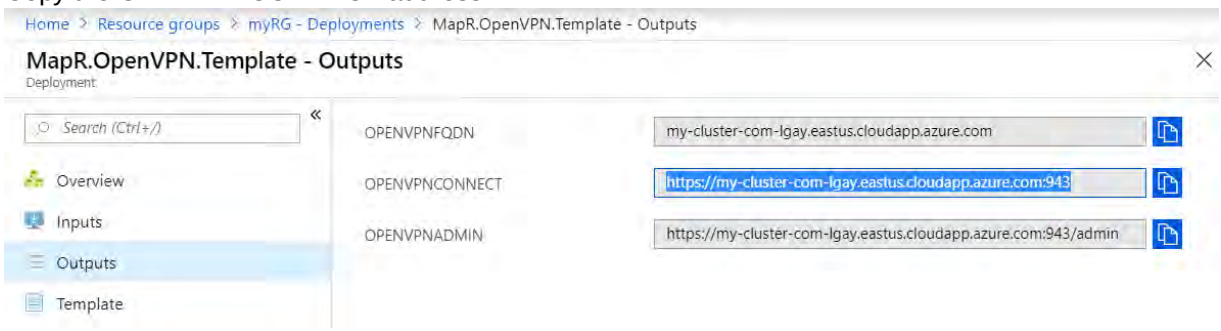
To connect to the cluster using OpenVPN:

1. Use the OpenVPN **Connect** command to display the **Login** screen.
2. In the **Server** field, paste the **OpenVPNIp** value from the **Outputs** tab.
3. In the **Username** and **Password** fields, enter the OpenVPN username and password you specified in the reference template.

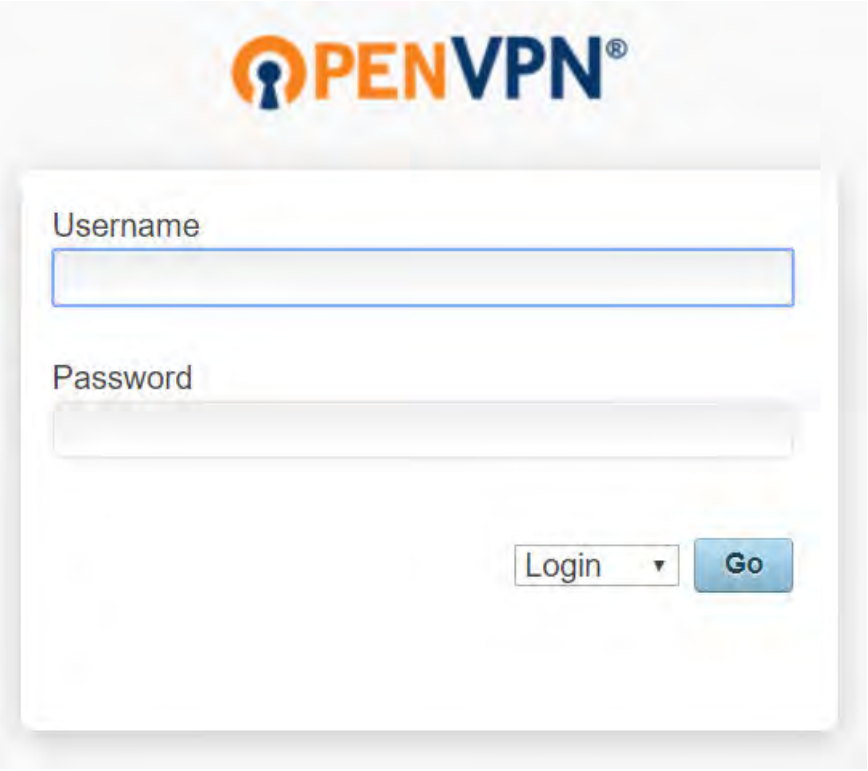
Using OpenVPN with Azure

For Azure clusters, you can use the OPENVPNCONNECT URL and your OpenVPN login user name and password to establish a secure connection to the OpenVPN server. Then you can download and install the OpenVPN client and use the client to connect to the cluster. To locate the Open VPN URL:

1. Navigate to the **Resource groups** menu, and click the link for your Resource group.
2. Click **Deployments > MapR.Open.VPN Template > Outputs**.
3. Copy the OPENVPNCONNECT address:



4. Paste the OPENVPNCONNECT address into a browser. The OpenVPN login dialog appears:



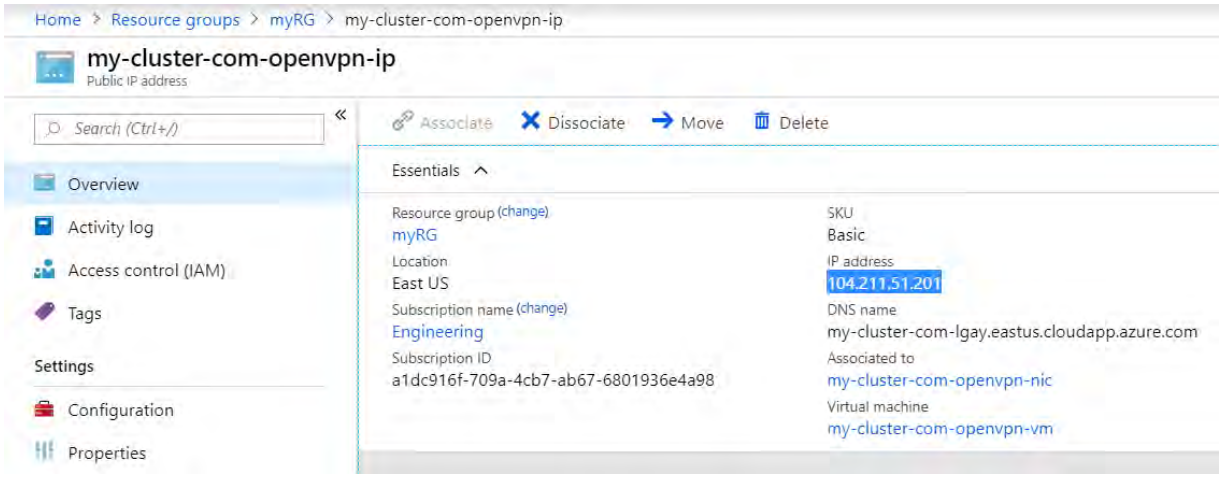
5. Log in using the OpenVPN login user name and password that you specified in Configure Global Access and Security on page 269. The OpenVPN download page appears:

last-updated: Apr 11, 2022

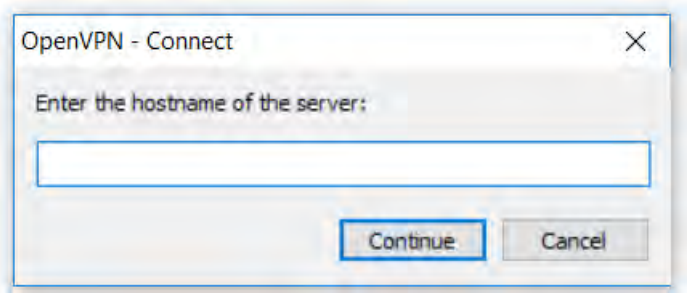
6. Click the appropriate link to download the OpenVPN installation file to your workstation.
7. Double-click the installation file to install the OpenVPN client.

To connect to the cluster using the OpenVPN public IP address:

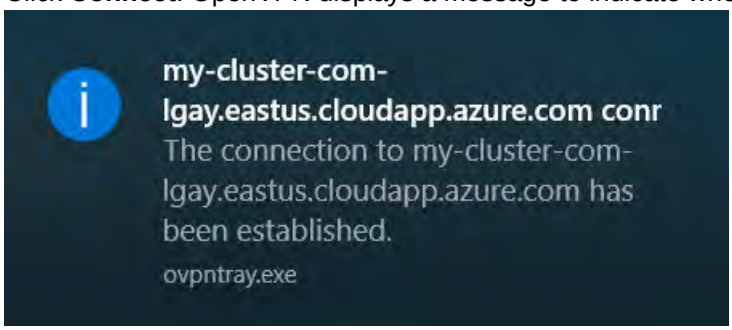
1. Locate the OpenVPN public IP address by navigating to **Resource Groups > myRG**.
2. In the list of resources, click the link for the public IP address. For example: `my-cluster-com-openvpn-ip`. IP address information is displayed:



3. In the system tray, open the OpenVPN Connection application, and select **Connect**. The OpenVPN Connect dialog appears:



4. Enter the OpenVPN IP address into the hostname field, and click **Continue**.
5. In the **Username** and **Password** fields, enter the OpenVPN username and password you specified in the reference template:Locate the OpenVPN public IP address by navigating to **Resource Groups > myRG**.
6. Click **Connect**. OpenVPN displays a message to indicate when you are connected:



Known Issues for Cloud-Based Clusters

Before installing a MapR cluster in the cloud, be sure to review these known issues:

Issue	Description
IN-2012	An issue in the Java version used by the MapR Installer prevents the Installer from supporting a domain suffix that starts with a number. Azure generates a random string for the domain suffix, so this problem can occur intermittently. **Workaround** The `mapr-setup.sh` script checks for the DNS Suffix during installation and fails with an error similar to the following when the suffix begins with a number. If you see this error message, create new nodes and retry the installation with a DNS Suffix that does not start with a number. `ERROR: The DNS Suffix 14x325z0ek4ezp1skptqsztkyg.dx.internal.cloudapp.net starts with a number which is incompatible with OpenJDK keytool application. This cluster is invalid. Recreate the cluster again; DNS Suffix is 14x325z0ek4ezp1skptqsztkyg.dx.internal.cloudapp.net; First character of DNS Suffix is: 1\n*.*`
IN-860	If you create a cluster using a MapR marketplace offering for Amazon AWS and use ephemeral drives, and you then shut down the cluster, most MapR services cannot be restarted. This happens because the disks are new, and the MapR cluster is not aware of the ephemeral drives. **Workaround:** Do not use clusters with ephemeral drives, or do not shut down the cluster. When the cluster is not in use, terminate it.
IN-974	If you request public IP addresses for the MapR Installer nodes in Amazon AWS, the IP addresses can change after a shutdown and restart of the cluster, and the MapR Installer status page continues to display the old public IP addresses. **Workaround:** None.

Links to AWS and Azure Documentation

Cloud-computing platforms add new features and capabilities almost daily.

For more information about using the Amazon or Microsoft cloud platforms:

- Amazon AWS Documentation
- Microsoft Azure Documentation

Installing Metering

This section describes how to install Metering by using the MapR Installer or manual steps.

Installing Metering Using the MapR Installer

This section describes how to install Metering by using the MapR Installer.

Metering is part of the `mapr-apiserver` package and is installed by default when you use the MapR Installer to install MapR 6.1 or later. Metering requires a minimal level of metrics collection to be enabled. On the **Monitoring** page of the MapR Installer web interface, you can choose between two configuration options for metrics collection:

- A "full collection configuration" installs collectd on all nodes in the cluster with a configuration suitable for monitoring all available node metrics. This configuration is the default setting.
- A "minimum collection configuration" installs collectd on all nodes in the cluster with a configuration suitable for supporting metering metrics collection (but not MapR Database table metrics). Note that when you choose this option, table metrics are not visible in the MapR Control System.

last-updated: Apr 11, 2022

For more information about Metering, see Metering on page 1320. To get started using the MapR Installer, see MapR Installer.

Installing Metering Using Manual Steps

This section describes how to install Metering by using manual steps.

For MapR 6.1 and later releases, no special steps are needed to install Metering. Following the manual installation steps in Installing without the MapR Installer on page 141 installs Metering. Note these considerations:

- Metering is included in the `mapr-apiserver` package and is installed when you install or upgrade to MapR 6.1.
- Metering requires a minimal level of metrics collection to be enabled. You must install collectd and OpenTSDB. See Step 8: Install Metrics Monitoring on page 162. Grafana does not need to be installed for metering and is optional for metrics monitoring in general.
- The manual steps for configuring metrics collection configure collectd to monitor all available node metrics. You can configure collectd to collect only metering metrics (and omit MapR Database table metrics) by using the `-EPcollectd -minimal` option with `configure.sh`. For example:

  ```
  /opt/mapr/server/configure.sh -C nodeA -Z nodeB -EPcollectd -minimal
  ```

  To change from the minimal collection configuration back to the full collection configuration, use the `-EPcollectd -all` option with `configure.sh`.

To use Metering, see Metering on page 1320.

Upgrading MapR Core or EEP Components

Depending on your current configuration, you may choose to upgrade the release version (core), ecosystem components, clients, or monitoring components.

Getting Started with Upgrades

Most upgrades involve moving from one version of core to another version and upgrading MapR Ecosystem Pack (EEP) components at the same time. To learn about the different methods you can use to execute this kind of upgrade, see Upgrade Workflows on page 285.

Upgrading an EEP

To upgrade an ecosystem component, you must upgrade the EEP to which it belongs. EEPs can be upgraded when you upgrade core or independently of a core upgrade. If your core release supports multiple EEP versions, you might be able to upgrade to a different EEP without upgrading core. See EEP Support and Lifecycle Status on page 5531.

If a component is supported by your current EEP, you can add the component at any time by using manual steps or the **Incremental Install** function of the MapR Installer. See Upgrading MapR Ecosystem Packs on page 335.

Upgrading MapR Clients

The MapR client is a part of core. You upgrade client nodes after you upgrade the cluster nodes but before enabling new features. See Planning Upgrades to MapR Clients on page 301.

last-updated: Apr 11, 2022

Upgrading MapR Monitoring

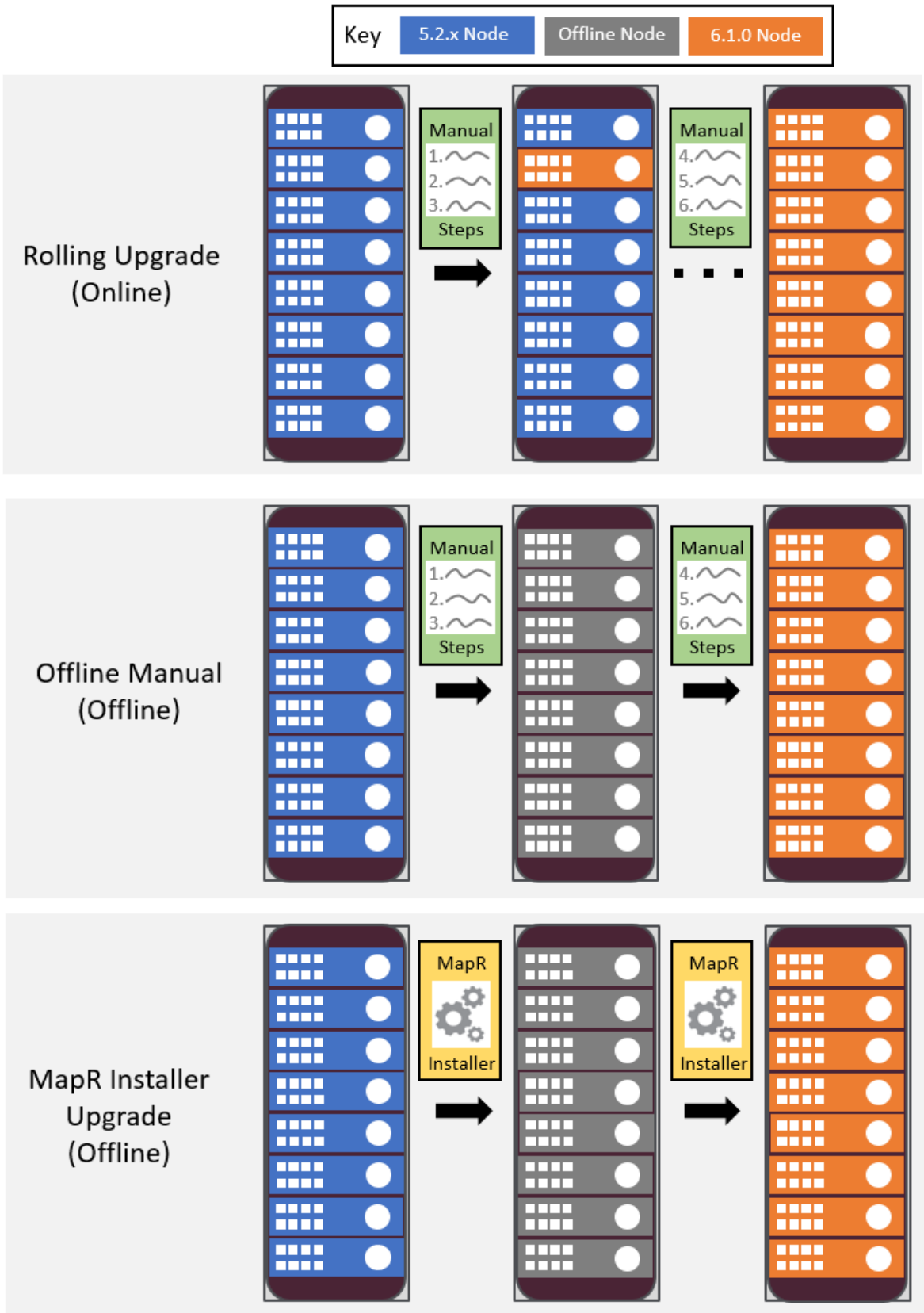
MapR Monitoring components are available as part of the MapR Ecosystem Pack (EEP) that you selected for the cluster. Once MapR Monitoring is installed, you can upgrade MapR Monitoring components as part of the EEP upgrade process. You can upgrade a EEP without having to upgrade core, provided the EEP you plan to upgrade to is supported by the current release. See Upgrading MapR Ecosystem Packs on page 335.

Upgrade Workflows

This section describes three common methods for upgrading from one MapR core release to another. The workflows in this section introduce you to the high-level steps for each method and provide links to pages showing more detail.

To view the basic steps for a specific workflow, click one of the workflows in the following illustration:

last-updated: Apr 11, 2022



1. Click here to view the workflow
2. Click here to view the workflow

last-updated: Apr 11, 2022

3. Click here to view the workflow

Workflow: Manual Rolling Upgrade
This page summarizes the steps for upgrading from MapR 5.2.x to MapR 6.1.x by using a manual rolling upgrade. In this workflow, the cluster to be upgraded is ; after the upgrade, the cluster will continue to be .

Manual Rolling Upgrade Summary

In a manual rolling upgrade, you upgrade the software one node at a time so that the cluster as a whole remains operational throughout the process. The manual rolling upgrade requires you to:

1. Perform pre-upgrade checks.
2. Perform a rolling upgrade of core.
3. Verify that all use cases are functional on the cluster.
4. Upgrade the EEP to 6.0.0.
5. Merge custom configuration settings.
6. Enable MapR 6.1.x features.
7. Perform post-upgrade checks.

The workflow later in this section provides more detail to help you get started with a manual rolling upgrade.

Considerations for Manual Rolling Upgrades

Before performing a manual rolling upgrade, note these considerations:

- Rolling upgrades only upgrade core packages, not ecosystem components. A rolling upgrade of ecosystem components is not supported.
- If you choose to do a rolling upgrade on a cluster with core and ecosystem components, the ecosystem components will continue to work during the rolling upgrade as long as the ecosystem components are not updated. If you choose to upgrade core and ecosystem components together, the ecosystem components might not function properly during the upgrade process.
- You can only perform a manual rolling upgrade from the following MapR versions:
  - MapR 5.2.x with EEP 3.0.1 or later
  - MapR 6.0.0 with EEP 4.0.x or 4.1.x
  - MapR 6.0.1 with EEP 5.0.x
  - MapR 6.1.0 with EEP 6.x.x
- If your cluster is running MapR 5.2.x but your EEP is not 3.0.1 or later (for example, you have EEP 2.0.3), you must upgrade to EEP 3.0.1 or later before you can do a rolling upgrade to MapR 6.1.x. See Upgrading MapR Ecosystem Packs on page 335.
- If you want to use a manual rolling upgrade from a pre-5.2 release to MapR 6.1.x, you must do multiple rolling upgrades, one of which upgrades your cluster to a MapR / EEP combination that can upgrade directly to MapR 6.1.x.
- After upgrading MapR Core to MapR 6.1.x, you must upgrade ecosystem components to EEP 6.0.0 or later, and this must be done before you enable MapR 6.1.x features.

Manual Rolling Upgrade Workflow

High-Level Steps	Detailed Information (review all items unless noted otherwise)
1. Understand MapR Core/MEP Dependencies 	• Operating System Support Matrix on page 5522 • Operating System Support Matrix (MapR 5.x) • EEP Support and Lifecycle Status on page 5531 • Component Versions for Released EEPs
2. Plan for the MapR Core Upgrade 	• Upgrading and Your License on page 295 • Installation and Upgrade Notes (MapR 6.0) • Installation and Upgrade Notes (MapR 6.0.1) • Installation and Upgrade Notes (MapR 6.1.0) on page 39 • Installation and Upgrade Notes (MapR 6.1.1) on page 74 • Planning Your MapR Core Upgrade on page 298
3. Plan for the EEP Upgrade 	• Planning MapR Ecosystem Pack (EEP) Upgrades on page 335
4. Perform Pre-Upgrade Steps for MapR Core 	• Preparing to Upgrade MapR Core on page 303
5. Prepare to Upgrade EEP Components 	• Preparing to Upgrade the MapR Ecosystem Pack on page 335
6. Set up Repositories 	• Setting Up Repositories on page 309
7. Perform the Manual Rolling Upgrade 	• Manual Rolling Upgrade Description on page 319 • Manual Rolling Upgrade Procedure on page 320

last-updated: Apr 11, 2022

High-Level Steps	Detailed Information (review all items unless noted otherwise)
8. Upgrade the EEP Components	• Upgrading the MapR Ecosystem Pack Without the MapR Installer on page 350
9. Perform Post-Upgrade Steps for EEP	• Finishing the MapR Ecosystem Pack Upgrade on page 372
10. Perform Post-Upgrade Steps for MapR Core	• Restart and Check Cluster Services • Manually Update Configuration Files • Upgrade Clients • Enable New Features • Update Hadoop Configuration File
11. Install Additional MapR Core Features	• Installing Additional MapR Core Features on page 335
12. Secure the Upgraded Cluster	• Securing the Upgraded Cluster on page 335

Workflow: Offline Manual Upgrade

This page summarizes the steps for upgrading from MapR 5.2.x to MapR 6.1.x by using the offline manual upgrade. In this workflow, the cluster to be upgraded is nonsecure; after the upgrade, the cluster will continue to be nonsecure, but can be secured by using manual steps.

Offline Manual Upgrade Summary

In an offline manual upgrade, cluster processes and the jobs that depend on them are stopped on all nodes so that packages can be updated. The offline upgrade process is simpler than a rolling upgrade, and usually completes faster. The offline manual upgrade requires you to:

1. Perform pre-upgrade checks.
2. Shut down the cluster.
3. Upgrade core to 6.1.x.
4. Upgrade the EEP to 6.0.0.
5. Merge custom configuration settings.
6. Start the cluster and perform post-upgrade checks.
7. Enable core 6.1.x features.

last-updated: Apr 11, 2022

The workflow later on this page provides more detail to help you get started with an offline manual upgrade.

Considerations for Offline Manual Upgrades

Before performing an offline manual upgrade, note the following considerations:

Note: After upgrading core to release 6.1.x, you must upgrade ecosystem components to an EEP that is compatible with your MapR 6.1.x release. This must be done before you enable MapR 6.1.x features. To determine the compatible EEPs, see EEP Support and Lifecycle Status on page 5531.

- You can perform an offline upgrade to MapR 6.1.xfrom the following core versions: 3.x, 4.0.x, 4.1, 5.x, and 6.x.
- The offline upgrade procedure requires an outage of the entire cluster. During the maintenance window, the administrator:
  - Stops all jobs on the cluster.
  - Stops all cluster services.
  - Upgrades packages on all nodes (which can be done in parallel).
  - Brings the cluster back online at once.

Offline Manual Upgrade Workflow

High-Level Steps	Detailed Information (review all items unless noted otherwise)
1. Understand MapR Core/EEP Dependencies ↓	• Operating System Support Matrix on page 5522 • Operating System Support Matrix (MapR 5.x) • EEP Support and Lifecycle Status on page 5531 • Component Versions for Released EEPs
2. Plan for the MapR Core Upgrade ↓	• Upgrading and Your License on page 295 • Installation and Upgrade Notes (MapR 6.0) • Installation and Upgrade Notes (MapR 6.0.1) • Installation and Upgrade Notes (MapR 6.1.0) on page 39 • Installation and Upgrade Notes (MapR 6.1.1) on page 74 • Planning Your MapR Core Upgrade on page 298
3. Plan for the EEP Upgrade ↓	• Planning MapR Ecosystem Pack (EEP) Upgrades on page 335

last-updated: Apr 11, 2022

High-Level Steps	Detailed Information (review all items unless noted otherwise)
4. Perform Pre-Upgrade Steps for MapR Core 	• Preparing to Upgrade MapR Core on page 303
5. Prepare to Upgrade EEP Components 	• Preparing to Upgrade the MapR Ecosystem Pack on page 335
6. Set up Repositories 	• Setting Up Repositories on page 309
7. Perform the Offline Manual Upgrade 	• Offline and Manual Upgrade Procedure on page 316
8. Upgrade the EEP Components 	• Upgrading the MapR Ecosystem Pack Without the MapR Installer on page 350
9. Perform Post-Upgrade Steps for EEP 	• Finishing the MapR Ecosystem Pack Upgrade on page 372
10. Perform Post-Upgrade Steps for MapR Core 	• Restart and Check Cluster Services • Manually Update Configuration Files • Upgrade Clients • Enable New Features • Update Hadoop Configuration File
11. Install Additional MapR Core Features 	• Installing Additional MapR Core Features on page 335

last-updated: Apr 11, 2022

High-Level Steps	Detailed Information (review all items unless noted otherwise)
12. Secure the Upgraded Cluster	• Securing the Upgraded Cluster on page 335

Workflow: MapR Installer Upgrade
This page summarizes the steps for upgrading from MapR 5.2.x to MapR 6.1.x by using the MapR Installer.

In this workflow, the cluster to be upgraded is nonsecure; after the upgrade, the cluster will continue to be nonsecure, but can be secured by using the **Incremental Install** function of the MapR Installer.

Installer Upgrade Summary

In an upgrade using the MapR Installer, the Installer shuts down MapR core on the entire cluster, upgrades and configures MapR core, starts MapR core, upgrades EEP components, and then starts the EEP components. Like the offline manual upgrade, the MapR Installer upgrade is an *offline* upgrade. The MapR Installer upgrade requires you to:

1. Plan for the upgrade.
2. Update the MapR Installer to version 1.10.
3. Launch the Installer and select **Version Upgrade**.
4. Complete the upgrade through the Installer.
5. Merge custom configuration settings.
6. Perform post-upgrade checks.

The workflow later on this page provides more detail to help you get started with the MapR Installer upgrade.

Considerations for MapR Installer Upgrades

Before upgrading using the MapR Installer, note these considerations:

- Upgrades using the MapR Installer are supported only from release 5.2.x with EEP 3.0.1 or later.
- If security is enabled on the cluster – either MapR default security or custom security – you cannot upgrade from release 5.2.x using the MapR Installer; you must use one of the manual upgrade workflows. Note that upgrades from secure release 6.0.0 or 6.0.1 clusters can be done using the MapR Installer, as described in Planning Your MapR Core Upgrade on page 298.
- Security settings cannot be changed during a version upgrade using the MapR Installer.
- Before upgrading using the MapR Installer, you must update the installer to version 1.10 or later.
- This procedure assumes that the cluster was originally installed using the MapR Installer or a MapR Installer Stanza. If the cluster was installed manually, you must use the manual steps to upgrade or use probe and import to generate the installer database.
- When upgrading to release 6.1 using the MapR Installer, EEP 3.0.1 or later is upgraded automatically to EEP 6.0.0.

MapR Installer Upgrade Workflow

High-Level Steps	Detailed Information (review all items unless noted otherwise)
1. Understand MapR Core/MEP Dependencies 	• Operating System Support Matrix (MapR 6.x) • Operating System Support Matrix (MapR 5.x) • EEP Support and Lifecycle Status on page 5531 • Component Versions for Released EEPs
2. Plan for the MapR Core Upgrade 	• Installation and Upgrade Notes (MapR 6.0) • Installation and Upgrade Notes (MapR 6.0.1) • Installation and Upgrade Notes (MapR 6.1.0) on page 39 • Installation and Upgrade Notes (MapR 6.1.1) on page 74 • Planning Your MapR Core Upgrade on page 298
3. Plan for the EEP Upgrade 	• Planning MapR Ecosystem Pack (EEP) Upgrades on page 335
4. Perform Pre-Upgrade Steps for MapR Core 	• Preparing to Upgrade MapR Core on page 303
5. Prepare to Upgrade EEP Components 	• Preparing to Upgrade the MapR Ecosystem Pack on page 335
6. Upgrade the MapR Installer to Version 1.10 	• Checking the MapR Installer Version • Updating the MapR Installer • MapR Installer
7. Halt Jobs and Applications 	• Upgrading MapR Core With the MapR Installer on page 308

last-updated: Apr 11, 2022

High-Level Steps	Detailed Information (review all items unless noted otherwise)
8. Launch the MapR Installer 	• Upgrading MapR Core With the MapR Installer on page 308 • MapR Installer
9. Select the Version Upgrade Option 	• Upgrading MapR Core With the MapR Installer on page 308
10. Complete the Upgrade Through the Installer 	• Online Help for MapR Installer Fields on page 5412
11. Perform Post-Upgrade Steps for MapR Core 	• Considerations for Upgrades Using the MapR Installer on page 322 • Step 2: Manually Update Configuration Files on page 325 • Step 3: Upgrade Clients on page 326
12. Perform Post-Upgrade Steps for EEP 	• Finishing the MapR Ecosystem Pack Upgrade on page 372
13. Secure the Upgraded Cluster	• Securing the Upgraded Cluster on page 335

Upgrading MapR Core

Describes the process of upgrading MapR core.

Upgrading MapR core typically includes upgrading:

- MapR Core
- Ecosystem Components
- MapR Clients

Upgrading MapR core means you will need to upgrade to a MapR Ecosystem Pack (EEP). For example, upgrading to release 6.2 requires you to upgrade to EEP 7.0.0 or later before you can enable release 6.2 features.

The steps for upgrading a EEP are in another section of this guide because EEPs can be upgraded independently of the core version. The following procedures prompt you when it is necessary to plan for or upgrade a EEP.

Upgrading MapR core consists of the following steps:

last-updated: Apr 11, 2022

1. Planning the Upgrade – Determine the upgrade method, when to upgrade, and whether ecosystem components or MapR clients need to be upgraded along with core.
2. Preparing to Upgrade – Prepare the cluster for upgrade while it is still operational. This includes pre-upgrade steps for MapR core and ecosystem components.
3. Upgrading the Cluster
   - Upgrading with the MapR Installer - Use a web interface that automates the upgrade of MapR core and ecosystem components.
   - Upgrading without the MapR Installer - Perform steps to upgrade core and manually upgrade each ecosystem component.
4. Finishing the Upgrade -Complete the post-upgrade steps for MapR core and any ecosystem components that you upgraded.
5. Upgrading MapR clients – Perform steps to upgrade the MapR client.

Note: In this document, the *existing* version refers to the release version that you are upgrading *from* and the *new* version refers to the release version that you are upgrading to.

Instructions in the following sections guide you through each upgrade step:

Upgrading and Your License
If you are upgrading from MapR version 5.0 or earlier, the Base License file must be manually updated on all nodes in your cluster.

Important: As of MapR version 5.1, because of the new license categories, you must manually update the Base License file. This applies to all upgrade methods (MapR Installer, MapR Installer Stanzas, or manual rolling upgrade).

To update the Base License file, copy the new Base License file from the `/opt/mapr/conf.new/` directory to the `/opt/mapr/conf/` directory on *every node in your cluster*.

```
cp /opt/mapr/conf.new/BaseLicense.txt /opt/mapr/conf/
```

Note: The BaseLicense.txt file must be copied before starting Warden; otherwise, Warden must be restarted.

MapR Event Store For Apache Kafka Module

If you are upgrading from the Enterprise Database Edition (M7) to the Enterprise Edition and want to use the MapR Event Store For Apache Kafka module, you must purchase additional licensing. In this case, you will be issued a new license to apply. See Managing Licenses on page 777 for more information.

License Editions

As of MapR 5.1, MapR licenses are categorized by new editions and modules that further define the features supported by an edition. See the following table for descriptions of the new license options:

License Edition	Description
Community Edition	An unlimited, free, community-supported MapR edition with one free NFS Gateway. This edition includes Hadoop, MapR Database, and MapR Event Store For Apache Kafka. However, real-time global replication of MapR Database tables or MapR Event Store For Apache Kafka is not included.

License Edition	Description
Enterprise Edition	MapR Edition that enables enterprise class features such as high availability, multi-tenancy, and disaster recovery. Each of the following modules for the Enterprise Edition unlocks a portion of the total platform capabilities: **Analytics** — Enables enterprise class features for analytic use cases, such as highly-available NFS and support for services such as YARN and MapReduce. **Database** — Enables enterprise class features for operational NoSQL database, with MapR Database JSON and binary tables, and real-time global database replication. **Streams** — Enables enterprise class features for publish/subscribe event streaming, with MapR Event Store For Apache Kafka and real-time global stream replication.

For more information, see What's Included.

License Mapping

With the release of 5.1 and MapR Event Store For Apache Kafka, the licensing model has been simplified, allowing more choice in which specific features are licensed on a cluster. See the following table to understand how the new MapR licenses correspond to the legacy license editions with which you are familiar:

Legacy Edition	New Edition & Module(s)
M3 or Community Edition	Community Edition. Starting in 5.1, the Community Edition includes MapR Event Store For Apache Kafka.
M5 or Enterprise Edition	Enterprise Edition with Hadoop Module.
M7 or Enterprise Database Edition	Enterprise Edition with Hadoop and Database modules.

MapR Metering Feature

Beginning with the MapR 6.1 release, MapR software supports metering. Annual subscriptions will continue to be offered, but metering gives you the option of purchasing a variable consumption plan that is based on usage.

For more information, see MapR Software Licensing.

Checking Your Cluster License

To view license information on your cluster, see Viewing the Licenses on the Cluster on page 778.

For MapR Data Platform licensing information, see Product Licensing on page 6581.

MapR POSIX Licenses

Beginning with MapR 6.0.1, customers who purchase MapR XD Distributed File and Object Store Enterprise Premier automatically receive the following licenses for free:

- MapR FUSE POSIX Basic Client
- MapR POSIX Client for Containerized Apps (required for MapR PACC and DSR functionality)

For more information, see the MapR Editions web page.

Related tasks

Viewing the Licenses on the Cluster on page 778
List the licenses on the cluster using either the Control System or the CLI.

Adding a License on page 777
Add a license through the Control System or the CLI.

Removing a License on page 780
Describes how to remove a license using the Control System and the CLI.

Related reference

license add on page 1680
Adds a license. Permissions required: `fc` or `a`.

license addcrl on page 1681
Adds a certificate revocation list (CRL). Permissions required: `fc` or `a`.

license apps on page 1682
Displays the features authorized for the current license. Permissions required: `login`

license list on page 1684
Lists licenses on the cluster. Permissions required: `login`. For best results, use the `-json` option when running the command.

license listcrl on page 1686
Lists certificate revocation lists (CRLs) on the cluster. Permissions required: `login`.

license remove on page 1687
Removes a license. Permissions required: `fc` or `a`.

license showid on page 1689
Displays the cluster ID for use when creating a new license. Permissions required: `login`.

MapR Core Upgrade Process

When you upgrade MapR core, you will also upgrade a number of cluster components.

The following cluster components are upgraded with MapR core:

- Storage Layer: MapR File System fileserver and Container Location Database (CLDB) services
- Cluster Management Services: ZooKeeper and Warden
- NFS server
- Web server, including the Control System user interface and REST API to cluster services
- The `maprcli` commands for managing cluster services
- Any new features and performance enhancements introduced with the new version.

When you upgrade MapR core, the following changes occur within the `/opt/mapr` directory:

- If required, additional folders are added.

- Product binaries are replaced by binaries associated with the new version.
- Existing configuration files remain in the active directory and default configuration files associated with the new version are installed in a new directory:

Default Configuration File Directories	Active Configuration File Directories
`/opt/mapr/conf.new`	`/opt/mapr/conf`
`/opt/mapr/conf/conf.d.new`	`/opt/mapr/conf/conf.d`

When MapR core includes an updated hadoop common version, a new hadoop directory is installed and the following changes also occur:

- The `/opt/mapr/conf/hadoop_version` file reflects the new hadoop version number.
- Paths in the `/opt/mapr/conf/conf.d/warden.<servicename>.conf` files reflect the new hadoop directory.
- The following symlinks in `/opt/mapr/lib` reflect the new hadoop version:
  - `hadoop-auth-2.x.x.jar`
  - `hadoop-yarn-api-2.x.x.jar`
  - `hadoop-yarn-client-2.x.x.jar`
  - `hadoop-yarn-common-2.x.x.jar`
  - `hadoop-common-2.x.x.jar`
- A new hadoop 2.x directory is created under `/opt/mapr/hadoop`.

  If a hadoop 2.x directory already exists, the configuration files from the existing hadoop 2.x directory will replace the default configuration files in the new 2.x hadoop directory thereby retaining the existing configuration. The existing hadoop 2.x directory is also moved into the following directory: `/opt/mapr/hadoop/OLD_HADOOP_VERSION`.

Related reference

Hadoop Protocol Versions for MapR Software on page 5597
Shows the Hadoop RPC protocol version and compatible MapR client versions for each MapR release.

Planning Your MapR Core Upgrade

Describes how to develop a successful plan for your upgrade process.

The key to a successful upgrade process is to plan the process ahead of time. This page helps you develop an upgrade process that fits the needs of your cluster and users.

Choosing a Cluster Upgrade Method

Supported upgrade methods are:

- Manual rolling upgrade
- Offline manual upgrade
- Offline upgrade using MapR Installer

Upgrade Workflows on page 285 describe these methods in more detail. The method you choose affects the flow of events while upgrading packages on nodes and the duration of the maintenance window.

The following table summarizes the MapR core upgrade paths supported for each method:

Upgrade From	Upgrade To	Offline	Rolling
3.x	MapR 6.1.0	Yes	No
4.x	MapR 6.1.0	Yes	No*
5.x	MapR 6.1.0	Yes	Yes**
6.x	MapR 6.1.0	Yes	Yes

*Rolling upgrades from 4.x to 5.x are supported, after which you may perform another rolling upgrade from 5.x to 6.x.

**MapR core by itself supports rolling upgrades from release 5.x to later releases. If your cluster includes MapR core and MapR ecosystem components, rolling upgrades to release 6.x are supported only from release 5.2.x and EEP 3.0.1 or later. Ecosystem components will continue to work during a rolling upgrade of MapR core as long as the ecosystem components are not updated. But the rolling upgrade of ecosystem components is not supported. If you choose to upgrade MapR core and ecosystem components together, the ecosystem components might not function properly during the upgrade process.

The following table summarizes upgrade support for MapR Ecosystem Packs (EEPs):

Upgrade From	Upgrade To	Offline	Rolling
Pre-MEP	EEP 6.0.0	Yes	No
EEP 1.1.x	EEP 6.0.0	Yes	No
EEP 2.x	EEP 6.0.0	Yes	No
EEP 3.0.1+	EEP 6.0.0	Yes	Yes*
EEP 4.x	EEP 6.0.0	Yes	Yes*
EEP 5.x	EEP 6.0.0	Yes	Yes*

*A rolling upgrade of MapR core is supported, and ecosystem components will continue to work during the rolling upgrade, but a rolling upgrade of ecosystem components is not supported.

Transition EEPs

The following EEPs are considered *transition EEPs* because you can upgrade these EEPs directly to release 6.1.0 using a rolling upgrade (for components that support a rolling upgrade):

- EEP 3.0.1 and later 3.0.x
- EEP 4.x
- EEP 5.x

Transition EEPs can coexist with release 6.1.0 only during the course of an upgrade. You must not enable release 6.1.0 new features while a transition EEP is in use on release 6.1.0. You must upgrade to EEP 6.0.0 before enabling new features for release 6.1.0.

Offline Upgrade

The offline upgrade process is simpler than a rolling upgrade, and usually completes faster. In an offline upgrade, MapR software processes and the jobs that depend on them are stopped on all nodes so that packages can be updated. Offline upgrade is the default upgrade method when other methods cannot be used.

Offline Upgrade Paths without the MapR Installer

last-updated: Apr 11, 2022

You can perform an offline upgrade from the following core versions:

- MapR 3.x
- MapR 4.0.x
- MapR 4.1
- MapR 5.x
- MapR 6.x

Note: After upgrading MapR core to release 6.0 or later, you must upgrade ecosystem components to an EEP that is compatible with your core 6.0 or later release. To determine the compatible EEPs, see EEP Support and Lifecycle Status on page 5531. This must be done before you enable core features.

During the maintenance window, the administrator:

- Stops all jobs on the cluster.
- Stops all cluster services.
- Upgrades packages on all nodes (which can be done in parallel).
- Brings the cluster back online at once.

Rolling Upgrade

In a manual rolling upgrade, you upgrade the MapR software one node at a time so that the cluster as a whole remains operational throughout the process. The fileserver service on each node goes offline while packages are upgraded, but its absence is short enough that the cluster does not raise the data-under-replication alarm.

The following restrictions apply to rolling upgrades:

- In release 6.0 and later, only manual rolling upgrades are supported. Scripted rolling upgrades are not supported.
- Rolling upgrades only upgrade core packages, not ecosystem components. A rolling upgrade of ecosystem components is not supported.
- If you choose to do a rolling upgrade on a cluster with core and ecosystem components, the ecosystem components will continue to work during the rolling upgrade as long as the ecosystem components are not updated. If you choose to upgrade core and ecosystem components together, the ecosystem components might not function properly during the upgrade process.
- The administrator should block off a maintenance window, during which only critical jobs are allowed to run and users expect longer-than-average run times.

Rolling Upgrade Paths

You can perform a manual rolling upgrade from only the following core versions:

- Release 5.2.x with EEP 3.0.1 or later
- Release 6.x with EEP 4.0.0 or later

Note: After upgrading core, you must upgrade ecosystem components to EEP 4.0.0 or later, and this must be done before you enable release 6.x or later features. To determine the EEP required by your release, see EEP Support and Lifecycle Status on page 5531.

last-updated: Apr 11, 2022

Updating the JDK

Check the JDK Support Matrix to verify that your JDK version is supported by the core version to which you are upgrading. Releases 6.0 and 6.1 require JDK 8. For more information, see the JDK Support Matrix.

Planning for Security

Security is not enabled by default for upgrades. During an upgrade, the security attributes of your cluster are preserved unless you decide to change them. Note that if you have configured security on a release 5.2.x cluster, you cannot use the MapR Installer or Stanzas to upgrade. You must upgrade manually. For information about custom security, see Customizing Security in MapR on page 1474.

Before upgrading core software, make sure that you have reviewed the list of known vulnerabilities in Security Vulnerabilities on page 6569. If a vulnerability applies to your release, contact your HPE support representative for a fix, and apply the fix immediately, if applicable.

Scheduling the Upgrade

Consider the following factors when scheduling the upgrade:

- When will preparation steps be performed? How much of the process can be performed before the maintenance window?
- What calendar time would minimize disruption in terms of workload, access to data, and other stakeholder needs?
- How many nodes need to be upgraded? How long will the upgrade process take for each node, and for the cluster as a whole?
- When should the cluster stop accepting new non-critical jobs?
- When (or will) existing jobs be terminated?
- How long will it take to clear the pipeline of current workload?
- Will other Hadoop ecosystem components (such as Hive) get upgraded during the same maintenance window?
- When and how will stakeholders be notified?

Planning Upgrades to MapR Clients

Determine if you need to upgrade MapR client nodes. You upgrade MapR client nodes after you upgrade the cluster nodes but before enabling new features.

MapR Client Nodes

On each MapR client node, upgrade to the client version that is compatible with the operations that you want to perform on the cluster. The following table shows which supported client operations are available based on the client version and the cluster version.

The following client operations are supported on 4.0.x and 5.x clusters and clients.

- File system access
- Submit MapReduce (MR) v1 job
- Submit MapReduce v2 applications

The following client operation is supported on 4.0.2 and higher clusters with 4.0.2 and higher clients. For example, this operation is not available on a 5.1 cluster with a 4.0.1 client.

last-updated: Apr 11, 2022

- Submit MapReduce version 2 applications with Resource Manager zero-configuration failover

MapR POSIX Client Nodes

On MapR POSIX client nodes, the only supported client operation is filesystem access. As of release 5.1, MapR FUSE-based POSIX clients are available in addition to MapR loopback NFS clients.

Note: Only releases 5.1 or higher support MapR FUSE-based POSIX clients.

MapR POSIX loopback NFS clients can be upgraded or a fresh install can be performed.

See Upgrading the MapR POSIX loopbacknfs Client on page 328 and Migrating to FUSE-based POSIX Clients for more information.

Note: Basic and Platinum POSIX client packages are recommended for fresh installation and for all new clusters.

Upgrade to the client version supported by your cluster, as shown in the following table. Upgrading to the 5.x client is recommended for 4.1 and 5.x clusters of MapR loopbacknfs POSIX nodes. If you plan on using FUSE-based POSIX clients, ensure that the cluster has been upgraded to release 5.1 or higher because the FUSE-based POSIX client can only connect to clusters running release v5.1 or higher.

The following table shows which MapR loopback NFS client versions are supported by which MapR data-fabric clusters. For example, the release 6.0 cluster supports 4.0.2, 4.1, 5.0, 5.1, and 5.2 loopback NFS clients.

Table

	6.x client	5.2 client	5.1 client	5.0 client	4.1 client	4.0.2 client	4.0.1 client
6.x cluster	Yes	Yes	Yes	Yes	Yes	Yes	N/A
5.2 cluster	Yes	Yes	Yes	Yes	Yes	Yes	N/A
5.1 cluster	Yes	Yes	Yes (recommended)	Yes	Yes	Yes	N/A
5.0 cluster	Yes	Yes	Yes (recommended)	Yes	Yes	Yes	N/A
4.1 cluster	Yes	Yes	Yes (recommended)	Yes	Yes	Yes	N/A
4.0.2 cluster	Yes	Yes	Yes	Yes	Yes	Yes	N/A

Determining Cross-Cluster Feature Support

MapR Data Platform supports features that operate on more than one cluster. Before you upgrade, consider the impact of the following cross-cluster features:

Volume Mirroring	Volume mirroring works from a lower MapR version to a higher MapR version irrespective of the features that you enable on the higher version. For example, you can mirror volumes from a release 6.1 cluster to a release 6.2 cluster irrespective of whether or not you have enabled the new features present in the release 6.2 version. However, volume mirroring from a higher release version to a lower release version works only when you enable identical set of features on both clusters. For example, you can mirror volumes from a release 6.2 cluster to a release 6.1 cluster only if you do not

last-updated: Apr 11, 2022

	enable new features that are present on the release 6.2 cluster.
Table Replication	Table replication works between clusters of different versions as long as both versions support MapR Database table replication. For example, you can replicate MapR Database binary tables from a release 6.2 cluster to a release 6.0 cluster. **Note:** As of release 5.2, MapR Database JSON table replication is also supported. You cannot replicate MapR Database JSON tables to a cluster that runs a version prior to release 5.2.

Planning for the MapR Ecosystem Pack

To plan for the MapR Ecosystem Pack (EEP), see Planning MapR Ecosystem Pack Upgrades.

What's Next

Go to Preparing to Upgrade MapR Core on page 303.

Preparing to Upgrade MapR Core

Complete these pre-upgrade steps for MapR core.

Upgrade a test cluster before upgrading your production cluster. After you have planned your upgrade process, prepare the cluster for upgrade while your existing cluster is fully operational. Prepare to upgrade as described in this section to minimize downtime and eliminate unnecessary risk. Design and run health tests and back up critical data. Performing these tasks during upgrading reduces the number of times you have to touch each node, but increases down-time during upgrade.

Complete the following pre-upgrade steps:

1. Verify System Requirements for All Nodes

Verify that all nodes meet the following minimum requirements for the new version of core software:

- **Software dependencies.** Package dependencies in the MapR distribution can change from version to version. If the new MapR version has dependencies that were not present in the older version, you must address them on all nodes before upgrading your software. Installing dependency packages can be done while the cluster is operational. See Packages and Dependencies for MapR Software. If you are using a package manager, you can specify a repository that contains the dependency package(s), and allow the package manager to automatically install them when you upgrade the MapR packages. If you are installing from package files, you must pre-install dependencies on all nodes manually.
- **Hardware requirements.** The newer version of packages might have greater hardware requirements. Hardware requirements must be met before upgrading.
- **OS requirements.** MapR OS requirements do not change frequently. If the OS on a node doesn't meet the requirements for the newer MapR version, plan to decommission the node and re-deploy it with an updated OS after the upgrade. See Operating System Support Matrix on page 5522 and Operating System Support Matrix (MapR 5.x).
- **Certificate requirements.** Recent versions of Safari and Chrome web browsers have removed support for older certificate cipher algorithms, including those used by some MapR versions. For more information about resolving this issue, see Unable to Establish a Secure Connection on page 6569.

2. Design Health Checks

Plan what kind of test jobs and scripts you will use to verify cluster health as part of the upgrade process. You will verify cluster health several times before, during, and after upgrade to ensure success at every step, and to isolate issues whenever they occur. Create both simple tests to verify that cluster services start and respond, as well as non-trivial tests that verify workload-specific aspects of your cluster.

Design Simple Tests

Here are a few examples of simple tests you can design to check node health:

- Use maprcli commands to see if any alerts exist and to verify that services are running as expected. For example:

```
# maprcli node list -columns svc
hostname
service                                                 ip
labnode55
nodemanager,cldb,fileserver,hoststats                   10.10.82.55
labnode56
nodemanager,fileserver,hoststats                        10.10.82.56
labnode57
fileserver,nodemanager,hoststats                        10.10.82.57
labnode58
fileserver,nodemanager,webserver,hoststats              10.10.82.58
                       ...lines deleted...
                       # maprcli alarm list
                       alarm state
description
                        entity    alarm name
alarm statechange time
                       1            One or more licenses is about to
expire within 25 days
CLUSTER   CLUSTER_ALARM_LICENSE_NEAR_EXPIRATION   1366142919009
                       1            Can not determine if service: nfs is
running. Check logs at: /opt/mapr/logs/nfsserver.log         labnode58
NODE_ALARM_SERVICE_NFS_DOWN            1366194786905
```

  In this example, you can see that an alarm is raised indicating that MapR software expects an NFS server to be running on node `labnode58`, and the node list of running services confirms that the nfs service is not running on this node.

- Batch create a set of test files.
- Submit a MapReduce application.
- Run simple checks on installed Hadoop ecosystem components. For example, run a Hive query.

Design Non-trivial Tests

Appropriate non-trivial tests are specific to your particular cluster workload. You may have to work with users to define an appropriate set of tests. Run tests on the existing cluster to calibrate expectations for “healthy” task and job durations. On future iterations of the tests, inspect results for deviations. Some examples:

- Run performance benchmarks relevant to the cluster’s typical workload.
- Run a suite of common jobs. Inspect for correct results and deviation from expected completion times.
- Test correct inter-operation of all components in the Hadoop stack and third-party tools.
- Confirm the integrity of critical data stored on the cluster.

last-updated: Apr 11, 2022

3. Verify Cluster Health

Run the test you designed in step 2 to verify the cluster health prior to upgrade.

- Run the suite of simple tests to verify that basic features of the MapR core are functioning correctly, and that any alarms are known and accounted for.
- Run the suite of non-trivial tests to verify that the cluster is running as expected for a typical workload, including integration with Hadoop ecosystem components and third-party tools.

Proceed with the upgrade only if the cluster is in an expected, healthy state.

4. Back Up Critical Data

Data in the MapR cluster persists across upgrades from version to version. However, as a precaution, you might want to back up critical data before upgrading. If you deem it practical and necessary, you can do any of the following:

- Copy data out of the cluster using `distcp` to a separate, non-Hadoop datastore.
- Mirror critical volume(s) into a separate MapR cluster, creating a read-only copy of the data which can be accessed via the other cluster.

When services for the new version are activated, the MapR File System will update data on disk automatically. The migration is transparent to users and administrators. Once the cluster is active with the new version, you cannot roll back.

5. Run Your Upgrade Plan on a Test Cluster

Before executing your upgrade plan on the production cluster, perform a complete *dry run* on a test cluster. You can perform the dry run on a smaller cluster than the production cluster, but make the dry run as similar to the real-world circumstances as possible. For example, install all Hadoop ecosystem components that are in use in production, and replicate data and jobs from the production cluster on the test cluster. The goals for the dry run are:

- Eliminate surprises. Get familiar with all upgrade operations you will perform as you upgrade the production cluster.
- Uncover any upgrade-related issues as early as possible so you can accommodate them in your upgrade plan. Look for issues in the upgrade process itself, as well as operational and integration issues that could arise after the upgrade.

6. Pause Cross-Cluster Operations

Complete the steps for each cross-cluster feature used by this cluster:

- **Volume Mirroring.** If volumes from another cluster are mirrored on this cluster, use one of the following options to stop the mirroring of volumes on this cluster:

Using the Control System	See Stopping the Mirror on page 920.
Using a `maprcli` command	Run the `maprcli volume mirror stop` command on this cluster. See volume mirror stop on page 2004.

- **Table Replication.** If source tables on this cluster are replicated to tables on another cluster, pause the replication of tables on this cluster. Use one of the following options for each source table on this cluster:

Using the Control System	1. Log in to the Control System and go to the source table information page. 2. On the Replications tab associated with the source table, select each replica and then click **Actions** > **Pause Replication** > **.**
Using a `maprcli` command	Run the `maprcli table replica pause` command. See table replica pause on page 1866.

Note: Once you have completed the MapR core upgrade and the Post-Upgrade Steps for MapR Core on page 322, you can resume cross-cluster operations.

7. Back up Configuration Files

If you are upgrading the MapR FUSE-based POSIX client on Ubuntu, create a backup copy of your custom settings in the `fuse.conf` file in `/opt/mapr/conf` directory. If you do not create a backup copy, you might lose your custom settings for the POSIX client because the new `fuse.conf` file with default settings will overwrite your current `fuse.conf` file with custom settings.

If you are upgrading the MapR FUSE-based POSIX client on other supported operating systems, during upgrade the software automatically sets up the `fuse.conf.backup` file in addition to the new `fuse.conf` file in the `/opt/mapr/conf` directory.

Consider creating the `env_override.sh` file to store custom settings for environmental variables. Upgrading to a new MapR release causes the `env.sh` file to be replaced and removes any custom settings. Creating the `env_override.sh` file can simplify the management of environmental variables. For more information, see About env_override.sh on page 2290.

8. Migrate from MapReduce Version 1

MapReduce version 1 (MRv1) is deprecated for MapR 6.0 or later. If you were previously using MRv1, you must prepare your cluster to run MapReduce version 2 (MRv2) applications before upgrading to core 6.0 or later:

- Ensure that the MapReduce mode on your cluster is set to `yarn`. MRv2 is an application that runs on top of YARN.
- Uninstall all packages associated with MRv1.

For more information about how to prepare your cluster to run MRv2 applications, see Migrating from MapReduce Version 1 to MapReduce Version 2 on page 307.

9. Migrate from Mahout and Storm

Mahout and Storm are not supported on core 6.0 or later. Before the upgrade, disable applications that use these components, and remove the Mahout and Storm packages. To view the ecosystem components supported on MapR releases, see Component Versions for Released EEPs.

10. Prepare to Upgrade the MapR Ecosystem Pack

Complete the pre-upgrade steps in Preparing to Upgrade the MapR Ecosystem Pack. Then return to this section.

What's Next

Go to Upgrading MapR Core With the MapR Installer on page 308 or Upgrading MapR Core Without the MapR Installer on page 309.

Migrating from MapReduce Version 1 to MapReduce Version 2

If you previously ran MRv1 jobs on your cluster, prepare your cluster to run YARN applications in MapR 6.0 or later.

Configuring the MapReduce Mode

Client and cluster nodes submit MapReduce applications to the YARN framework (`yarn` mode) unless you configure them to use the classic framework (`classic` mode). Because MapReduce version 1 (MRv1) is deprecated for MapR 6.0 and later, you can no longer submit MRv1 jobs using `classic` mode. If you previously configured your client and cluster nodes to use `classic` mode, you must prepare your cluster to run YARN applications in MapR 6.0 or later. Before upgrading to MapR 6.0 or later, ensure that the MapReduce mode is set to `yarn` in the environment variable, on client nodes, and on the cluster.

Configure the MapReduce mode for the following components:

Component	How to Change the Mode to `yarn`:
Environment Variable	Set the MapReduce mode in an environment variable: **1.** Open a terminal on the client node. **2.** Enter the following command on the shell: `export MAPR_MAPREDUCE_MODE=yarn`
Client	In the hadoop_version file on a MapR client node, verify the MapReduce mode is set to `yarn`. **1.** Open the hadoop_version file in the following location: `/opt/mapr/conf/` **2.** If the default_mode parameter is set to `classic`, change it to `yarn`: `default_mode=yarn`.
Cluster	Run the following command to display the cluster-wide mode: `maprcli cluster mapreduce get`. If the cluster is set to `classic` mode, use the command line or the MapR Control System to set it back to the `yarn`default for all nodes in the cluster: • To set the cluster's MapReduce mode using maprcli: **1.** Run the following command: `maprcli cluster mapreduce set -mode yarn`. • To set the cluster's MapReduce mode in the MapR Control System: **1.** Log in to the MapR Control System. **2.** Perform one of the following operations: • In the header area, click the link that contains the current MapReduce version. • In the **System Settings** view of the Navigation Pane, click **MapReduce Version.** **3.** In the **Configure MapReduceMode** dialog, select the `yarn` option for the cluster. **4.** Click **OK**.

last-updated: Apr 11, 2022

Uninstalling MRv1 Packages

After verifying that the MapReduce mode is set to `yarn`, uninstall the packages associated with MRv1:

- `mapr-jobtracker`
- `mapr-tasktracker`
- `mapr-metrics`

The following table lists uninstall commands by operating system. Use these commands to uninstall the above packages:

Operating System	Uninstall Command
Ubuntu	`apt-get remove <package name>`
CentOS/RedHat:	`yum remove <package name>`
SLES	`zypper rm <package name>`

Upgrading MapR Core With the MapR Installer

If the cluster that you want to upgrade was installed using the MapR Installer, use the MapR Installer to upgrade MapR core.

Before you begin, review Planning Your MapR Core Upgrade on page 298, and verify that your cluster is prepared for an upgrade. In some cases, a maintenance update can be performed rather than a version upgrade. For more information, see Performing a Maintenance Update on page 5447

Warning: The **Version Upgrade** operation is an *offline* operation. Service failures, job failures, or the loss of customized configuration files can occur if you do not perform the steps to prepare the cluster for an upgrade.

Note: To upgrade to release 6.0 or later using the MapR Installer, release 5.2.x and EEP 3.0.1 or later must be installed on your cluster. If security is enabled on the cluster, you cannot use the MapR Installer or Stanzas to upgrade. You must perform the upgrade manually.

1. Update the MapR Installer. For more information, see Updating the MapR Installer on page 5409.
   This step ensures that the MapR Installer has access to the latest packages.
2. Halt jobs and applications. Stop accepting new jobs and applications, and stop YARN applications.

   ```
   # yarn application -list
   # yarn application -kill <ApplicationId>
   ```

   You might also need specific commands to terminate custom applications.
3. Launch the MapR Installer URL (`https://<hostname/IPaddress>:9443`)
4. Select the **Version Upgrade** option to complete the upgrade through the MapR Installer. The installer allows you to specify the core version, the MapR Ecosystem Pack (EEP) version, and the components and services you want to install.

last-updated: Apr 11, 2022

Note:

- Incorporate a brief delay when you use the **Version Upgrade** screen. After specifying the core version, the EEP version, and the services you need, wait a minute or two before clicking **Next** to advance to the next screen. This delay gives the Installer time to process the selections that you made.
- Do not refresh the browser page during the upgrade sequence. Doing so can cause errors. For more information, see IN-1915 in MapR Installer Known Issues.
- If you upgrade a non-secure cluster to release 6.0.1 or later and metrics monitoring is enabled, the MapR Installer asks you to specify a password for the Grafana administrator ID (`admin`). You must specify a password. For more information about Grafana password requirements, see Logging on to Grafana on page 1382.
- If the MapR Installer indicates that the version upgrade failed, or if the following error message is displayed, use the **Import State** command to revert the cluster to the last known state. See Importing or Exporting the Cluster State on page 5447.

```
Custom secure cluster < MapR core 6.0.0 cannot be upgraded
```

5. Once the upgrade through the MapR Installer is complete, perform the post-upgrade steps. See Finishing the MapR Core Upgrade on page 322.

Upgrading MapR Core Without the MapR Installer
You can upgrade MapR core without using the MapR Installer.

First, you perform an offline or rolling upgrade of the MapR core manually. Next, you configure the new version to enable support of MapR core features. Finally, you upgrade ecosystem components manually.

When upgrading, you can install packages from the following sources:

- The MapR Internet repository
- A local repository
- Individual package files

Go to the next topic to begin setting up repositories.

When you are finished upgrading MapR core, you will need to follow the steps to upgrade the ecosystem components, as described in Upgrading the MapR Ecosystem Pack Without the MapR Installer.

Note: MapR 6.0 and later releases are only compatible with EEP 4.0.0 and later EEPs (see EEP Support by MapR Core Version). If you upgrade MapR core to MapR 6.0 or later, you must install or upgrade to EEP 4.0.0 or later. Clusters with EEP 3.0.1 or later are allowed to upgrade to MapR 6.0 or later, but such clusters must not enable new features, add new services, add new nodes, or make configuration changes until EEP 3.0.1 or later is upgraded to EEP 4.0.0 or later.

Then proceed to Finishing the MapR Core Upgrade on page 322.

Setting Up Repositories
This section describes how to set up internet and local repositories for each operating system.

Both internet repositories and local repositories can be set up. In addition, package files can be downloaded, stored locally, and then the software can be installed from the files. The following sections describes how to do both for each operating system.

Platform-specific upgrade repositories are hosted at: https://package.mapr.hpe.com/releases/v<version>/<platform>/mapr-v<version>GA-upgrade.<rpm/deb>.tgz.

See MapR Repositories and Packages on page 128 for more information about repositories and packages.

Using the MapR Internet Repository
It is usually easiest to install a MapR cluster from an Internet repository.

The MapR repository on the Internet provides all of the packages required to install a cluster using native tools such as `yum` on RedHat, Oracle Linux, or CentOS, or `apt-get` on Ubuntu, or `zypper` on SUSE. Installing from the MapR repository is generally the easiest installation method, but requires the greatest amount of bandwidth. With this method, each node is connected to the Internet to download the required packages.

When setting up a repository for the new version, leave in place the repository for the existing version because you might need to use it again.

Prepare packages and repositories on every node, or on a single node if keyless SSH is set up for the root user.

Set up repositories by completing the steps for your RedHat/Oracle Linux/CentOS, SUSE, or Ubuntu distribution.

For information on repositories and packages for MapR software and Hadoop Ecosystem tools, see MapR Repositories and Packages on page 128.

Repositories for Core Software

HPE hosts `rpm` and `deb` repositories for installing the core software using Linux package management tools. For every release of the core software, a repository is created for each supported platform.

Platform-specific upgrade repositories are hosted at: https://package.mapr.hpe.com/releases/v<version>/<platform>/mapr-v<version>GA-upgrade.<rpm/deb>.tgz.

Repositories for Ecosystem Tools

HPE hosts `rpm` and `deb` repositories for installing ecosystem tools, such as Flume, Hive, Myriad, Oozie, Pig, and Sqoop. At any given time, the recommended versions of ecosystem tools that work with the latest version of core software are available here.

These platform-specific repositories are hosted at the following locations:

- https://package.mapr.hpe.com/releases/MEP/MEP-<version>/<platform>

  Note: The MEP-<version> directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

Set Up the Internet Repository: RHEL/CentOS
This section describes how to set up a `maprtech.repo` file before adding an Internet repository.

Before upgrading your cluster software, you need to set up or update a `maprtech.repo` file. The `baseurl` properties of the `maprtech.repo` specify the URLs of the packages you want to install. To access the URLs for all release packages, see Packages and Dependencies for MapR Software.

1. Change to the `root` user or use `sudo`.

2. Create a text file called `maprtech.repo` in the `/etc/yum.repos.d/` directory with the following content, replacing `<version>` with the version of software that you want to install.

```
[maprtech]
name=MapR Technologies
baseurl=https://package.mapr.com/releases/v<version>/redhat/
     enabled=1
     gpgcheck=0
     protect=1

[maprecosystem]
name=MapR Technologies
baseurl=https://package.mapr.com/releases/MEP/MEP-<version>/redhat
    enabled=1
    gpgcheck=0
    protect=1
```

3. If your connection to the Internet is through a proxy server, set the `http_proxy` environment variable before installation.
   - Method 1:

     ```
     http_proxy=http://<host>:<port>
     export http_proxy
     ```

   - Method 2: Set the value for the `http_proxy` environment variable by adding settings to the `/etc/yum.conf` file.

     ```
     proxy=http://<host>:<port>
     proxy_username=<username>
     proxy_password=<password>
     ```

4. Download and install EPEL (Extra Packages for Enterprise Linux) if you use the `mapr-metrics` service; otherwise, skip this step.
   - RHEL/CentOS 6.x:

     ```
     wget http://download.fedoraproject.org/pub/epel/6/x86_64/
     epel-release-6-8.noarch.rpm
     rpm -Uvh epel-release-6*.rpm
     ```

   - RHEL/CentOS 7.0

     ```
     wget http://dl.fedoraproject.org/pub/epel/
     epel-release-latest-7.noarch.rpm
     ```

Set Up the Internet Repository: SUSE

This section describes how to add an internet repository using a zypper command.

As root or sudo, you add the repository for the latest packages using a zypper command to specify URLs to the packages. See the Packages and Dependencies for MapR Software for the URLs for all release packages.

1. Change to the root user or use `sudo`.

last-updated: Apr 11, 2022

2. Add the repository for MapR packages, replacing <version> with the software version that you want to install.
   ```
   zypper ar https://package.mapr.hpe.com/releases/v<version>/suse/ maprtech
   ```
3. Add the repository for ecosystem packages.
   ```
   zypper ar https://package.mapr.hpe.com/releases/MEP/MEP-<version>/suse/
   maprecosystem
   ```
4. If your connection to the Internet is through a proxy server, set the `http_proxy` environment variable before installation.
   ```
   http_proxy=http://<host>:<port>
   export http_proxy
   ```
5. Update the system package index.
   ```
   zypper refresh
   ```

Set Up the Internet Repository: Ubuntu

This section describes how to add repositories to the `sources.list`.

As `root` or using `sudo`, you add the repositories to the `sources.list`. The `sources.list` specifies the URLs to the packages that you want to install. See Packages and Dependencies for MapR Software on page 68 for the URLs for all release packages.

1. Change to the `root` user or use `sudo`.
2. Add the following lines to `/etc/apt/sources.list`, replacing `<version>` with the version that you want to install.

   Release 7.0.0 (with EEP 8.1.0) and later
   ```
   deb https://package.mapr.hpe.com/releases/v7.0.0/ubuntu/ binary bionic
   deb https://package.mapr.hpe.com/releases/MEP/MEP-8.1.0/ubuntu/ binary
   bionic
   ```

   Release 5.2.1 through 6.2.0
   ```
   deb https://package.mapr.hpe.com/releases/v<version>/ubuntu/ binary
   trusty
   deb https://package.mapr.hpe.com/releases/MEP/MEP-<version>/ubuntu/
   binary trusty
   ```

   Release 5.2 and earlier
   ```
   deb https://package.mapr.hpe.com/releases/<version>/ubuntu/ mapr optional
   deb https://package.mapr.hpe.com/releases/MEP/MEP-<version>/ubuntu/
   binary trusty
   ```
3. Update the package indexes:
   ```
   apt-get update
   ```

4. If your connection to the Internet is through a proxy server, add the following lines to `/etc/apt/apt.conf`:

```
Acquire {
Retries "0";
HTTP {
Proxy "http://<user>:<password>@<host>:<port>";
};
};
```

Using a Local Repository
If you install a MapR cluster from a local repository, you do not need an internet connection.

You can set up a local repository on each node to provide access to installation packages. With this method, nodes do not require internet connectivity. The package manager on each node installs from packages in the local repository. To set up a local repository, nodes need access to a running web server to download the packages.

Set up repositories by completing the steps for your RedHat/Oracle Linux/CentOS, SUSE, or Ubuntu distribution.

Platform-specific upgrade repositories are hosted at: https://package.mapr.hpe.com/releases/v<version>/<platform>/mapr-v<version>GA-upgrade.<rpm/deb>.tgz.

For more information about repositories and packages for MapR software and Hadoop Ecosystem tools, see MapR Repositories and Packages on page 128.

Set Up the Local Repository: RHEL/CentOS

To create a local repository, download your files from the Internet and then add the repositories to each node in the cluster.

You create a local repository from files that you download from the internet, and then add the repositories to each node in the cluster. The files that you download differ from version to version. Using a `maprtech.repo` file, you specify URLs of the RHEL/CentOS packages. See the Packages and Dependencies for MapR Software for the URLs for all release packages.

1. Log in as root on the node or use `sudo`.
2. Create the following directory if it does not exist: `/var/www/html/yum/base`
3. On a computer that is connected to the internet, download the following files, substituting the appropriate `<version>` and `<datestamp>`:

   For example:

```
https://package.mapr.hpe.com/<product_package>.rpm.tgz
```

4. Copy the files to `/var/www/html/yum/base` on the node, and extract them there.

   For example:

```
tar -xvzf <product_package>.rpm.tgz
```

5. Create the base repository headers.

```
createrepo /var/www/html/yum/base
```

6. Verify that the content of the new `/var/www/html/yum/base/repodata` directory contains `filelists.xml.gz`, `other.xml.gz`, `primary.xml.gz`, and `repomd.xml`.

7. On each node, create a text file called `maprtech.repo` in `/etc/yum.repos.d`.

```
[maprtech]
name=MapR Technologies, Inc.
baseurl=http://<host>/yum/base
enabled=1
gpgcheck=0
```

Set Up the Local Repository: SUSE

To create a local repository, download your files from the internet and then add the repositories to each node in the cluster.

You create a local repository from files that you download from the internet, and then add the repositories to each node in the cluster. The files that you download differ from version to version. See the Packages and Dependencies for MapR Software for the URLs for all versions.

1. Login as root on the node or use `sudo`.
2. Create the following directory if it does not exist: `/var/www/html/zypper/base`
3. On a computer that is connected to the internet, download the following files, substituting the appropriate `<version>` and `<datestamp>`.

   For example:

```
https://package.mapr.hpe.com/releases/<version>/suse/
mapr-<version>GA.rpm.tgz
```

4. Copy the files to `/var/www/html/zypper/base` on the node, and extract them there.

   For example:

```
tar -xvzf <product_package>.rpm.tgz
```

5. Create the base repository headers.

   `createrepo /var/www/html/zypper/base`
6. Verify that the new `/var/www/html/zypper/base/repodata` directory contains the following files.

   `filelists.xml.gz`, `other.xml.gz`, `primary.xml.gz`, `repomd.xml`
7. Add the repositories for core and the ecosystem packages to each node in the cluster.

   `zypper ar http://<host>/zypper/base/ maprtech`

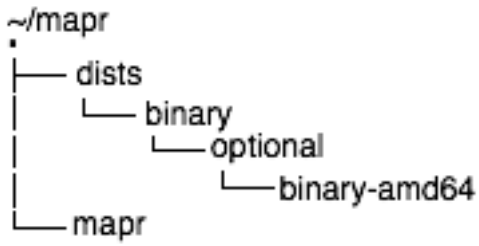
Set Up the Local Repository: Ubuntu

To create a local repository, download your files from the internet and then add the repositories to each node in the cluster.

You create a local repository from files that you download from the internet, and then add the repositories to each node in the cluster. The files that you download differ from version to version. See the Packages and Dependencies for MapR Software for the URLs for all versions.

1. Login as root on the machine where you will set up the repository.
2. Change to the directory `/root` and create the following directories within it:

last-updated: Apr 11, 2022



3. On a computer that is connected to the internet, download the following files, substituting the appropriate `<version>` and `<datestamp>`.

   For example:

   ```
   https://package.mapr.hpe.com/releases/v<version>/ubuntu/
   mapr-v<version>GA-upgrade.deb.tgz
   ```

4. Copy the files to `/root/mapr/mapr` on the node, and extract them there.

   For example:

   ```
   tar -xvzf <product_package>.deb.tgz
   ```

5. Navigate to the `/root/mapr/` directory.

6. Use `dpkg-scanpackages` to create `Packages.gz` in the `binary-amd64` directory.

   ```
   dpkg-scanpackages . /dev/null | gzip -9c > ./dists/binary/optional/
   binary-amd64/Packages.gz
   ```

7. Move the entire `/root/mapr` directory to the default directory served by the HTTP server (for example, `/var/www`) and make sure the HTTP server is running.

8. Add the following line to `/etc/apt/sources.list` on each node, replacing <host> with the IP address or hostname of the node where you created the repository.

   ```
   deb http://<host</mapr binary optional
   ```

9. On each node update the package indexes (as root or with `sudo`).

   ```
   apt-get update
   ```

10. Use `apt-get` to install MapR software and Hadoop ecosystem components from the local repository on each node.

Set Up Individual Package Files

This section describes how to upgrade MapR software using individual package files.

You can download package files, store them locally, and then install MapR from the files. This option is typically used to upgrade clusters that are not connected to the internet.

Platform-specific upgrade repositories are hosted at: https://package.mapr.hpe.com/releases/v<version>/<platform>/mapr-v<version>GA-upgrade.<rpm/deb>.tgz.

See MapR Repositories and Packages on page 128 for more information about repositories and packages.

To upgrade MapR using individual package files, you need to first pre-install the MapR package dependencies on each node. See the Packages and Dependencies for MapR Software for the dependency

last-updated: Apr 11, 2022

packages required for the cluster services that you are installing. Manually download the packages and install them.

1. Using a machine connected to the internet, download the tarball for the MapR components and the Hadoop ecosystem components, substituting the appropriate `<platform>`, `<version>`, and `<datestamp>`.

   For example: `https://package.mapr.hpe.com/releases/v<version>/ubunto/mapr-v<version>GA-upgrade.deb.tgz`

2. Extract the tarball to a local directory, either on each node or on a local network accessible by all nodes.

   ```
   tar -xvzf <product_package>.tgz
   ```

Upgrading MapR Core
The following topics describe offline and manual upgrades, and manual rolling upgrades.

Note: After completing your upgrade, see Post-Upgrade Steps for MapR Core on page 322

Offline and Manual Upgrade Procedure
The offline, manual upgrade procedure is suitable for upgrading small clusters. On large clusters, these steps are commonly performed on all nodes in parallel using scripts or remote management tools.

This procedure assumes that you have planned and prepared for the upgrade as described earlier. This procedure also assumes that the cluster meets prerequisites, including the correct JDK for the core version to which you are upgrading. For more information, see the JDK Support Matrix.

Note: An offline upgrade is performed as the `root` user or with sudo.

At the end of this procedure, you use `yum update` or `zypper update` on RHEL/CentOS or SLES to upgrade the packages. Ignore any warnings that certain packages are not installed. Packages will be upgraded correctly, and no additional packages will be installed.

1. Notify stakeholders of the impending upgrade, and then stop accepting new jobs and applications. Terminate running jobs and applications by running `maprcli` commands on appropriate nodes in the cluster.

   For YARN applications, use the following commands:

   ```
   # yarn application -list
            # yarn application -kill <ApplicationId>
   ```

2. Disconnect NFS mounts. Unmount the MapR NFS share from all clients connected to it, including other nodes in the cluster. This allows all processes accessing the cluster via NFS to disconnect gracefully.

   For example, if the cluster is mounted at `/mapr`: `# umount /mapr`

3. Display the services on each node in the cluster, and stop ecosystem component services on the nodes.

   ```
   # maprcli node list -columns hostname,csvc
   # maprcli node services -multi '[{ "name": "hue", "action": "stop"},
   { "name": "oozie", "action": "stop"}, { "name": "hs2", "action":
   "stop"}]' -nodes <hostnames>
   ```

4. If a MapR POSIX client service is running, stop the service:

last-updated: Apr 11, 2022

- For the mapr-loopbacknfs service:

```
service mapr-loopbacknfs stop
```

- For the FUSE-based POSIX basic service:

```
service mapr-posix-client-basic stop
```

- For the FUSE-based POSIX platinum service:

```
service mapr-posix-client-platinum stop
```

5. Determine where CLDB and ZooKeeper services are installed.
6. Stop Warden on CLDB nodes first, and then on all remaining nodes.
7. Stop ZooKeeper on all nodes where it is installed.
8. Ensure that no stale cluster processes are running. If so, stop the processes:

```
ps -ef | grep mapr
pkill -u mapr
```

9. Remove any existing patches:
   a. Run one of the following commands to determine if a patch is installed.
      - RHEL/CentOS and SLES: `rpm -qa mapr-patch`
      - Ubuntu: `dpkg -l | grep mapr-patch`

      If the command displays no output, no patch is installed.
   b. If one or more patches are installed, run one of the following commands to remove the patches:
      - RHEL/CentOS or SLES: `sudo rpm -e mapr-patch`
      - Ubuntu: `sudo apt-get -y remove mapr-patch`
10. Upgrade core packages by installing the appropriate package key.
    - RHEL/CentOS: `sudo rpm --import https://package.mapr.hpe.com/releases/pub/maprgpg.key`
    - SLES: No package key needed.
    - Ubuntu: `wget -O - https://package.mapr.hpe.com/releases/pub/maprgpg.key | sudo apt-key add -`
11. Upgrade these core component and MapR hadoop common packages on all nodes where packages exist.

    Components to upgrade are:
    - `mapr-cldb`
    - `mapr-core`
    - `mapr-core-internal`

- `mapr-fileserver`
- `mapr-gateway`
- `mapr-hadoop-core`
- `mapr-historyserver`
- `mapr-mapreduce2`
- `mapr-nfs`
- `mapr-nodemanager`
- `mapr-resourcemanager`
- `mapr-webserver`
- `mapr-zookeeper`
- `mapr-zk-internal`

When using `yum update` or `zypper update`, do not use a wildcard such as `mapr-*` to upgrade all MapR packages, which could erroneously include Hadoop ecosystem components such as `mapr-hive` and `mapr-pig`.

- RHEL/CentOS:

```
yum update mapr-cldb mapr-core mapr-core-internal mapr-gateway
mapr-fileserver mapr-hadoop-core mapr-historyserver mapr-mapreduce2
mapr-nfs mapr-nodemanager mapr-resourcemanager mapr-webserver
mapr-zookeeper mapr-zk-internal
```

- SLES:

```
zypper update mapr-cldb mapr-compat-suse mapr-core mapr-core-internal
mapr-gateway mapr-fileserver mapr-hadoop-core mapr-historyserver
mapr-mapreduce2 mapr-nfs mapr-nodemanager mapr-resourcemanager
mapr-webserver mapr-zookeeper mapr-zk-internal
```

- Ubuntu: First get a list of the MapR packages installed on the node, and then run `apt-get install` on the listed packages.

```
# dpkg --list | grep "mapr" | grep -P "^ii"| awk '{ print $2}'|tr "\n"
" "
# apt-get install <package-list>
```

12. Verify that packages were installed successfully on all nodes. Confirm that there were no errors during installation, and check that `/opt/mapr/MapRBuildVersion` contains the expected value.

For example:

```
# cat /opt/mapr/MapRBuildVersion
6.1.0.xxxxxxxxxxxxxx.GA
```

See Post-Upgrade Steps for MapR Core on page 322

Manual Rolling Upgrade Description
In a manual rolling upgrade, you upgrade the MapR software one node at a time so that the cluster as a whole remains operational throughout the process.

Note: Rolling upgrades to MapR 6.0 or later are supported only for clusters running MapR 5.2.x with EEP 3.0.1 or later. If you are upgrading from an earlier version, upgrade MapR core manually and offline. See Offline and Manual Upgrade Procedure on page 316.

Before you Upgrade

Before you begin a manual rolling upgrade, perform the following steps:

- Determine the upgrade groups. To see a list of services on each node, run the following command:

```
maprcli node list -columns hostname,csvc
```

- If the cluster is secure, the cluster admin user must have a security ticket created before running the upgrade. Otherwise, some upgrade commands will not run.

Group Upgrade Order

Upgrade cluster nodes in groups based on the services running on each node. Upgrade groups of nodes in the following order:

Group	Nodes in this Group
1	Each node only has ZooKeeper. This establishes a stable ZooKeeper quorum on the new version, which remains active through the rest of the upgrade process.
2	Each node only has a MapR gateway (`mapr-gateway-x.x.x`), or it has ZooKeeper. When upgrading from MapR 6.0 or 6.0.1 to MapR 6.1.x or later, you must upgrade the MapR gateway before fileserver.
3	Each node only has fileserver or it has fileserver, MapR gateway, and ZooKeeper.
4	Each node only has NodeManager or it has NodeManager, fileserver, MapR gateway, and ZooKeeper.
5	Each node only has ResourceManager or it has ResourceManager, NodeManager, fileserver, MapR gateway, and ZooKeeper. When you upgrade nodes in this group, upgrade nodes with the standby ResourceManagers before you upgrade the node with the active ResourceManager.
6	Each node has ResourceManager, NodeManager, fileserver, MapR gateway, and ZooKeeper.
7	Each node only has CLDB server or it has CLDB server, ResourceManager, NodeManager, fileserver, MapR gateway, and ZooKeeper. When you upgrade nodes in this group, upgrade nodes with the secondary CLDB before you upgrade the node with the primary CLDB.

Package Upgrade Order

When you upgrade each node, upgrade existing packages in the following order:

last-updated: Apr 11, 2022

- On all operating systems except SLES, upgrade the `mapr-core` package first. Subsequent packages can be done in any order.
- On SLES, upgrade the `mapr-compat-suse` package first and the `mapr-core` package second. Subsequent packages can be done in any order.

The following is a list of the primary packages:

- `mapr-cldb`
- `mapr-compat-suse` (if upgrading on SLES)
- `mapr-core-internal`
- `mapr-core`
- `mapr-fileserver`
- `mapr-gateway`
- `mapr-hadoop-core`
- `mapr-historyserver`
- `mapr-mapreduce2`
- `mapr-nfs`
- `mapr-nodemanager`
- `mapr-resourcemanager`
- `mapr-webserver`
- `mapr-zk-internal`
- `mapr-zookeeper`

What's Next

See Manual Rolling Upgrade Procedure on page 320

Manual Rolling Upgrade Procedure

Describes how to manually upgrade each node to the latest version of MapR packages.

Complete the following upgrade steps for each node in each upgrade group.

1. Download the archive file from https://package.mapr.hpe.com/releases/.
2. Extract the archive file. When you upgrade each package, be sure to specify the full path to the files in this local directory.

   ```
   tar -xzvf <archive file>
   ```

3. Run commands to determine if a patch is installed. If the commands display no output, no patch is installed.
   - RHEL/CentOS and SLES: `rpm -qa mapr-patch`
   - Ubuntu: `dpkg -l | grep mapr-patch`

4. Get the default MapReduce mode for the cluster.

   ```
   maprcli cluster mapreduce get
   ```

5. Stop CLDB if it is running on the node, before putting that node in maintenance mode. Else, the maintenance mode operation is not permitted

   ```
   maprcli node services -name cldb -action stop -nodes mapr-<node>
   ```

6. Set the node to maintenance mode.

   ```
   sudo maprcli node maintenance -nodes <hostname> -timeoutminutes 30
   ```

7. Notify the CLDB that the node is going to be upgraded.

   ```
   sudo maprcli notifyupgrade start -node <hostname>
   ```

8. Stop Warden.

   ```
   sudo service mapr-warden stop
   ```

9. If ZooKeeper is installed on the node, stop ZooKeeper.

   ```
   service mapr-zookeeper stop
   ```

10. Remove any patches installed on the node.
    - RHEL/CentOS or SLES: `sudo rpm -e mapr-patch`
    - Ubuntu: `sudo apt-get -y remove mapr-patch`

11. Upgrade each MapR package on the node based on the defined package upgrade order by running this command for each package:
    - RHEL/CentOS or SLES: `sudo rpm --quiet --force --nosignature -U </FullPathToPackage/PackageName.rpm>`
    - Ubuntu: `sudo dpkg --force-all -i </FullPathToPackage/PackageName.deb>`

      Note: During the upgrade process on Ubuntu, the system displays *dpkg* warnings about overwriting. You can ignore these warnings.

12. Configure the node.

    ```
    sudo /opt/mapr/server/configure.sh -R
    ```

13. If ZooKeeper is installed on the node, start ZooKeeper.

    ```
    service mapr-zookeeper start
    ```

14. Start Warden.

    ```
    sudo service mapr-warden start
    ```

15. Check that the CLDB is running. If output is displayed, the CLDB is running. If not, start CLDB.

    ```
    maprcli node list
    ```


last-updated: Apr 11, 2022

16. Unset maintenance mode on the node, notify the CLDB about the upgraded version, and about the finished status of the upgrade process.

```
sudo maprcli node maintenance -nodes <hostname> -timeoutminutes 0
sudo maprcli config save -values {mapr.targetversion:"`cat /opt/mapr/
MapRBuildVersion`"}
sudo maprcli notifyupgrade finish -node <hostname>
```

17. Wait for the containers to synchronize, run the following command, and check that there is no output.

```
/opt/mapr/server/mrconfig info containers resync local
```

    No output signifies that the containers are synchronized.

See Post-Upgrade Steps for MapR Core on page 322.

Finishing the MapR Core Upgrade
This section provides post-upgrade steps for MapR core.

Post-Upgrade Steps for MapR Core
After upgrading MapR core, several manual steps are required.
Considerations for Upgrades Using the MapR Installer
If you use the MapR Installer to upgrade the cluster, some post-upgrade steps for MapR core do not need to be performed. This page describes the post-upgrade steps that are performed by the MapR Installer and the steps that you must perform manually.

This information is relevant to the workflow for MapR Installer upgrades. The information on this page also applies to upgrades performed using MapR Installer Stanzas.

Post-Upgrade Step for MapR Core	Performed by MapR Installer?	Notes
Step 1: Restart and Check Cluster Services on page 323	Yes, with exceptions. See the notes.	Usually, you do not need to perform this step. When you use the MapR Installer to upgrade, the Installer runs `configure.sh` on each node, starts ZooKeeper and Warden, and sets the new cluster version. If the `env_override.sh` file is present, the Installer uses environment variables from the `env_override.sh` file. If you use the MapR Installer to upgrade, you need to perform this step manually only if you make significant changes to configuration files *after* the upgrade (see Step 2: Manually Update Configuration Files on page 325).
Step 2: Manually Update Configuration Files on page 325	No	You might need to perform this step. The MapR Installer uses default configuration-file settings and does NOT update configuration files. You must perform this step after upgrading using the MapR Installer only if you have made configuration-file customizations that you want to migrate to the upgraded cluster. Depending on the customization, you might also need to restart services or even the full cluster after updating configuration files.
Step 3: Upgrade Clients on page 326	No	You need to perform this step. Enabling some new features, such as `mfs.feature.name.container.size.control` can cause client failures if the features are enabled before the clients are upgraded.

Post-Upgrade Step for MapR Core	Performed by MapR Installer?	Notes
Step 4: Enable New Features on page 330	Yes	You do not need to perform this step. The MapR Installer enables new features as part of a version upgrade.
Step 5: Update Hadoop Configuration File on page 334	Yes	You do not need to perform this step. The MapR Installer updates the Hadoop configuration file, if necessary, as part of a version upgrade.

Step 1: Restart and Check Cluster Services
After upgrading core using either a manual offline or rolling upgrade method (not upgrading with the MapR Installer) and upgrading your ecosystem components, configure and restart the cluster and services.

Note: This task is applicable only to manual offline and rolling upgrade methods.

Important: Before restarting cluster services, upgrade any existing ecosystem packages to versions compatible with the upgraded MapR release. For more information, see EEP Components and OS Support.

This procedure configures and restarts the cluster and services, including ecosystem components, remounts the NFS share, and checks that all packages have been upgraded on all nodes.

After finishing this procedure, run non-trivial health checks, such as performance benchmarks relevant to the cluster's typical workload or a suite of common jobs. It is a good idea to run these types of checks when the cluster is idle. In this procedure, you configure each node in the cluster without changing the list of services that will run on the node. If you want to change the list of services, do so after completing the upgrade. After you have upgraded packages on all nodes, perform this procedure on all nodes to restart the cluster. Upon completion of this procedure, core services are running on all nodes.

1. Merge any custom edits that you made to your cluster environment variables into the new `/opt/mapr/conf/env_override.sh` file before restarting the cluster. This is because the upgrade process replaces your original `/opt/mapr/conf/env.sh` file with a new copy of `env.sh` that is appropriate for the data-fabric release to which you are upgrading. The new `env.sh` does not include any custom edits you might have made to the original `env.sh`. However, a backup of your original `env.sh` file is saved as `/opt/mapr/conf/env.sh<timestamp>`. Before restarting the cluster, you must add any custom entries from `/opt/mapr/conf/env.sh<timestamp>` into `/opt/mapr/conf/env_override.sh`, and copy the updated `env_override.sh` to all other nodes in the cluster. See About env_override.sh on page 2290.

2. On each node in the cluster, remove the `maprusertícket` file. For manual upgrades, the file must be removed to ensure that impersonation works properly. The `maprusertícket` file is re-created automatically when you restart Warden. For more information, see Installation and Upgrade Notes (MapR 6.1.0) on page 39.
   ```
   # rm /opt/mapr/conf/maprusertícket
   ```

3. If you are upgrading from core 6.0.x to 6.2.0, create the `ssl_truststore.pem` and `ssl_keystore.pem` files. These files are used by the Data Access Gateway, Grafana, and Hue components. This step is necessary only for manual upgrades because upgrades performed with the MapR Installer distribute the files automatically. Use these commands:
   a) Use the `manageSSLKeys.sh` utility to generate the files:
   ```
   /opt/mapr/server/manageSSLKeys.sh convert -N my.cluster.com /opt/mapr/conf/ssl_truststore /opt/mapr/conf/ssl_truststore.pem

   /opt/mapr/server/manageSSLKeys.sh convert -N my.cluster.com /opt/mapr/conf/ssl_keystore /opt/mapr/conf/ssl_keystore.pem
   ```

b) Copy the generated `ssl_keystore.pem` and `ssl_truststore.pem` files to the `/opt/mapr/conf/` directory on all the nodes in the cluster.

4. On each node in the cluster, run `configure.sh` with the `-R` option:
   ```
   # /opt/mapr/server/configure.sh -R -HS <hostname>
   ```

5. If ZooKeeper is installed on the node, start it:
   ```
   # service mapr-zookeeper start
   ```

6. Start Warden.
   ```
   # service mapr-warden start
   ```

7. Run a simple health-check targeting the filesystem and MapReduce services only. Address any issues or alerts that might have come up at this point.

8. Set the new cluster version in the `/opt/mapr/MapRBuildVersion` file by running the following command on any node in the cluster:
   ```
   # maprcli config save -values {mapr.targetversion:"`cat /opt/mapr/MapRBuildVersion`"}
   ```

9. Verify the new cluster version:
   For example:
   ```
   # maprcli config load -keys mapr.targetversion
   mapr.targetversion
   6.1.0.xxxxxxxxxx.GA
   ```

10. Remount the MapR NFS share:
    The following example assumes that the cluster is mounted at `/mapr`: `# mount -o hard,nolock <hostname>:/mapr /mapr`

11. Run commands, as shown in the example, to check that the packages have been upgraded successfully:
    Check the following:
    - All expected nodes show up in a cluster node list, and the expected services are configured on each node.
    - A master CLDB is active, and all nodes return the same result.
    - Only one ZooKeeper service claims to be the ZooKeeper leader, and all other ZooKeepers are followers.

    For example:
    ```
    # maprcli node list -columns hostname,csvc
    hostname configuredservice ip
    centos55 nodemanager,cldb,fileserver,hoststats 10.10.82.55
    centos56 nodemanager,cldb,fileserver,hoststats 10.10.82.56
    centos57 fileserver,nodemanager,hoststats,resourcemanager 10.10.82.57
    centos58 fileserver,nodemanager,webserver,nfs,hoststats,resourcemanager
    10.10.82.58
    ...more nodes...

    # maprcli node cldbmaster
    cldbmaster
    ServerID: 8851109109619685455 HostName: centos56
    ```

```
# service mapr-zookeeper status
Redirecting to /bin/systemctl status mapr-zookeeper.service
●mapr-zookeeper.service - MapR Technologies, Inc. zookeeper service
   Loaded: loaded (/etc/systemd/system/mapr-zookeeper.service; enabled;
vendor preset: disabled)
   Active: active (running) since Wed 2021-05-26 09:18:54 PDT; 1 months
9 days ago
  Process: 2215 ExecStart=/opt/mapr/initscripts/zookeeper start
(code=exited, status=0/SUCCESS)
 Main PID: 2510 (java)
    Tasks: 0 (limit: 410335)
   Memory: 4.5M
   CGroup: /system.slice/mapr-zookeeper.service

●2510 /usr/lib/jvm/java-11-openjdk-11.0.9.11-3.el8_3.x86_64/bin/
java -Dzookeeper.log.dir=/opt/mapr/zookeeper/zookeeper-3.5.6/
logs -Dzookeeper.lo>

May 26 09:18:53 <node> systemd[1]: Starting
MapR Technologies, Inc. zookeeper service...
May 26 09:18:53 <node> su[2459]: (to mapr) root on none
May 26 09:18:53 <node> su[2459]: pam_unix(su:session):
session opened for user mapr by (uid=0)
May 26 09:18:53 <node> zookeeper[2215]: JMX disabled by user request
May 26 09:18:53 <node> zookeeper[2215]: Using
config: /opt/mapr/zookeeper/zookeeper-3.5.6/conf/zoo.cfg
May 26 09:18:54 <node> zookeeper[2215]: Starting zookeeper ... STARTED
May 26 09:18:54 <node> su[2459]: pam_unix(su:session):
session closed for user mapr
May 26 09:18:54 <node> systemd[1]: Started
MapR Technologies, Inc. zookeeper service.
```

Step 2: Manually Update Configuration Files
After upgrading MapR core using a manual offline or rolling upgrade method, update your configuration files.

Note: This task is applicable to all manual upgrade methods: offline and rolling upgrades.

Important: After upgrading MapR but before enabling new features, all nodes in your cluster must be upgraded, and all configuration files must be updated.

To manually update the configuration files:

1. On all nodes, manually merge new configuration settings from the `/opt/mapr/conf.new/warden.conf` file into the `/opt/mapr/conf/warden.conf` file.
2. For secure clusters: On all nodes, manually copy `/opt/mapr/conf.new/mapr.login.conf` to the `/opt/mapr/conf/` directory, and set the file permissions to 0644. This file contains Zookeeper security information.
3. On all nodes, manually merge new configuration settings from the files in the `/opt/mapr/conf/conf.d.new/` directory to the files in the `/opt/mapr/conf/conf.d/ directory`.

4. Manually merge the port and authentication configuration information in the `/opt/mapr/conf/web.conf` directory from the pre-6.0 release version to the `/opt/mapr/apiserver/conf/properties.cfg` file of the upgraded release version.

   For example, the following from the `/opt/mapr/conf/web.conf` file from the pre-6.0 version must be manually copied over to the `/opt/mapr/apiserver/conf/properties.cfg` file of the new version:

   ```
   # HTTPS Settings
   mapr.webui.https.port=8443
   mapr.rest.auth.methods=kerberos,basic // if kerberos auth
   ```

5. Manually merge new configuration settings in `/opt/mapr/conf/fuse.conf` file with custom settings in `/opt/mapr/conf/fuse.conf.backup` file and restart FUSE for the settings to take effect.

   After the upgrade, on all supported operating systems other than Ubuntu, the new `fuse.conf` file and a backup copy of the `fuse.conf` file from prior version named `fuse.conf.backup` are available in the `/opt/mapr/conf` directory. You can find the new parameters with default values in the new `fuse.conf` file and your custom settings from the prior version in the `fuse.conf.backup` file. On Ubuntu, you can find the new `fuse.conf` file in the `/opt/mapr/conf` directory and by default, there is no backup copy of the `fuse.conf` file from prior version unless you created one before upgrade.

Note: When hadoop common is updated along with MapR core, a new directory is created for the new hadoop common version and the configuration files in the existing `/opt/mapr/hadoop/hadoop-2.x.x` directory are automatically copied into the active directory associated with the new hadoop 2.x.x directory. For example, when you upgrade from 4.1 to 5.x, configuration files that were in the hadoop-2.5.1 directory are copied into the hadoop-2.7.1 directory.

Step 3: Upgrade Clients
After you upgrade your cluster, you may also need to upgrade your MapR client or MapR POSIX client.

When you upgrade the cluster, consider if your MapR client or your MapR POSIX client needs to be upgraded as well. See Planning Upgrades to MapR Clients on page 301.

Additionally, MapR POSIX loopbacknfs clients can be migrated to FUSE-based POSIX clients on Basic POSIX client packages. MapR POSIX loopbacknfs clients can *not* be migrated to Platinum POSIX client packages.

See Migrating to FUSE-based POSIX Clients for more information.

Note: Basic and Platinum POSIX client packages are recommended for fresh installation and for all new clusters.

Upgrading the MapR Client

Depending on which MapR client you want to update, you will either need to install and reconfigure or perform a package upgrade.

To get a newer version of the Windows or Mac OS X MapR client, install the newer MapR client and reconfigure it. To get a newer version of the Linux MapR client, perform a package upgrade.

Upgrading the MapR Client on a Linux Server

This section describes how to upgrade the MapR client on a Linux Server.

To upgrade the MapR client on an RHEL, SLES, or Ubuntu server, you must upgrade the `mapr-client` package. When you upgrade the MapR client packages on the server, the configuration files in the `/opt/mapr/hadoop/hadoop-2.x.x` directory are automatically copied into the active directory associated with the Hadoop 2.x.x directory.

1. Remove any currently installed client patches. For example:

- **On RedHat and CentOS**

```
yum remove mapr-patch-client-<version>
```

- **On Ubuntu**

```
apt-get remove mapr-patch-client-<version>
```

- **On SLES**

```
zypper remove mapr-patch-client-<version>
```

2. Configure the repository to point to the target release and operating system.
3. Run the following command to upgrade the client package:
   - On RedHat / CentOS: `yum update mapr-client`
   - On SLES: `zypper update mapr-client`
   - On Ubuntu: `apt-get install mapr-client`

Related tasks

Upgrading the MapR loopbacknfs POSIX Client on a Linux Server on page 328
This section describes how to upgrade the MapR loopbacknfs POSIX Client on a Linux server.

Upgrading the MapR Client on Windows

This section provides instructions on how to upgrade the MapR client on Windows.

When you upgrade the MapR client on Windows, you need to rename the existing client directory, install the new version, and then merge the configuration files.

1. Rename the existing client installation directory. For example, you can rename `\opt\mapr` to `\opt_old\mapr.`
2. Complete the installation steps in Installing the MapR Client on Windows on page 397.
3. To retain existing configurations and accepts new defaults, merge the contents of the following directories in the previous installation with the ones in the new installation:
   - `%MAPR_HOME%\hadoop\hadoop-2.x.x\etc\hadoop`
   - `%MAPR_HOME%\hadoop\hadoop-0.20.0\conf`

Upgrading the MapR Client in Mac OS X

This section describes how to upgrade the MapR client on Mac OS X.

When you upgrade the MapR client on Mac OS X, you need to rename the existing client directory, install the new version, and then merge the configuration files.

1. Rename the existing client installation directory. For example, you can rename `/opt/mapr` to `/opt_old/mapr.`
2. Complete the installation steps in Installing the MapR Data Platform Client on Mac OS X on page 395.
3. To retain existing configurations and accept the new defaults, merge the contents of the following directories in the previous installation with the ones in the new installation:
   - `opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop`

- `opt/mapr/hadoop/hadoop-0.20.0/conf`

Upgrading the MapR POSIX loopbacknfs Client

Perform a package upgrade to get a newer version of the MapR POSIX loopbacknfs Client.

To get a newer version of the MapR POSIX loopback NFS client, perform a package upgrade. MapR POSIX loopback NFS client can be upgraded or a fresh install can be performed.

Additionally, MapR POSIX loopbacknfs client can be migrated to a FUSE-based POSIX Basic client. MapR POSIX loopback NFS clients can *not* be migrated to FUSE-based Platinum POSIX clients.

See Migrating to FUSE-based POSIX Clients for more information.

Note: Basic and Platinum POSIX client packages are recommended for fresh installation and for all new clusters.

Upgrading the MapR loopbacknfs POSIX Client on a Linux Server

This section describes how to upgrade the MapR loopbacknfs POSIX Client on a Linux server.

To upgrade the MapR loopbacknfs POSIX client on a RHEL, SLES, or Ubuntu server, you must upgrade the `mapr-loopbacknfs` package.

Note: To migrate the MapR POSIX Client to a FUSE-based POSIX client, see Migrating to FUSE-based POSIX Clients.

1. Stop the `mapr-loopbacknfs` service.

```
service mapr-loopbacknfs stop
```

2. Remove any currently installed client patches. For example:
   - On RedHat / CentOS: `yum remove mapr-patch-loopbacknfs`
   - On SLES: `zypper remove mapr-patch-loopbacknfs`
   - On Ubuntu: `apt-get remove mapr-patch-loopbacknfs`
3. Upgrade the `mapr-loopbacknfs` package.
   - On RedHat / CentOS: `yum update mapr-loopbacknfs`
   - On SLES: `zypper update mapr-loopbacknfs`
   - On Ubuntu: `apt-get install mapr-loopbacknfs`
4. Update the cluster configuration information in the `mapr-loopbacknfs.new` file.

```
vi /usr/local/mapr-loopbacknfs/initscripts/mapr-loopbacknfs.new
```

5. Copy `mapr-loopbacknfs.new` to the `mapr-loopbacknfs` script

```
cp
/usr/local/mapr-loopbacknfs/initscripts/mapr-loopbacknfs.new
/usr/local/mapr-loopbacknfs/initscripts/mapr-loopbacknfs
```

6. Start the loopbacknfs service.

```
service mapr-loopbacknfs start
```

7. Check the status of the loopbacknfs service.

```
service mapr-loopbacknfs status
```

Troubleshooting MapR loopbacknfs POSIX Client Upgrades

If you are having difficulty upgrading the MapR POSIX client, it might be due to a shared-memory-segment lock.

If the `mapr-loopbacknfs` service fails to start after an upgrade, use the following steps to determine if a shared-memory-segment lock was the cause of the failure:

1. Open the `loopbacknfs.log` file. The loopbacknfs.log file is in the following directory: `/usr/local/mapr-loopbacknfs/logs/`
2. Check for the following string: `Create/Attach to shm failed`
3. If the string is present, perform the following steps:
   a) Run the following command to identify the `shmid` of the lock: `ipcs -m | grep 0x0000161c`
   b) Run the following command to remove the lock:`ipcrm -m <shm id>`
   c) Start the `mapr-loopbacknfs` service.

Migrating to FUSE-Based POSIX Clients

This section describes how to migrate from the loopbacknfs POSIX client to the FUSE-based POSIX basic client.

The FUSE-based Basic or Platinum POSIX client package is recommended for fresh installation and for all new clusters. If you are currently running the loopbacknfs POSIX client, you can migrate to the FUSE-based POSIX basic client package. By default, the system will only allow migrating the existing loopbacknfs POSIX client to the FUSE-based POSIX basic client. If you choose to upgrade using the existing loopbacknfs POSIX client licenses, by default, the system upgrades the selected nodes to the (paid) FUSE-based POSIX basic client. You cannot use existing licenses to migrate to the Platinum POSIX client package.

Before upgrading, ensure that the cluster has been upgraded to release 5.1.0 or higher because the FUSE-based POSIX client can only connect to clusters running release 5.1.0 or higher.

To migrate to the FUSE-based POSIX Basic client:

1. Stop the existing loopbacknfs POSIX service, unmount the mountpoint, and uninstall the `mapr-loopbacknfs` POSIX client.
   For more information, see MapR loopbacknfs POSIX Client.
2. Run the following command to remove the loopbacknfs service node from the cluster:

```
/opt/mapr/bin/maprcli node remove -service nfsserver -nodes <node-name>
```

   If the host has multiple services, run the following command:

```
/opt/mapr/bin/maprcli node remove -hostids <service host id>
```

3. Install the FUSE-based POSIX basic client package.
   For more information, see Installing a POSIX FUSE Client Package on Your Machine. When you install the FUSE-based POSIX client package, the licenses for the loopbacknfs POSIX client are automatically applied to the new client package.

4. Verify the FUSE-based POSIX client license.
   For more information, see Configuring the MapR FUSE-Based POSIX Client on page 1240.

Upgrading the MapR FUSE POSIX Client on a Linux Server

This section describes how to upgrade the MapR FUSE POSIX client on a Linux server.

To upgrade the MapR FUSE POSIX client on a RHEL, SLES, or Ubuntu server, you must upgrade the `mapr-posix-client-basic` or `mapr-posix-client-platinum` package.

1. Stop the `mapr-posix-client-basic` or `mapr-posix-client-platinum` service. For example:

   ```
   service mapr-posix-client-basic stop
   ```

2. Remove any currently installed client patches. For example:
   - On RHEL / CentOS: `yum remove mapr-patch-posix-client-basic`
   - On SLES: `zypper remove mapr-patch-posix-client-basic`
   - On Ubuntu: `apt-get remove mapr-patch-posix-client-basic`
3. Upgrade the `mapr-posix-client-basic` or `mapr-posix-client-platinum` package:
   - On RHEL / CentOS: `yum update mapr-patch-posix-client-basic`
   - On SLES: `zypper update mapr-patch-posix-client-basic`
   - On Ubuntu: `apt-get install mapr-patch-posix-client-basic`
4. Update the cluster configuration information in the `/opt/mapr/conf/fuse.conf` file:

   ```
   vi /opt/mapr/conf/fuse.conf
   ```

5. Copy any new parameters from `fuse.conf.new` to the `/opt/mapr/conf/fuse.conf` file.
6. Start the FUSE POSIX service:

   ```
   service mapr-patch-posix-client-basic start
   ```

7. Check the status of the FUSE POSIX service:

   ```
   service mapr-patch-posix-client-basic status
   ```

Step 4: Enable New Features

Describes the new features to enable after upgrading MapR core without the MapR Installer using a manual offline or rolling upgrade method.

This task applies to all manual upgrade methods: offline, rolling, and manual rolling upgrades. After a successful manual upgrade, administrators have the option to enable new features that are not enabled by default. During a fresh install, these features are enabled automatically.

Before Enabling New Features

Before enabling new features, review these important notes:

- You can obtain a list of features for your currently installed software by using the following command:

```
maprcli cluster feature list
```

- Before enabling new features, you must upgrade all nodes in the cluster and all clients that access the cluster. Do NOT enable new features if your release 6.0 or later cluster is configured with EEP 3.0.x. For release 6.0 or later, upgrade all nodes to EEP 4.0 or later before you enable release 6.0 or later features. EEP 3.0.x is supported only for upgrade purposes. See EEP Support and Lifecycle Status on page 5531.
- The `maprcli config save` command is no longer available for enabling features.

How to Enable New Features

You enable new features by using the `maprcli cluster feature enable` command. For more information about this command, see maprcli cluster commands.

HPE recommends that you enable *all* new features. Use the following command:

```
maprcli cluster feature enable -all
```

Feature Summary

The following table describes considerations for enabling some features. The table does not represent a complete list of MapR features:

Feature	Feature Name	Available as of Release	Description
Data-at-Rest Encryption	`mfs.feature.dare`	6.1	Enables support for encrypting data at rest on the MapR cluster. See Enabling Encryption of Data at Rest on page 1413 for more information.
Data Tiering	`mfs.feature.storage.tiering.support`	6.1	Enables support for offloading data to different storage tiers. See Enabling Tiering on page 957 for more information.
Name Container Threshold	`mfs.feature.name.container.size.control`	6.0.1	Enables support for setting a limit on the size of data stored in the name container for a volume.
Directcopy for Autosetup Replication, Change Data Capture and Secondary Index	`mfs.feature.db.streams.v6.support`	6.0	Enables the following: • MapR Database tables and MapR Event Store For Apache Kafka to use the directcopy option with the autosetup replication feature. • MapR Database table Change Data Capture (CDC) feature. • MapR Database Secondary Index feature.

last-updated: Apr 11, 2022

Feature	Feature Name	Available as of Release	Description
Enforce Guaranteed Minimum Replication	`mfs.feature.enforce.min.replication`	6.0	Enables support for enforcing minimum number of replicas for (read-write) volumes during write operations. **Note:** Do not enable this feature before upgrading all the nodes in the cluster. If you enable this feature before upgrading all the nodes, MapR File System shuts down on the nodes that have not yet been upgraded.
CLDB Snapshot Improvements	`mfs.feature.snapshotdb.lite`	6.0	This feature stays disabled even after you enable it, till you perform a CLDB failover. The feature is enabled only after the CLDB failover is complete, after which you can experience significant performance improvements for snapshot create and delete operations.
External IPs for CLDB	`cldb.feature.external.ip`	6.0	Enables support for external IP addresses and port forwarding. Set the environment variables (as described here) before enabling this feature. After enabling this feature, perform a CLDB failover to allow MapR File System to re-register.
Container Identity Reuse	`cldb.feature.cid.reuse`	5.2.1	Support for container identity reuse.
Fast inode Scan for Mirroring	`mfs.feature.fastinodescan.support`	5.2.1	Enables fast mirroring when there are large numbers of files with few changes.
Streams Connect Support	`mfs.feature.streams.connect.support`	5.2.1	Enables support for Kafka Connect in the distributed mode.
Extended Attributes	`mfs.feature.fileace.support`	5.2	Enables support for adding, retrieving, and removing extended attributes on files and directories.
Hardlinks	`mfs.feature.hardlinks.support`	5.2	Enables support for retrieving hard links on files.
Access Control Expressions for MapR File System	`mfs.feature.fileace.support`	5.1	Enables the setting of Access Control Expressions on filesystem and whole volume data.
MapR Event Store For Apache Kafka and MapR Database as a document database	`mfs.feature.db.json.support`	5.1	Enables the use of MapR Streams and MapR Database as a Document Database on page 507.
MapR Auditing	`mfs.feature.audit.support`	5.0	Logs audit records of cluster-administration operations and operations on directories, files, and tables.

Feature	Feature Name	Available as of Release	Description
MapR Volume Upgrade	`mfs.feature.volume.upgrade` `mfs.feature.rwmirror.support`	5.0	Enables support for promotable mirrors on both old-format and new-format volumes.
MapR Database Table Replication	`mfs.feature.db.repl.support`	4.1	Enables support for MapR Database table replication.
Promotable Mirror Volumes	`mfs.feature.rwmirror.support`	4.0.2	Enables support for promotable mirror volumes.
Reduce On-Disk Container Size	`cldb.reduce.container.size`	4.0.2	Reduces the space required on-disk for each container. The reduction of the on-disk container size takes effect after the CLDB service restarts or fails over. **Note:** After enabling this feature on a cluster with more than a million containers, it may take some time for the initial failover to complete, as the CLDB rewrites container location tables and storage pool container map tables. However, this delay does not reoccur with any subsequent failovers.
Bulk Loading of Data to MapR Database Tables	`mfs.feature.db.bulkload.support`	3.1.1	Enables support for bulk loading of data to MapR Database tables. Used when upgrading from MapR version 3.1 or earlier.

last-updated: Apr 11, 2022

Feature	Feature Name	Available as of Release	Description

Access Control Expressions and Table Region Merges	`mfs.feature.db.ace.support`
	`mfs.feature.db.regionmerge.support`
	`mfs.feature.filecipherbit.support`
		3.1	The following features enable support for Managing Access Control Expressions on page 1448 (ACEs) and table region merge on page 1847. Used when upgrading from MapR version 3.0.x.
			`mfs.feature.db.ace.support`
			`mfs.feature.db.regionmerge.support`
			These features are automatically enabled with a fresh install or when you upgrade from a version earlier than 3.0.x.
			Important: After enabling ACEs for MapR tables, table access is enforced by table ACEs instead of the filesystem. As a result, all newly created tables are owned by root and have their mode bits set to 777.
			The following feature enables encryption of network traffic to or from a file, directory, or MapR Database table. This feature is enabled after you enable security features on your cluster.
			`mfs.feature.filecipherbit.support`
			Warning: Clusters with active security features experience job failures until this configuration value is set.

Step 5: Update Hadoop Configuration File
When manually upgrading (manual offline and manual rolling upgrade) from version 4.x to version 6.x and switching from classic mode to YARN mode, you must manually update the Hadoop configuration file.

This task is applicable only if you are upgrading from any version of 4.x to any version of 6.x and you did not convert from classic mode to YARN mode before the upgrade. In addition, this step is applicable for only the manual offline upgrade and the manual rolling upgrade. This step is not applicable when using the MapR Installer upgrade method.

1. Change directory to **/opt/mapr/conf** directory.

```
cd /opt/mapr/conf/
```

2. Edit the Hadoop configuration file

```
vim hadoop_version
```


last-updated: Apr 11, 2022

3. Change the `yarn_version` parameter to the actual version of Hadoop. For example:

```
yarn_version=2.7.0
```

Installing Additional MapR Core Features
Some features can require the installation of additional packages after an upgrade to a new release.

Installing Monitoring After an Upgrade

MapR Monitoring, also known as the Spyglass initiative, provides the ability to collect, store, and view metrics and logs for nodes, services, and jobs/applications. You can only install MapR Monitoring after you upgrade ecosystem components.

- To install MapR Monitoring without the MapR Installer, see Step 8: Install Metrics Monitoring on page 162 and Step 9: Install Log Monitoring on page 165.
- To install MapR Monitoring with the MapR Installer, launch the MapR Installer URL (`https://<hostname/IPaddress>:9443`) and select the Incremental Install option. With the incremental installation option, you can install new packages and also install MapR Monitoring.

Securing the Upgraded Cluster
Nonsecure clusters can be secured after the upgrade process.

If your cluster was nonsecure before you upgraded it, the cluster will remain nonsecure after the upgrade. If you used the manual-rolling or offline-manual workflows to upgrade and you want to add security, see Getting Started with MapR Security on page 1405.

If you used the MapR Installer workflow to upgrade, and you want to add security, see Using the Enable MapR Secure Cluster Option on page 5427.

Upgrading MapR Ecosystem Packs

Describes how to upgrade MapR Ecosystem Packs (EEPs), either as part of a MapR core upgrade or to take advantage of a new EEP for the current version of MapR core.

An MapR Ecosystem Pack (EEP) provides a set of ecosystem components that are fully tested by HPE to be interoperable except where noted. For more information about EEPs, see MapR Ecosystem Packs.

Planning MapR Ecosystem Pack (EEP) Upgrades
The set of ecosystem components that you run in the cluster must all belong to the same EEP.

As of release 5.2, you must install ecosystem components as part of an EEP. You will be offered packs to install that contain selected component versions. After upgrading, you may want to upgrade to a more recent EEP to get the latest patch releases or newer versions of ecosystem components.

Most MapR core versions support multiple EEPs, but the set of ecosystem components that you run in the cluster must all belong to the same EEP. You cannot selectively upgrade components. When you upgrade an EEP, all components are replaced with the versions contained in the newly selected EEP.

To compare ecosystem component versions across EEPs, see Component Versions for Released EEPs.

For EEP lifecycle information, see EEP Support and Lifecycle Status on page 5531.

For details about the ecosystem components available in each EEP and the list of EEPs supported by your MapR core version, see the EEP Release Notes on page 5658.

For API or behavioral changes associated with new ecosystem components, see the documentation for the individual component under Ecosystem Components.

Preparing to Upgrade the MapR Ecosystem Pack
Complete these pre-upgrade steps for each ecosystem component in the EEP that you want to upgrade.

For the components provided in each EEP, see the EEP Release Notes on page 5658.

These steps are intended primarily for ecosystem-component upgrades performed manually (without the MapR Installer). However, you need to perform some pre-upgrade steps even if you are upgrading using the MapR Installer.

Stopping each service is optional if you are using the MapR Installer because the Installer stops all services before upgrading. But the MapR Installer does NOT back up configuration files.

Important: Regardless of the upgrade method that you use, follow the pre-upgrade steps for backing up your configuration files before upgrading. The upgrade process replaces your current configuration files with new configuration files that contain default values for the release to which are upgrading. Any custom settings are lost and must be migrated manually as part of the post-upgrade steps.

Pre-Upgrade Steps for Drill
Complete the following steps before you upgrade Drill with or without the MapR Installer.

Starting in Drill 1.11, Drill is automatically secured when installed in a release 6.x cluster with the default MapR security configuration. The default security configuration uses MapR security (`mapr` tickets) to provide authentication, authorization, and encryption for cluster security. See Securing Drill for more information.

Note: The default security feature introduced in release 6.0 is not supported with Drill-on-YARN.

Note: See Component Versions for Released EEPs for version support in each EEP release.

Preserving Custom Security When Upgrading Core and Drill

You can perform a manual upgrade, for example from a secured release 5.2 cluster to 6.0, to preserve your security settings from 5.2. When you upgrade, a special file, /opt/mapr/conf/.upgrade_from, checks for security settings. If security is set, the same settings carry over to 6.x.

During a custom upgrade, Drill configurations are not carried over. You must either reconfigure all of your Drill settings, or save your previous settings and then override the default Drill settings. You can manually secure Drill either by copying over the old drill-override.conf file into /opt/mapr/drill/drill-<version>/conf or by updating the /opt/mapr/drill/drill-<version>/conf/drill-distrib.conf file with the security settings.

After you upgrade, you must run configure.sh -R to configure the cluster. When you run the configuration script, it adds a .customSecure file to the /opt/mapr/conf directory. This file calls the internal ecosystem scripts, but does not configure Drill.

If you decide you want to enable the default security option, you can do so by running configure.sh with the -secure and -forceSecurityDefaults flags, as shown:

```
/opt/mapr/server/configure.sh -forceSecurityDefaults [ -unsecure | -secure ]
 -C <CLDB_node> -Z <ZK_node>
```

Running configure.sh with these flags secures the cluster and supported ecosystem components. The internal Drill configuration script configures Drill security in the drill-distrib.conf and distrib-env.sh files. See Securing Drill.

Drill Management Service

Drill can run under the MapR Warden service or under YARN. You can upgrade Drill and continue to run Drill under the MapR Warden service. If you are currently running Drill under the MapR Warden service, you can migrate Drill to run under YARN. If you want to migrate Drill to run under YARN, see Migrate Drill to Run Under YARN.

If you are upgrading Drill to run under Warden, Drill should preserve your storage plugin and configuration files when you upgrade. However, you should backup and restore your configuration files and UDF JAR files.

Pre-Upgrade Steps

last-updated: Apr 11, 2022

Complete the following steps on Drill servers and clients before you upgrade Drill:

1. Optionally, back up storage plugin configurations:
   a. Open the Drill Web Console. The Drill node that you use to access the Web Console is a node that is currently running the Drillbit process.
   b. Click the **Storage** tab.
   c. Click **Update** next to a storage plugin.
   d. Copy the configuration to a text file, and save the file.
   e. Repeat steps **c** and **d** for each storage plugin configuration that you want to save.
2. To stop the Drillbit service on all nodes, issue the following command:

```
maprcli node services -name drill-bits -action stop -nodes <node
hostnames separated by a space>
```

Pre-Upgrade Steps for Flume
Complete the following steps before you upgrade Flume with or without the MapR Installer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You need to back up configuration files for Flume if you made configuration changes that you want to carry over to the next version.

1. Locate configuration files in `/opt/mapr/flume/flume-<version>/conf/`.
2. Copy the configuration files to a location outside the MapR installation directory.
   After upgrading, you can reapply changes to the updated Flume using the backup.

Pre-Upgrade Steps for HBase
Complete the following steps before you upgrade HBase with or without the MapR Installer.

1. **Upgrade your HBase Java applications.**
   Check your HBase applications for Java APIs that are no longer supported in HBase 1.1. See HBase Java API Support. Then, update the applications to use APIs supported by HBase 1.1 and recompile your applications with HBase 1.1.
2. **Take a snapshot of the HBase volume.** This step is applicable if you are upgrading with the MapR Installer or upgrading manually.
   The snapshot creates a backup of the volume data that you can use to recover your data in the event that corruption occurs during the upgrade process. For more information, see Creating Volume Snapshots on page 947.
3. **Optional: Create a backup copy of any configuration files that contain customized values.** This step is applicable if you are upgrading with the MapR Installer or upgrading manually.
   The configuration files are located in `/opt/mapr/hbase/hbase-<version>/conf/`. Copy any that you want to back up to another location. If you plan to upgrade with the MapR installer, copy files to a location that is outside the MapR installation directory. After upgrading, you can reapply changes to the updated HBase installation using the backup.

Considerations for Upgrading to HBase 1.1.13
Before upgrading to HBase 1.1.13, familiarize yourself with aspects of the EEP 6.3.0 HBase implementation that can affect an upgrade.

last-updated: Apr 11, 2022

Removing the Mappings Property

A release 6.0.x cluster installed using the MapR Installer contains an `hbase.table.namespace.mappings` property in the `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/core-site.xml` file. For example:

```
<property>
   <name>hbase.table.namespace.mappings</name>
   <value>*:/</value>
</property>
```

Release 6.0.x clusters support MapR Database, but they do not support HBase. Therefore, if you upgrade the cluster manually from Release 6.0.x, you need to remove this property in order to support HBase connections to both HBase and MapR Database.

If you do not remove the property, you might see the following message:

```
This client is configured to use MapR tables only. HBase status is not
available. MapR cluster status
can be viewed using the 'maprcli dashboard info' command or the UI.
```

How HBase Configuration Files Are Preserved During an Upgrade

Starting with EEP 6.3.0, existing HBase configuration files are automatically saved during an upgrade. This table describes what happens to HBase configuration files during an upgrade.

When you upgrade from HBase Version	To Version	HBase Configuration Files
1.1.8	1.1.13	Are overwritten by new configuration files.
1.1.13	1.1.13 (with patches)[1]	Are saved (not overwritten).

[1]An upgrade from HBase 1.1.13 to HBase 1.1.13 is a valid upgrade path if patches have been added to HBase 1.1.13 and you want to apply the patches.

This example shows a listing of the configuration files that are saved after an upgrade from HBase 1.1.8 to HBase 1.1.13:

```
[mapr@node2 etc]$ ls /opt/mapr/hbase/
hbase-1.1.13 hbase-1.1.8.201904050941 hbaseversion

$ ls /opt/mapr/hbase/hbase-1.1.8.201904050941/conf/
hadoop-metrics2-hbase.properties hbase-env.sh hbase-policy.xml
hbase-site.xml log4j.properties regionservers
warden.hbaserest.conf warden.hbaserest.conf.template
warden.hbasethrift.conf warden.hbasethrift.conf.template
```

Applications and Security

Existing HBase applications might need to be modified to work properly with HBase 1.1.13 in a secure MapR cluster:

HBase API Client	No changes
HBase REST	No changes if the old client used PAM (can now be changed to mapr-sasl header)
HBase Thrift over HTTP	No change if the old client used PAM (can now be changed to mapr-sasl header)

HBase Thrift over Socket	No changes

Configuring the Default Database for HBase Clients

EEP 6.3.0 introduced some minor changes to default database configuration. For details, see Configure the Default Database for HBase Clients on page 3389.

Pre-Upgrade Steps for HBase Client
Complete the following steps before you upgrade HBase Client with or without the MapR Installer.

If you made configuration changes that you want to carry over to the next version of HBase Client, you need to back up configuration files.

1. Copy the configuration files in `/opt/mapr/hbase/hbase-<version>/conf/` to a location outside the MapR installation directory.
2. After upgrading, you can reapply changes to the updated HBase Client by merging the configuration files back into `/opt/mapr/hbase/hbase-<version>/conf/`. See Post-Upgrade Steps for HBase Client on page 373 for more information.

Related concepts

Considerations for Upgrading to HBase 1.1.13 on page 337
Before upgrading to HBase 1.1.13, familiarize yourself with aspects of the EEP 6.3.0 HBase implementation that can affect an upgrade.

Pre-Upgrade Steps for Hive
Complete the following steps before you upgrade Hive with or without the MapR Installer.

You need to back up the metastore database in case an error occurs during the Hive upgrade. You also need to back up configuration files if you made configuration changes that you want to carry over to the next version.

1. Back up the metastore database.
   ```
   mysqldump -u<user> -p<passwd> <metastore_db_name> -r metastore-db-dump.sql
   ```
2. Copy the configuration files in `/opt/mapr/hive/hive-<version>/conf/` to a location outside the MapR installation directory.
   After upgrading, you can reapply changes to the updated Hive installation using the backup.
3. Stop Hive services.
   ```
   maprcli node services -name hivemeta -action stop -nodes <list of hive
   nodes>
   maprcli node services -name hs2 -action stop -nodes <list of hive nodes>
   maprcli node services -name  hcat -action stop -nodes <list of hive
   nodes>
   ```

Preserving the Hive Configuration
Starting from EEP-6.0.0, preserving of user configuration logic is built into Hive.

- For a minor version update (for example, `Hive-2.1-1803` to `Hive-2.1-1808`), user configuration from a previous version is copied to a folder with an old version timestamp and is also copied to a new version `conf` folder.
- For a major version update (for example, `Hive-2.1-1803` to `Hive-2.3-1808`), user configuration from a previous version is **only** copied to a folder with an old version timestamp.

Starting from EEP-5.0.2 and EEP-6.0.1, a logic of preserving user Warden files configuration for Hive Metastore, HiveServer2 and WebHCat are built into Hive.

- For a minor version update (for example, `Hive-2.1-1808` to `Hive-2.1-1901`), user configuration of Warden files from a previous version is copied to a folder with an old version timestamp and is also preserved in the `MAPR_HOME/conf/conf.d/` folder.
- For a major version update (for example, `Hive-2.1-1808` to `Hive-2.3-1901`), user configuration from a previous version is **only** copied to a folder with an old version timestamp.

Pre-Upgrade Steps for HttpFS
Complete the following steps before you upgrade HttpFS with or without the MapR Installer.

Stop the HttpFS service using the following command:

```
maprcli node services -name httpfs -action stop -nodes <ip_address>
```

For EEP 6.3.4 and earlier, back up existing configuration files if you made HttpFS configuration changes that you want to carry over to the next version. Typically, the following configuration files contain changes:

- `/opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/webapps/webhdfs/WEB-INF/web.xml`
- `/opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/conf/server.xml`
- `/opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/conf/tomcat-users.xml`
- `/opt/mapr/httpfs/httpfs-1.0/etc/hadoop/httpfs-site.xml`

To back up configuration files, copy the files to a location outside the MapR installation directory. After upgrading, you can reapply changes to the updated httpFS installation using the backup.

Note: To verify EEP and core version support, see EEP Support and Lifecycle Status on page 5531.

Preserving HttpFS Configuration
Preserving of user configuration logic is built into HttpFS.

- User configuration from a previous version is copied to a folder with an old version timestamp and is also copied to a new version `conf` folder.

Pre-Upgrade Steps for Hue
Complete the following steps before you upgrade Hue with or without the MapR Installer.

1. Stop the Hue service:
   ```
   maprcli node services -name hue -action stop -nodes <ip_address>
   ```
2. Create a Hue database dump as a JSON object:

   For MySQL, PostgreSQL, or Oracle

   ```
   source /opt/mapr/hue/hue-<version>/build/env/bin/activate
   hue dumpdata > ~/dump-hue-<version>.json
   deactivate
   ```

   For SQLite

   ```
   cd /opt/mapr/hue/hue-<version>/desktop
   sqlite3 desktop.db .dump > ~/dump-hue-<version>-sqlite.bak
   ```
3. Copy the configuration properties from `/opt/mapr/hue/hue-<version>/desktop/conf/` to a location outside your MapR installation directory.

   After upgrading, you can reapply changes to the updated Hue installation using the backup.

Preserving Hue Configuration
Starting from EEP-6.0.0, preserving of user configuration logic is built into Hue.

- For a major version update (for example, `Hue-3.2-1803` to `Hive-4.2-1808`), user configuration from a previous version is **only** copied to a folder with an old version timestamp (`/HUE_HOME/hue-3.12.0.201707281202`).

Pre-Upgrade Steps for Impala
Complete the following steps before you upgrade Impala with or without the MapR Installer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Stop the Impala server, statestore, and catalog services on all Impala nodes.

1. Run the following command to stop the Impala server on each Impala node:

```
$ maprcli node services -name impalaserver -action stop -nodes <node IP
addresses separated by a space>
Example:$ maprcli node services -name impalaserver -action stop -nodes
10.10.30.166
```

2. Run the following command to stop the Impala statestore instances:

```
$ maprcli node services -name impalastore -action stop -nodes <node IP
addresses separated by a space>
Example:$ maprcli node services -name impalastore -action stop -nodes
10.10.30.166
```

3. Run the following command to stop the Impala catalog instances:

```
$ maprcli node services -name impalacatalog -action stop -nodes <node IP
addresses separated by a space>
Example:$ maprcli node services -name impalacatalog -action stop -nodes
10.10.30.166
```

Preserving Impala Configuration
Starting from EEP 6.0.0, preserving of user configuration logic is built into Impala.

- For a minor version update (for example, `Impala-2.10-1803` to `Impala-2.10-1808`), user configuration from a previous version is copied to a folder with an old version timestamp and is also copied to a new version `conf` folder.
- For a major version update (for example, `Impala 2.7.0-1710` to `Impala-2.10-1808`), user configuration from a previous version is **only** copied to a folder with an old version timestamp.

Pre-Upgrade Steps for MapR Data Access Gateway
Complete the following steps before you upgrade the MapR Data Access Gateway with or without the MapR Installer.

Stop the service:

```
maprcli node services -nodes <node name> -name data-access-gateway -action
stop
```

Pre-Upgrade Steps for Livy

Complete the following steps before you upgrade Livy with or without the MapR Installer.

1. Stop the Livy service if it is installed:

   ```
   maprcli node services -name livy -action stop -nodes <ip_address>
   ```

2. If necessary, back up configuration files:

   If you made configuration changes that you want to carry over to the next version, you need to back up the configuration files. Typically, the following configuration files contain changes:

   - `/opt/mapr/livy/livy-<version>/conf/livy.conf`
   - `/opt/mapr/livy/livy-<version>/conf/livy-env.sh`

   To back up configuration files, copy the files to a location outside the MapR installation directory. After upgrading, you can reapply changes to the updated Livy installation using the backup.

Livy is included in the MapR EEP repositories beginning with EEP 4.0.0. Before EEP 4.0.0, Livy was included as a package called `mapr-hue-livy`, and released only as a part of Hue. The configuration files for `mapr-hue-livy` have a different location:

- `/opt/mapr/hue-livy/hue-livy-<version>/conf/livy.conf`
- `/opt/mapr/hue-livy/hue-livy-<version>/conf/livy-env.sh`

Note:

- Starting from EEP-6.0, for Livy upgrades, from Livy 0.3 and above, user configuration files are saved during upgrade.
- For Livy upgrades from `mapr-hue-livy 3.12`, user configuration files are NOT saved during the upgrade.
- For manual Livy upgrades from `mapr-hue-livy 3.12` on Ubuntu, you need to remove the old `mapr-hue livy` package manually.

Pre-Upgrade Steps for MapR Monitoring

Complete the following steps before you upgrade MapR Monitoring Components with or without the MapR Installer.

During an upgrade using the MapR Installer, a script backs up many of the configuration files. However, whether or not you are upgrading manually or by using the MapR Installer, it is a best practice to back up the files manually. Manual backups can help in case an error occurs or the specific file you customized is not automatically backed up by the script.

Before performing the pre-upgrade steps, note these important considerations:

- The MapR Monitoring upgrade is an offline upgrade and *not* a rolling upgrade.
- This upgrade procedure is customized for the MapR implementation of the monitoring components. Because the MapR implementation has a narrow focus and there are numerous components, the upgrade steps are simplified. MapR upgrade documentation does *not* include all of the upgrade steps that are included in the vendor documentation for each component. Before starting the upgrade process, consider familiarizing yourself with the vendor-upgrade steps to determine if your environment requires extra measures to protect data and configurations.
  - Elasticsearch upgrade
  - Kibana upgrade
  - Search Guard upgrade

last-updated: Apr 11, 2022

- Grafana upgrade

- This upgrade sequence does not implement security in the MapR Monitoring components. If the cluster you are upgrading is secure and you are upgrading to a new version of Elasticsearch, the security keys will be deleted when you upgrade the monitoring packages. You must regenerate the keys and copy them to the appropriate nodes after upgrading. The Post-Upgrade Steps for MapR Monitoring on page 379 provide links to the installation procedures containing this information.

1. Before backing up configuration files, ensure that your Elasticsearch and Kibana indexes are not affected by the upgrade:

   Note: This step assumes that log monitoring is configured. You can skip this step if your cluster is not configured for log monitoring.

   a) If you are using Elasticsearch version 2.x, upgrade your Elasticsearch index to version 6. For upgrade information, see: https://www.elastic.co/guide/en/elasticsearch/reference/current/reindex-upgrade.html

   You need to upgrade your Elasticsearch index if your cluster is running a EEP in the range 1.1 through 3.0.x. See the following table. EEPs 1.1 through 3.0.x use Elasticsearch version 2.3.3. If your cluster is running a EEP in the range 4.0.0 through 5.0.x, you are using Elasticsearch 5.4.1, and you do NOT need to upgrade the index. For more information about Elasticsearch / Search Guard version information, see this website.

MapR Core	EEP	Elasticsearch Version	SearchGuard Version	Kibana Plugin Version
6.1.0	6.1.0	6.5.3.0	24.0	17
6.1.0	6.0.x	6.2.3	23.0	14
6.0.x	4.0.0 through 5.0.2	5.4.1	N/A	N/A
5.2.x	3.0.5 and earlier	2.3.3	N/A	N/A

   For more information about the MapR Monitoring component versions included in each EEP, see Component Versions for Released EEPs on page 5586.

   b) Create a snapshot of the Kibana index to capture index information before the upgrade. This information will be restored after the upgrade. For snapshot information, see https://www.elastic.co/guide/en/elasticsearch/reference/5.6/modules-snapshots.html.

2. Before you upgrade metric monitoring components, create a backup of the configuration files to a location outside your MapR installation directory. The following configuration file-lists include files that are commonly used for configuration and may not include every file that you may have customized.
   - Collectd configuration files:
     - /opt/mapr/conf/conf.d/warden.collectd.conf
     - /opt/mapr/collectd/collectd-<version>/etc/collectd.conf
     - /etc/logrotate.d/collectd
   - Grafana configuration files:
     - /opt/mapr/conf/conf.d/warden.grafana.conf
     - /opt/mapr/grafana/grafana-<version>/etc/grafana/grafana.ini
     - /opt/mapr/grafana/grafana- <version>/etc/grafana/ldap.toml

last-updated: Apr 11, 2022

- OpenTSDB configuration files:
  - /opt/mapr/conf/conf.d/warden.opentsdb.conf
  - /opt/mapr/opentsdb/opentsdb-<version>/etc/opentsdb/opentsdb.conf
  - /opt/mapr/opentsdb/opentsdb-<version>/etc/opentsdb/logback.xml
  - opt/mapr/opentsdb/opentsdb-<version>/bin/tsdb_cluster_mgmt.sh (This file is not automatically backed up.)

3. Before you upgrade log monitoring components, create a backup of the following files to a location outside your MapR installation directory. The following configuration file lists include files that are commonly used for configuration and may not include every file that you may have customized.
   - Kibana configuration files:
     - /opt/mapr/conf/conf.d/warden.kibana.conf
     - /opt/mapr/kibana/kibana-<version>/etc/conf/kibana.js
   - fluentd configuration files:
     - /opt/mapr/conf/conf.d/warden.fluentd.conf
     - /opt/mapr/fluentd/fluentd-<version>/etc/fluentd/fluentd.conf
     - /opt/mapr/fluentd/fluentd-<version>/etc/fluentd/es_config.conf
     - /opt/mapr/fluentd/fluentd-<version>/etc/fluentd/maprfs_config.conf
     - /opt/mapr/fluentd/fluentd-<version>/etc/fluentd/grok-patterns
     - /etc/logrotate/fluentd
   - Elasticsearch configuration files:
     - /opt/mapr/conf/conf.d/warden.elasticsearch.conf
     - /opt/mapr/elasticsearch/elasticsearch-<version>/etc/elasticsearch/elasticsearch.yml
     - /opt/mapr/elasticsearch/elasticsearch-<version>/etc/elasticsearch/logging.yml
     - /opt/mapr/elasticsearch/elasticsearch-<version>/etc/elasticsearch/curator.yml
     - /opt/mapr/elasticsearch/elasticsearch-<version>/etc/elasticsearch/curator_actions/delete_indices.yml (This file is not automatically backed up.)
     - /opt/mapr/elasticsearch/elasticsearch-<version>/var/lib/MaprMonitoring/ (This directory is the default location for Elasticsearch index data. You must back up this directory unless you specified a non-default location using the `-ESDB` parameter with `configure.sh` during installation.)

4. Stop all MapR monitoring services on the cluster.
   a) To stop collectd, run the following command:

```
maprcli node services -name collectd -nodes <space separated list of
hostname/IPaddresses> -action stop
```

b) To stop Grafana, run the following command:

```
maprcli node services -name grafana -nodes <space separated list of
hostname/IPaddresses> -action stop
```

c) To stop OpenTSDB, run the following command:

```
maprcli node services -name opentsdb -nodes <space separated list of
hostname/IPaddresses> -action stop
```

d) To stop Kibana, run the following command:

```
maprcli node services -name kibana -nodes <space separated list of
hostname/IPaddresses> -action stop
```

e) To stop fluentd, run the following command:

```
maprcli node services -name fluentd -nodes <space separated list of
hostname/IPaddresses> -action stop
```

f) To stop Elasticsearch, run the following command:

```
maprcli node services -name elasticsearch -nodes <space separated
list of hostname/IPaddresses> -action stop
```

Pre-Upgrade Steps for MapR Object Store with S3-Compatible API
Complete the following steps before you upgrade the MapR Object Store with S3-Compatible API with or without the MapR Installer.

Stop the service:

```
maprcli node services -nodes <node name> -name objectstore -action stop
```

Pre-Upgrade Steps for MapR Event Store For Apache Kafka Tools
Complete the following steps before you manually upgrade MapR Event Store For Apache Kafka Tools.

Kafka REST

Run the following command to stop the Kafka REST service on each node:

```
maprcli node services -name kafka-rest -action stop -nodes <list of Kafka
REST service nodes>
```

To see how configuration files are saved during an upgrade, see Saving Kafka REST Configurations on page 3872.

Kafka Connect

Run the following command to stop the Kafka Connect service on each node:

```
maprcli node services -name kafka-connect -action stop -nodes <list of
Kafka Connect service nodes>
```

To see how configuration files are saved during an upgrade, see Saving Kafka Connect Configurations on page 3941.

Pre-Upgrade Steps for Oozie
Complete the following steps before you upgrade Oozie with or without the MapR Installer.

1. Stop any jobs or coordinators that are in a RUNNING or SUSPENDED state.
2. **For upgrades without the MapR Installer:** Stop the Oozie service.

   ```
   # maprcli node services -name oozie -action stop -nodes <node names>
   ```

3. (OPTIONAL) Back up Oozie configuration files and save them to a location outside the MapR installation. Configuration properties are located in /opt/mapr/oozie/oozie-<version>/conf/. After upgrading, you can reapply changes to the updated Oozie installation using the backup.
4. **For upgrades without the MapR Installer:** Remove the old share libraries and examples from the following directories:

   ```
   maprfs:///oozie/share
   maprfs:///user/${user.name}/examples
   ```

5. **For upgrades from EEP 3.0.x to EEP 4.0 (or later):** If you are upgrading Oozie 4.3.0 on a secure cluster and you have not already configured SSL on your Oozie server and client, you must perform the following steps:
   a) Configuring Oozie on a Secure Cluster on page 3990
   b) Configuring Oozie Clients to Use SSL (Oozie 4.3.0) on page 3991

Preserving Oozie Configuration

Starting from 1808 release, EEP 4.1.2, EEP 5.0.1, EEP 6.0.0 and newer versions, in case of a version update, configuration from a previously installed version of Oozie is stored in a folder with an old version timestamp.

Pre-Upgrade Steps for Pig
Complete the following steps before you upgrade Pig with or without the MapR Installer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You need to back up configuration properties files if you made configuration changes that you want to carry over to the next version.

1. Locate the files in `/opt/mapr/pig/pig-<version>/conf/` directory.
2. Copy the files to a location outside the MapR installation.
   After upgrading, you can reapply changes to the updated Pig installation using the backup.

Pre-Upgrade Steps for Sentry
Complete the following steps before you upgrade Sentry with or without the MapR Installer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You must stop the Sentry service before you upgrade. If you modified the configuration files in your current installation of Sentry, you can back up the configuration files to preserve the changes that you want to carry over to the next version.

To stop the Sentry service, issue one of the following commands:

- If the warden.sentry.conf exists in the /opt/mapr/conf/conf.d/ directory:

```
maprcli node services -name sentry -action stop -nodes <list of Sentry
service nodes>
```

- If the warden.sentry.conf does not exist in the /opt/mapr/conf/conf.d/ directory:

```
/opt/mapr/sentry/sentry-<SENTRY_VERSION>/bin/sentry-daemon.sh stop
```

To back up the configuration files, copy the files in `/opt/mapr/sentry/sentry-<version>/conf/` to a location outside of the MapR installation directory. After upgrading, you can reapply changes to the updated Sentry installation using the backup.

Preserving Sentry Configuration
Starting from EEP 6.0.0, preserving of user configuration logic is built into Sentry.

- For a minor version update (for example, `Sentry-1.7-1803` to `Sentry-1.7-1808`), user configuration from a previous version is copied to a folder with an old version timestamp and is also copied to a new version `conf` folder.
- For a major version update (for example, `Sentry-1.6-1707` to `Sentry-1.7-1808`), user configuration from a previous version is **only** copied to a folder with an old version timestamp.

Upgrading from EEP 6.3.1 and EEP 7.0.0
If you are upgrading from Sentry in EEP 6.3.1 or EEP 7.0.0 to Sentry in the latest EEP version, you must manually back up the `/conf` and `/logs` directories located in SENTRY_HOME. After the upgrade completes, add those directories back into the SENTRY_HOME directory.

You can locate the `/conf` and `/logs` directories in the Sentry installation directory, as shown:

```
ll /opt/mapr/sentry/sentry-1.7.0
total 76
drwxr-xr-x 2 mapr mapr  4096 Jan  5 13:34 bin
-rw-r--r-- 1 mapr mapr 15211 Jan  5 10:26 CHANGELOG.txt
drwxr-xr-x 2 mapr mapr  4096 Jan  5 13:34 conf
drwxr-xr-x 2 mapr mapr  4096 Jan  5 13:34 conf.d
drwxr-xr-x 2 mapr mapr  4096 Jan  5 13:34 conf.new
drwxr-xr-x 4 mapr mapr 12288 Jan  5 13:34 lib
-rw-r--r-- 1 mapr mapr 16000 Jan  5 10:26 LICENSE.txt
drwxr-xr-x 2 mapr mapr  4096 Jan  5 13:34 logs
-rw-r--r-- 1 mapr mapr   388 Jan  5 10:26 NOTICE.txt
-rw-r--r-- 1 mapr mapr  1580 Jan  5 10:26 README.md
drwxr-xr-x 3 mapr mapr  4096 Jan  5 13:34 scripts
```

For additional information, see Sentry on page 3817.

Pre-Upgrade Steps for Spark
Complete the following steps before you upgrade Spark with or without the MapR Installer.
Pre-Upgrade Steps for Spark Standalone

1. Copy configuration files from `/opt/mapr/spark/spark-<version>/conf` to a location outside of the MapR installation directory.

   For example, if Spark SQL is configured to work with Hive, copy the `/opt/mapr/spark/spark-<version>/conf/hive-site.xml` file to a backup directory.

2. Shut down the spark-master, spark-historyserver services (if the spark-historyserver is running).

```
maprcli node services -nodes <node-ip> -name spark-master -action stop
maprcli node services -nodes <node-ip> -name spark-historyserver -action
stop
```

3. As the `mapr` user, stop the secondary instances:

   For Spark 2.x:

```
/opt/mapr/spark/spark-<version>/sbin/stop-slaves.sh
```

   For Sparx 3.x:

```
/opt/mapr/spark/spark-<version>/sbin/stop-workers.sh
```

Pre-Upgrade Steps for Spark on YARN

1. Copy configuration files from `/opt/mapr/spark/spark-<version>/conf` to a location outside of the MapR installation directory.

   For example, if Spark SQL is configured to work with Hive, copy the `/opt/mapr/spark/spark-<version>/conf/hive-site.xml` file to a backup directory.

2. Shut down the spark-historyserver services (if the spark-historyserver is running):

```
maprcli node services -nodes <node-ip> -name spark-historyserver -action
stop
```

Preserving Spark Configuration

Starting from EEP 6.0.0, in case of a version update, configuration from a previously installed version of Spark is stored in a folder with an old version timestamp.

Pre-Upgrade Steps for Sqoop1

Review the following information before you upgrade Sqoop1 with or without the MapR Installer.

Preserving the Sqoop1 Configuration

Starting from EEP 6.0.0, in case of a version update, the configuration from a previously installed version of Sqoop is stored in a folder with an old version timestamp.

After an upgrade from Sqoop 1.4.6 to 1.4.7, configuration files are saved and overwritten by new configuration files. After an update from Sqoop 1.4.7 to a newer package of 1.4.7, configuration files are saved (but not overwritten). The following example shows a listing of the configuration files that are saved after an update from Sqoop 1.4.7 (EEP 7.0.0) to newer package of Sqoop 1.4.7 (EEP 7.0.1):

```
ls /opt/mapr/sqoop/
sqoop-1.4.7  sqoop-1.4.7.202009100127
ls /opt/mapr/sqoop/sqoop-1.4.7.202009100127/conf/
oraoop-site-template.xml  sqoop-env-template.cmd  sqoop-env-template.sh
sqoop-site-template.xml  sqoop-site.xml
```

Pre-Upgrade Steps for Sqoop2

Complete the following steps before you upgrade Sqoop2 with or without the MapR Installer.

1. Back up configuration files. Copy configuration files from `/opt/mapr/sqoop/sqoop-<version>/conf/sqoop.properties` to a location outside the MapR installation.
   For example, you can copy the file to a `/tmp` directory:

```
cp /opt/mapr/sqoop/sqoop-<version>/conf/sqoop.properties /tmp/
```

2. Delete the Sqoop2 repository (`/opt/mapr/sqoop/repository`).

   Note: Because of SQOOP-1593, you also should not create a backup of the Sqoop2 repository before you upgrade.

3. **For upgrades to be performed without the MapR Installer:** Stop each Sqoop2 server node:

   ```
   maprcli node services -name sqoop2 -action stop -nodes <space delimited
   list of nodes>
   ```

Pre-Upgrade Steps for Tez

Complete the following steps before you upgrade Tez with or without the MapR Installer.

If you made any configuration changes that you want to carry over to the next version, you need to back up the configuration properties files:

1. Locate the files in the `/opt/mapr/tez/tez-<version>/conf/` and `/opt/mapr/tez/tez-<version>/tomcat/apache-tomcat-<version>/webapps/tez-ui/config/` directories.
2. Copy the files to a location outside the MapR installation directory.
3. Delete the old `/apps/tez` directory on the MapR File System layer.

   ```
   hadoop fs -rm -r /apps/tez
   ```

   After upgrading, you can reapply changes to the updated Tez installation using the backup.

Preserving the Tez Configuration

Starting from EEP 6.0.0, Tez assists you in preserving the user configuration.

- For a minor version update (for example, `Tez-0.8-1803` to `Tez-0.8-1808`), the user configuration from a previous version is copied to a folder with an old version timestamp and also copied to a new version `conf` folder.
- For a major version update (for example, `Tez-0.8-1803` to `Tez-0.9-1808`), the user configuration from a previous version is **only** copied to a folder with an old version timestamp.

CentOS and SLES OS	• For upgrades from EEP 3.x to EEP 6.0.0, no additional steps are needed to preserve the user configuration. • For upgrades from EEP 4.0.0/4.1.0 to EEP 6.0.0, preserving the user configuration works only for configuration files (such as `tez-site.xml` and `configs.js`), but the Tomcat service is still present from the previous Tez version. As a precondition for upgrade from EEP 4.0.0/4.1.0, manually stop the Tomcat service and remove the `tez-0.8/` directory. • For upgrades from EEP 4.1.1/5.0.0 to EEP 6.0.0, no additional steps are needed to preserve the user configuration.
Ubuntu	• For upgrades from EEP 3.x to EEP 6.0.0, no additional steps are needed to preserve the user configuration.

last-updated: Apr 11, 2022

- For upgrades from EEP 4.0.0/4.1.0 to EEP 6.0.0, preserving the user configuration works only for configuration files (such as `tez-site.xml` and `configs.js`), but the Tomcat service is still present from the previous Tez version. As a precondition for upgrade from EEP 4.0.0/4.1.0, manually stop the Tomcat service and remove the `tez-0.8/` directory.
- For upgrades from EEP 4.1.1/5.0.0 to EEP 6.0.0, you need to preserve the user configuration manually.

Upgrading the MapR Ecosystem Pack With the MapR Installer
If the cluster that you want to upgrade was installed using the MapR Installer, you can use the MapR Installer to upgrade the MapR Ecosystem Pack (EEP).

1. Verify that all ecosystem components are prepared for an upgrade.

   Warning: Service failures, job failures, or the loss of customized configuration files can occur if you do not perform the steps to prepare ecosystem components for an upgrade.

2. Update the MapR Installer. For more information, see Updating the MapR Installer on page 5409.
   This step ensures that the MapR Installer has access to the latest packages.

3. Halt jobs and applications. Stop accepting new jobs and applications, and stop YARN applications.

   ```
   # yarn application -list
   # yarn application -kill <ApplicationId>
   ```

   You might also need specific commands to terminate custom applications.

4. Launch the MapR Installer URL (`https://<hostname/IPaddress>:9443`).

5. Select the **Incremental Install** option.

6. Select the **EEP Version** to which you want to upgrade, and complete the upgrade through the MapR Installer.

7. Once the upgrade through the MapR Installer is complete, perform the post-upgrade steps. See Finishing the MapR Ecosystem Pack Upgrade.

Upgrading the MapR Ecosystem Pack Without the MapR Installer
After you upgrade MapR core without using the MapR Installer, you need to upgrade ecosystem components with manual steps. First, verify that your repository is configured to use an MapR Ecosystem Pack (EEP) that is supported by your cluster version. Then, upgrade each component manually.

Note: If you installed the cluster with the MapR Installer, do not use the following steps to upgrade your ecosystem components. Instead, see Upgrading MapR Core With the MapR Installer on page 308.

Prerequisite: Set up the EEP Repository

Complete the following steps on each node in the cluster when you upgrade without the MapR Installer:

1. Verify that each node can access the ecosystem packages associated with the EEP version that you want to use. For information on how to setup the ecosystem repositories or to manually download each package, see Setting Up Repositories on page 309.

2. Update the repository cache to get the latest list of available packages:
   - On RedHat/CentOS:
     ```
     # yum clean all
     ```
   - On SLES:
     ```
      # zypper refresh
     ```
   - On Ubuntu:
     ```
     # apt-get update
     ```

Manually Upgrade Ecosystem Components

Review the EEP Release Notes on page 5658 to determine the list of ecosystem components available in the EEP that you have selected. Then, complete the manual upgrade steps for each component that you want to upgrade.

Upgrading AsyncHBase Libraries

This section describes how to upgrade the AsyncHBase Libraries without the MapR Installer.

To upgrade to a more recent version of the AsyncHBase library, install the new version. See Installing AsyncHBase Libraries.

Note: AsyncHBase 1.7 is binary compatible with AsyncHBase 1.6.

Upgrading Drill

This section describes how to upgrade Drill without the MapR Installer.

Before you upgrade Drill, complete the pre-upgrade steps.

Complete the following steps on the Drill server and client nodes as root or using sudo to upgrade Drill without the MapR Installer:

1. To eliminate cached packages and files, issue the following command:
   ```
   yum clean all
   ```
2. To upgrade Drill, issue the command appropriate for your system on each Drill node:
   - RedHat/CentOS
     ```
     yum update mapr-drill
     ```
   - Ubuntu
     ```
     sudo apt-get install mapr-drill
     ```
   - SLES
     ```
     zypper update mapr-drill
     ```

     Note: SLES is supported as of Drill 1.9.0-1703 and Drill 1.10.0-1703.

3. Complete the post-upgrade steps for Drill.

Upgrading Flume
This section describes how to upgrade Flume without the MapR Installer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Flume installs into separate directories named after the version, such as `/opt/mapr/flume/flume-<version>/`.

To upgrade with the package manager, run one of the following commands:

On RedHat and CentOS:	`yum update mapr-flume`
On Ubuntu:	`apt-get install mapr-flume`
On SLES:	`zypper update mapr-flume`

To apply custom configurations to the new version, migrate any custom configuration settings into the new default files in the `conf` directory (`/opt/mapr/flume/flume-<version>/conf/`). See Post-Upgrade Steps for Flume on page 372.

Upgrading HBase
This section describes how to upgrade HBase without the MapR Installer.

Upgrading an established deployment of HBase requires planning and consideration before beginning the upgrade process. Below are items to consider as you plan to upgrade:

- **Check for version interoperability and perform any required MapR cluster upgrade first.** To see which versions of HBase are supported in each MapR release, see the HBase Release Notes on page 5855 and Interoperability Matrices on page 5519. If you also plan to upgrade the core release as part of upgrading your HBase cluster, upgrade core first (see Upgrading MapR Core on page 294). After successfully upgrading core, upgrade the HBase component.
- **Perform health checks.** Perform health checks and address any concerns before upgrading HBase. As a start, run `hbck` to check for any inconsistencies in HBase data. For usage details, refer to the Apache HBase Reference Guide.

  ```
  hbase hbck
  ```

- **Review the cluster service layout.** While planning to upgrade, it is a good time to review your cluster service layout and determine if the right services are running on the right set of nodes. For example, as your cluster grows, you will tend to isolate cluster-management services from compute services on separate nodes. Review Planning the Cluster on page 107 and Installing HBase on page 183 for details on planning the service layout.
- **Consider migration of data, maintenance of HBase services, and any version-specific considerations that apply to you.** For details, refer to the Apache HBase Reference Guide.
- **Perform a test upgrade.** Because the upgrade process takes HBase services offline and requires careful planning, perform a test upgrade on a development cluster to make sure you understand the process. After you have experienced success on a dev cluster, proceed with your production cluster.

To upgrade, complete the upgrade steps for the version of HBase to which you want to upgrade.

Upgrade from HBase 1.1.8

Complete the following steps to upgrade HBase 1.1.8 to 1.1.13 or later:

1. Ensure that you have completed the pre-upgrade steps.
2. Use the following commands to upgrade the packages:

   After configuring repositories so that the version you want to install is available, you can use a package manager to install from the repository. The upgrade process removes all but the following directories in the current HBase directory: `conf` and `logs`.

To upgrade with a package manager:

On RedHat and CentOS

To upgrade an HBase region server node:

```
yum update
mapr-hbase
mapr-hbase-regio
nserver
```

To upgrade an HBase master node:

```
yum update
mapr-hbase
mapr-hbase-maste
r
```

To upgrade an HBase client node:

```
yum update
mapr-hbase
```

On Ubuntu

To upgrade an HBase region server node:

```
apt-get install
mapr-hbase
mapr-hbase-regio
nserver
```

To upgrade an HBase master node:

```
apt-get install
mapr-hbase
mapr-hbase-maste
r
```

To upgrade an HBase client node:

```
apt-get install
mapr-hbase
```

On SLES

To upgrade an HBase region server node:

```
zypper update
mapr-hbase
```

```
mapr-hbase-regio
nserver
```

To upgrade an HBase master node:

```
zypper update
mapr-hbase
mapr-hbase-maste
r
```

To upgrade an HBase client node:

```
zypper update
mapr-hbase
```

If you have additional HBase services or libraries installed, you should also upgrade those packages to match the HBase version you are upgrading to.

- Upgrade HBase Thrift Gateway.
- Upgrade the AsynchHbase Libraries.
- To upgrade the libhbase libraries, see Using the libhbase Library on page 3398.

3. Migrate any custom configuration settings to the configuration files within the `conf` directory:

```
/opt/mapr/hbase/hbase-<version>/conf/
```

4. Run `configure.sh -R` on all of the upgraded HBase nodes:

```
$ /opt/mapr/server/configure.sh -R
```

5. Complete the post-upgrade steps.

Related concepts

Considerations for Upgrading to HBase 1.1.13 on page 337
Before upgrading to HBase 1.1.13, familiarize yourself with aspects of the EEP 6.3.0 HBase implementation that can affect an upgrade.

Upgrading HBase Client and Tools

This section describes how to upgrade HBase Client and other tools without the MapR Installer.

Note that release 6.0.x provides Apache HBase-compatible APIs and client interfaces but does not support HBase as an ecosystem component. MapR 6.1.0 supports HBase with EEP 6.3.0 and later.

Service	Description
HBase Client	Upgrading the HBase Client upgrades both the HBase Shell and the HBase APIs supported for use with MapR Database binary tables and MapR Database JSON tables.
HBase Thrift Gateway	HBase Thrift Gateway includes an API and a service that accepts Thrift requests to connect to MapR Database tables. See Upgrading HBase Thrift Gateway on page 355 for upgrade information.

Service	Description
HBase REST Gateway	HBase REST Gateway includes an API and a service that accepts REST requests to connect to MapR Database tables. See Upgrading HBase REST Gateway on page 355 for upgrade information.
AsyncHBase Libraries	AsyncHBase library provides asynchronous Java APIs to access MapR Database tables. See Upgrading AsyncHBase Libraries on page 351 for upgrade information.

Upgrading HBase Thrift Gateway

Complete the following steps to upgrade the HBase Thrift Gateway:

1. Run the following command to upgrade the HBase Thrift package:

On CentOS / Red Hat	`yum update mapr-hbasethrift mapr-hbase`
Ubuntu	`apt-get update mapr-hbasethrift mapr-hbase`
SLES	`zypper update mapr-hbasethrift mapr-hbase`

2. Run the `configure.sh` script with the `-R` option on the node where you upgraded the HBase Thrift package:

```
/opt/mapr/server/configure.sh -R
```

Upgrading HBase REST Gateway

Complete the following steps to upgrade the HBase REST Gateway:

1. Run the following command to upgrade the HBase REST package:

On CentOS / Red Hat	`yum update mapr-hbase-rest mapr-hbase`
Ubuntu	`apt-get update mapr-hbase-rest mapr-hbase`
SLES	`zypper update mapr-hbase-rest mapr-hbase`

2. Run configure.sh on the node where you upgraded the HBase Thrift package:

```
/opt/mapr/server/configure.sh -R
```

Upgrading AsyncHBase Libraries

This section describes how to upgrade the AsyncHBase Libraries without the MapR Installer.

last-updated: Apr 11, 2022

To upgrade to a more recent version of the AsyncHBase library, install the new version. See Installing AsyncHBase Libraries.

Note: AsyncHBase 1.7 is binary compatible with AsyncHBase 1.6.

Upgrading Hive
This section describes how to upgrade Hive without the MapR Installer.

Use one of the following methods to upgrade the Hive components on all nodes where Hive is installed.

To:

- Upgrade with a package manager, install new packages from the repository:
  - On RedHat and CentOS:

```
yum update mapr-hive mapr-hiveserver2 mapr-hivemetastore
mapr-hivewebhcat
```

  - On Ubuntu:

```
apt-get install mapr-hive mapr-hiveserver2 mapr-hivemetastore
mapr-hivewebhcat
```

  - On SLES:

```
zypper update mapr-hive mapr-hiveserver2 mapr-hivemetastore
mapr-hivewebhcat
```

- Manually remove a prior version and manually install the latest version in the repository, run the package manager twice, first to remove the old version, and again to install the new version.

  Note: In this case, configurations are not preserved automatically.

  - On RedHat and CentOS:

```
yum remove mapr-hive mapr-hiveserver2 mapr-hivemetastore
mapr-hivewebhcat
yum install mapr-hive mapr-hiveserver2 mapr-hivemetastore
mapr-hivewebhcat
```

  - On Ubuntu:

```
apt-get remove mapr-hive mapr-hiveserver2 mapr-hivemetastore
mapr-hivewebhcat
apt-get install mapr-hive mapr-hiveserver2 mapr-hivemetastore
mapr-hivewebhcat
```

  - On SLES:

```
zypper remove mapr-hive mapr-hiveserver2 mapr-hivemetastore
mapr-hivewebhcat
zypper install mapr-hive mapr-hiveserver2 mapr-hivemetastore
mapr-hivewebhcat
```

To apply custom configurations to the new version, migrate any custom configuration settings into the new default files in the `conf` directory. See Post-Upgrade Steps for Hive on page 373.

last-updated: Apr 11, 2022

Upgrading from Hive 2.1 to Hive 2.3 with Oracle DB used in Metastore
This section describes how the different upgrade scenarios from Hive 2.1 to Hive 2.3.

Column type verification

You need to first check your current Oracle DB schema and understand your upgrade scenario.

All the examples below use the Oracle SQL*Plus tool to execute SQL statements. Use the `DESCRIBE <Table name>;` to command to check the Oracle table information for following Hive metastore tables:

- `COLUMNS_V2`
- `SD_PARAMS`
- `TABLE_PARAMS`
- `SERDE_PARAMS`

Table

Table	Column	Possible value of column type	
		Scenario I	Scenario II
(1)	(2)	(3)	(4)
COLUMNS_V2	TYPE_NAME	CLOB	VARCHAR2(4000)
SD_PARAMS	PARAM_VALUE	CLOB	VARCHAR2(4000)
TABLE_PARAMS	PARAM_VALUE	CLOB	VARCHAR2(4000)
SERDE_PARAMS	PARAM_VALUE	CLOB	VARCHAR2(4000)

If column `TYPE_NAME` in the `COLUMNS_V2` table has `VARCHAR2(4000)` as the data type, then you have to perform upgrade scenario I. If column `TYPE_NAME` in the `COLUMNS_V2` table has a data type `CLOB`, then you have to perform upgrade scenario II.

All columns types must belong to the same upgrade scenarios, in other words all your columns types must be `VARCHAR2` or `CLOB`.

Use upgrade scenario I

Upgrading to Hive 2.3 (EEP 6.1.0 and above)

To upgrade from Hive-2.1 to Hive 2.3, first download Hive 2.3 from the EEP 6.1.0 package repository and perform the upgrade according to the common upgrade instructions.

Upgrading to Hive 2.3 (before EEP 6.1.0)

To upgrade Hive 2.1 to Hive 2.3 (before EEP 6.1.0), edit the `upgrade-2.1.0-to-2.2.0.oracle.sql` file:

```
nano $HIVE_HOME/scripts/metastore/upgrade/oracle/
upgrade-2.1.0-to-2.2.0.oracle.sql
```

Remove the `@039-HIVE-12274.oracle.sql;` line from the upgrade script and then perform the upgrade according to the common upgrade instructions.

Use upgrade scenario II

Upgrade to Hive 2.3 (EEP 6.1.0 and above)

last-updated: Apr 11, 2022

1. Replace the content of `@039-HIVE-12274.oracle.sql;` file to:

```
-- change PARAM_VALUE to CLOBs
ALTER TABLE COLUMNS_V2 ADD (TEMP CLOB);
UPDATE COLUMNS_V2 SET TEMP=TYPE_NAME;
ALTER TABLE COLUMNS_V2 DROP COLUMN TYPE_NAME;
ALTER TABLE COLUMNS_V2 RENAME COLUMN TEMP TO TYPE_NAME;

ALTER TABLE TABLE_PARAMS ADD (TEMP CLOB);
UPDATE TABLE_PARAMS SET TEMP=PARAM_VALUE, PARAM_VALUE=NULL;
ALTER TABLE TABLE_PARAMS DROP COLUMN PARAM_VALUE;
ALTER TABLE TABLE_PARAMS RENAME COLUMN TEMP TO PARAM_VALUE;

ALTER TABLE SERDE_PARAMS ADD (TEMP CLOB);
UPDATE SERDE_PARAMS SET TEMP=PARAM_VALUE, PARAM_VALUE=NULL;
ALTER TABLE SERDE_PARAMS DROP COLUMN PARAM_VALUE;
ALTER TABLE SERDE_PARAMS RENAME COLUMN TEMP TO PARAM_VALUE;

ALTER TABLE SD_PARAMS ADD (TEMP CLOB);
UPDATE SD_PARAMS SET TEMP=PARAM_VALUE, PARAM_VALUE=NULL;
ALTER TABLE SD_PARAMS DROP COLUMN PARAM_VALUE;
ALTER TABLE SD_PARAMS RENAME COLUMN TEMP TO PARAM_VALUE;

-- Expand the hive table name length to 256
ALTER TABLE TBLS MODIFY (TBL_NAME VARCHAR2(256));
ALTER TABLE NOTIFICATION_LOG MODIFY (TBL_NAME VARCHAR2(256));
ALTER TABLE PARTITION_EVENTS MODIFY (TBL_NAME VARCHAR2(256));
ALTER TABLE TAB_COL_STATS MODIFY (TABLE_NAME VARCHAR2(256));
ALTER TABLE PART_COL_STATS MODIFY (TABLE_NAME VARCHAR2(256));
ALTER TABLE COMPLETED_TXN_COMPONENTS MODIFY (CTC_TABLE VARCHAR2(256));

-- Expand the hive column name length to 767
ALTER TABLE COLUMNS_V2 MODIFY (COLUMN_NAME VARCHAR(767));
ALTER TABLE PART_COL_PRIVS MODIFY (COLUMN_NAME VARCHAR2(767));
ALTER TABLE TBL_COL_PRIVS MODIFY (COLUMN_NAME VARCHAR2(767));
ALTER TABLE SORT_COLS MODIFY (COLUMN_NAME VARCHAR2(767));
ALTER TABLE TAB_COL_STATS MODIFY (COLUMN_NAME VARCHAR2(767));
ALTER TABLE PART_COL_STATS MODIFY (COLUMN_NAME VARCHAR2(767));
```

2. Add the following line to the `$HIVE_HOME/scripts/metastore/upgrade/oracle/upgrade-2.1.0-to-2.2.0.oracle.sql` file after the `@038-HIVE-10562.oracle.sql;` line:

```
@039-HIVE-12274.oracle.sql;
```

3. Perform upgrade according to the common upgrade instructions.

Upgrade to Hive 2.3 (before EEP 6.1.0)

1. Replace the content of `@039-HIVE-12274.oracle.sql;` file to the same as in the previous scenario.

2. Make sure that the following line is present in the `$HIVE_HOME/scripts/metastore/upgrade/oracle/upgrade-2.1.0-to-2.2.0.oracle.sql` file:

```
@039-HIVE-12274.oracle.sql;
```

3. Perform upgrade according to the common upgrade instructions.

last-updated: Apr 11, 2022

Upgrading HttpFS
This section describes how to upgrade HttpFS without the MapR Installer.

For EEP 6.3.4 and later, run one of the following commands to upgrade the HttpFS using package manager and install the new packages from the repository:

- On RedHat/CentOS:

```
yum update mapr-httpfs
```

- On Ubuntu:

```
apt-get install mapr-httpfs
```

- On SLES:

```
zypper update mapr-httpfs
```

For EEP 6.3.3 and earlier, run the following set of commands as `root` or with `sudo` to upgrade the HttpFS packages:

- On RedHat/CentOS

```
yum remove mapr-httpfs
rm -rf /opt/mapr/httpfs/
yum install mapr-httpfs
```

- On Ubuntu

```
apt-get autoremove mapr-httpfs
rm -rf /opt/mapr/httpfs/
apt-get install mapr-httpfs
```

- On SLES

```
zypper remove mapr-httpfs
rm -rf /opt/mapr/httpfs/
zypper install mapr-httpfs
```

 **Note:** To verify EEP and core version support, see EEP Support and Lifecycle Status on page 5531.

Upgrading Hue
This section describes how to upgrade Hue without the MapR Installer.

Execute the following commands as a user with admin permissions:

1. Run one of the following commands to upgrade Hue using a package manager:
   - On Ubuntu:

```
apt-get install mapr-hue
```

   - On RedHat/CentOS:

```
yum update mapr-hue
```

- On SLES:

```
zypper update mapr-hue
```

2. For EEP 4.0.0 and later, update Livy using the steps in Upgrading Livy on page 361. For EEP releases earlier than EEP 4.0.0, run one of the following commands to upgrade Hue-livy using a package manager:
   - On Ubuntu:

```
apt-get install mapr-hue-livy
```

   - On RedHat/CentOS:

```
yum update mapr-hue-livy
```

   - On SLES:

```
zypper update mapr-hue-livy
```

Upgrading Impala

This section describes how to upgrade Impala without the MapR Installer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

1. Upgrade the `mapr-impala` package on all Impala nodes in the cluster:

```
$ sudo yum update mapr-impala
```

2. In `/opt/mapr/impala/impala-<version>/conf/env.sh`, complete the following steps:
   a. Verify that the Statestore address is set to the address where you plan to run the statestore service.

```
IMPALA_STATE_STORE_HOST=<IP address hosting statestore>
```

   b. Change the Catalog service address to the address where you plan to run the catalog service.

```
CATALOG_SERVICE_HOST=<IP address hosting catalog service>
```

   Refer to Additional Impala Configuration Options for a list of other options that you can modify in `env.sh`.

3. Verify that `hive-site.xml` has the following property configured on all nodes:

```
<property>
    <name>hive.metastore.uris</name>
    <value>thrift://<metastore_server_host>:9083</value>
</property>
```

4. To upgrade the statestore service, issue the following command:

```
$ sudo yum update mapr-impala-statestore
```

5. To upgrade the catalog service, issue the following command:

```
$ sudo yum update mapr-impala-catalog
```

Note: It is recommended (not required) that you install the catalog service on the same node as the statestore service.

6. To upgrade the impala server, issue the following command:

```
$ sudo yum update mapr-impala-server
```

7. Run `configure.sh` to refresh the node configuration.

```
/opt/mapr/server/configure.sh -R
```

Upgrading the MapR Data Access Gateway

This section describes how to upgrade the MapR Data Access Gateway without the MapR Installer.

Complete the following steps to upgrade the MapR Data Access Gateway without the MapR Installer.

1. Install the new package using the command that is appropriate for your distribution:

RedHat/CentOS	`yum update mapr-data-access-gateway`
Ubuntu	`apt-get install mapr-data-access-gateway`
SLES	`zypper update mapr-data-access-gateway`

2. On the CLDB master node, use the `manageSSLKeys.sh` script to generate the p12 keystore file, which enables OpenSSL communication channel for the MapR gRPC service. For example:

```
/opt/mapr/server/manageSSLKeys.sh convert -k -N <cluster_name> /opt/mapr/
conf/ssl_keystore /opt/mapr/conf/ssl_keystore.pem
```

3. Copy the `/opt/mapr/conf/ssl_keystore.p12` and `/opt/mapr/conf/ssl_keystore.pem` files to all other nodes that contain the MapR Data Access Gateway.

Upgrading Livy

This section describes how to upgrade Livy without the MapR Installer.

Run the following command to upgrade Livy using a package manager:

- On RHEL/CentOS:

```
yum update mapr-livy
```


last-updated: Apr 11, 2022

- On Ubuntu:

```
apt-get install mapr-livy
```

- On SLES:

```
zypper update mapr-livy
```

Note: Livy is included in EEP repositories beginning with EEP 4.0.0. Before EEP 4.0.0, it was included as a package called mapr-hue-livy and released only as a part of Hue.

Upgrading MapR Monitoring
Complete the following steps to upgrade MapR Monitoring without the MapR Installer.

Note: Before performing the following steps, make sure that you have completed the Pre-Upgrade Steps for MapR Monitoring on page 342.

Execute the following commands as `root` or using `sudo`.

1. Upgrade the following metric monitoring packages wherever they are installed on the cluster: `mapr-collectd`, `mapr-grafana`, and `mapr-opentsdb`.

   For example, on a three node cluster, you could run the following commands to upgrade metrics monitoring packages:

   - For CentOS/RedHat:
     - Node A:

```
yum upgrade mapr-collectd mapr-grafana
```

     - Node B:

```
yum upgrade mapr-collectd mapr-opentsdb
```

     - Node C:

```
yum upgrade mapr-collectd
```

   - For Ubuntu:
     - Node A:

```
apt-get install mapr-collectd mapr-grafana
```

     - Node B:

```
apt-get install mapr-collectd mapr-opentsdb
```

     - Node C:

```
apt-get install mapr-collectd
```

   - For SLES:

- Node A:

```
zypper update mapr-collectd mapr-grafana
```

- Node B:

```
zypper update mapr-collectd mapr-opentsdb
```

- Node C:

```
zypper update mapr-collectd
```

2. Upgrade the following log monitoring packages wherever they are installed on the cluster: `mapr-fluentd`, `mapr-elasticsearch`, and `mapr-kibana`.

   For example, on a three node MapR cluster, you can run the following commands to upgrade log monitoring packages:

   - For CentOS/RedHat:
     - Node A:

```
yum upgrade mapr-fluentd mapr-elasticsearch
```

     - Node B:

```
yum upgrade mapr-fluentd mapr-elasticsearch
```

     - Node C:

```
yum upgrade mapr-fluentd mapr-elasticsearch mapr-kibana
```

   - For Ubuntu:
     - Node A:

```
apt-get install mapr-fluentd mapr-elasticsearch
```

     - Node B:

```
apt-get install mapr-fluentd mapr-elasticsearch
```

     - Node C:

```
apt-get install mapr-fluentd mapr-elasticsearch mapr-kibana
```

   - For SLES:
     - Node A:

```
zypper update mapr-fluentd mapr-elasticsearch
```

- Node B:

```
zypper update mapr-fluentd mapr-elasticsearch
```

- Node C:

```
zypper update mapr-fluentd mapr-elasticsearch mapr-kibana
```

Reinstalling Monitoring Components After an Upgrade
During an upgrade from EEP 6.x to EEP 7.0.0 or EEP 7.0.1, some monitoring components do not get updated because of an error in the fourth digit of the package version. This page provides a workaround for the issue.

The issue (known issue ES-77 or FLUD-55) is fixed in EEP 7.1.0 and later.

This issue can occur during manual upgrades or upgrades performed using the Installer. The affected components can include any or all of the following:

- Elasticsearch
- Fluentd
- Grafana
- Kibana

Follow these steps to identify, remove, and reinstall the packages that were not upgraded:

1. After upgrading, use one of the following commands to check the package versions of the installed monitoring components:

OS	Command
RHEL/CentOS	`yum list installed`
SLES	`zypper packages --installed-only`
Ubuntu	`apt list --installed`

2. Identify the packages that were not upgraded. The following table shows the desired versions for each package for EEPs 7.0.0 and 7.0.1. For more version information, see Component Versions for Released EEPs on page 5586.

Package	Desired Version	
	EEP 7.0.0	EEP 7.0.1
`mapr-elasticsearch`	6.8.8.0	6.8.8.0
`mapr-fluentd`	1.10.3.0	1.10.3.0
`mapr-grafana`	6.7.4.0	6.7.4.0
`mapr-kibana`	6.8.8.0	6.8.8.0

3. Before removing the packages that were not upgraded:

last-updated: Apr 11, 2022

a) Ensure that you have backed up any configuration files and indexes as described in Pre-Upgrade Steps for MapR Monitoring on page 342. For Elasticsearch in particular, you must back up the default location for Elasticsearch index data unless you specified a non-default location using the `-ESDB` parameter with `configure.sh` during installation.

b) Export or make backup copies of any custom dashboards you configured for Grafana or Kibana.

4. Manually uninstall the packages that were not upgraded by using one of the following commands. For example, to uninstall the Elasticsearch package:
   - RHEL/CentOS:
     - `yum remove mapr-elasticsearch`
   - SLES:
     - `zypper remove mapr-elasticsearch`
   - Ubuntu:
     - `apt remove mapr-elasticsearch --purge`
5. Install the desired packages for EEP 7.0.0 or EEP 7.0.1 using *one* of the following methods:
   - **Method 1 – Using the Installer**

     Using the Installer, perform an incremental installation. See Using the Incremental Install Function on page 5443.
   - **Method 2 – Manual Reinstall**

     For Grafana, reinstall the package and run `configure.sh` using the commands shown in Step 8: Install Metrics Monitoring on page 162.

     For Elasticsearch, Fluentd, and Kibana, reinstall the packages and run `configure.sh` using the commands shown in Step 9: Install Log Monitoring on page 165.

Upgrading the S3 Gateway
Upgrading from S3 gateway version 1.0.x to 2.0.0 is not supported.
Edge Node Upgrade

Complete the following steps to upgrade the S3 gateway on an Edge node:

1. Stop the MapR Object Store service:

```
sudo /opt/mapr/objectstore-client/objectstore-client-<version>/bin/
objectstore.sh stop
```

2. Upgrade the S3 gateway:

```
# Ubuntu
sudo apt-get install mapr-objectstore-client

# RedHat/CentOS
sudo yum update mapr-objectstore-client

# SLES
sudo zypper update mapr-objectstore-client
```

3. Run `configure.sh`:

```
sudo /opt/mapr/objectstore-client/objectstore-client-<version>/bin/
configure.sh
-c -u <userName> -g <groupName> --path /path/to/filesystem/
```

4. Restart the service:

```
sudo /opt/mapr/objectstore-client/objectstore-client-<version>/bin/
objectstore.sh start
```

Server Node Upgrade

Complete the following steps to upgrade the S3 gateway on a server node:

1. Stop the MapR Object Store service:

```
maprcli node services -nodes <nodename> -name objectstore -action stop
```

2. Upgrade the S3 gateway:

```
# Ubuntu
sudo apt install mapr-objectstore-client mapr-objectstore-gateway

# RedHat/CentOS
sudo yum install -f  mapr-objectstore-client mapr-objectstore-gateway

# SLES
sudo zypper update mapr-objectstore-client mapr-objectstore-gateway
```

3. Run `configure.sh -R`:

```
/opt/mapr/server/configure.sh -R
```

4. Restart the service:

```
maprcli node services -nodes <node name> -name objectstore -action
          restart
```

Upgrading the MapR Event Store For Apache Kafka Python Client

This section describes how to upgrade the MapR Event Store For Apache Kafka Python Client without the MapR Installer.

To install the MapR Event Store For Apache Kafka Python Client using the Python Software Foundation, run the following command as `root` or using `sudo`:

```
pip
install -upgrade --global-option=build_ext --global-option="--library-dirs=/
opt/mapr/lib" --global-option="--include-dirs=/opt/mapr/include/"
mapr-streams-python
```

OR:

```
pip
install -U --global-option=build_ext --global-option="--library-dirs=/opt/
```

```
mapr/lib" --global-option="--include-dirs=/opt/mapr/include/"
mapr-streams-python
```

Alternatively, you can install the MapR Event Store For Apache Kafka Python Client via the MapR package repository:

```
https://package.mapr.hpe.com/releases/MEP/<MEP version>/mac/
mapr-streams-python-<version>.tar.gz
```

Upgrading MapR Event Store For Apache Kafka Tools
Complete the following steps to upgrade MapR Event Store For Apache Kafka Tools without the Installer.

If you are upgrading from Kafka 2.1.1 to 2.6.1, you may first want to review Changes in Kafka 2.6.1 on page 3863. To upgrade, run the following commands as `root` or using `sudo`:

1. Stop the service using `maprcli node services -name <name_service> - action stop - nodes <space delimited list of Kafka tools server nodes>`:
2. Run one of the following commands to upgrade the Kafka REST Proxy for MapR Streams:
   - On Ubuntu:

     ```
     apt-get install mapr-kafka-rest
     ```

   - On RedHat/CentOS:

     ```
     yum update mapr-kafka-rest
     ```

   - On SLES:

     ```
     zypper update mapr-kafka-rest
     ```

3. Run one of the following commands to upgrade the Kafka Connect for MapR Event Store For Apache Kafka - HDFS Connector:
   - On Ubuntu:

     ```
     apt-get install mapr-kafka-connect-hdfs
     ```

   - On RedHat/CentOS:

     ```
     yum update mapr-kafka-connect-hdfs
     ```

   - On SLES:

     ```
     zypper update mapr-kafka-connect-hdfs
     ```

4. Run one of the following commands to upgrade the Kafka Connect for MapR Event Store For Apache Kafka - JDBC Connector:
   - On Ubuntu:

     ```
     apt-get install mapr-kafka-connect-jdbc
     ```

- On RedHat/CentOS:

```
yum update mapr-kafka-connect-jdbc
```

- On SLES:

```
zypper update mapr-kafka-connect-jdbc
```

5. Run `configure.sh -R` on each node where you installed Kafka components to complete the configuration: `/opt/mapr/server/configure.sh -R`

Apply custom configurations to the new version by migrating any custom configuration settings into the new default files in the **conf** directory. See Post-Upgrade Steps for MapR Event Store For Apache Kafka Tools on page 379 for more information.

Upgrading Oozie

This section describes how to upgrade Oozie without the MapR Installer.

Run the following command to upgrade Oozie using a package manager:

- On RedHat/CentOS:

```
yum update mapr-oozie
```

- On Ubuntu:

```
apt-get install mapr-oozie
```

- On SLES:

```
zypper update mapr-oozie
```

Upgrading Pig

This section describes how to upgrade Pig without the MapR Installer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Complete the following steps to upgrade Pig without the MapR Installer.

Run the following command to upgrade Pig using a package manager:

On RedHat and CentOS	`yum update mapr-pig`
On Ubuntu	`apt-get install mapr-pig`
On SLES	`zypper update mapr-pig`

To apply custom configurations to the new version, migrate any custom configuration settings into the new default files in the `conf` directory. See Post-Upgrade Steps for Pig on page 381.

Upgrading Sentry

This section describes how to upgrade Sentry without the MapR Installer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Before you complete the following step to upgrade Sentry without the MapR Installer, verify that you have completed the pre-upgrade steps for Sentry.

To upgrade Sentry, issue the command appropriate for the system:

RedHat/CentOS	`yum update mapr-sentry`
Ubuntu	`apt-get install mapr-sentry`
SLES	`zypper update mapr-sentry`

Upgrading Spark Standalone

This section describes how to upgrade Spark Standalone without the MapR Installer.

1. Install the Spark packages.
   - On Ubuntu:
     ```
     apt-get install mapr-spark mapr-spark-master mapr-spark-historyserver
     ```
   - On RedHat/CentOS:
     ```
     yum update mapr-spark mapr-spark-master mapr-spark-historyserver
     ```
   - On SLES:
     ```
     zypper upgrade mapr-spark mapr-spark-master mapr-spark-historyserver
     ```
2. (Optional): Starting in the EEP 4.0 release, you can install the MapR Spark Thrift Server package.

On Ubuntu	`apt-get insall mapr-spark-thriftserver`
On RedHat / CentOS	`yum install mapr-spark-thriftserver`
On SLES	`zypper install mapr-spark-thriftserver`

Upgrading Spark on YARN

This section describes how to upgrade Spark on YARN without the MapR Installer.

The following instructions explain how to upgrade an existing installation of Spark. Spark will be installed in a new subdirectory under `/opt/mapr/spark`.

1. Install the Spark packages.

   Note: You only need to upgrade the mapr-spark-historyserver if your previous installation included this package.

last-updated: Apr 11, 2022

On Ubuntu	`apt-get install mapr-spark mapr-spark-historyserver`
On RedHat / CentOS	`yum update mapr-spark mapr-spark-historyserver`
On SLES	`zypper upgrade mapr-spark mapr-spark-historyserver`

2. (Optional): Starting in the EEP 4.0 release, you can install the MapR Spark Thrift Server package.

On Ubuntu	`apt-get insall mapr-spark-thriftserver`
On RedHat / CentOS	`yum install mapr-spark-thriftserver`
On SLES	`zypper install mapr-spark-thriftserver`

Upgrading Sqoop1

This section describes how to upgrade Sqoop1 without the MapR Installer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Use one of the following methods to upgrade Sqoop1 without the MapR Installer.

To:

- Upgrade with a package manager and install new packages from the repository, run the following command:

On RedHat and CentOS	`yum update mapr-sqoop`
On Ubuntu	`apt-get install mapr-sqoop`
On SLES	`zypper update mapr-sqoop`

- Manually remove a prior version and then manually install the latest version in the repository, run the following commands:

On RedHat and CentOS	`yum remove mapr-sqoop` `yum install mapr-sqoop`
On Ubuntu	`apt-get remove mapr-sqoop` `apt-get install mapr-sqoop`

last-updated: Apr 11, 2022

On SLES	`zypper remove mapr-sqoop` `zypper install mapr-sqoop`

To apply custom configurations to the new version, migrate any custom configuration settings into the new default files in the `conf` directory (`/opt/mapr/sqoop/sqoop-<version>/conf/`). See Post-Upgrade Steps for Sqoop1 on page 383.

Upgrading Sqoop2

This section describes how to upgrade Sqoop2 without the MapR Installer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Execute the following commands as `root` or using `sudo`:

1. On each Sqoop2 server node, upgrade the `mapr-sqoop2-server` packages.

On Ubuntu	`apt-get install mapr-sqoop2-server`
On RedHat and CentOS	`yum update mapr-sqoop2-server`
On SLES	`zypper update mapr-sqoop2-server`

2. On each Sqoop2 client node, install `mapr-sqoop2-client.`

On Ubuntu	`apt-get install mapr-sqoop2-client`
On RedHat / CentOS	`yum update mapr-sqoop2-client`
On SLES	`zypper update mapr-sqoop2-client`

Upgrading Tez

This section describes how to upgrade Tez without the MapR Installer.

Complete the following steps to upgrade Tez without the MapR Installer.

Use the following method to upgrade Tez on all the nodes where Tez is installed.

- Upgrade with a package manager, install new packages from the repository.

On RedHat and CentOS	`yum update mapr-tez`
On Ubuntu	`apt-get install mapr-tez`
On SLES	`zypper update mapr-tez`

To apply custom configurations to the new version, migrate any custom configuration settings into the new default files in the `conf` directory. See Post-Upgrade Steps for Tez on page 384.

last-updated: Apr 11, 2022

Finishing the MapR Ecosystem Pack Upgrade

Complete the post-upgrade steps for each ecosystem component that was upgraded.

Post-Upgrade Steps for Drill

Complete the following steps after you upgrade Drill with or without the MapR Installer.

1. Configuration files from the previous installation now reside in `/opt/mapr/drill/OLD_DRILL_VERSIONS`. If you have made any changes to configuration files in the previous version, compare and restore your previous configurations in the `/opt/mapr/drill/drill-<version>/conf` directory. Also, copy over any UDF or custom storage or format plugin JAR files that you added to the previous Drill directory.

   Note: The `drill-override.conf` contains your ZooKeeper configuration and any other options specified in the file. The `drill-env.sh` file contains any options that you modified, such as Drill memory allocation. The `logback.xml` file contains changes you may have made to use Lilith.

2. Run `configure.sh` to refresh the node configuration.

   ```
   $ /opt/mapr/server/configure.sh -R
   ```

   Note: Drill should be configured and running on the node. You can use one of the following methods to verify that the Drillbit service is running on the node:

   - Issue the following command to verify the status of the Drillbit service from the command line:

     ```
     jps
     ```

   - Log in to the Control System at `https://<host name>:8443` and click **Services** to verify the status of the Drillbit service.

   You should see the Drillbit listed as a service running on the node.

3. Enter the following URL in a web browser to access the Drill Web Console and verify that your storage plugin configurations were preserved during the upgrade:

   ```
   http://<IP address or host name>:8047/storage
   ```

   If your storage plugins were not preserved, use the back up that you took before the upgrade to restore them.

   Note: You can start/stop/restart the Drillbit service on one or more nodes using the Control System or the following command:

   ```
   $ maprcli node services -name drill-bits -action start|restart|
   stop -nodes <node host names separated by a space>
   ```

   Use the host name if possible. Using host names instead of IP addresses is a best practice.

   You can access the Drill log files in `/opt/mapr/drill/drill-<version>/logs/drillbit.log`.

Post-Upgrade Steps for Flume

Complete the following steps after you upgrade Flume with or without the MapR Installer.

last-updated: Apr 11, 2022

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Migrate Custom Configurations (optional).

Migrate any custom configuration settings into the new default files in the conf directory (`/opt/mapr/flume/flume-<version>/conf/`).

Post-Upgrade Steps for HBase

Complete the following steps after you upgrade HBase without the MapR Installer.

If you upgrade using the MapR Installer, running `configure.sh` is not necessary.

Run `configure.sh -R`.

```
/opt/mapr/server/configure.sh -R
```

After you run the `configure.sh -R` command, the HBase Thrift, HBase REST, HBase Master, and HBase RegionServer services are started automatically.

Related concepts

Considerations for Upgrading to HBase 1.1.13 on page 337
Before upgrading to HBase 1.1.13, familiarize yourself with aspects of the EEP 6.3.0 HBase implementation that can affect an upgrade.

Post-Upgrade Steps for HBase Client

Complete the following steps after you upgrade HBase Client with or without the MapR Installer.

Merge custom configuration files with the new default files (optional).

1. If you backed up HBase Client configuration files into a location outside the installation directory before upgrading HBase Client, you must retrieve them if you want to save your custom configuration.
2. Merge HBase Client configuration files from with the new default files in `/opt/mapr/hbase/hbase-<version>/conf/`. Be sure not to simply copy over the configuration files: to avoid overwriting the default files, conduct a merge.

Related concepts

Considerations for Upgrading to HBase 1.1.13 on page 337
Before upgrading to HBase 1.1.13, familiarize yourself with aspects of the EEP 6.3.0 HBase implementation that can affect an upgrade.

Post-Upgrade Steps for Hive

Complete the following steps after you upgrade Hive with or without the MapR Installer.

1. Migrate Hive Configuration.

   Migrate any custom configuration settings into the Hive 2.3 version in the `/opt/mapr/hive/hive-2.3/conf/` directory.

2. Update the Hive Metastore.

   For upgrades from the old version of Hive to the new version, run the `schematool` command with the `-upgradeSchema` option.

   Note: Review and, if necessary, perform the steps described in Troubleshooting Hive Upgrade Issues on page 374 before running this command.

   ```
   /opt/mapr/hive/hive-<version>/bin/schematool -dbType
   <metastore_database> -upgradeSchema
   ```

   For example, for upgrades from Hive 2.1 to 2.3 on MySQL, run the following command:

   ```
   /opt/mapr/hive/hive-2.3/bin/schematool -dbType mysql -upgradeSchema
   ```

3. Run `configure.sh -R`.

   ```
   /opt/mapr/server/configure.sh -R
   ```

   This step enables Warden to recognize the newly installed services.

4. Verify that the metastore database update completed successfully. You can use the following diagnostic tests:
   - Run the `show tables` command in Hive and make sure it returns a complete list of all your Hive tables.
   - Perform simple SELECT operations on Hive tables that existed before the upgrade.
   - Perform filtered SELECT operations on Hive tables that existed before the upgrade.

Troubleshooting Hive Upgrade Issues

This section describes how to troubleshoot inconsistencies in an underlying database after creating tables with the `datanucleus.schema.autoCreateAll` property.

The `datanucleus.schema.autoCreateAll` property creates tables gradually. After creating tables using this property, if you upgrade to Hive 2.3 using `schematool`, the `schematool` command will not be able to verify all the necessary tables because all the necessary tables were not created by the `datanucleus.schema.autoCreateAll` property.

1. Determine the version from which you are upgrading.
2. Start MySQL:

   ```
   mysql -u <user> -p <password>
   ```

3. Run the following commands in your MySQL command line.

Upgrade from Hive 1.2	`USE metastore;`
	`SOURCE /opt/mapr/hive/hive-2.1/scripts/metastore/upgrade/mysql/hive-schema -1.2.0.mysql.sql;`
	`SOURCE /opt/mapr/hive/hive-2.1/scripts/metastore/upgrade/mysql/hive-txn-schema`

```
        -0.13.0.mysql.sql;
```

Upgrade from Hive 1.0

```
USE metastore;
SOURCE /opt/mapr/hive/hive-2.1/
scripts/metastore/upgrade/mysql/
hive-schema-0.14.0.
mysql.sql;
```

Upgrade from Hive 0.13

```
USE metastore;
SOURCE /opt/mapr/hive/hive-2.1/
scripts/metastore/upgrade/mysql/
hive-schema-0.13.0.mysql.sql;
```

Running SOURCE for the version before the upgrade makes the underlying database consistent for running `schematool` command with the `-upgradeSchema` option.

Post-Upgrade Steps for HttpFS

Complete the following steps after you upgrade HttpFS with or without the MapR Installer.

1. Migrate any custom HttpFS configuration settings into the new default files.
   - The following examples show configuration file locations:
     - `/opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/webapps/webhdfs/WEB-INF/web.xml`
     - `/opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/conf/server.xml`
     - `/opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/conf/tomcat-users.xml`
     - `/opt/mapr/httpfs/httpfs-1.0/etc/hadoop/httpfs-site.xml`
2. Run the `configure.sh` script after making configuration changes:

```
sudo bash /opt/mapr/server/configure.sh -R
```

Post-Upgrade Steps for Hue

Complete the following steps after you upgrade Hue with or without the MapR Installer:

1. To configure the Hue-livy package after upgrading, see Integrate Hue 3.10+ With Spark 2 on page 3663.
2. After upgrading on Ubuntu, remove the `mapr-hue-base` package:

```
apt-get remove mapr-hue-base
```

3. Copy the changes that you made for required services in your existing `hue.ini` file into the latest version of the file:

```
/opt/mapr/hue/hue-<version>/desktop/conf/hue.ini
```

Note: Hue 3.9 uses the old Query editor to work with Hive and Impala queries, and introduces new Spark Notebooks. Hue 3.10+ uses Notebooks as a replacement for the old Query editor for Hive and Impala. If you are upgrading to Hue 3.10+, and you want to have access to your saved queries in the old Hive or Impala Query editor, you need to configure Hue to use the old Query editor. To do this, set the `use_new_editor` property in the `hue.ini` file to `false`. For example:

```
[desktop]
...
  # Choose whether to show the new SQL editor.
  use_new_editor=false
```

4. If you use SQLite as the Hue database, load its backup:
   a) If the Hue node runs on Ubuntu, install sqlite3:

```
apt-get install sqlite3
```

   b) Run the following commands:

```
cd /opt/mapr/hue/hue-<new_version>/desktop
mv desktop.db desktop.db.old
sqlite3 desktop.db < ~/dump-hue-<old_version>-sqlite.bak
sqlite3 desktop.db
DELETE FROM django_content_type;
```

5. Update the old database schema so that it is compatible with the new upgraded version:
   a) For Hue 4.3+:

```
source /opt/mapr/hue/hue-<new_version>/bin/activate
hue migrate --run-syncdb --fake-initial
deactivate
```

   For example, run the following commands to update the database schema to make it compatible with Hue 4.3+:

```
source /opt/mapr/hue/hue-4.3.0/bin/activate
hue migrate --run-syncdb --fake-initial
deactivate
```

b) For Hue version up to Hue 4.2:

```
source /opt/mapr/hue/hue-<new_version>/bin/activate
hue syncdb --noinput
hue migrate --merge
deactivate
```

For example, run the following commands to update the database schema to make it compatible with Hue 4.2:

```
source /opt/mapr/hue/hue-4.2.0/bin/activate
hue syncdb --noinput
hue migrate --merge
deactivate
```

If you are using MySQL, PostgreSQL, or Oracle, and you have trouble with the database during the Hue upgrade, you can restore your data from the backup that you created during the Pre-Upgrade Steps for Hue:

```
source /opt/mapr/hue/hue-<new_version>/bin/activate
hue loaddata --ignorenonexistent ~/dump-hue-<old_version>.json
deactivate
```

6. **For upgrades performed without the MapR Installer:** If you are using Hadoop MRv1, complete the following steps to establish communication between Hue and the JobTracker processes:
   a) Remove existing Hue plugins from the MapReduce lib directory:

```
rm /opt/mapr/hadoop/hadoop-0.20*/lib/hue-plugins-*.jar
```

   b) Copy new Hue plugins to the MapReduce lib directory:

```
cp /opt/mapr/hue/hue-<version>/desktop/libs/hadoop/java-lib/
hue-plugins-*.jar /opt/mapr/hadoop/hadoop-0.20*/lib/
```

   For example, run the following commands to copy the Hue plugin for Hue 3.10+:

```
cp /opt/mapr/hue/hue-3.10.0/desktop/libs/hadoop/java-lib/
hue-plugins-*.jar /opt/mapr/hadoop/hadoop-0.20*/lib/
```

   c) Restart the JobTracker services:

```
maprcli node services -jobtracker restart -nodes <ip_addresses>
```

7. Run `configure.sh -R`:

```
/opt/mapr/server/configure.sh -R
```

If you do not complete this step, Hue may fail to start and the Control System may still display references to the Hue version that you upgraded from.

8. Restart the Hue service:

```
maprcli node services -name hue -action restart -nodes <ip_address>
```


last-updated: Apr 11, 2022

Post-Upgrade Steps for Impala

Complete the following steps after you upgrade Impala with or without the MapR Installer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Start the Impala, statestore, and catalog services. Always restart the Impala statestore service prior to restarting the Impala server on nodes in the cluster.

1. Issue the following command to start the Impala statestore service on the designated host in the cluster:

```
$ sudo maprcli node services -name impalastore -action start -nodes
<node IP addresses separated by a space>
Example:$ sudo maprcli node services -name impalastore -action
start -nodes 10.10.30.166
```

2. Issue the following command to start the catalog service on the designated host in the cluster:

```
$ sudo maprcli node services -name impalacatalog -action start -nodes
<node IP addresses separated by a space>
Example:$ sudo maprcli node services -name impalacatalog -action
start -nodes 10.10.30.166
```

3. Issue the following command to start the Impala server on each Impala node:

```
$ sudo maprcli node services -name impalaserver -action start -nodes
<node IP addresses separated by a space>
Example:$ sudo maprcli node services -name impalaserver -action
start -nodes 10.10.30.166
```

Note: Check the Impala log files for errors if the services do not start successfully. You can locate log files in the following Impala installation directory on each node:

```
/opt/mapr/impala/impala-<version>/logs
```

Post-Upgrade Steps for MapR Data Access Gateway

Complete the following steps after you upgrade the MapR Data Access Gateway with or without the MapR Installer.

1. Run `configure.sh -R`:

```
/opt/mapr/server/configure.sh -R
```

2. Restart the service:

```
maprcli node services -nodes <node name> -name
data-access-gateway -action restart
```

Post-Upgrade Steps for Livy

Complete the following steps after you upgrade Livy with or without the MapR Installer.

Optional: Migrate Custom Configurations

Transfer any custom configuration settings into the new default files in the `conf` directory (`/opt/mapr/livy/livy-<version>/conf/`).

Post-Upgrade Steps for MapR Monitoring
Complete the following steps after you upgrade MapR Monitoring Components with or without the MapR Installer.

1. After you upgrade monitoring components, add customized properties from the configuration files that you backed up before the upgrade to the files in the new installation directories.

   Backups of many of the MapR Monitoring component configuration files are stored in the `/opt/mapr/<component>/<component>-<new_version>/etc` directory and its subdirectories. During the backup of a configuration file, the upgrade script appends the component version number to the filename. For example, the backup filename for `collectd.conf` is `collectd.conf-5.5.1`. Therefore, if you did not manually back up the configuration files before upgrading MapR Monitoring components, you may be able to retrieve the configuration.

2. On each node in the cluster, run `configure.sh` with the `-R` option.

   ```
   /opt/mapr/server/configure.sh -R
   ```

3. If you created a snapshot of the Kibana index as described in Pre-Upgrade Steps for MapR Monitoring on page 342, restore the snapshot to ensure that you have access to index information that was present before the upgrade. See https://www.elastic.co/guide/en/elasticsearch/reference/5.6/modules-snapshots.html.

4. If you need to configure the MapR Monitoring components for security, follow the steps in the installation procedures to generate the necessary files and distribute them across the cluster:
   - Step 8: Install Metrics Monitoring on page 162
   - Step 9: Install Log Monitoring on page 165

Post-Upgrade Steps for MapR Object Store with S3-Compatible API
Complete the following steps after you upgrade the MapR Object Store with S3-Compatible API with or without the MapR Installer.

1. Run `configure.sh -R`:

   ```
   /opt/mapr/server/configure.sh -R
   ```

2. Restart the service:

   ```
   maprcli node services -nodes <node name> -name objectstore -action
   restart
   ```

Post-Upgrade Steps for MapR Event Store For Apache Kafka Tools
Complete the following steps after manually upgrading MapR Event Store For Apache Kafka Tools.

Kafka REST Proxy

The following post-upgrade steps are applicable when upgrading Kafka REST Proxy.

1. Review your configuration files and modify as needed:

   ```
   /opt/mapr/kafka-rest-<version>/config
   ```


last-updated: Apr 11, 2022

2. Run the `configure.sh` file. For example:

```
/opt/mapr/server/configure.sh -R
```

3. Start the Kafka REST service

```
maprcli node services -name kafka-rest -action start -nodes < list of
Kafka REST service nodes >
```

To see how configuration files are saved during an upgrade, see Saving Kafka REST Configurations on page 3872.

Kafka Connect

The following post-upgrade steps are applicable when upgrading Kafka Connect.

1. Review your configuration files and modify as needed:

```
/opt/mapr/kafka-<version>/config
```

2. Run the `configure.sh` file. For example:

```
/opt/mapr/server/configure.sh -R
```

3. Start the Kafka Connect service

```
maprcli node services -name kafka-connect -action start -nodes <list of
Kafka Connect service nodes>
```

To see how configuration files are saved during an upgrade, see Saving Kafka Connect Configurations on page 3941.

Post-Upgrade Steps for Oozie

Complete the following steps after you upgrade Oozie with or without the MapR Installer.

1. Add customized properties from the configuration files that you saved before the upgrade to the files in the new Oozie `conf` directory: `/opt/mapr/oozie/oozie-<version>/conf/.`

2. If your Oozie installation is configured to use a MySQL or Oracle database and you are upgrading to a new Oozie version, copy the JDBC driver jar file for MySQL or Oracle to the following directory:

```
/opt/mapr/oozie/oozie-<oozie version>/libext
```

3. Run `configure.sh -R`:

```
# /opt/mapr/server/configure.sh -R
```

4. Verify that Oozie is in the 'Running' state and then run the following command to upgrade the database schema:

```
# /opt/mapr/oozie/oozie-<version>/bin/ooziedb.sh upgrade -run
```

5. Restart the Oozie server:

```
maprcli node services -name oozie -action restart -nodes <node names>
```

6. Start any Oozie coordinators that you stopped before the upgrade.

7. If needed, update the Oozie shared libraries, as described in Updating the Oozie Shared Libraries on page 4000.

Post-Upgrade Steps for Pig
Complete the following steps after you upgrade Pig with or without the MapR Installer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Migrate Custom Configurations (optional).

Migrate any custom configuration settings into the new default files in the conf directory (`/opt/mapr/pig/pig-<version>/conf/`).

Post-Upgrade Steps for Sentry
Complete the following steps after you upgrade Sentry with or without the MapR Installer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

If you saved the configuration files from the previous installation of Sentry, you can migrate the saved settings into the new default files. Use the schema tool to upgrade the database schema and then start the Sentry service.

1. (Optional) Migrate the settings in the saved configuration files from the previous version of Sentry into the new default files in `/opt/mapr/sentry/sentry-<version>/conf/`.

2. To upgrade the database schema using the schema tool, issue the following command:

```
/opt/mapr/sentry/sentry-<SENTRY_VERSION>/bin/sentry --command
schema-tool --conffile <sentry-site.xml> --dbType
<db-type> --upgradeSchema
```

3. To start the Sentry service, issue one of the following commands
   - If the warden.sentry.conf exists in the `/opt/mapr/conf/conf.d/` directory:

```
maprcli node services -name sentry  -action start -nodes <list of
Sentry service nodes>
```

   - If the warden.sentry.conf does not exist in the `/opt/mapr/conf/conf.d/` directory:

```
/opt/mapr/sentry/sentry-<SENTRY_VERSION>/bin/sentry-daemon.sh  start
<sentry-site.xml>
```

Post-Upgrade Steps for Spark
Complete the following steps after you upgrade Spark with or without the MapR Installer.

Post-Upgrade Steps for Spark Standalone Mode

1. (Optional) Migrate Custom Configurations.
   Migrate any custom configuration settings into the new default files in the conf directory (`/opt/mapr/spark/spark-<version>/conf`).
2. If Spark SQL is configured to work with Hive, copy the `hive-site.xml` file into the `conf` directory (`/opt/mapr/spark/spark-<version>/conf`).
3. Run the following commands to configure the secondary instances:
   a) For Spark 2.x:
      Copy the `/opt/mapr/spark/spark-<version>/conf/slaves.template` into `/opt/mapr/spark/spark-<version>/conf/slaves`.
      For Spark 3.x:
      Copy the `/opt/mapr/spark/spark-<version>/conf/workers.template` into `/opt/mapr/spark/spark-<version>/conf/workers`.
   b) Add the hostnames of the Spark worker nodes. Put one worker node hostname on each line.
      For example:

      ```
      localhost
      worker-node-1
      worker-node-2
      ```

4. Run configure.sh -R.
5. Restart all the spark secondary instances as the `mapr` user:
   For Spark 2.x:

   ```
   /opt/mapr/spark/spark-<version>/sbin/start-slaves.sh spark://
   <comma-separated list of spark master hostname: port>
   ```

   For Spark 3.x:

   ```
   /opt/mapr/spark/spark-<version>/sbin/start-workers.sh spark://
   <comma-separated list of spark master hostname: port>
   ```

6. Delete the old Spark directory from `/opt/mapr/spark`. For example, if you upgraded from Spark 2.1.0 to 2.3.1, you need to delete `/opt/mapr/spark/spark-2.1.0`.
   Starting with the EEP 6.1.0 release, for Spark 2.2.1 and later versions, after an upgrade the old directory is automatically removed. Only the new directory and the directory with the timestamp is present.

Post-Upgrade Steps for Spark on YARN

1. (Optional) Migrate Custom Configurations.
   Migrate any custom configuration settings into the new default files in the conf directory (`/opt/mapr/spark/spark-<version>/conf`). Also, if you previously configured Spark to use the Spark JAR file from a location on the MapR File System, you need to copy the latest JAR file to the MapR File System and reconfigure the path to the JAR file in the `spark-defaults.conf` file. See Configure Spark JAR Location on page 4033.

2. If Spark SQL is configured to work with Hive, copy the `hive-site.xml` file into the conf directory (`/opt/mapr/spark/spark-<version>/conf`).
3. Run configure.sh -R.
4. Delete the old Spark directory from `/opt/mapr/spark`. For example, if you upgraded from Spark 2.1.0 to 2.3.1, you need to delete `/opt/mapr/spark/spark-2.1.0`.

   Starting with the EEP 6.1.0 release, for Spark 2.2.1 and later versions, after an upgrade the old directory is automatically removed. Only the new directory and the directory with the timestamp is present.

Post-Upgrade Steps for Sqoop1
Complete the following steps after you upgrade Sqoop1 with or without the MapR Installer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Migrate Custom Configurations.

Migrate any custom configuration settings into the new default files in the `conf` directory (`/opt/mapr/sqoop/sqoop-<version>/conf/`).

Post-Upgrade Steps for Sqoop2
Complete the following steps after you upgrade Sqoop2 with or without the MapR Installer.

1. Perform one of the following steps:
   a) Manually update the `sqoop.properties` file with new options. On each Sqoop2 server node, set the following property in `sqoop.properties` file:

   ```
   org.apache.sqoop.repository.schema.immutable=false
   org.apache.sqoop.connector.autoupgrade=true
   org.apache.sqoop.driver.autoupgrade=true
   ```

   b) Run `configure.sh -R`:

   ```
   /opt/mapr/server/configure.sh -R
   ```

   c) Stop Sqoop2:

   ```
   maprcli node services -name sqoop2 -action stop -nodes <space
   delimited list of Sqoop2 server nodes>
   ```

   d) Use the upgrade utility to update the sqoop.properties file. On each Sqoop2 server node, run the upgrade utility.

   ```
   /opt/mapr/sqoop/sqoop-<version>/bin/sqoop2-tool upgrade
   ```

   When the upgrade utility completes successfully, the following message displays: `Tool class org.apache.sqoop.tools.tool.UpgradeTool has finished correctly.`

   Note: In Sqoop 1.99.6, the sqoop.properties file is in the following directory: `/opt/mapr/sqoop/sqoop-<version>/server/conf/`. In Sqoop 1.99.7, the sqoop.properties file is in the following directory: `/opt/mapr/sqoop/sqoop-<version>/conf/`.
2. Optionally, add customized properties from the `sqoop.properties` file that you saved before the upgrade to the new `sqoop.properties` file.

last-updated: Apr 11, 2022

3. If Sqoop is not running, start the Sqoop Server.

```
maprcli node services -name sqoop2 -action start -nodes <space delimited
list of Sqoop2 server
      nodes>
```

Post-Upgrade Steps for Tez
Complete the following steps after you upgrade Tez with or without the MapR Installer.

After a minor version update, for example from `Tez-0.9-1808` to `Tez-0.9-1901`, no changes to the user configuration, `tez-site.xml` file, are applied. To apply the latest changes manually, see the Tez Release Notes on page 6481.

1. (Optional) Migrate any custom configuration settings into the new default files in the `/opt/mapr/tez/tez-<old version>/conf/` directory.
2. Reconfigure the Hive-on-Tez User Interface. This is necessary because the old tomcat folder gets removed from the cluster during the upgrade procedure. For details, see Hive-on-Tez User Interface on page 3506.
3. If you are using the MapR Installer, no additional steps are required. For manual installation, you need to configure Hive and Tez. See Configuring Hive and Tez on page 3502.

Preparing the Cluster for a Maintenance Update

This section identifies how to prepare for applying either a minor update or a patch.

Depending on the task you need to perform, see the following topics:

To prepare for a minor update of the core version:	Preparing to Upgrade MapR Core on page 303
To prepare to apply a patch:	Verify Cluster Readiness for a Patch

For more information about maintenance updates, see Performing a Maintenance Update on page 5447.

Performing a Maintenance Update

Perform a maintenance update when you want to upgrade to a new patch version of MapR core or apply a patch.

A maintenance update is an update to your installed MapR software that does not require configuration-file changes. Performing a maintenance update has no effect on the ecosystem packages (EEP components). You perform a maintenance update when you want to do either or both of the following:

- **Update to a new patch version of MapR core.** For example, you can perform a maintenance update to change your MapR core version from MapR 6.1.0 to MapR 6.1.1. You cannot use a maintenance update to change your MapR core version from a minor version, such as 6.1, to another minor version, such as 6.2. Use the **Version Upgrade** button for minor-version upgrades. The **Version Upgrade** button also permits an upgrade to a patch version of MapR core.
- **Apply a patch.** The **Maintenance Update** page is one of several installer screens that offer the **Patch file** option. See Applying a Patch Using the MapR Installer on page 438.

You cannot perform a maintenance update if your current EEP version is incompatible with the selected MapR core version. For example, you cannot do a maintenance update from MapR 6.1.0 and EEP 6.3.0 to MapR 6.1.1 because EEP 6.3.0 is not compatible with MapR 6.1.1. For EEP and core compatibility information, see EEP Support and Lifecycle Status on page 5531.

Note: The maintenance update is an offline update (not a rolling update).

You perform a maintenance update using the MapR Installer. To perform a maintenance update:

1. Verify that your installed EEP is supported by the core version you plan to select for the maintenance update. To check your EEP version, see Checking the EEP Version on page 5413. For EEP and core compatibility information, see EEP Support and Lifecycle Status on page 5531.
2. Update the MapR Installer to the latest supported version. See Updating the MapR Installer on page 5409.
3. Prepare the cluster for a maintenance update by referring to one or both of these topics:
   - Preparing to Upgrade MapR Core on page 303
   - Verify Cluster Readiness for a Patch
4. Start the MapR Installer. For more information, see MapR Installer on page 5395.
5. Click the **Maintenance Update** button.
6. Change the MapR core version, or install a MapR core patch, or both.

   Important: During patch-file installation, do not refresh the browser page while the patch file is being uploaded. Doing so can interrupt the upload process.
7. Click **Next** to complete the update.

Related concepts

Checking the EEP Version on page 5413
Some MapR Installer operations require you to know the version of the currently installed MapR Ecosystem Pack (EEP). You can check the EEP version easily from within the MapR Installer user interface or derive the EEP version from your repository information.

MapR Installer Updates on page 5481
MapR Installer updates provide new features or bug fixes.

Related reference

EEP Support and Lifecycle Status on page 5531
This page shows the EEPs that are supported for different core releases and the current lifecycle status for each EEP.

Setting Up Clients and Services

Describes how to set up and use interfaces to an cluster from a client computer.

MapR packages are contained in two different repositories:

- **Core packages**: Contains the API server, the webserver, CLDB, the core MapR package, file server, the NFS servers, the gateway, various POSIX and thin clients, and ZooKeeper. The latest versions of these packages are at https://package.mapr.com/releases/v6.2.0/.
- **EEP (previously *MEP*) packages**: These are the ecosystem packages and contain Drill, Hadoop, Hive, Livy, Pig, Spark, Tez, and Yarn. Available versions of these packages are at https://package.mapr.com/releases/MEP/.

MapR provides the following interfaces for working with a cluster from a client computer:

Direct Access NFS™

Describes how to configure Direct Access NFS to mount the MapR File System to a local directory.

Use Direct Access NFS™ to mount the MapR filesystem locally as a directory on a Mac, Linux, or Windows computer.

last-updated: Apr 11, 2022

See Managing the MapR NFS Service on page 1176 for more information.

Installing MapR NFS
Describes how to install the NFS service on a node.

The following sections describe how to install the NFSv3 server, NFSv4 server, and the NFS client.

Installing the NFSv3 Server

- Install the NFSv3 server package.
  To install, run the following command:

Red Hat or CentOS	`yum install mapr-nfs`
Ubuntu	`apt-get install mapr-nfs`
SLES	`zypper install mapr-nfs`

  If the NFS server is installed without fileserver on a node, the node will be placed in the `/nfsserver` topology. If the fileserver is installed at a later time, the node will be moved to the `/data` topology, which is the default for fileserver nodes.

Installing the NFSv4 Server

The NFSv4 server can only be installed on MapR 6.1 or later clusters. Both NFSv4 and NFSv3 servers cannot run on the same node. If you have the NFS client running on an edge node, you can use that client to connect to the MapR NFS server on clusters running either 5.2, where NFSv3 server can be installed, or 6.1, where NFSv4 or NFSv3 can be installed.

1. Download, if necessary, and install the `nfs-utils` package, if it is already not installed, on the host where you plan to install the NFSv4 server.
2. Modify the `/etc/yum.repos.d/mapr_ecosystem.repo` file on CentOS/SLES or `/etc/apt/sources.list` file on Ubuntu to add the following:
   - CentOS:

```
[mapr-dev-ecosystem]
name=mapr-dev-ecosystem
baseurl=http://artifactory.devops.lab/artifactory/eco-rpm/releases/
opensource/redhat
enabled=1
gpgcheck=0
```

   - Ubuntu:

```
deb http://artifactory.devops.lab/artifactory/eco-deb binary trusty
#opensource.repo
```

3. Ensure that `rpc.statd` is running on the node.
   To verify, run the following command:

```
ps -ef| grep rpc.st
    rpcuser 18889 1 0 01:04 ? 00:00:00 /sbin/rpc.statd
    root 27016 6933 0 01:25 pts/0 00:00:00 grep color=auto rpc.st
```

   If it is already not running, execute the following to start it:

```
/sbin/rpc.statd
```

4. Install NFSv4 server package.
   To install, run the following command:

Red Hat or CentOS	`yum install mapr-nfs4server`
Ubuntu	`apt-get install mapr-nfs4server`
SLES	`zypper install mapr-nfs4server`

   The `mapr-nfsganesha` package is also installed as a dependency package. If NFS server is installed without fileserver on a node, the node is in `/nfsserver` topology. If fileserver is installed at a later time, the node is moved to `/data` topology, which is the default for fileserver nodes.

5. Run `configure.sh` on page 2053 utility with the `-u` and `-g` options to configure the services to run under user `mapr` and the group of the `mapr` user.

   Important: This step is required only if you are configuring NFSv4 server to work with Kerberos.

Installing the NFS Client

- To install the NFS client, run the following command:

Red Hat or CentOS	`sudo yum install nfs-utils`
Ubuntu	`sudo apt-get install nfs-common`
SLES	`sudo zypper install nfs-client`

Note: NFSv3 clients cannot connect to the NFSv4 server because the NFSv4 server only supports v4 protocol.

Mounting NFS on the MapR Filesystem on a Cluster Node

Refer to Accessing Data with NFS v4 on page 1193 and Mounting NFS to MapR File System on a Cluster Node on page 1185 for steps to mount NFS to a MapR filesystem.

last-updated: Apr 11, 2022

Before You Start Using MapR NFS

Make sure the following conditions are met before using the MapR NFS gateway:

- The stock Linux NFS service must not be running. Linux NFS and MapR NFS cannot run concurrently.
- MapR NFSv3 and NFSv4 should not be installed on the same node.
- The lock manager (nlockmgr) must be disabled.
- On Red Hat and CentOS v6.0 and higher, the `rpcbind` service must be running.

  You can use the command `ps ax | grep rpcbind` to check.
- On Red Hat and CentOS v5.x and lower, and on Ubuntu and SLES, the `portmapper` service must be running.

  You can use the command `ps ax | grep portmap` to check.
- The `mapr-nfs` package for NFSv3 or `mapr-nfs4server` package for NFSv4 must be present and installed.

  You can list the contents in the `/opt/mapr/roles` directory to check for `nfs` in the list.
- Make sure you have applied a Community Edition (M3) license or an Enterprise Edition (M5) license (paid or trial) to the cluster.

  See Adding a License.
- Make sure the MapR NFS service is started.

  See Starting, Stopping, and Restarting MapR NFSv3 on page 1184 or Starting, Stopping, and Restarting MapR NFSv4 on page 1216.
- Verify that the primary group of the user listed for `mapr.daemon.user` in the `/opt/mapr/conf/daemon.conf` file is `mapr.daemon.group`.

  Restart Warden after any changes to `daemon.conf`.

For information about mounting the cluster using:

- NFSv3, see Accessing Data with NFS v3 on page 1183.
- NFSv4, see Accessing Data with NFS v4 on page 1193.

For information on upgrading your cluster, see Upgrading MapR Core on page 294.

Warning: To preserve compatibility with 32-bit applications and system calls, MapR-NFS uses 32-bit inode numbers by default. On 64-bit clients, this default forces the client's 64-bit inode numbers to be hashed down to 32 bits. Hashing 64-bit inodes down to 32 bits can potentially cause inum conflicts. To change the default behavior to 64-bit inode numbers, set the value of the `Use32BitFileId` property to 0 in the `nfsserver.conf` file, then restart the NFS server.

MapR Client

Describes how to install the MapR client to run Hadoop commands, jobs, and applications from a client machine.

You can use the client to:

- Submit MapReduce applications.
- Submit YARN applications.

- Run `hadoop fs` on page 5364, and `hadoop mfs` on page 5373 commands.

The method that you use to submit the Hadoop commands on Mac and Windows clients is different from the method that is used on Linux machines. For more information, see Running Hadoop Commands on a Mac and Windows Client.

Installing the MapR Client
This section describes how to prepare the client machine for the installation process.

CAUTION: Do not attempt to install MapR Ecosystem Pack (EEP) service components on client machines. Client machines do not have the service-management framework required to host the service components.

Before you install the MapR client, perform the following steps:

- **Verify that the operating system on the machine where you plan to install the client is supported.** For a list of operating systems that are compatible with the MapR clients, see MapR Client Support Matrix on page 5598.
- **Verify that the machine where you plan to install the client is not a cluster node.** The MapR client is intended for use on a computer that has no other MapR server software installed.
- **Ensure that the hostname of the machine is set to a fully qualified DNS name**. This is critical as else configure.sh will fail to generate SSL keys.
- **Obtain connectivity information and cluster setup requirements.** When you use `configure.sh` to configure the client, you will need to know the following details:
  - The cluster name. You will need the cluster name when you specify the `-N` parameter.
  - The IP addresses and ports of the CLDB nodes on the cluster. You will need this information when you specify the CLDB nodes with the `-C` parameter.
  - If one or more nodes in the cluster run the ResourceManager, you may need to specify the hostname or IP address for each ResourceManager nodes using the `-RM` parameter. If the cluster is configured to use zero-configuration failover, do not specify the ResourceManager nodes. If the cluster is not configured to use zero-configuration failover, specify each ResourceManager node.
  - Determine if the cluster is secure. If the cluster is secure, you will need to specify the `-secure` parameter when you run `configure.sh`.
  - If a node in the cluster runs the HistoryServer, note the hostname for the HistoryServer. You must specify each HistoryServer node using the `-HS` parameter.
- **Add the hostname mapping.** In the `/etc/hosts` file of the client machine, add a mapping between the CLDB nodes in the cluster and the IP addresses of those nodes.

  For example, add the IP address 10.10.82.22 and CLDB node name centos22 on the Mac OSX where you installed the client:

```
127.0.0.1       localhost
255.255.255.255 broadcasthost
::1             localhost
fe80::1%lo0     localhost
10.10.82.22     centos22
```

- **Configure repositories for the client.** The client nodes also need to have the MapR repositories configured in order to pull the client packages. See MapR Repositories and Packages on page 128.

last-updated: Apr 11, 2022

To install the client, obtain the MapR packages for your operating system at https://package.mapr.hpe.com/releases/ and complete the installation steps described in one of the subsequent topics.

Installing the MapR Client on CentOS, RedHat, Oracle Linux
This section describes how to install the MapR client on CentOS, RedHat, Oracle Linux.

1. Remove any previous MapR software. You can use `rpm -qa | grep mapr` to get a list of installed MapR packages, then type the packages separated by spaces after the `rpm -e` command:

   ```
   rpm -qa | grep mapr
   rpm -e mapr-fileserver mapr-core
   ```

2. Install the MapR package key. The package key must be installed before you can install MapR packages. For more information, see Step 1: Install the Package Key on page 142:

   ```
   rpm --import https://package.mapr.hpe.com/releases/pub/maprgpg.key
   ```

3. Install the MapR client for your target architecture:

   ```
   yum install mapr-client.x86_64
   ```

4. To use this client with a secure cluster or clusters, copy the `ssl_truststore` file from the `/opt/mapr/conf` directory on the cluster to the `/opt/mapr/conf` directory on the client.

   If this client will connect to multiple clusters, merge the `ssl_truststore` files with the `/opt/mapr/server/manageSSLKeys.sh` tool. See Managing Secure Clusters on page 1482 for details on how to connect to a secure cluster.

5. Run `configure.sh` to configure the client. In the following examples, the `-N` parameter specifies the cluster name, the `-c` (lowercase) parameter specifies a client configuration, the `-secure` parameter is added if the cluster is secure, the `-C` (uppercase) parameter specifies the CLDB nodes, and the `-HS` parameter specifies the HistoryServer node. To ensure that the client can connect in the event of a CLDB node failure, all CLDB nodes are specified. For more information about the syntax, parameters, and behavior of `configure.sh`, see configure.sh.

   Non-secure cluster example

   ```
   /opt/mapr/server/configure.sh -N
   my.cluster.com -c -C
   mynode01:7222,mynode02:7222,mynode03
   :7222 -HS mynode02
   ```

   Secure cluster example

   ```
   /opt/mapr/server/configure.sh -N
   my.cluster.com -c -secure -C
   mynode01:7222,mynode02:7222,mynode03
   :7222 -HS mynode02
   ```

Note:

If the cluster was configured with a cluster-admin `user:group` that is different from the default `mapr:mapr` value, you must include options to specify the cluster-admin user and group information when you run `configure.sh` to configure the client.

If the cluster-admin user ID is present on the client node, include these options:

- `-u`
- `-g`

If the cluster-admin user ID is not present on the client node, include these options:

- `-u`
- `-g`
- `--create-user | -a`
- `-U`
- `-G`

The following table describes each option:

Option	Description
`-u`	The user name under which MapR services run.
`-g`	The group name under which MapR services run.
`--create-user | -a`	Creates a local user to run MapR services, using the specified user either from the `-u` parameter, or from the environment variable `$MAPR_USER`.
`-U`	The user ID to use when creating `$MAPR_USER` with the `--create-user` or `-a` option; corresponds to the `-u` or `--uid` option of the `useradd` command in Linux.
`-G`	The group ID to use when creating `$MAPR_USER` with the `-create-user` or `-a` option; corresponds to the `-g` or `-gid` option of the `useradd` command in Linux.

6. At the end of the client installation, run the maprlogin password command to create a valid ticket to connect to the cluster.

Related concepts

Managing Secure Clusters on page 1482
Provides procedures that will enable you to use MapR clusters securely.

Related tasks

Step 1: Install the Package Key on page 142
Before you install MapR packages, you must install the package key.

Related reference

configure.sh on page 2053
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

Installing the MapR Client on SLES

This section describes how to install the MapR Client on SLES.

last-updated: Apr 11, 2022

1. Remove any previous MapR software. You can use `rpm -qa | grep mapr` to get a list of installed MapR packages:

   ```
   rpm -qa | grep mapr
   ```

   Then type the package names separated by spaces after the `zypper rm` command. For example:

   ```
   zypper rm mapr-fileserver mapr-core
   ```

2. Run the following command to install the MapR client:

   ```
   zypper install mapr-client
   ```

3. To use this client with a secure cluster or clusters, copy the `ssl_truststore` file from the `/opt/mapr/conf` directory on the cluster to the `/opt/mapr/conf` directory on the client.

   If this client will connect to multiple clusters, merge the `ssl_truststore` files with the `/opt/mapr/server/manageSSLKeys.sh` tool.

4. Run `configure.sh` to configure the client. In the following examples, the `-N` parameter specifies the cluster name, the `-c` (lowercase) parameter specifies a client configuration, the `-secure` parameter is added if the cluster is secure, the `-C` (uppercase) parameter specifies the CLDB nodes, and the `-HS` parameter specifies the HistoryServer node. To ensure that the client can connect in the event of a CLDB node failure, all CLDB nodes are specified. For more information about the syntax, parameters, and behavior of `configure.sh`, see configure.sh.

   Secure cluster example

   ```
   /opt/mapr/server/configure.sh -N
   my.cluster.com -c -secure -C
   mynode01:7222,mynode02:7222,mynode03
   :7222 -HS mynode02
   ```

   Non-secure cluster example

   ```
   /opt/mapr/server/configure.sh -N
   my.cluster.com -c -C
   mynode01:7222,mynode02:7222,mynode03
   :7222 -HS mynode02
   ```


last-updated: Apr 11, 2022

Note:

If the cluster was configured with a cluster-admin `user:group` that is different from the default `mapr:mapr` value, you must include options to specify the cluster-admin user and group information when you run `configure.sh` to configure the client.

If the cluster-admin user ID is present on the client node, include these options:

- `-u`
- `-g`

If the cluster-admin user ID is not present on the client node, include these options:

- `-u`
- `-g`
- `--create-user | -a`
- `-U`
- `-G`

The following table describes each option:

Option	Description
`-u`	The user name under which cluster services run.
`-g`	The group name under which cluster services run.
`--create-user | -a`	Creates a local user to run cluster services, using the specified user either from the `-u` parameter, or from the environment variable `$MAPR_USER`.
`-U`	The user ID to use when creating `$MAPR_USER` with the `--create-user` or `-a` option; corresponds to the `-u` or `--uid` option of the `useradd` command in Linux.
`-G`	The group ID to use when creating `$MAPR_USER` with the `-create-user` or `-a` option; corresponds to the `-g` or `-gid` option of the `useradd` command in Linux.

5. At the end of the client installation, run the `maprlogin password` command to create a valid ticket to connect to the cluster.

Related concepts

Managing Secure Clusters on page 1482
Provides procedures that will enable you to use MapR clusters securely.

Related reference

configure.sh on page 2053
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

Installing the MapR Client on Ubuntu

This section describes how to install the MapR client on Ubuntu.

last-updated: Apr 11, 2022

1. Remove any previous MapR client software.

   You can use `dpkg --list | grep mapr` to get a list of installed MapR packages, then type the packages separated by spaces after the `dpkg -r` command. Example: `dpkg -l | grep mapr`.

   ```
   dpkg -r mapr-core mapr-fileserver
   ```

2. Update your Ubuntu repositories. For example:

   ```
   apt-get update
   ```

3. Run the following command to install the MapR client:

   ```
   apt-get install mapr-client
   ```

4. To use this client with a secure cluster or clusters, copy the `ssl_truststore` file from the `/opt/mapr/conf` directory on the cluster to the `/opt/mapr/conf` directory on the client.

   If this client will connect to multiple clusters, merge the `ssl_truststore` files with the `/opt/mapr/server/manageSSLKeys.sh` tool.

5. Run `configure.sh` to configure the client. In the following examples, the `-N` parameter specifies the cluster name, the `-c` (lowercase) parameter specifies a client configuration, the `-secure` parameter is added if the cluster is secure, the `-C` (uppercase) parameter specifies the CLDB nodes, and the `-HS` parameter specifies the HistoryServer node. To ensure that the client can connect in the event of a CLDB node failure, all CLDB nodes are specified. For more information about the syntax, parameters, and behavior of `configure.sh`, see configure.sh.

   Secure cluster example

   ```
   /opt/mapr/server/configure.sh -N
   my.cluster.com -c -secure -C
   mynode01:7222,mynode02:7222,mynode03
   :7222 -HS mynode02
   ```

   Non-secure cluster example

   ```
   /opt/mapr/server/configure.sh -N
   my.cluster.com -c -C
   mynode01:7222,mynode02:7222,mynode03
   :7222 -HS mynode02
   ```

Note:

If the cluster was configured with a cluster-admin `user:group` that is different from the default `mapr:mapr` value, you must include options to specify the cluster-admin user and group information when you run `configure.sh` to configure the client.

If the cluster-admin user ID is present on the client node, include these options:

- `-u`
- `-g`

If the cluster-admin user ID is not present on the client node, include these options:

- `-u`
- `-g`
- `--create-user | -a`
- `-U`
- `-G`

6. At the end of the client installation, run the maprlogin password command to create a valid ticket to connect to the cluster.

Related concepts

Managing Secure Clusters on page 1482
Provides procedures that will enable you to use MapR clusters securely.

Related reference

configure.sh on page 2053
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

Installing the MapR Data Platform Client on Mac OS X

This section describes how to install the MapR Data Platform client on Mac OS X.

Limitation: Under OS X, the `getgroups` command returns a maximum of 16 groups for a user. If the Mac OS user for which you are installing the client attempts to read or write to a MapR Data Platform filesystem resource as a member of a group that was not included in the list of 16 groups returned by `getgroups`, file permission errors may result.

1. Create the `/opt` directory: `sudo mkdir -p /opt`
2. Download the file for the version that you want to install:

   ```
   https://package.mapr.hpe.com/releases/<version>/mac/<mapr-client package name>
   ```

3. Open the **Terminal** application.
4. Extract `mapr-client-<version>.tar.gz` into the `/opt` directory:

   ```
   sudo tar -C /opt -zxf mapr-client-<version>.tar.gz*
   ```

5. Before running `configure.sh`, make sure that `JAVA_HOME` is set correctly for the client in the following script: `/opt/mapr/conf/env.sh`

   For example:

   ```
   $ export JAVA_HOME=$(/usr/libexec/java_home)
   ```

6. To use this client with a secure cluster or clusters, copy the `ssl_truststore` file from the `/opt/mapr/conf` directory on the cluster to the `/opt/mapr/conf` directory on the client.

   If this client will connect to multiple clusters, merge the `ssl_truststore` files with the `/opt/mapr/server/manageSSLKeys.sh` tool.

7. Run `configure.sh` to configure the client. In the following examples, the `-N` parameter specifies the cluster name, the `-c` (lowercase) parameter specifies a client configuration, the `-secure` parameter is added if the cluster is secure, the `-C` (uppercase) parameter specifies the CLDB nodes, and the `-HS` parameter specifies the HistoryServer node. To ensure that the client can connect in the event of a CLDB node failure, all CLDB nodes are specified. For more information about the syntax, parameters, and behavior of `configure.sh`, see configure.sh.

   Non-secure cluster example

   ```
   sudo /opt/mapr/server/
   configure.sh -N
   my.cluster.com -c -C
   mynode01:7222,mynode02:7222,mynode03
   :7222 -HS nodeA
   ```

   Secure cluster example

   ```
   sudo /opt/mapr/server/
   configure.sh -N
   my.cluster.com -c -secure -C
   mynode01:7222,mynode02:7222,mynode03
   :7222
   ```

   Note:

   If the cluster was configured with a cluster-admin `user:group` that is different from the default `mapr:mapr` value, you must include options to specify the cluster-admin user and group information when you run `configure.sh` to configure the client.

   If the cluster-admin user ID is present on the client node, include these options:

   - `-u`
   - `-g`

   If the cluster-admin user ID is not present on the client node, include these options:

   - `-u`
   - `-g`
   - `--create-user | -a`
   - `-U`
   - `-G`

last-updated: Apr 11, 2022

8. At the end of the client installation, run the maprlogin password command to create a valid ticket to connect to the cluster.

For information about running Hadoop commands on Mac OS X, see Running Hadoop Commands on a Mac and Windows Client on page 417.

Related concepts

Managing Secure Clusters on page 1482
Provides procedures that will enable you to use MapR clusters securely.

Related reference

configure.sh on page 2053
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

Installing the MapR Client on Windows

Installing the MapR Data Platform client makes it possible to access the MapR filesystem from a Windows workstation.

Compatibility with Network Address Translation (NAT) Adapters

In VM environments, the MapR client on Windows works with a single NAT virtual adapter as long as it is the only virtual adapter configured for the VM. If you want to use more than one adapter, you must use other types of virtual adapters. If you use multiple NAT adapters in your VM environment, your jobs and file-system operations will fail.

Use these steps to install the client:

1. Make sure Java is installed on the computer and that the `JAVA_HOME` environment variable is set correctly.

   Note: The path that you set for the JAVA_HOME environment variable must not include spaces.

2. Create the directory `\opt\mapr` on your `c:` drive (or another hard drive of your choosing). You can use Windows Explorer, or type the following at the command prompt: `mkdir c:\opt\mapr`
3. Set the `MAPR_HOME` environment variable to `c:\opt\mapr`
4. Open the command line.
5. Use the following command to navigate to `MAPR_HOME`: `cd %MAPR_HOME%`
6. Download the file for the version that you want to install into `MAPR_HOME`:

   ```
   https://package.mapr.hpe.com/releases/<version>/windows/<package name>
   ```

7. Extract the archive by right-clicking on the file and selecting **Extract All...**
8. From the command line, run `configure.bat` to configure the client. In the following examples, the `-N` parameter specifies the cluster name, the `-c` (lowercase) parameter specifies a client configuration, the `-secure` parameter is added if the cluster is secure, the `-C` (uppercase) parameter specifies the CLDB nodes, and the `-HS` parameter specifies the HistoryServer node. To ensure that the client can connect in the event of a CLDB node failure, all CLDB nodes are specified. For more information about the syntax, parameters, and behavior of `configure.sh`, see configure.sh.

   For details about the syntax, parameters, and behavior of configure.bat, see configure.sh.

   Non-secure cluster example

   ```
   server\configure.bat -N
   my.cluster.com -c -C
   ```


last-updated: Apr 11, 2022

```
mynode01:7222,mynode02:7222,mynode03
:7222 -HS nodeA
```

Secure cluster example

```
server\configure.bat -N
my.cluster.com -c -secure -C
mynode01:7222,mynode02:7222,mynode03
:7222
```

9. To use this client with a secure cluster or clusters, copy the `ssl_truststore` file from the `/opt/mapr/conf` directory on the cluster to the `c:\opt\mapr\conf` directory on the client.

   If this client will connect to multiple clusters, you must merge the `ssl_truststore` files on the server by using the `/opt/mapr/server/manageSSLKeys.sh` tool, and then copy the merged file to `c:\opt\mapr\conf` on the client. For an example of merging the `ssl_truststore` files, see step 3 in Configuring Secure Clusters for Running Commands Remotely on page 1484.

   See Managing Secure Clusters on page 1482 for details on how to connect to a secure cluster.

10. At the end of the client installation, run the maprlogin password command to create a valid ticket to connect to the cluster.

For more information about running Hadoop commands on Windows, see Running Hadoop Commands on a Mac and Windows Client on page 417.

Related concepts

Managing Secure Clusters on page 1482
Provides procedures that will enable you to use MapR clusters securely.

Configuring the Windows Client

You can use the MapR client on Windows.

This section describes how to configure the MapR client on Windows:

Configuring MapR Client User on Windows

Before you use the Windows Client, configure it with information from your cluster.

Before running applications on the Windows Client, configure the `core-site.xml` file with the UID, GID, and user name of the cluster user that will be used to access the non-secure cluster.

Note: If you are on secure cluster, this configuration is not needed because on secure clusters, the username is available through the ticket.

Complete the following steps:

1. Obtain the UID and GID that has been set up for your user account. To determine the correct UID and GID values for your username, log into a cluster node and type the `id` command. In the following example, the UID is 1000 and the GID is 2000:

   ```
   $ id
   uid=1000(juser) gid=2000(juser)
   groups=4(adm),20(dialout),24(cdrom),46(plugdev),105(lpadmin),119(admin),1
   22(sambashare),2000(juser)
   ```

2. Add the following parameters to the `core-site.xml` files that correspond to the version of the hadoop commands that you plan to run:

```
<property>
  <name>hadoop.spoofed.user.uid</name>
  <value>{UID}</value>
</property>
<property>
  <name>hadoop.spoofed.user.gid</name>
  <value>{GID}</value>
</property>
<property>
  <name>hadoop.spoofed.user.username</name>
  <value>{id of user who has UID}</value>
</property>
```

Warning: You must use the *numeric* values for `UID` and `GID`, not the text names.

Note: When wire-level security is implemented on Windows, spoofing is not supported. This greatly increases the security of the cluster, but the `core-site.xml` file settings above then have no effect.

For MapReduce version 2 or other applications that run on YARN, the `core-site.xml` file(s) that you need to edit is located at: `%MAPR_HOME%\hadoop\hadoop-2.x.x\etc\hadoop\core-site.xml`.

Configure the Windows Client to Submit MapReduce Version 2 Applications
This section describes how to add a property that allows you to run MapReduce version 2 applications from a Windows Client.

- Before running any MapReduce version 2 (MRv2) applications from a Windows client, add the following property to the `mapred-site.xml` file:

```
<property>
<name>mapreduce.app-submission.cross-platform</name>
<value>true</value>
</property>
```

The `mapred-site.xml` file for MRv2 applications is located in the following directory:

```
%MAPR_HOME%\hadoop\hadoop-2.x.x\etc\hadoop\mapred-site.xml
```

MapR POSIX Clients

Describes how to install the POSIX loopback NFS, and the FUSE-based POSIX clients.

This section contains instructions for installing the MapR POSIX Clients. The MapR software provides a POSIX loopback NFS client package, and FUSE-based POSIX Basic and Platinum client packages. Each FUSE-based POSIX client package implies a specific MapR filesystem throughput optimization of n/G per second. These clients can be installed and used according to the same principles as the POSIX loopback NFS client.

Overview of loopback NFS and FUSE-based POSIX Clients

The NFS version 3.0 protocol does not have secure data transit, nor authentication capabilities to authenticate users who connect to the MapR NFS version 3.0 server. Any client and any user can connect to the MapR NFS server, remotely.

Loopback NFS tightens security and ensures that only authenticated users can access the NFS server. The loopback NFS client runs on the same node as the NFS server, and can connect to only that NFS server based on the ticket generated. Remote clients cannot access the server.

> **Tip:** For enhanced performance and reliability, use a FUSE-based POSIX client that works similar to the loopback NFS server in terms of security and authentication, and always set the MAPR_SUBNETS enivronment variable.

Note: NFS version 4.0 has in-built security and authentication. Remote clients and users can access MapR NFS version 4.0 servers securely.

Installing the mapr-loopbacknfs Package
MapR POSIX clients enable application servers, web servers, and other client nodes and applications to read and write directly and securely to a MapR cluster.

Consider installing `mapr-loopbacknfs` if you need a secure POSIX client. You can install the `mapr-loopbacknfs` client on any client node, even your laptop, if you have Linux installed. A client node must have a supported Linux OS distribution and must be outside the MapR cluster, not running `mapr-fileserver` or other Hadoop services. You cannot install the MapR POSIX client on a Windows or Mac OS X machine.

As a POSIX client, `mapr-loopbacknfs` can use any of the 10 POSIX connections provided by the Basic license. If you need more POSIX client connections, consider upgrading to a Platinum license, as described in Preparing for Installation (MapR POSIX Client) on page 400.

Attention: Note that the Installer installs `mapr-loopbacknfs` on all nodes in the cluster when **Enable NFS** is not specified.

To install `mapr-loopbacknfs` on your machine, perform the following steps for your version of Linux, as the `root` user or using `sudo`. The package is installed to the `/usr/local/mapr-loopbacknfs` directory.

- **For CentOS, RHEL, or Oracle Linux**

```
[root@ip-<ip_address> ~] # yum install mapr-loopbacknfs
```

- **For SLES**

```
# zypper install mapr-compat-suse
```

```
# zypper install mapr-loopbacknfs
```

- **For Ubuntu**

```
# sudo apt-get install mapr-loopbacknfs
```

Installing FUSE-based POSIX Client Packages
Describes how to install a FUSE-based Basic or Platinum POSIX client package.

FUSE-based POSIX clients allow app servers, web servers, and other client nodes and apps to read and write data directly and securely to a MapR cluster like a Linux filesystem.

Preparing for Installation (MapR POSIX Client)
To install the MapR POSIX Client on a node, you must meet certain requirements.

The MapR POSIX client can be installed on any node if you have Linux installed. You cannot install the MapR POSIX client on a Windows or Mac OS X machine. The client requires Java 1.7 or later to be installed on your system.

last-updated: Apr 11, 2022

POSIX Client Package Summary

Two separate POSIX client packages are provided, each with different performance tiers. Each package implies a specific MapR File System throughput optimization of n/GB (bytes) per second where n=1 for Basic, and n=5 for Platinum POSIX client. These clients can be installed and used according to the same principles as the POSIX loopback NFS client. The following table lists the packages.

	Basic POSIX Client	Platinum POSIX Client
Name	MapR POSIX Client Basic	MapR POSIX Client Platinum
Number of Clients	Up to 10 free	Free
Performance	Up to 1GB (Byte)/sec	Up to 5GB (Byte)/sec (with HT disabled)
MapR Package	`mapr-posix-client-basic`	`mapr-posix-client-platinum`

Client-Side Hardware Requirements

To accommodate the POSIX client, your hardware should meet the following requirements:

	Basic	Platinum
Hyper-threading*	Off	Off
Physical CPU(s) (with HT disabled)	1	2
Core(s) per socket	8	8
Socket(s)	1	2
Processor speed	2.2 GHz	2.60 GHz
Memory Click Speed	>=1333 MHz	>=1666 MHz
NICs	10 Gbps (2 GB/sec)	40 Gbps (5 GB/sec)

* Disabling hyper-threading (HT) improves performance.

Linux Kernel Tuning Recommendations

If the client connects to the servers over a 40GigE switch, you should set the following parameters in `/etc/sysctl.conf` to 16 MB on all the nodes to achieve maximum throughput.

Note: This setting is not required, but if the network has a large capacity, this setting allows the OS to buffer large chunks of data for transmission, which improves throughput.

- `net.core.rmem.max`
- `net.core.rmem_default`
- `net.core.wmem_max`
- `net.core.wmem_default`
- `net.ipv4.tcp_rmem`
- `net.ipv4.tcp_wmem`
- `net.ipv4.tcp_mem`

Installing the FUSE-Based POSIX Client

Describes how to install the FUSE-based POSIX client package on your system.

last-updated: Apr 11, 2022

FUSE-based POSIX clients require the FUSE kernel module. Run the following command to load the kernel module:

```
modprobe fuse
```

You can install the FUSE-based POSIX client on any node, including cluster nodes.

To install the mapr-posix-client-* package on your machine, where * refers to the basic or the platinum client package, perform the following steps for your version of Linux, as the `root` user, or using `sudo`. The package is installed to the `/opt/mapr/bin/` directory.

1. Run the following command to install the POSIX client package on your machine.
   - **For CentOS, RHEL, or Oracle Linux**

     ```
     # yum install mapr-posix-client-*, where * is either the basic or the
     platinum package
     For example: yum install mapr-posix-client-basic
     ```

     Troubleshooting: On Oracle Linux, you must also install the `compat-openssl10` package to get the POSIX client running:

     ```
     yum install compat-openssl10
     ```

   - **For SLES**

     ```
     # zypper install mapr-posix-client-*, where * is either the basic or
     the platinum package
     For example: zypper install mapr-posix-client-basic
     ```

   - **For Ubuntu**

     ```
     # sudo apt-get install mapr-posix-client-*, where * is either the
     basic or the platinum package
     For example: apt-get install mapr-posix-client-basic
     ```

2. To use this client with a secure cluster or clusters, copy the following files from the `/opt/mapr/conf` directory on the cluster to the `/opt/mapr/conf` directory on the client.
   - `ssl_truststore`
   - `ssl-client.xml`

   If this client will connect to multiple clusters, merge the `ssl_truststore` files with the `/opt/mapr/server/manageSSLKeys.sh` tool. You must perform the merging on the cluster. See Managing Secure Clusters on page 1482 for details on how to connect to a secure cluster.

3. Run `configure.sh` to set this node as a client node.
   - For a fresh installation of the POSIX client, run `configure.sh` as follows:

     ```
     /opt/mapr/server/configure.sh -N <clustername> -C <CLDBhost> -Z
     <ZooKeeperhost> -c
     ```

- When reinstalling the POSIX client, run `configure.sh` with the -R option to reuse the existing configuration.

```
/opt/mapr/server/configure.sh -c -R
```

Note: To use the POSIX client with a secure cluster, add the `-secure` option when running `configure.sh`. However, do NOT add the `-secure` option when running `configure.sh` with the -R option.

4. At the end of the client installation, run the maprlogin password command to create a valid ticket to connect to the cluster.

To configure, start, and mount the client:

- Configure the POSIX client
- Start the POSIX client
- Mount the cluster

About the MapR Persistent Application Client Container (PACC)

This container gives you seamless access to MapR cluster services.

This topic introduces the MapR Persistent Application Client Container (PACC), including its function, benefits, components, and applications.

The MapR (PACC) is a Docker-based container image that includes a container-optimized MapR client. The PACC provides seamless access to MapR Converged Data Platform services, including MapR File System, MapR Database, and MapR Event Store For Apache Kafka. The PACC makes it fast and easy to run containerized applications that access data in MapR.

FUSE POSIX Client for File-Based Applications

To support persistent, file-based applications, the MapR PACC includes a FUSE-based MapR POSIX Client, optimized for containers, that allows app servers, web servers, and other applications to read and write data directly to the MapR filesystem. If your cluster has a MapR POSIX Client for Containers license, the PACC can connect with MapR 5.1 or later clusters.

Traditionally, all file data created by containers is lost when a container is terminated, which can happen during an application or hardware failure. By using the POSIX client within the PACC, applications can reliably persist file data directly to the MapR filesystem, where it can be re-attached to the container in the event of application or hardware failures.

Support for Microservice Applications

To support stateful microservice applications, the MapR PACC also contains a container-optimized version of the MapR client, which includes libraries for accessing MapR Database and MapR Event Store For Apache Kafka.

Secure Access

The MapR PACC is designed to provide access to a secure cluster for all MapR Converged Platform data services. Users can pass a MapR ticket file into the container at runtime. All data access, whether to MapR File System, MapR Database, or MapR Event Store For Apache Kafka, is authorized and audited according to the authenticated identity of the ticket file.

last-updated: Apr 11, 2022

PACC Contents

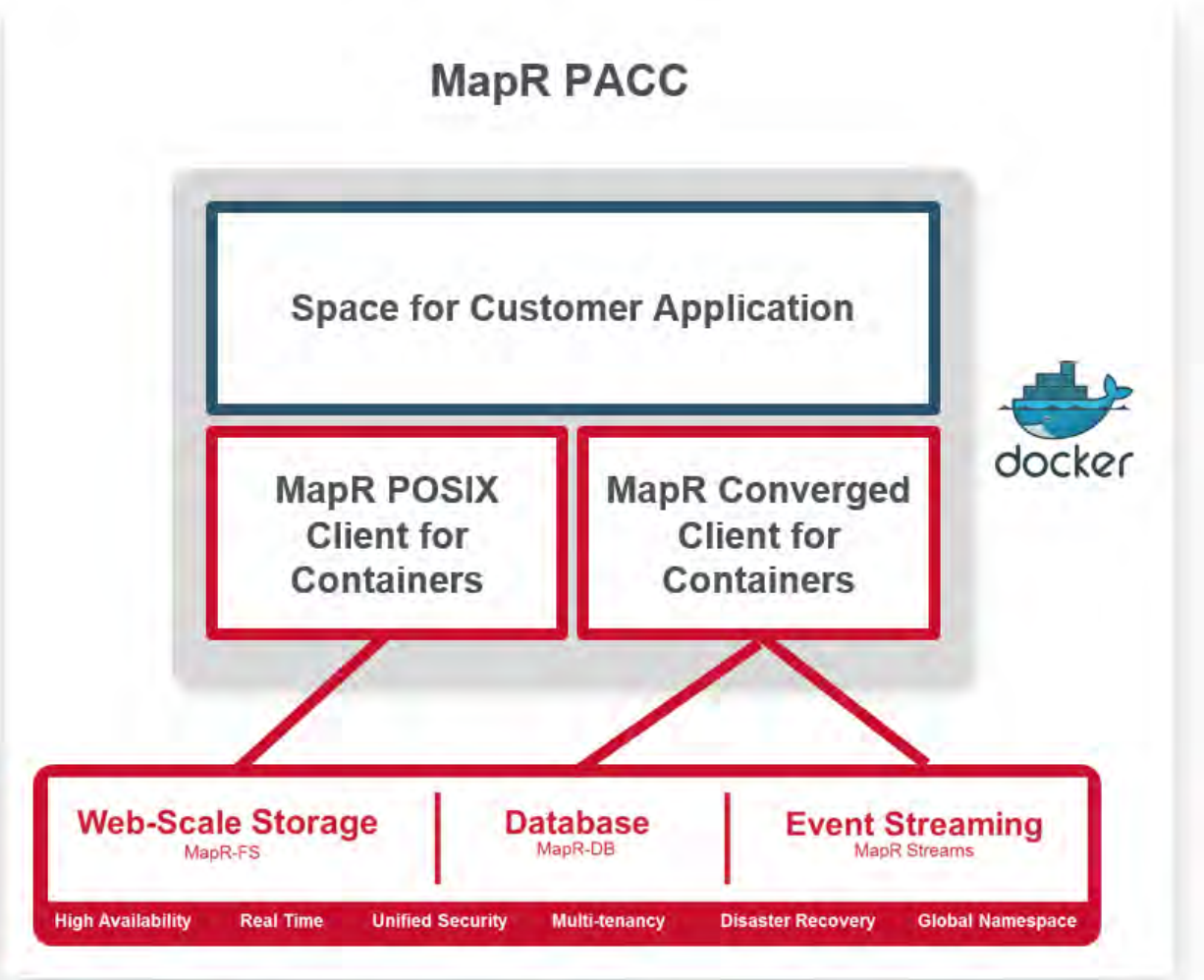
The PACC includes the following components:

- MapR Database Client[1]
- MapR Event Store For Apache Kafka Client
- POSIX Client for Containers
- Hadoop Client with YARN[2]
- HBase Client[2]
- Hive Client[2]
- Pig Client[2]
- Python
- Java
- Curl, Wget, Openssl, NFS-common, etc

[1]The MapR Database client includes support for MapR Database binary tables and MapR Database JSON tables.

[2]Included only if specified and only in MapR PACC images created using `mapr-setup.sh`.

The following diagram illustrates the contents of the PACC, and how it allows applications to access MapR Converged Data Platform services.



Pre-Built and User-Created Images

To get started with the MapR PACC, you can take advantage of pre-built Docker images or create your own images to include site-specific environmental parameters:

To . . .	See this topic
See a list of the MapR pre-built Docker images	Extending a MapR PACC on page 408
Create your own images containing MapR software	Creating a MapR PACC Image Using mapr-setup.sh on page 409

Before Deploying the MapR PACC

Perform a series of checks on the platform and MapR cluster before deploying the MapR PACC.

Before you deploy a MapR PACC, or an application container based on the PACC, perform the following checks on the platform and MapR cluster.

On the platform where you plan to deploy...	On the MapR cluster...

last-updated: Apr 11, 2022

• Verify that Docker 1.12.5 or later is installed and the Docker daemon is up and running. For download instructions, see this Docker website.
• Verify that you have sufficient disk space for use by the container.
  • 1 GB for the pre-built images
  • 1.5 GB for the user-created images
	• If you plan to connect to the MapR filesystem using the POSIX client, you *must* verify that you have a license for the MapR POSIX Client for Containers. No license is required to connect to MapR Database or MapR-Streams. To obtain the license, do one of the following:
	  • Clusters registered with the MapR Enterprise Trial license can connect unlimited POSIX Clients for Containers.
	  • For production use of the POSIX Client for Containers, contact your HPE sales representative.
	• Obtain the information that you will use to set up the container. You need to gather:
	  • The cluster name.
	  • The time zone of the container. If no time zone is specified, the container uses UTC as the default time zone.
	  • The IP addresses of the cluster nodes running the CLDB.
	  • A ticket if you need to connect to a secure cluster.

Leveraging MapR File System Storage for Application Persistence

The data associated with containers is typically not persistent. If you specify a MapR mount path using `MAPR_MOUNT_PATH`, any data written to the mounted directory will be persisted to the MapR filesystem. For example, you could create a symlink to the Apache log directory and persist all log files in the MapR filesystem.

If you specify a MapR mount path and if your hardware or application fails, re-launching the container in any location using the same Docker runtime environment variables will result in all pre-created data being available for use.

Security Considerations

You should be aware of security considerations for secure and non-secure clusters when using MapR PACCs. See Security Considerations for the MapR PACC.

Security Considerations for the MapR PACC

This section describes key considerations for using Docker containers with secure and non-secure MapR clusters.

Secure Clusters

Docker containers, like other virtualization technologies, allow client access from user identities that are not controlled by central IT. As a result, these technologies can be problematic when used with clusters that are not secure (where trust is based on trusting the client). Therefore, HPE suggests that you use secure clusters with MapR PACCs.

MapR PACCs, and applications built from them, are launched with a MapR ticket that contains the application's identity from the perspective of the cluster. On secure clusters, the user identity, user ID (`UID`), and group ID (`GID`) are specified in the MapR ticket and passed to the MapR filesystem for cluster communication. The ticket ensures that operations, such as authorization and auditing, are performed as the authenticated MapR user. A different ticket should be created for each container that is launched. The user's identity should be the identity of the user who accesses data.

last-updated: Apr 11, 2022

All access from Docker containers to the MapR cluster requires a MapR ticket be present inside the container. Users or administrators should generate a MapR ticket for each container prior to launch, and pass the ticket into the container at runtime. The MapR ticket *must* be generated for the user that your applications access the cluster as. You should create a container user with the same `MAPR_CONTAINER_USER`, `MAPR_CONTAINER_GROUP`, `MAPR_CONTAINER_GID`, and `MAPR_CONTAINER_UID` runtime environment variables.

Always use service or user tickets, not impersonation tickets. The ticket type and lifetime should consider the lifetime of the application being deployed. Use of impersonation tickets may allow rogue applications running in containers to impersonate arbitrary users (including `root` or `mapr`) and gain access to any data in the cluster.

The MapR ticket file location in the container is set with the `MAPR_TICKETFILE_LOCATION` environment variable, which is set at runtime for the user specified in `MAPR_CONTAINER_USER`. The ticket file must always be stored in `/tmp`. For example: `/tmp/mapr_ticket`.

In case of loss or breach, you can revoke tickets.

Non-Secure Clusters

On non-secure clusters, you can restrict access by running the application inside the container as a user with appropriate privileges on the MapR cluster. This is controlled using runtime environment variables.

Note: HPE recommends that you do not use either `mapr` or `root` users.

- **`MAPR_CONTAINER_USER`** and **`MAPR_CONTAINER_UID`** specify:
  - The default user invoked when starting the container
  - The user that the user application inside the container will run as
- **`MAPR_CONTAINER_GID`** represents the `GID` that the application inside the Docker container will run as
- **`MAPR_CONTAINER_GROUP`** represents the group that the application inside the Docker container will run as

Related Information

For more information related to security topics discussed in this section, see:

- Managing Secure Clusters on page 1482 —secure cluster details
- Managing Users and Groups on page 752 —MapR user roles
- Using the docker run Command —Docker container variable details
- Tickets
  - Managing Tickets on page 1424—using tickets
  - maprlogin on page 2130 —originating tickets
  - Generating a MapR User Ticket on page 1426 —generating tickets
  - How Tickets Work on page 1426 —revoking a user's existing valid tickets

Writing Applications to Use the MapR PACC

This section describes a number of resources for developing applications to use the MapR MapR Persistent Application Container Client.

Developing applications to use the MapR PACC is the same as developing applications on other MapR-supported platforms. To get started, see these topics:

last-updated: Apr 11, 2022

- MapR File System and Applications
- MapR Database and Applications
- MapR Event Store For Apache Kafka and Applications

To build a container for your application that can leverage MapR Data Platform services, you will create a Dockerfile referencing the MapR PACC which installs your application and its dependencies. You can then build and launch your application using the same runtime variables described later in this section.

Extending a MapR PACC
You can use a MapR PACC to create a new Docker image.

These pre-built Docker container base images – called MapR Persistent Application Client Containers (PACCs) – are available in the maprtech/pacc public repository:

Table

PACC Repository and Tag	Container OS			
	CentOS 6.8	CentOS 7.x	Ubuntu 14.04	Ubuntu 16.04
6.1.0	N/A	`maprtech/pacc:6.1.0_6.0.0_centos7` or `maprtech/pacc:latest` or `maprtech/pacc`	N/A	`maprtech/pacc:6.1.0_6.0.0_ubuntu16`
6.0.1	N/A	`maprtech/pacc:6.0.1_5.0.0_centos7`	N/A	`maprtech/pacc:6.0.1_5.0.0_ubuntu16`
6.0.0	N/A	`maprtech/pacc:6.0.0_4.0.0_centos7`	N/A	`maprtech/pacc:6.0.0_4.0.0_ubuntu16`
5.2.2	`maprtech/pacc:5.2.2_3.0.1_centos6`	`maprtech/pacc:5.2.2_3.0.1_centos7`	`maprtech/pacc:5.2.2_3.0.1_ubuntu14`	`maprtech/pacc:5.2.2_3.0.1_ubuntu16`
5.2.1	`maprtech/pacc:5.2.1_3.0_centos6`	`maprtech/pacc:5.2.1_3.0_centos7`	`maprtech/pacc:5.2.1_3.0_ubuntu14`	N/A
5.2.0	`maprtech/pacc:5.2.0_2.0_centos6`	`maprtech/pacc:5.2.0_2.0_centos7`	`maprtech/pacc:5.2.0_2.0_ubuntu14`	N/A

While you cannot modify a MapR-provided Docker image directly, you can build a custom image that is based on a MapR Persistent Application Client Container (PACC). The following example shows a custom Dockerfile that is used to create a new Docker image. In this example, an application has a JAR file that takes a producer as a parameter and runs a custom function.

The example has two parts. In Part 1, the custom Dockerfile uses the Docker `FROM` command to download a MapR PACC to a container on the user platform. A directory is created, and a JAR file is copied into the container so that it can be run in Java. The `CMD` command starts the application inside the container. In Part 2, the custom Dockerfile is built using the `docker build` command.

Part 1. Creating a Custom Dockerfile

```
FROM maprtech/pacc:5.2.0_2.0_centos6

# Copy jar to container
RUN mkdir -p /usr/share/mapr-apps/
COPY mapr-streams-examples-1.0-SNAPSHOT-jar-with-dependencies.jar /usr/
```


last-updated: Apr 11, 2022

```
share/mapr-apps/mapr-app-001.jar

# Run producer application in container
CMD [ "java", "-cp", "$MAPR_CLASSPATH:/usr/share/mapr-apps/
mapr-app-001.jar", "com.mapr.examples.Run", "producer" ]
```

Part 2. Building a Custom Docker Image From the Dockerfile

```
docker build -t <new docker image> .
  Note: Above needs to be run in the same directory as Dockerfile
  Make sure the image is created and no issue building Docker image.
docker images -a
```

Creating a MapR PACC Image Using `mapr-setup.sh`

This section describes how to download and run the `mapr-setup.sh` script to create a MapR Persistent Application Container Client (PACC) image.

To create a MapR PACC image using `mapr-setup.sh`:

1. Before using `mapr-setup.sh`, review these topics to understand important prerequisites and security considerations:
   - Before Deploying the MapR PACC on page 405
   - Security Considerations for the MapR PACC on page 406
2. Download the `mapr-setup.sh` script from mapr.com to a Linux or Mac OS X platform where Docker 1.12.5 or later is installed.

   Note: Running `mapr-setup.sh` on Windows is not supported.
3. Run the mapr-setup.sh script with the `docker client` command to create the Docker image:

   ```
   ./mapr-setup.sh docker client
   ```
4. Answer the command-line prompts to provide the information needed to configure the image. The following table describes each prompt. If you press **Enter** without specifying a value, `mapr-setup.sh` uses the default value shown in the square brackets ([]):

Parameter	Notes
`Build MapR client image? (y/n) [y]`	Press y to continue or n to exit the script.
`Image OS class (centos7, ubuntu16) [<local OS>]:`	Specify the base operating system on which to build the image. **Note:** SLES is not currently supported.
`Docker FROM base image name:tag [centos:centos7]:`	Specify the starting image used to create the new image. If necessary, you can enter your own tag and image name to choose a base image already created for your installation.
`MapR core version [6.x.x]:`	Specify the MapR core version that matches the version of the MapR cluster you want to access using the PACC. For the supported MapR core values, see Table 1 in Extending a MapR PACC on page 408. If you want to install the Hadoop Client with YARN, you must select `5.2.1` or later.

last-updated: Apr 11, 2022

`MapR MEP version [x.x.x]:`	Specify the EEP version that matches the EEP version of the MapR cluster you want to access using the PACC. Supported values are `2.0` or later. If you want to install the Hadoop Client with YARN, you must select EEP `3.0` or later. For more information about EEPs, see EEP Components and OS Support.
`Install Hadoop YARN client (y/n) [n]:`	Choose whether to install the Hadoop Client with YARN. Note that the Hadoop Client with YARN requires MapR core version 5.2.1 and EEP 3.0 or later. If you choose No, the script installs the POSIX (FUSE), MapR Database, and MapR Streams clients. The script does not install the Hadoop Client with YARN and does not ask if you want to install the Hive, Pig, and Streams clients.
`Add POSIX (FUSE) client to container? (y/n) [y]:`	Choose whether to install the POSIX (FUSE) client.
`Add HBase client to container? (y/n) [n]:`	Choose whether to install the HBase client.
`Add Hive client to container? (y/n) [n]:`	Choose whether to install the Hive client.
`Add Pig client to container? (y/n) [n]:`	Choose whether to install the Pig client.
`Add Spark client to container? (y/n) [n]:`	Choose whether to install the Spark client.
`Add MapR Streams clients to container? (y/n) [y]:`	Choose whether to install the MapR Streams clients.
`MapR client image tag name [<name>]:`	Accept the software-provided name for the image, or provide your own name. This is the name you will use to run the image to create the MapR PACC. The script automatically provides a name. For example: `maprtech/pacc:6.0.0_4.0.0_centos7_yarn_fuse_hbase_hive_pig_streams`
`Container network mode (bridge|host) [bridge]:`	Select the Docker network mode. For more information, see the Docker documentation.
`Container memory: specify host XX[kmg] or 0 for no limit [0]:`	Specify the maximum amount of memory (in kilobytes, megabytes, or gigabytes) that Docker allows the container to access. For example: • 2g • 4096m • 0 Accepting the default (0), means there is no restriction on memory, and the container can use as much memory as the platform makes available.

last-updated: Apr 11, 2022

5. Press **Enter** after the last prompt. The script creates the image and notifies you of successful creation. For example:

```
    Complete!

    ...Success

    Stopped service mapr-posix-client-container

    ...Success

     ---> 170362a5a82d
    Removing intermediate container 8f100b9d6d9b
    Step 7/8 : ENTRYPOINT /opt/mapr/installer/docker/mapr-setup.sh
container
     ---> Running in f98e5cde91ed
     ---> 7099a990a422
    Removing intermediate container f98e5cde91ed
    Step 8/8 : CMD start
     ---> Running in f6ae4139ab41
     ---> 01ca2ab6d0d3
    Removing intermediate container f6ae4139ab41
    Successfully built 01ca2ab6d0d3

    Edit '/root/thinclient/docker_images/client/mapr-docker-client.sh'
to set
       MAPR_CLUSTER and MAPR_CLDB_HOSTS and then execute it to start the
container
```

`mapr-setup.sh` creates the `mapr-docker-client.sh` sample-run file and displays the location of the file:

```
Edit '/root/thinclient/docker_images/client/mapr-docker-client.sh' to set
MAPR_CLUSTER and MAPR_CLDB_HOSTS and then execute it to start the
container
```

`mapr-docker-client.sh` contains environment variables for the image and makes it easy for you to start the container.

6. Edit the `mapr-docker-client.sh` script file. At a minimum, you must provide the `MAPR_CLUSTER` name and the `MAPR_CLDB_HOSTS` information. For example:

```
MAPR_CLUSTER=my.cluster.com
MAPR_CLDB_HOSTS=perfnode13[4-9].perf.lab
```

Note: To specify multiple entries, you can use a comma-separated list of CLDB hosts or an expression like the expressions described in "What expressions can I use to specify multiple nodes?" in the MapR Installer FAQ on page 5421.

You may wish to provide other values. You can:

- Start the FUSE client by specifying the `MAPR_MOUNT_PATH`.
- For a secure cluster, use a ticket by specifying a `MAPR_TICKETFILE_LOCATION`. For more information about security parameters, see Running the MapR PACC Using Docker on page 413.
- For secure and non-secure clusters, follow security best practices by specifying these parameters:
  - `MAPR_CONTAINER_USER`

last-updated: Apr 11, 2022

- MAPR_CONTAINER_GROUP
- MAPR_CONTAINER_UID
- MAPR_CONTAINER_GID

- Set environment variables, such as MAPR_CLASSPATH, MAPR_HOME, PATH, and others.
- Set the container time zone by specifying MAPR_TZ. The default is UTC.
- Add the POSIX mount by uncommenting the MAPR_MOUNT_PATH parameter and specifying a mount path value. The MAPR_MOUNT_PATH parameter is commented out for Mac builds but not for Linux builds. If you uncomment the parameter, you can ignore the following error message:

```
Started service mapr-posix-client-container [FAILED]
```

7. Run the mapr-docker-client.sh file to start the container:

```
./docker_images/client/mapr-docker-client.sh
```

The script uses the current user name to create a user for cluster access. This user is created so that you can run your application as a MapR client user:

```
Testing for cluster user account...

Enter MapR cluster user name: robertjones

User 'robertjones' does not exist. Creating new cluster user account...

Enter 'robertjones' uid: 502
 Enter 'robertjones' group name: users
 Enter 'robertjones' password: <password>
...Success

Configuring MapR client ( -c -C perfnode134.perf.lab -N
my.cluster.com)...

create /opt/mapr/conf/conf.old
Configuring Hadoop-2.7.0 at /opt/mapr/hadoop/hadoop-2.7.0
Done configuring Hadoop
CLDB node list: perfnode134.perf.lab:7222
Zookeeper node list:
Node setup configuration: hbinternal
Log can be found at: /opt/mapr/logs/configure.log

...Success
```

The successful completion of this step results in a user prompt that is inside the newly running container. Take care not to exit this prompt inadvertently, as doing so terminates the container.

8. Open a new session to the Docker host, and use the `docker ps` and `docker inspect` commands to inspect the container. Do not try to run the `docker ps` and `docker inspect` commands from the user prompt created in Step 7. You must open a new session to the Docker host to avoid terminating the container:

```
docker ps
docker inspect <container-run-ID>
```

Running the MapR PACC Using Docker

This section describes and provides examples for using the `docker run` command to run a pre-built MapR container image.

To run a pre-built MapR container image, you:

1. Select a PACC or an application built from the PACC.
2. Determine if your MapR cluster is secure by viewing the contents of the file `/opt/mapr/conf/mapr-clusters.conf`. For example, the following shows a non-secure cluster:

```
my.cluster.com secure=false ip-172-24-11-84
```

   If your cluster is secure, generate a MapR service ticket by following the instructions in Generating a Service Ticket on page 1428.

3. Use the `docker run` command to run the container. You can run the command from a Linux prompt, Windows command line, or a Mac terminal.
4. Verify that the container was created and is connected to the cluster.

Note: You run user-created MapR images from the `mapr-client.sh` script file. See Creating a MapR PACC Image Using mapr-setup.sh on page 409.

Using the `docker run` Command

Here is the general syntax for the `docker run` command:

```
docker run -it -e MAPR_CLUSTER=<cluster-name> -e
MAPR_TZ=<time-zone> -e MAPR_CLDB_HOSTS=<cldb-list> -e
MAPR_CONTAINER_USER=<user-name> -e MAPR_CONTAINER_PASSWORD=<password> -e
MAPR_CONTAINER_UID=<uid> -e MAPR_CONTAINER_GID=<gid> -e
MAPR_CONTAINER_GROUP=<group-name> -e MAPR_TICKETFILE_LOCATION=/tmp/
mapr_ticket -v <ticket-file-host-location>:/tmp/mapr_ticket:ro -e
MAPR_MOUNT_PATH=<path_to_fuse_mount_point> --cap-add SYS_ADMIN --cap-add
SYS_RESOURCE --device /dev/fuse --security-opt apparmor:unconfined
<image-name>
```

The following table describes the keys and variables used in the syntax:

Note: Pay special attention to the mandatory parameters. If you neglect to specify all mandatory parameters, the `docker run` command will fail.

Key	Variable	Mandatory/Optional	Description
`MAPR_CLUSTER`	`<cluster-name>`	Mandatory	The name of the MapR cluster to which the container will connect.

last-updated: Apr 11, 2022

Key	Variable	Mandatory/Optional	Description
MAPR_CLDB_HOSTS	<cldb-list>	Mandatory	CLDB host IP addresses separated by a comma. For example: `(hostname[:port_no] [,hostname[:port_no]...])`
MAPR_CONTAINER_USER	<user-name>	Mandatory	The user that the user application inside the Docker container will run as. This configuration is functionally equivalent to the Docker native `-u` or `--user`. Do not use Docker `-u` or `--user`, as the container needs to start as the `root` user to bring up FUSE before switching to the MAPR_CONTAINER_USER. The user specified here is the user that all storage operations on the MapR cluster will be performed as. Therefore, HPE recommends not using `root` or `mapr`. For secure clusters, this user must match the user in the MapR ticket passed via MAPR_TICKETFILE_LOCATION. This user also owns the `/opt/mapr` directory tree.
MAPR_CONTAINER_PASSWORD	<password>	Optional	The password of the user running inside the container. If not specified, it defaults to the <user-name>.
MAPR_TZ	<time-zone>	Optional	The time zone inside the container. For a list of time-zone settings, see this website. The default is UTC.
MAPR_CONTAINER_UID	<uid>	Optional	The UID that the application inside the Docker container will run as. This is a companion to the MAPR_CONTAINER_USER option. If a UID is not provided, the default is UID 1000. Providing a UID is strongly recommended. For secure clusters, this UID must match the UID specified in the MapR ticket file.
MAPR_CONTAINER_GID	<gid>	Optional	The GID that the application inside the Docker container will run as. This is a companion to the MAPR_CONTAINER_USER option. If a GID is not provided, the default is GID 1000. Providing a GID is strongly recommended. For secure clusters, this GID must match the GID specified in the MapR ticket file.
MAPR_CONTAINER_GROUP	<group-name>	Optional	The group that the application inside the Docker container will run as. This is a companion to the MAPR_CONTAINER_USER option. If a group name is not provided, the default is `users`. Providing a group name is strongly recommended.

last-updated: Apr 11, 2022

Key	Variable	Mandatory/Optional	Description
			For secure clusters, the group must match the group specified in the MapR ticket file.
`MAPR_TICKETFILE_LOCATION`	`/tmp/mapr_ticket`	Optional (required only for a secure cluster)	The location inside the container where the ticket file resides. For more information about MapR tickets, see Managing Tickets.
`MAPR_MOUNT_PATH`	`<path-to-fuse-mount-point>`	Optional (required only for FUSE client use)	The path to the FUSE mount point. If this parameter is not specified, the FUSE client is disabled.
`-v`	`<ticket-file-host-location>:/tmp/mapr_ticket:ro`	Optional (required only for a secure cluster)	The location of the ticket on the host where you are running the container, and the desired location of the ticket file in the Docker container. The `docker run` command maps the location on the host with the location inside the container. `ro` means read-only. `-v` refers to a volume mount. Make sure the owner and group on the host ticket file match the UID and GID specified in the MapR ticket file.
`--cap-add`	`SYS_ADMIN`	Optional (required only for FUSE use)	A parameter that is needed for the FUSE process to start inside the container, as `root` access to the FUSE device is required.
`--cap-add`	`SYS_RESOURCE`	Optional (required only for FUSE use)	A parameter that is required for the FUSE process to start.
`--device`	`/dev/fuse`	Optional (required only for FUSE use)	A parameter that is required to mount the FUSE device.
	`<image-name>`	Mandatory	The name of the container image to run. This is either the MapR Persistent Application Client Container (PACC) or a custom application container built from the PACC.
`--security-opt`	`apparmor:unconfined`	Optional (required only on Ubuntu hosts)	A parameter that is required for FUSE on Ubuntu hosts. For more information, see Docker-16429.

Example `docker run` Commands

Here are four examples for using the `docker run` command:

- Secure Cluster with FUSE-Based POSIX Client
- Secure Cluster without FUSE-Based POSIX Client
- Non-Secure Cluster with FUSE-Based POSIX Client
- Non-Secure Cluster without FUSE-Based POSIX Client

The following command generates a service ticket on the cluster or a client that is valid for 30 days. (For more `maprlogin` command examples, see maprlogin Command Examples).

```
maprlogin generateticket -type service -cluster cluster1 -duration
30:0:0 -out /tmp/bobs_ticket -user bob
```


last-updated: Apr 11, 2022

The ticket can be copied from `/tmp/bobs_ticket` to `/user/tickets/bobs_ticket` on the container host and used in the following `docker run` commands for secure clusters:

Secure Cluster with FUSE-Based POSIX Client

```
docker run -it -e MAPR_CLUSTER=cluster1 -e MAPR_CLDB_HOSTS=CLDB_1,CLDB_2 -e
MAPR_CONTAINER_USER=bob -e MAPR_TICKETFILE_LOCATION=/tmp/mapr_ticket -v  /
user/tickets/bobs_ticket:/tmp/mapr_ticket:ro -e MAPR_MOUNT_PATH=/
mapr --cap-add SYS_ADMIN --cap-add SYS_RESOURCE --device /dev/fuse maprtech/
pacc:5.2.1_3.0_centos7
```

Secure Cluster without FUSE-Based POSIX Client

```
docker run -it -e MAPR_CLUSTER=cluster1 -e MAPR_CLDB_HOSTS=CLDB_1,CLDB_2 -e
MAPR_CONTAINER_USER=bob -e MAPR_TICKETFILE_LOCATION=/tmp/mapr_ticket -v  /
user/tickets/bobs_ticket:/tmp/mapr_ticket:ro maprtech/pacc:5.2.1_3.0_centos7
```

Non-Secure Cluster with FUSE-Based POSIX Client

In a non-secure cluster, specifying the MAPR_CONTAINER_USER, MAPR_CONTAINER_GROUP, MAPR_CONTAINER_UID, and MAPR_CONTAINER_GID is strongly recommended, and these values must match the user credentials on the server:

```
docker run -it --cap-add SYS_ADMIN --cap-add SYS_RESOURCE --device /dev/
fuse -e MAPR_CLUSTER=cluster1 -e MAPR_CLDB_HOSTS=CLDB_1,CLDB_2 -e
MAPR_CONTAINER_USER=bob -e MAPR_CONTAINER_GROUP=dev -e
MAPR_CONTAINER_UID=10000 -e MAPR_CONTAINER_GID=10000 -e MAPR_MOUNT_PATH=/
mapr maprtech/pacc:5.2.1_3.0_centos7
```

Non-Secure Cluster without FUSE-Based POSIX Client

In a non-secure cluster, specifying the MAPR_CONTAINER_USER, MAPR_CONTAINER_GROUP, MAPR_CONTAINER_UID, and MAPR_CONTAINER_GID is strongly recommended, and these values must match the user credentials on the server:

```
docker run -it -e MAPR_CLUSTER=cluster1 -e MAPR_CLDB_HOSTS=CLDB_1,CLDB_2 -e
MAPR_CONTAINER_USER=bob -e MAPR_CONTAINER_GROUP=dev -e
MAPR_CONTAINER_UID=10000 -e MAPR_CONTAINER_GID=10000 maprtech/
pacc:5.2.1_3.0_centos7
```

Tip:

To re-launch a container, you can use these Docker commands:

```
# docker ps -a
# docker start <container-run-ID>
```

Use `docker start -i` if you need to start with an interactive shell.

Verifying the Launch of the MapR PACC

After running the `docker run` command, you should see the `Starting services` message. For example:

```
Starting services (mapr-posix-client-container)...
Started service mapr-posix-client-container
...Success
$
```

When the installation is successful, the client connects to the cluster, storage is mounted, and the FUSE POSIX client is started automatically. Use the `ls $MAPR_MOUNT_PATH` command to test the connection to the cluster. This command should return the cluster name. For example:

```
$ ls $MAPR_MOUNT_PATH
cluster1
```

To display some directories on the cluster, use this command:

```
$ ls $MAPR_MOUNT_PATH/cluster1
apps var user hbase opt tmp
```

MapR PACC Sample Application
These examples demonstrate how to deploy and run a MapR application into a container.

For an example of deploying and running a MapR application into a container, see:

Getting Started with a Client Container (blog)

MapR PACC Known Issues
This topic describes some known issues that you should be aware of while troubleshooting.

Issue	Description
DOC-148	On Mac OS X, using `mapr-setup.sh` to create a Docker image can generate the following error when the ping to a hostname fails: `ERROR: Hostname <hostname> cannot be resolved. Correct the problem and retry mapr-setup.sh` **Workaround:** To enable remote login on the Mac, select the **Remote Login** option in **System Preferences > Sharing**. Then retry the `mapr-setup.sh` command.

INFO-47	When running PACC or Zeppelin Docker images, starting or restarting the FUSE client incorrectly reports FAILED. Docker generates an error like the following:
	`mapr-posix-client-container [FAILED]`
	If you can access the MapR filesystem from your client, then ignore the error. You can also confirm a successful FUSE client start by checking `/opt/mapr/logs/posix-client-container.log`. The following shows a successful start:
	`Mon Oct 16 10:49:56 PDT 2017: Mounting posix-client-container / mapr --log_path /opt/mapr/logs --client_lib_path /tmp -o allow_other -o big_writes -o auto_unmount -o async_dio -o max_background=64 -o auto_inval_data at /mapr ...`
	`Starting fuse with 1 libraries`
	`Mon Oct 16 10:49:56 PDT 2017: Result:0`
	`Mon Oct 16 10:50:06 PDT 2017: Running /etc/init.d/ mapr-posix-client-container status`
	`Mon Oct 16 10:50:06 PDT 2017: posix-client-container is mounted at /mapr.`
	The error is due to incorrect handling of a MapR script's return exit code.

Running Hadoop Commands on a Mac and Windows Client

The location from which you run Hadoop commands depends on your machine.

When you run hadoop commands on the Mac and Windows client, use the Hadoop 2 version to run MapReduce version 2 applications.

last-updated: Apr 11, 2022

To run the..	Run the command from this location:
Hadoop 2 version of the command	On Windows: `%MAPR_HOME%\hadoop\hadoop-2.x.x\bin` On Mac: `/opt/mapr/hadoop/hadoop-2.x.x/bin`

On Linux installations, the installer creates symlinks to the hadoop directories by default. On Mac, you can create the symlinks. Once the symlinks are created, you can specify the version of the hadoop command as mentioned in the Hadoop command documentation.

Important: Notes Specific to Windows

- The user that runs hadoop commands from the hadoop 2 directory cannot have a space or a hyphen (-) in the username.
- The native Hadoop library is not present on Windows. Therefore, the `hadoop fs -getmerge` command is not available.

Create Symlinks to Hadoop Directories for the Mac Client
Run Hadoop commands using the hadoop2 keywords.

Perform the following steps to create hadoop2 symlinks in the `usr/local/bin` directory for a MapR client on Mac OS X:

1. Run the following commands as root to create the symlinks:

```
ln -s /opt/mapr/hadoop/hadoop-2.x.x/bin/hadoop /usr/local/bin/hadoop2
ln -s /opt/mapr/bin/hadoop /usr/local/bin/hadoop
```

Note: In the command above, replace hadoop-2.x.x with the actual hadoop 2 version number that you have installed.

2. Add the Hadoop binaries to the PATH environment variable. For example, add the following text to the user login shell script such as ~/.bashrc:

```
export PATH=/opt/mapr/bin:/opt/mapr/hadoop/hadoop-2.x.x/bin:/opt/mapr/
hadoop/hadoop-0.20.2/bin:${PATH}
```

Note: In the text above, replace hadoop-2.x.x with the actual hadoop 2 version number that you have installed.

Now, you can run hadoop commands using the `hadoop1` and `hadoop2` keywords.

Installing the MAST Gateway

Describes how to install the MapR Automated Storage Tiering (MAST) Gateway service.

The MAST Gateway acts as the centralized entry point for all the operations that need to be performed on the tiered storage. The MAST Gateway can be installed (with or without MapR File System) on specific hosts on the cluster with access to the 3rd party cloud storage (for cold tier operations) or on the edge node. Before you install the MAST Gateway, review the Pre-Installation Considerations on page 419.

See Configuring the MAST Gateway Service on page 1259 after installing the MAST Gateway.

Installing the MAST Gateway Using the MapR Installer

When you install release 6.1 or later using the MapR Installer on page 5395, select the **MapR-XD: Cloud Scale Data Platform auto-provisioning template** to install the MAST Gateway automatically on all the nodes.

Installing the MAST Gateway from the Command-line

- Run the following command on the node where you want to install the MAST Gateway:

CentOS	`yum install mapr-mastgateway`
Ubuntu	`apt-get install mapr-mastgateway`
SLES	`zypper install mapr-mastgateway`

Installing Additional MAST Gateways from the Command-line

If you install a new MAST Gateway on a cluster already performing tiering operations (using other installed MAST Gateways), perform the following steps to force CLDB to rebalance utilization of all the MAST Gateways including the newly added MAST Gateway:

1. Install the `mapr-mastgateway` package on the node.
2. Run `configure.sh` on page 2053 with the `-R` option to register the MAST Gateway with the CLDB.
   For example:

   ```
   /opt/mapr/server/configure.sh -R
   ```

   After this command runs, newly created volumes are assigned to this MAST Gateway.
3. Run the following command to force CLDB to reassign existing volumes to the least utilized MAST Gateways:

   ```
   /opt/mapr/server/mrconfig mastgateway refreshvolassignment <volume-name>
   ```

   You must run this command once for each volume to reassign. HPE recommends running this command for all volumes if MAST Gateway is either newly added to the cluster or permanently removed from the cluster. When this command runs, CLDB reassigns the volume tiering operation to the least utilized MAST Gateway, which might be the newly added MAST Gateway, to force rebalancing.

   For more information, see `mastgateway refreshvolassignment` on page 2167.

Pre-Installation Considerations

Lists the recommendations that you must follow before installing MAST Gateways.

By default, the MAST Gateway uses 16 threads for volume and file offload and recall operations and another 16 threads for handling internal operations and other operations such as reads (which triggers automatic recall requests), writes, etc. Each thread processes offload or recall of a container (associated with a volume). Each MAST Gateway can process one or more volumes (and associated containers) simultaneously depending on the number of threads available for processing the containers associated with the volumes. Each volume is assigned to a MAST Gateway for a tiering operation irrespective of the number of containers associated with the volume.

For example, suppose you have a volume with 5 containers. The MAST Gateway allocates 5 threads, one per container, to process the offload of that volume's data; the other 11 threads are available for other tiering-related operations on other tiering-enabled volumes. However, if you have a volume with 20 containers. The MAST Gateway allocates all 16 threads to process the offload of that volume's data and as threads are freed, other unprocessed containers associated with the volume are processed. Now, suppose

last-updated: Apr 11, 2022

that you have configured multiple MAST Gateways for the volume that has 20 containers. . Volume offload is then distributed among the multiple MAST gateways, leading to enhanced performance of the cluster. If you have multiple large volumes with multiple containers, MapR recommends more than one MAST Gateway to process all the containers associated with all the volumes.

If you have a limited number of nodes that can access the cold tier (because of controlled access to WAN, proxy setup, etc.), install and run MAST Gateway on only those nodes and set up proxy server parameters in the `mastgateway.conf` file. See step 5 in Configuring the MAST Gateway Service on page 1259 for more information on the configuration parameters to set for using a proxy server. On the other hand, if all the cluster nodes can access the tier, then consider the following before deploying the MAST Gateway:

1. A single MAST Gateway can offload at around 300 MB/sec at full throttle. So, compute the minimum number of MAST Gateways based on network capacity of the connection to the tier.
2. If you expect many volume offloads and recall operations to get triggered at the same time, consider installing MAST Gateways on a few more nodes or adding more MAST Gateways at a later time. See Installing the MAST Gateway on page 418 for information.

In general, you must allocate at least 2GB of memory for the MAST Gateway operations. The memory consumption can increase during heavy load. See settings for configuring memory for MAST Gateway in Step 7 for Configuring the MAST Gateway Service on page 1259.

Note: Before installing MAST Gateways, you must ensure that the system time on all the cluster nodes is the same. If the system time on CLDB and file server nodes are different, the mtime rule for migrating data might not work as intended. If you see a time skew alarm in the cluster, resolve the alarm immediately to prevent catastrophic failures.

Supported Clients

To manually perform tiering-related operations on a volume, you can use the following:

- The maprcli command
- The REST API

To manually perform tiering-related operations on a file, you can use the following:

- The FUSE-based POSIX client client
- The loopbacknfs POSIX client client
- The NFS client
- The hadoop command
- The maprcli command
- The REST API

You must use clients from MapR v6.1 for accessing tiered volumes and performing tiering operations. You cannot use mixed mode clients to access and run tiering jobs on tiered volumes.

Enabling Soft Mount and Setting the Timeout

Describes how to enable soft mount, and set the RPC timeout for MapR components.

By default, all MapR File System, MapR Database, and MapR-Streams operations never timeout as they wait (hard mount behavior) for the operation to succeed and/or the server to respond. You can configure a soft mount behavior by setting the values for the following parameters in the `core-site.xml` or `hbase-site.xml` file:

`fs.mapr.hardmount`	Specifies whether or not to enable hard mount. Value can be: • `true` - enable hard mount • `false` - disable hard mount The default value is `true`.
`fs.mapr.rpc.timeout`	This parameter is valid for MapR 6.0.0 and earlier. Specifies the RPC timeout value in seconds. The default value is 300 seconds. The value cannot be less than 30 seconds. If the value is greater than 300 seconds, TCP keepalive probes are sent to prevent the TCP socket from timing out. If value is below 300 seconds, the RPCs will timeout after the specified time.
`streams.rpc.timeout.ms`	This parameter is new as of MapR 6.0.1. Specifies the RPC timeout value in milliseconds. The default value is 300000 milliseconds. The value cannot be lower than 30000 milliseconds. If the value is greater than 300000 milliseconds, TCP keepalive probes are sent to prevent the TCP socket from timing out. If value is below 30000 milliseconds, the RPCs will timeout after the specified time.

These parameter settings affect all clients.

Note: For MapR-Streams, these parameters can be set as configuration properties when constructing the Consumer or Producer Java object. For more information, see MapR-Streams.

Enabling Soft Mount

1. Open the `core-site.xml` or `hbase-site.xml` file and add the parameter as follows:

```
<property>
  <name>fs.mapr.hardmount</name>
  <value>false</value>
  <description>enabling soft mount by setting value to false</
description>
</property>
```

2. Save and close the file.

last-updated: Apr 11, 2022

Setting RPC Timeout

1. Open the `core-site.xml` or `hbase-site.xml` file and add the parameter as follows:

   As of MapR 6.0.1:

   ```
   <property>
     <name>streams.rpc.timeout.ms</name>
     <value>300000</value>
     <description>RPC timeout value</description>
   </property>
   ```

   For MapR 6.0.0 and earlier:

   ```
   <property>
     <name>fs.mapr.rpc.timeout</name>
     <value>30</value>
     <description>RPC timeout value</description>
   </property>
   ```

2. Save and close the file.

Troubleshooting

Describes changes to `core-site.xml` file to troubleshoot issues.

fs.mapr.bind.retries Parameter

If there are issues related to unavailability of port, set the value for `fs.mapr.bind.retries` configuration parameter in `core-site.xml` file to `true`. If `true`, the client tries to bind during client initialization for 5 minutes before failing. By default, the `fs.mapr.bind.retries` configuration parameter is set to `false`.

For example, your entry in `core-site.xml` file should look similar to the following:

```
<property>
  <name>fs.mapr.bind.retries</name>
  <value>true</value>
  <description>Bind during client initialization for 5 minutes</description>
</property>
```

fs.mapr.bailout.on.library.mismatch Parameter

When running any application with older versions of the MapR JARs, the system could hang if the older JARs link to the native library installed on cluster nodes that have been updated to a newer MapR version. The `fs.mapr.bailout.on.library.mismatch` parameter detects mismatched libraries, fails the job, and logs an error message. The parameter is enabled by default. You can disable the parameter on all the YARN nodes and resubmit the job for the job to continue to run. To disable the parameter, you must set it to `false` in the `core-site.xml` file.

For example, to disable, your entry in the `core-site.xml` file should look similar to the following:

```
<property>
  <name>fs.mapr.bailout.on.library.mismatch</name>
  <value>false</value>
  <description>Disabling to continue running jobs</description>
</property>
```

libMapRClient.so Binary

The `libMapRClient.so` binary is in `/opt/mapr/lib` directory and also bundled in `maprfs-XXX.jar` file. All the applications that include the JAR also have `libMapRClient.so` binary. If there are multiple `libMapRClient.so` on a machine and if you know the location of all the JARs, you can run the following commands to check the mapr version of a binary:

```
# jar tvf mapr-<XXX>.jar | grep libMapRClient.so
# jar xvf mapr-<XXX>.jar com/mapr/fs/native/Linux/x86_64/libMapRClient.so
# cd com/mapr/fs/native/Linux/x86_64/
# strings libMapRClient.so | grep mapr-version
# cd /opt/mapr/lib
# strings libMapRClient.so | grep mapr-version
```

This is useful in determining if there are old binaries installed on the system.

Setting Up the Control System

Describes how to configure and access the Control System.

The Control System allows you to manage the cluster (including nodes, volumes, users, and alarms) through a comprehensive graphical user interface with all the functionality of the command line or REST APIs.

Installing the Web Server and API Server

In prior releases, the `mapr-webserver` package contained both the Control System UI static files and the server running the Java application. Starting from v6.0, the UI static files are in `mapr-webserver`. The `mapr-apiserver` runs the server that sends the queries. The apiserver allows you to perform cluster administration programmatically.

When you install `mapr-webserver`, the `mapr-apiserver` is automatically installed because of the dependency on the `mapr-apiserver` to perform the queries. If `mapr-webserver` is installed, you can use the graphical user interface to manage your cluster. You can also install the `mapr-apiserver` independently to run APIs or web clients that query or programmatically access MapR File System, MapR Database, and other components; however, without the webserver, the Control System will not be available on this node to perform administrative tasks using the UI.

To install the webserver and/or apiserver, see Installing MapR and MapR Ecosystem Components on page 128.

- If you install using the MapR Installer, by default, the installer selects one instance of the `mapr-webserver` and `mapr-apiserver` to install. You can specify additional webserver and/or apiserver instances to install in the *Configure Service Layout* page.
- If you install manually, run the appropriate command on the node to install the `mapr-webserver` and/or `mapr-apiserver` packages. For more information on the command to run, see Step 3: Install Cluster Service Packages on page 150. After you install the packages, run the following commands:
  - `/opt/mapr/server/configure.sh -R`
  - `maprcli node services -nodes <nodes> -name apiserver -action start`

For the purposes of high availability, the recommendation is to run at least 2 instances of the webserver and 2 instances of the apiserver.

Configuring Metrics and Logging to Enable Metrics Visualization

During installation using the MapR Installer, you can configure metrics and logging using settings on the **Monitoring** page of the MapR Installer user interface. The metrics collection infrastructure must be installed because the Control System relies on these metrics to provide graphs and charts. If the metrics collection infrastructure is not installed, you cannot visualize the metrics in the panes on the Control System. If you did not install metrics collection or logging during your initial installation, you can add it later by selecting the feature during an Incremental Install.

Configuring SameSite Cookie Support

The SameSite attribute of the Set-Cookie HTTP response header allows you to declare if your cookie should be restricted to a first-party or same-site context.

Edit the following section in the `/opt/mapr/apiserver/conf/web.xml` file to set the SameSite cookie:

```
<session-config>
   <cookie-config>
     <http-only>true</http-only>
     <max-age>86400</max-age>
     <name>MAPR.APISERVER.JSESSIONID</name>
     <secure>true</secure>
     <comment>__SAME_SITE_LAX__</comment>
   </cookie-config>
   <session-timeout>30</session-timeout>
 </session-config>
```

Set it to one of the following values:

_SAME_SITE_STRICT_	Cookies will only be sent in a first-party context and not be sent along with requests initiated by third party websites.
_SAME_SITE_LAX_	Cookies are allowed to be sent with top-level navigations and will be sent along with GET request initiated by the third party website. This is the default value.
_SAME_SITE_NONE_	Cookies will be sent in all contexts, that is, sending cross-origin is allowed.

For more information, see SameSite cookies.

Browser Compatibility

The Control System is web-based, and works with the following browsers:

- Chrome 58 and later
- Safari 11.x for v6.0.1

  Safari 10.x for v6.0

  Note: Safari Private Window is not supported.
- Firefox 53 and later
- Microsoft Edge 15, 16, and 17

Note: If you encounter the following error on Firefox 79 and above:

```
Secure Connection Failed
Error code: SEC_ERROR_REUSED_ISSUER_AND_SERIAL
```

then delete the Control System certificates as described to resolve this error.

Launching the Control System

To use the Control System, navigate to the host that is running the WebServer in the cluster. Control System access to the cluster is typically using HTTP on port 8080 or using HTTPS on port 8443. You should disable pop-up blockers in your browser to allow MapR to open help links in new browser tabs.

The first time you open the Control System using HTTPS from a new browser, the browser alerts you that the security certificate is unrecognized. This is normal behavior for a new connection. Add an exception in your browser to allow the connection to continue.

Configuring the Control System

Describes the configuration of Control System properties.

The Control System properties can be configured in the `/opt/mapr/apiserver/conf/properties.cfg` file. For example:

```
ojai.cache.size=64
mapr.webui.https.port=8443
doc.url=https://docs.datafabric.hpe.com/
proxy.zkservices=elasticsearch,opentsdb
activity.metrics.thread.pool.size=10
```

The properties are as follows:

activity.metrics.thread.pool.size

Default Value: 10

Description: Denotes the number of threads used to query table metrics.

authentication.pam.service

Default Value: `mapr-admin`

Description: The file to use for PAM authentication.

doc.url

Default Value: https://docs.datafabric.hpe.com/

Description: The URL to the MapR documentation.

log.sensitive.keys

Default Value: Not Applicable

Description: The properties to exclude from the logs. For example, to hide SMTP or LDAP passwords in the logs, specify the properties for the passwords as follows:

```
log.sensitive.keys=<mapr.smtp.sender.password>;<mapr.ldap.binddnpasswd>
```

If you do not specify this property, passwords are not hidden in the logs.

mapr.rest.auth.methods

Default Value: `basic`

Description: The authentication methods to use for MapR REST calls. Add `kerberos` to this setting to enable SPNEGO.

mapr.webui.http.port

Default Value: 8081

last-updated: Apr 11, 2022

	Description: The port to use to connect to the Control System.
mapr.webui.https.port	*Default Value:* 8443 *Description:* The port to use to securely connect to the Control System.
ojai.cache.size	*Default Value:* 64 *Description:* The size of the cache in MB that the OJAI controller uses.
proxy.zkservices	*Default Value:* `elasticsearch,opentsdb` *Description:* The proxy to use for ZooKeeper services
requestHeaderSize	*Default Value:* 8KB *Description:* The size of the request header.
ssl.exclude-ciphers	*Default Value:* `TLS_DHE.,TLS_EDH` *Description:* The encryption ciphers that should *not* be used for communication.
ssl.exclude-protocols	*Default Value:* `SSLv3,TLSv1,TLSv1.1` *Description:* The security protocols that should *not* be used for communication.
ssl.keystore	*Default Value:* `/opt/mapr/conf/ssl_keystore` *Description:* The path to the SSL keystore.
ssl.truststore	*Default Value:* `/opt/mapr/conf/ssl_truststore` *Description:* The path to the SSL truststore.

Note: You must restart the apiserver for the changes to take effect. For example, run the following command to restart the apiserver:

```
maprcli node services -action restart -name apiserver -nodes `hostname`
```

Configuring Authentication

Lists the authentication methods supported by the Control System and the apiserver.

Both the Control System and apiserver (that processes REST API calls) require one of the following method of authentication:

- Basic authentication (with a username and password) on secure and non-secure clusters. Refer to documentation for information on setting up username, password, and permissions for accessing the Control System and REST API calls.
- Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO) on secure clusters to authenticate REST calls to the Control System and access the resources directly. Refer to documentation for information on configuring SPNEGO for the Control System and REST API calls.
- Pluggable Authentication Modules (PAM) for password verification. Refer to documentation for information on configuring PAM for the Control System and REST API calls.

For information on using the Control System, `maprcli` commands, and the REST APIs, refer to topics under 6.1 Administration on page 752.

Configuring Impersonation

Lists the process to permit the `mapr` user to impersonate other users.

last-updated: Apr 11, 2022

Impersonation, also known as identity assertion, is one user (the `mapr` super user) accessing data and submitting jobs on behalf of another user.

Note: Only the `mapr` user can impersonate other users.

For secure clusters, to have a request processed as an impersonated user:

1. The user submitting the request must be the `mapr` user and the request should have the HTTP header X-MAPR-IMPERSONATED-USER, passed in the request.

   The value of the header is the username of the impersonated user.

2. The header must also include `"Authorization: Basic <base64_encoding_of_userID:pwd>"` for the apiserver to authorize the request.

   Here userID is `mapr` and the password is the PAM Linux password for `mapr` user on the node on which the apiserver is running.

For example:

```
curl -XPOST -H "Accept: application/json" -H  "X-MAPR-IMPERSONATED-USER:
m7user1" -H "Authorization: Basic bWFwcjptYXBy" -k https://10.20.30.40:8443/
rest/table/create?path=%2Ftmp%2FsrcC -v
```

For a non-secure cluster, MapR requires a file for the user to impersonate in the `/opt/mapr/conf/proxy` directory. The logged-in user is allowed to impersonate only if the `/opt/mapr/conf/proxy/<user_to_impersonate>` file is present. By default, this file is created during installation for the `mapr` user and the `root` user. If the file is not present, HTTP 403 is returned to the client if the client attempts to impersonate a user who does not have the file.

Migrating to MapR

Provides instructions for migrating business-critical data and applications from an Apache Hadoop cluster to a MapR cluster.

This guide provides instructions for migrating business-critical data and applications from an Apache Hadoop cluster to a MapR cluster.

The MapR distribution is 100% API-compatible with Apache Hadoop, and migration is a relatively straightforward process. The additional features available in MapR provide new ways to interact with your data. In particular, MapR provides a fully read/write storage layer that can be mounted as a filesystem via NFS, allowing existing processes, legacy workflows, and desktop applications full access to the entire cluster.

Migration consists of planning, deployment, and migration of components, applications, data, and nodes.

See the https://docs.datafabric.hpe.com/home/ReleaseNotes/c_relnotes_intro.html for up-to-date information about migration issues.

Planning and Initial Deployment

There are a number of considerations to take into account before migrating from Apache Hadoop to MapR Hadoop.

The first phase of migration is planning. In this phase you will identify the requirements and goals of the migration, identify potential issues in the migration, and define a strategy.

The requirements and goals of the migration depend on a number of factors:

- Data migration: can you move your datasets individually, or must the data be moved all at once?

last-updated: Apr 11, 2022

- Downtime: can you tolerate downtime, or is it important to complete the migration with no interruption in service?
- Customization: what custom patches or applications are running on the cluster?
- Storage: is there enough space to store the data during the migration?

The MapR Hadoop distribution is 100% plug-and-play compatible with Apache Hadoop, so you do not need to make changes to your applications to run them on a MapR cluster. MapR Hadoop automatically configures compression and memory settings, task heap sizes, and local volumes for shuffle data.

Initial Deployment

The initial MapR deployment phase consists of installing, configuring, and testing the MapR cluster and any ecosystem components (such as Hive or Pig) on an initial set of nodes. Once you have the MapR cluster deployed, you will be able to begin migrating data and applications.

To deploy the MapR cluster on the selected nodes, see the Installing MapR and MapR Ecosystem Components on page 128

Component Migration

This section describes how to migrate customized components to MapR Hadoop.

MapR Hadoop features the complete Hadoop distribution including components such as Hive. There are a few things to know about migrating Hive, or about migrating custom components you have patched yourself.

Custom Components

If you have applied your own patches to a component and wish to continue to use that customized component with the MapR distribution, you should keep the following considerations in mind:

- **MapR libraries**: All Hadoop components must point to MapR for the Hadoop libraries. Change any absolute paths. Do not hardcode `hdfs://` or `maprfs://` into your applications. This is also true of Hadoop ecosystem components that are not included in the MapR Hadoop distribution (such as Cascading). For more information see Working with MapR File System.
- **Component compatibility**: Before you commit to the migration of a customized component (for example, customized HBase), check the MapR release notes to see if HPE has issued a patch that satisfies your business requirements. HPE publishes a list of Hadoop common patches and MapR patches with each release and makes those patches available for HPE customers to take, build, and deploy.
- **ZooKeeper coordination service**: Certain components depend on ZooKeeper. When you migrate your customized component from the HDFS cluster to the MapR cluster, make sure it points correctly to the MapR ZooKeeper service.

Hive Migration

You can continue to use Hive tables in a MapR cluster.

Hive facilitates the analysis of large datasets stored in the Hadoop filesystem by organizing that data into tables that can be queried and analyzed using a dialect of SQL called HiveQL. The schemas that define these tables and all other Hive metadata are stored in a centralized repository called the *metastore*.

If you would like to continue using Hive tables developed on an HDFS cluster in a MapR cluster, you can import Hive metadata from the metastore to recreate those tables in MapR. Depending on your needs, you can choose to import a subset of table schemas or the entire metastore in one go:

last-updated: Apr 11, 2022

- **Importing table schemas into a MapR cluster**: Use this procedure to import a subset of the Hive metastore from an HDFS cluster to a MapR cluster. This method is preferred when you want to test a subset of applications using a smaller subset of data.
- **Importing an entire Hive metastore into a MapR cluster**: Use the following procedure to import an entire Hive metastore from an HDFS cluster to a MapR cluster. This method is preferred when you want to test all applications using a complete dataset. You will need to redirect all of links that formerly pointed to the HDFS (`hdfs://<namenode>:<port number>/<path>`) to point to MapR File System (`maprfs:///<path>`).

  MySQL is a very popular choice for the Hive metastore and is used in the following example. If you are using another RDBMS, consult the relevant documentation.

1. Ensure that both Hive and your database are installed on one of the nodes in the MapR cluster. For step-by-step instructions on setting up a standalone MySQL metastore, see Using MySQL for the Hive Metastore.
2. On the HDFS cluster, back up the metastore to a file.

   ```
   mysqldump [options] \--databases db_name... > filename
   ```

3. Ensure that queries in the dumpfile point to the MapR File System rather than HDFS. Search the dumpfile and edit all of the URIs that point to `hdfs://` so that they point to `maprfs:///` instead.
4. Import the data from the dumpfile into the metastore running on the node in the MapR cluster:

   ```
   mysql [options] db_name < filename
   ```

Using Hive with MapR volumes

MapR File System does not allow moving or renaming across volume boundaries. Be sure to set the Hive Scratch Directory and Hive Warehouse Directory in the same volume where the data for the Hive job resides before running the job. For more information, see How Hive Handles Scratch Directories on MapR in Hive Directories.

HBase Migration

The MapR Hadoop distribution includes HBase, with a number of MapR-exclusive enhancements.

HBase is the Hadoop database, which provides random, real-time read/write access to very large datasets. The MapR Hadoop distribution includes HBase and is fully integrated with MapR enhancements for speed, usability, and dependability. MapR provides a volume (normally mounted at `/hbase`) to store HBase data.

- **HBase bulk load jobs**: If you are currently using HBase bulk load jobs to import data into the HDFS, make sure to load your data into a path under the `/hbase` volume.
- **Compression**: The HBase write-ahead log (WAL) writes many tiny records, and compressing it would cause massive CPU load. Before using HBase, turn off MapR compression for directories in the HBase volume.

Migrating between Apache HBase and MapR Database Binary Tables

You can use the CopyTable tool to migrate data from an Apache HBase table to a MapR Database binary table.

MapR Database tables can be parsed by the Apache CopyTable tool (`org.apache.hadoop.hbase.mapreduce.CopyTable`).

Before You Start

Before migrating your tables to another platform, consider the following points:

last-updated: Apr 11, 2022

- **Schema Changes**. Apache HBase and MapR Database binary tables have different limits on the number of column families. When you are migrating to MapR Database binary tables, you may be interested in changing your table's schema to take advantage of the increased availability of column families.
- **API Mappings**: When you are migrating from Apache HBase to MapR Database tables, examine your current HBase applications to verify the APIs and HBase Shell commands used are fully supported.
- **Namespace Mapping**: If the migration will take place over a period of time, be sure to plan your table namespace mappings in advance to ease the transition. See Mapping to HBase Table Namespaces on page 430 for more information.
- **Implementation Limitations**: MapR Database binary tables do not support HBase coprocessors. If your existing Apache HBase installation uses coprocessors, plan any necessary modifications in advance. MapR Database binary tables support a subset of the regular expressions supported in Apache HBase. Check your existing workflow and HBase applications to verify you are not using unsupported regular expressions.

When migrating to MapR Database binary tables, change your Apache HBase client to the MapR client by installing the version of the `mapr-hbase` package that matches the version of Apache HBase on your source cluster.

See Installing MapR Software for information about MapR installation procedures, including setting up the proper repositories.

Mapping to HBase Table Namespaces

This section describes mapping table namespaces between Apache HBase tables and MapR Database binary tables.

The MapR implementations of the HBase Java API and libhbase differentiate between Apache HBase tables and MapR Database tables according to table names. In certain cases, such as migrating code from Apache HBase tables to MapR Database tables, users need to force the API they are using to access a MapR Database table, even though the table name could map to an Apache HBase table. The `hbase.table.namespace.mappings` property allows you to map Apache HBase table names to MapR Database tables. This property is typically set in the configuration file `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/core-site.xml`.

In general, if a table name includes a slash (`/`), the name is assumed to be a path to a MapR Database table, because slash is not a valid character for Apache HBase table names. In the case of "flat" table names without a slash, namespace conflict is possible, and you might need to use table mappings.

Table Mapping Naming Conventions

A table mapping takes the form *`name`*`:`*`map`*, where *`name`* is the table name to redirect and *`map`* is the modification made to the name. The value in *`name`* can be a literal string or contain the `*` wildcard. When mapping a name with a wild card, the mapping is treated as a directory. Requests to tables with names that match the wild card are sent to the directory in the mapping.

When mapping a name that is a literal string, you can choose from two different behaviors:

- End the mapping with a slash to indicate that this mapping is to a directory. For example, the mapping `mytable1:/user/aaa/` sends requests for table `mytable1` to the full path `/user/aaa/mytable1`.
- End the mapping without a slash, which creates an alias and treats the mapping as a full path. For example, the mapping `mytable1:/user/aaa` sends requests for table `mytable1` to the full path `/user/aaa`.

last-updated: Apr 11, 2022

Mappings and Table Listing Behaviors

When you use the `list` command without specifying a directory, the command's behavior depends on two factors:

- Whether a table mapping exists
- Whether Apache HBase is installed and running

Here are three different scenarios and the resulting `list` command behavior for each.

- There is a table mapping for *, as in `*:/tables`. In this case, the `list` command lists the tables in the mapped directory.
- There is no mapping for *, and Apache HBase is installed and running. In this case, the `list` command lists the HBase tables.
- There is no mapping for *, and Apache HBase is not installed or is not running.
  - For HBase 0.98.12, the shell will try to connect to an HBase cluster but it will return an error instead.
  - For HBase 1.1 or above, if the `mapr.hbase.default.db` property in the hbase-site.xml is set to `hbase`, the list command will return an error stating that HBase is not available. If the `mapr.hbase.default.db` property is set to `maprdb`, list command will list the MapR Database tables under the user's home directory.

Example 1: Map all HBase tables to MapR Database tables in a directory

In this example, any flat table name `foo` is treated as a MapR Database table in the directory `/tables_dir/foo`.

```
<property>
        <name>hbase.table.namespace.mappings</name>
        <value>*:/tables_dir</value>
        </property>
```

Example 2: Map specific Apache HBase tables to specific MapR Database tables

In this example, the Apache HBase table name `mytable1` is treated as a MapR Database table at `/user/aaa/mytable1`. The Apache Hbase table name `mytable2` is treated as a MapR Database table at `/user/bbb/mytable2`. All other Apache HBase table names are treated as stock Apache HBase tables.

```
<property>
        <name>hbase.table.namespace.mappings</name>
        <value>mytable1:/user/aaa/,mytable2:/user/bbb/</value>
        </property>
```

Example 3: Combination of specific table names and wildcards

Mappings are evaluated in order. In this example, the flat table name `mytable1` is treated as a MapR Database table at `/user/aaa/mytable1`. The flat table name `mytable2` is treated as a MapR Database table at `/user/bbb/mytable2`. Any other flat table name `foo` is treated as a MapR Database table at `/tables_dir/foo`.

```
<property>
        <name>hbase.table.namespace.mappings</name>
        <value>mytable1:/user/aaa/,mytable2:/user/bbb/,*:/tables_dir</value>
        </property>
```

Compression Mappings

MapR Database binary tables support the LZ4, LZF, and ZLIB compression algorithms.

When you create a MapR Database binary table with the Apache HBase API or the HBase shell and specify the LZ4, LZO, or SNAPPY compression algorithms, the table uses the LZ4 compression algorithm.

When you describe a MapR Database binary table's schema through the HBase API, the LZ4 and OLDLZF compression algorithms map to the LZ4 compression algorithm.

Copying Data

Note: The Apache CopyTable tool launches a MapReduce application. The nodes on your cluster must have the correct version of the `mapr-hbase` package installed. To ensure that your existing HBase applications and workflow work properly, install the `mapr-hbase` package that provides the same version number of HBase as your existing Apache HBase.

Launch the CopyTable tool with the following command, specifying the full destination path of the table with the `--new.name` parameter:

```
hbase org.apache.hadoop.hbase.mapreduce.CopyTable
-Dhbase.zookeeper.quorum=<ZooKeeper IP Address>
-Dhbase.zookeeper.property.clientPort=5181 --new.name=/user/john/foo/
mytable01
```

This example migrates the existing Apache HBase table `mytable01` to the MapR Database tables `/user/john/foo/mytable01`. On the node in the MapR cluster where you will launch the CopyTable tool, modify the value of the `hbase.zookeeper.quorum` property in the `hbase-site.xml` file to point at a ZooKeeper node in the source cluster. Alternately, you can specify the value for the `hbase.zookeeper.quorum` property from the command line. This example specifies the value in the command line.

1. Create the destination table. This example uses the HBase shell. The maprcli and the Control System are also viable methods.

   ```
   [user@host]$ hbase shell
   HBase Shell; enter 'help<RETURN>' for list of supported commands.
   Type "exit<RETURN>" to leave the HBase Shell

   hbase(main):001:0> create '/user/john/foo/mytable01', 'usernames',
   'userpath'
   0 row(s) in 0.2040 seconds
   ```

2. Exit the HBase shell.

   ```
   hbase(main):002:0> exit
   [user@host]
   ```

3. From the command line, use the CopyTable tool to migrate data.

   ```
   [user@host] hbase
   org.apache.hadoop.hbase.mapreduce.CopyTable -Dhbase.zookeeper.quorum=zkno
   de1,zknode2,zknode3 --new.name=/user/john/foo/mytable01 mytable01
   ```

Verifying Migration

After copying data to the new tables, verify that the migration is complete and successful. In increasing order of complexity:

1. Verify that the destination table exists. From the HBase shell, use the `list` command, or use the `hadoop fs -ls /user/john/foo` command from a Linux prompt:

```
hbase(main):006:0> list '/user/john/foo'
TABLE
/user/john/foo/mytable01
1 row(s) in 0.0770 seconds
```

2. Check the number of rows in the source table against the destination table with the `count` command:

```
hbase(main):005:0> count '/user/john/foo/mytable01'
30 row(s) in 0.1240 seconds
```

3. Hash each table, then compare the hashes.

Decommissioning Apache HBase Nodes

To decommission nodes running Apache HBase, follow these steps for each node:

1. From the HBase shell, disable the Region Load Balancer by setting the value of `balance_switch` to `false`:

```
hbase(main):001:0> balance_switch false
```

2. Leave the HBase shell by typing `exit`.

3. Run the *graceful stop* script to stop the HBase RegionServer:

Warning: The `graceful_stop.sh` script does not look up the hostname for an IP number. Do not pass an IP number to the script. Check the list of RegionServers in the Apache HBase Master UI to determine the hostname for the node being decommissioned.

```
[user@host] cd /opt/mapr/hbase/hbase-<hbase-version>
./bin/graceful_stop.sh <hostname>
```

Application Migration

Before you migrate your applications to the MapR Hadoop distribution, consider testing your applications using a small subset of data.

In this phase, you will migrate your applications to the MapR cluster test environment. The goal of this phase is to get your applications running smoothly on the MapR cluster using a subset of data. Once you have confirmed that all applications and components are running as expected you can begin migrating your data.

Migrating your applications from HDFS to MapR is relatively easy. MapR Hadoop is 100% plug-and-play compatible with Apache Hadoop, so you do not need to make changes to your applications to run them on a MapR cluster.

Application Migration Guidelines Keep the following considerations in mind when you migrate your applications:

last-updated: Apr 11, 2022

- **MapR Libraries**: Ensure that your applications can find the libraries/configs it is expecting. Make sure the java classpath includes the path to `maprfs.jar` and the `java.library.path` includes `libMapRClient.so`
- **MapR Storage**: Every application must point to MapR File System (`maprfs:///`) rather than the HDFS (`hdfs://`). If your application uses fs.default.name then it will work automatically. If you have hardcoded HDFS links into your applications, you must redirect those links so they point to MapR File System. Setting a default path of `maprfs:///` tells your applications to use the cluster specified in the first line of mapr-clusters.conf. You can also specify a specific cluster with `maprfs:///mapr/<cluster name>/`.
- **Permissions**: The distcp command does not copy permissions; permissions defined in HDFS do not transfer automatically to MapR File System. MapR uses a combination of access control lists (ACLs) to specify cluster or volume-level permissions and file permissions to manage directory and file access. You must define these permissions in MapR when you migrate your customized components, applications, and data. For more information, see Managing Permissions.
- **Memory**: Remove explicit memory settings defined in your applications. If memory is set explicitly in the application, the jobs may fail after migration to MapR.

Generally, the best approach to migrating your applications to MapR is to import a small subset of data and test and tune your application using that data in a test environment before you import your production data.

The following procedure offers a simple roadmap for migrating and running your applications in a MapR cluster test environment.

1. Copy over a small amount of data to the MapR cluster. Use the hadoop distcp hftp command to copy over a small number of files:

   ```
   $ hadoop distcp hftp://namenode1:50070/foo maprfs:///bar
   ```

   You must specify the namenode IP address, port number, and source directory on the HDFS cluster. For more information, see Copying Data from Apache Hadoop
2. Run the application.
3. Add more data and test again.
4. When the application is running to your satisfaction, use the same process to test and tune another application.

Data Migration

After you migrate your applications to the MapR cluster, you can copy your data from the Apache Hadoop HDFS to the MapR cluster.

Once you have installed and configured your MapR cluster in a test environment and migrated your applications to the MapR cluster you can begin to copy over your data from the Apache Hadoop HDFS to the MapR cluster.

Use any of the following methods to copy data from an HDFS cluster to a MapR cluster:

Method	Description
hdfs:// protocol	You can use the hadoop distcp command with the `hdfs://` protocol to copy data from an HDFS cluster into a MapR cluster. Use this method if the HDFS cluster and the MapR cluster use the same RPC protocol version. For all other scenarios, use the `webhdfs://` protocol or NFS gateway to copy data to a MapR cluster.

Method	Description
webhdfs:// protocol	You can use the hadoop distcp command with the `webhdfs://` protocol to copy data from an HDFS cluster into a MapR cluster.
NFS	You can mount a MapR cluster to an HDFS cluster via NFS mount and then use the hadoop distcp command to copy data between the two clusters.

Using the hdfs:// Protocol
This section describes how to copy data from an HDFS cluster to a MapR cluster using the hdfs:// protocol.

Before you can copy data from an HDFS cluster to a MapR cluster using the `hdfs://` protocol, you must configure the MapR cluster to access the HDFS cluster. To do this, complete the steps listed in Configuring a MapR Cluster to Access an HDFS Cluster for the security scenario that best describes your HDFS and MapR clusters and then complete the steps listed under Verifying Access to an HDFS Cluster.

You also need the following information:

- `<NameNode>`: the IP address or hostname of the NameNode in the HDFS cluster
- `<NameNode Port>`: the port for connecting to the NameNode in the HDFS cluster
- `<HDFS path>`: the path to the HDFS directory from which you plan to copy data
- `<MapR-FS path>`: the path in the MapR cluster to which you plan to copy HDFS data
- `<file>`: a file in the HDFS path

To copy data from HDFS to MapR File System using the `hdfs://` protocol, complete the following steps:

1. Run the following hadoop command to determine if the MapR cluster can read the contents of a file in a specified directory on the HDFS cluster:

   ```
   hadoop fs -cat <NameNode>:<NameNode port>/<HDFS path>/<file>
   ```

   For example:

   ```
   hadoop fs -cat hdfs://nn1:8020/user/sara/contents.xml
   ```

2. If the MapR cluster can read the contents of the file, run the distcp command to copy the data from the HDFS cluster to the MapR cluster:

   ```
   hadoop distcp hdfs://<NameNode>:<NameNode Port>/<HDFS path> maprfs://
   <MapR-FS path>
   ```

   For example:

   ```
   hadoop distcp hdfs://nn1:8020/user/sara maprfs:///user/sara
   ```

   Note the required triple slashes in `maprfs:///`

Using the webhdfs:// Protocol
This section describes how to copy data from an HDFS cluster to a MapR cluster using the webhdfs:// protocol.

Before you can copy data from an HDFS cluster to a MapR cluster using the `webhdfs://` protocol, you must configure the MapR cluster to access the HDFS cluster. To do this, complete the steps listed in

Configuring a MapR Cluster to Access an HDFS Cluster for the security scenario that best describes your HDFS and MapR clusters and then complete the steps listed under Verifying Access to an HDFS Cluster.

To copy data from HDFS to MapR File System using the `webhdfs://` protocol, complete the following steps:

1. The HDFS cluster must have WebHDFS enabled. Verify that the following parameter exists in the `hdfs-site.xml` file and that the value is set to `true`.

   ```
   <property>
   <name>dfs.webhdfs.enabled</name>
   <value>true</value>
   </property>
   ```

   You also need the following information:

   - `<NameNode>`: the IP address or hostname of the NameNode in the HDFS cluster
   - `<NameNode HTTP Port>`: the HTTP port on the NameNode in the HDFS cluster
   - `<HDFS path>`: the path to the HDFS directory from which you plan to copy data
   - `<MapR-FS path>`: the path in the MapR cluster to which you plan to copy HDFS data

2. Run the following command from a node in the MapR cluster to copy data from HDFS to MapR File System using `webhdfs://`:

   ```
   hadoop distcp webhdfs://<NameNode>:<NameNode HTTP Port>/<HDFS path>
   maprfs:///<MapR-FS path>
   ```

   For example:

   ```
   hadoop distcp webhdfs://nn2:50070/user/sara maprfs:///user/sara
   ```

   Note the required triple slashes in `maprfs:///`.

Using NFS

This section describes how to copy data from an HDFS cluster to a MapR cluster using NFS.

If NFS is installed on the MapR cluster, you can mount the MapR cluster to the HDFS cluster and then copy files from one cluster to the other using `hadoop distcp`. If you do not have NFS installed and a mount point configured, see Accessing Data with NFS and Setting Up MapR NFS.

To perform a copy using `distcp` via NFS, you need the following information:

- `<MapR NFS Server>`: the IP address or hostname of the NFS server in the MapR cluster
- `<maprfs_nfs_mount>`: the NFS export mount point configured on the MapR cluster; default is /mapr
- `<hdfs_nfs_mount>`: the NFS mount point configured on the HDFS cluster
- `<NameNode>`: the IP address or hostname of the NameNode in the HDFS cluster
- `<NameNode Port>`: the port on the NameNode in the HDFS cluster
- `<HDFS path>`: the path to the HDFS directory from which you plan to copy data
- `<MapR-FS path>`: the path in the MapR cluster to which you plan to copy HDFS data

To copy data from HDFS to MapR File System using NFS, complete the following steps:

1. Issue the following command to mount the MapR cluster to the HDFS NFS mount point:

   ```
   mount <MapR NFS Server>:/<maprfs_nfs_mount> /<hdfs_nfs_mount>
   ```

   For example:

   ```
   mount 10.10.100.175:/mapr /hdfsmount
   ```

2. Issue the following command to copy data from the HDFS cluster to the MapR cluster:

   ```
   hadoop distcp hdfs://<NameNode>:<NameNode Port>/<HDFS path> file:///
   <hdfs_nfs_mount>/<MapR-FS path>
   ```

   For example:

   ```
   hadoop distcp hdfs://nn1:8020/user/sara/file.txt file:///hdfsmount/user/
   sara
   ```

3. Issue the following command from the MapR cluster to verify that the file was copied to the MapR cluster:

   ```
   hadoop fs -ls /<MapR-FS path>
   ```

   For example:

   ```
   hadoop fs -ls /user/sara
   ```

Node Migration

You can add decommissioned HDFS data nodes to your MapR cluster.

Once you have loaded your data and tested and tuned your applications, you can add decommission HDFS data-nodes and add them to the MapR cluster.

This is a three-step process:

- **Decommissioning nodes on an Apache Hadoop cluster**: The Hadoop decommission feature enables you to gracefully remove a set of existing data-nodes from a cluster while it is running, without data loss. For more information, see the Hadoop Wiki FAQ.
- **Meeting minimum hardware and software requirements**: Ensure that every data-node you want to add to the MapR cluster meets the hardware, software, and configuration requirements.
- **Adding Nodes to a MapR cluster**: You can add those data-nodes to the MapR cluster. For more information, see Adding Nodes to a Cluster.

Applying a Patch

You can apply a patch by using the MapR Installer, by using the command line (a manual process), or by using an Installer Stanza.

Downloading a Patch

MapR patches can be downloaded from a secure FTP server.

To download the latest MapR patches for supported versions:

last-updated: Apr 11, 2022

1. Navigate to the secure FTP server at https://sftp.mapr.com.
2. Log in using `maprpatches` for your **Login ID**. Leave the **Password** field blank.
3. Click **Login**.

For patch information, visit the Support notices of known issues, some with patches or workarounds.

Applying a Patch Using the MapR Installer

The MapR Installer automates much of the work involved in applying MapR patches.

For clusters with many nodes, using the MapR Installer can save you time and reduce the likelihood of errors when compared with other methods of applying patches. With the MapR Installer, you can apply a patch during a:

- New installation of MapR software
- Maintenance update
- Version upgrade
- Incremental Install

Note: Applying a patch using the MapR Installer is an offline update (not a rolling update). Also, you cannot use the MapR Installer to apply a patch to an edge node or a client node.

To apply a patch using the MapR Installer:

1. Obtain the patch from MapR Support. See Applying a Patch on page 437.
2. Ensure that the cluster is ready for a patch update. For more information, see Verify Cluster Readiness for a Patch. Then return to this procedure.
3. Start the MapR Installer. For more information, see Installer.
4. Select the **Patch file** option:

If . . .	Then . . .
MapR software is not yet installed on the cluster (new installation)	Click the **Patch file** option under the **MapR Version** field on the **Version & Services** page.
MapR software is already installed (maintenance update or version upgrade)	Do one of the following: • Click the **Maintenance Update** button. If you are performing a maintenance update, the Patch file option appears on the **Maintenance Update** page. For more information, see Performing a Maintenance Update on page 5447. • Click the **Version Upgrade** button on the MapR Installer page. If you are performing a version upgrade, the **Patch file** option appears under the **MapR version** on the **Upgrade Version & Services** page. • Click the **Incremental Install** button. The Patch file option appears on the **Version & Services** page.

The installer prompts you to select the patch.

5. Select the patch file, and click **Choose**. The installer verifies that the core version of the installed core (or the core version you are upgrading to) matches the core version of the patch file name. The installer also ensures that the patch file starts with *mapr-patch*, ends with *rpm* or *deb*, and does not include text such as *client* or *nfs* (to ensure that it is a core patch file). The installer does not ensure that the patch you are applying is a patch number higher than the one that is already installed (if a patch is already installed).

6. Make other installer selections as needed. The patch is uploaded and will be installed in the background after the installer has applied any core packages.

   Note: The next time you run the MapR Installer on the cluster, the installer shows the updated patch version on the **Incremental Install** page or, if you enable patch installation, on the **Version Upgrade** or **Maintenance Update** page.

Applying a Patch Manually

MapR patches are version-specific and cumulative. Each patch contains the code fixes that were included in the previous patch for that release version.

Before applying a patch, note these considerations:

- For patch-download information, see Applying a Patch on page 437.
- The steps for patching the Control System are different from the steps for patching MapR core. See Special Considerations for the Control System Patches on page 444.
- You can also apply a patch using the MapR Installer. See Applying a Patch Using the MapR Installer on page 438.
- A patch for a given software version can be removed, and an older patch for the same MapR software version can be installed. However, rolling back a cluster from a newer release version to an older version is not supported.
- Different types of patches are available, and some patches can only be installed on specific nodes:

Patch Type	Install on These Nodes
`mapr-patch`	All control and data nodes
`mapr-patch-client`	MapR client nodes only
`mapr-patch-loopbacknfs`	Any node where POSIX is supported
`mapr-patch-nfs4server`	Any node where NFSv4 is supported
`mapr-patch-posix-client-basic`	Any node where FUSE POSIX is supported
`mapr-patch-posix-client-platinum`	Any node where FUSE POSIX is supported

Applying a patch is a three-step process:

1. Step 1: Verify Cluster Readiness for a Patch on page 440

last-updated: Apr 11, 2022

2. Step 2: Apply the Patch to Data Nodes on page 441
3. Step 3: Apply the Patch to CLDB Nodes on page 442

When you apply a patch to the cluster, the patched files along with original files (non-patched) are copied to the `/opt/mapr/.patch` folder. In the `/opt/mapr/.patch` folder, the file ending with `.0` is the original file (non-patched) and the file ending with `.<patch_number>` is the patched version. Therefore, if there is a file under `/opt/mapr/.patch/lib/`, you can compare that with the corresponding file under `/opt/mapr/lib/` by using the `md5sum` command to verify that the patch was successfully deployed.

If you need more information or if you encounter any problems with patch installation, contact HPE Support.

Step 1: Verify Cluster Readiness for a Patch
Before you apply a patch, check that the cluster is ready for a patch to be applied. In addition to the prerequisites, consider verifying that the cluster utilizes best practices which will facilitate a more optimal patch installation.

Patch Install Prerequisites

Before you apply a patch on the cluster, verify that all CLDB nodes are running and that container 1 is fully replicated on each CLDB node.

Run `maprcli dump containerinfo -ids 1 -json`. In the output, all CLDBs should be listed under `ActiveServers` and each node should report a VALID state.

For example:

```
...
      "data":[
             {
                "ContainerId":1,
                "Epoch":3,
                "Master":"<masterCLDB_IP>:5660--3-VALID",
                "ActiveServers":{
                  "IP:Port":[
                      "<masterCLDB_IP>:5660--3-VALID",
                      "<slaveCLDB_IP>:5660--3-VALID",
                      "<slaveCLDB_IP>:5660--3-VALID"
                   ]
              },
                  "InactiveServers":{
                   },
                  "UnusedServers":{
              },
...
```

Note: RESYNC state will display when container 1 is not fully replicated on that node. You must wait until each CLDB node has a VALID state for container 1 before proceeding with the patch installation.

For more information, see dump containerinfo on page 1616

Best Practices for Patch Installation

Failure to follow the best practices may, in some cases, impact the speed in which the patch installation completes. Check to see if your cluster abides by the following best practices:

The volume min replication setting should be greater than or equal to 2 for CLDB volume.	This ensures that container 1 always has at least two valid copies. Run the following command to list the current replication setting: `maprcli dump volumeinfo -volumename mapr.cldb.internal -json` In the output, the "VolumeMinReplication" parameter lists the current replication setting for the named volume. For more information, see maprcli dump volumeinfo.
No under replicated volumes should exist on the cluster.	Run the following command to check for under-replicated volumes: `maprcli alarm list` For more information, see maprcli alarm list.

Each CLDB node should be configured to have a minimum of 3 disks in its storage pool.	Run the following command on each CLDB node to get a list of the disks configured for each storage pool:
	`mrconfig sp list [-v]`
	In this example output, there are three disks associated with SP1:
	`ListSPs resp: status 0:2 No. of SPs (2), totalsize 4562260 MB, totalfree 4537550 MB`
	`SP 0: name SP1, Online, size 2736933 MB, free 2724749 MB, path /dev/sdb, log 200 MB, port 5660, guid a3055a6db41f285b005883bbd701c1e5, clusterUuid -5009075714600063565-100367519220387605, disks /dev/sdb /dev/sdd /dev/sde`
	For more information, see mrconfig sp list.

Step 2: Apply the Patch to Data Nodes

When applying a patch manually, apply the patch to nodes dedicated to storing and processing data prior to applying the patch on nodes that run the CLDB. This includes nodes that run the Fileserver for storage and processing components such as the NodeManager and the HBase client.

Note: For clusters with more than 100 data nodes, it is a best practice to apply the patch in batches. Also, wait a few minutes before proceeding to the next batch of nodes.

On each data node:

1. Stop the MapR Warden and ZooKeeper (if installed) services:
   a) To stop MapR Warden, run the following command:
   ```
   sudo service mapr-warden stop
   ```
   b) If Zookeeper is installed on the node, run this command:
   ```
   sudo service mapr-zookeeper stop
   ```


last-updated: Apr 11, 2022

2. If there is already a patch installed on the cluster, run one of the following commands to uninstall it:
   - On CentOS/Redhat: `sudo rpm -e mapr-patch`
   - On SLES: `sudo zypper remove mapr-patch`
   - On Ubuntu: `sudo apt-get -y remove mapr-patch`
3. Install the patch using one of the following commands:
   - On CentOS/RedHat: `sudo rpm -ivh mapr-patch-<new_patch_number>.rpm`
   - On SLES: `sudo zypper install mapr-patch-<new_patch_number>.rpm`
   - On Ubuntu: `sudo dpkg -i mapr-patch-<new_patch_number>.deb`
4. Start the MapR Warden and ZooKeeper (if installed) services:
   a) If ZooKeeper is installed on the node, run this command to start ZooKeeper:

   ```
   sudo service mapr-zookeeper start
   ```

   b) To start Warden, run this command:

   ```
   sudo service mapr-warden start
   ```

5. To verify that the patch was installed successfully, run one of the following commands:
   - On CentOS/Redhat or SLES: `sudo rpm -ql mapr-patch-<new_patch_number>`
   - On Ubuntu: `sudo dpkg -l | grep mapr-patch-<new_patch_number>`

Step 3: Apply the Patch to CLDB Nodes
When applying a patch manually, apply the patch to CLDB secondary nodes prior to applying the patch on the primary CLDB node. After you apply a patch to a CLDB node, you must verify that container 1 is fully replicated before proceeding to apply the patch to the next CLDB node.

For large clusters with many containers, when you do not patch CLDB nodes in the prescribed order, there may be a considerable delay before the cluster can process client operations. For smaller clusters, this is not critical as the cluster can generally start accepting client operations in about 5 minutes.

Complete the following steps on each CLDB secondary node and then on the CLDB primary node:

1. Stop the MapR Warden and ZooKeeper (if installed) services:
   a) To stop MapR Warden , run the following command:

   ```
   sudo service mapr-warden stop
   ```

   b) If ZooKeeper is installed on the node, run this command:

   ```
   sudo service mapr-zookeeper stop
   ```

2. If there is already a patch installed on the cluster, run one of the following commands to uninstall it:
   - On CentOS/RedHat: `sudo rpm -e mapr-patch`
   - On SLES: `sudo zypper remove mapr-patch`
   - On Ubuntu: `sudo apt-get -y remove mapr-patch`

3. Install the patch using one of the following commands:
   - On CentOS/RedHat: `sudo rpm -ivh mapr-patch-<new_patch_number>.rpm`
   - On SLES: `sudo zypper install mapr-patch-<new_patch_number>.rpm`
   - On Ubuntu: `sudo dpkg -i mapr-patch<new_patch_number>.deb`
4. Start the MapR Warden and ZooKeeper (if installed) services:
   a) If ZooKeeper is installed on the node, run this command to start ZooKeeper:

   ```
   sudo service mapr-zookeeper start
   ```

   b) To start Warden, run this command:

   ```
   sudo service mapr-warden start
   ```

5. To verify that the patch was installed successfully, run one of the following commands:
   - On CentOS/RedHat or SLES: `sudo rpm -ql mapr-patch-<new_patch_number>`
   - On Ubuntu: `sudo dpkg -l | grep mapr-patch-<new_patch_number>`
6. Verify that the CLDB node that you patched is running and that container 1 on that node is fully replicated.

   Run `maprcli dump containerinfo -ids 1 -json`.

   In the output, the CLDB node that you just patched should be listed under ActiveServers, and should report a VALID state for container 1.

   For example:

   ```
   ...
         "data":[
                {
                   "ContainerId":1,
                   "Epoch":3,
                   "Master":"<masterCLDB_IP>:5660--3-VALID",
                   "ActiveServers":{
                     "IP:Port":[
                         "<masterCLDB_IP>:5660--3-VALID",
                         "<slaveCLDB_IP>:5660--3-VALID",
                         "<slaveCLDB_IP>:5660--3-VALID"
                      ]
                },
                    "InactiveServers":{
                     },
                    "UnusedServers":{
                },
   ...
   ```

   Note: The RESYNC state will display when container 1 is not fully replicated on that node. You must wait until the CLDB node that you just patched has a VALID state for container 1.

   For more information, see dump containerinfo on page 1616

last-updated: Apr 11, 2022

Applying a Patch Using an Installer Stanza

Applying a patch using an Installer Stanza leverages the automation provided by the patch-install capability of the web-based MapR Installer.

To apply a patch using an Installer Stanza, you specify a file name and directory in the `environment.patch_location` parameter in the Stanza (YAML) file. Then you issue the `install` command to run the Stanza.

Note: Applying a patch using the MapR Installer is an offline update (not a rolling update).

For information about the `install` command, see Installing or Upgrading MapR Core Using an Installer Stanza on page 5509. For information about the Stanza parameters, see Working with MapR Installer Stanza Files on page 5504.

Rolling Back a Patch

Removing a previously installed patch is a manual process. You can revert to a previous version of the patch by first removing the current patch and then installing the previous version.

Always test patch installs in a test environment before applying patches to production environments. If a newly installed patch for a given software version delivers unexpected behavior, you can remove the patch or install an older patch for the same data-fabric software version. To roll back to a previous patch version, use the steps in Step 2: Apply the Patch to Data Nodes on page 441, but install the desired older version of the patch instead of the latest version.

The latest patch version (for example, version n) and the previous patch version (version `n-1`), are always available on sftp.mapr.com. To log in, specify `maprpatches` for the **Login ID**, and leave the **Password** field blank.

For technical assistance in removing a patch and restoring the functionality that existed before the patch was installed, open a case with HPE Support.

Note: Rolling back a cluster from a newer data-fabric software version to an older version is not supported. See Applying a Patch Manually on page 439.

Special Considerations for the Control System Patches

Patches for the Control System are handled differently from patches for cluster data nodes and CLDB nodes.

The Control System software is updated as a package rather than a patch file. While MapR core patches typically include a prefix such as `mapr-patch` or `mapr-patch-client`, or `mapr-patch-posix-client-basic`, Control System software is updated as a new package and does not use the MapR core patch mechanism.

To identify a Control System patch package, look for `mapr-apiserver` or `mapr-webserver` in the package name, and use these steps to update your currently installed packages:

1. Stop the `apiserver` service on all Control System nodes:

```
$ maprcli node services -filter [csvc==apiserver] -name
apiserver -action stop
```

2. Upgrade the existing `mapr-apiserver` and `mapr-webserver` packages. For example:
   - On CentOS/RedHat or SLES:

```
$ rpm -Uvh <path to new mapr-apiserver>
$ rpm -Uvh <path to new mapr-webserver>
```


last-updated: Apr 11, 2022

- On Ubuntu:

```
$ dpkg -i <path to new mapr-apiserver>
$ dpkg -i <path to new mapr-webserver>
```

3. Run `configure.sh` to update the configuration for the new packages:

```
$ /opt/mapr/server/configure.sh -R --noRecalcMem
```

4. Start the `apiserver` service on all Control System nodes:

```
$ maprcli node services -filter [csvc==apiserver] -name
apiserver -action start
```

Special Considerations for FUSE POSIX Patches

Patches for some features, such as the FUSE POSIX client, can require post-installation steps.

When you install a FUSE POSIX patch, a new and backup copy of the `fuse.conf` file are created in the `/opt/mapr/conf` directory. These files are called:

- `fuse.conf.new`
- `fuse.conf.old`

You can find the new parameters in the `fuse.conf.new` file. If needed, you can copy the new parameters to your existing `fuse.conf` file and restart FUSE for the settings to take effect.

For FUSE POSIX configuration information, see Configuring the MapR FUSE-Based POSIX Client on page 1240.

Applying a Patch to a MapR POSIX Client

This procedure enables you to apply a patch to any of the MapR POSIX clients, which include the MapR loopbacknfs POSIX client, the FUSE-based POSIX basic client, and the FUSE-based POSIX platinum client.

1. Before applying a patch to a FUSE-based POSIX client, review Special Considerations for FUSE POSIX Patches on page 445.
2. Use one of the following commands to stop the POSIX client service:
   - For the `mapr-loopbacknfs` service:

```
service mapr-loopbacknfs stop
```

   - For the FUSE-based POSIX basic service:

```
service mapr-posix-client-basic stop
```

   - For the FUSE-based POSIX platinum service:

```
service mapr-posix-client-platinum stop
```

3. Remove any currently installed client patches:

last-updated: Apr 11, 2022

For the `mapr-loopbacknfs` service:	• On Red Hat / CentOS:
	`yum remove mapr-patch-loopbacknfs-<old_patch_number>.rpm`
	• On SLES:
	`zypper remove mapr-patch-loopbacknfs-<old_patch_number>.rpm`
	• On Ubuntu:
	`apt-get remove mapr-patch-loopbacknfs-<old_patch_number>.deb`

For the FUSE-based POSIX basic service:	• On Red Hat / CentOS:
	`yum remove mapr-patch-posix-client-basic-<old_patch_number>.rpm`
	• On SLES:
	`zypper remove mapr-patch-posix-client-basic-<old_patch_number>.rpm`
	• On Ubuntu:
	`apt-get remove mapr-patch-posix-client-basic-<old_patch_number>.deb`

For the FUSE-based POSIX platinum service:	• On Red Hat / CentOS:
	`yum remove mapr-patch-posix-client-platinum-<old_patch_number>.rpm`
	• On SLES:
	`zypper remove mapr-patch-posix-client-platinum-<old_patch_number>.rpm`
	• On Ubuntu:
	`apt-get remove mapr-patch-posix-client-platinum-<old_patch_number>.deb`

last-updated: Apr 11, 2022

4. Apply the new patch:

For the `mapr-loopbacknfs` service:	• On Red Hat / CentOS:
	`sudo rpm -i mapr-patch-loopbacknfs-<new_patch_number>.rpm`
	• On SLES:
	`sudo zypper install mapr-patch-loopbacknfs-<new_patch_number>.rpm`
	• On Ubuntu:
	`sudo dpkg -i mapr-patch-loopbacknfs-<new_patch_number>.deb`

For the FUSE-based POSIX basic service:	• On Red Hat / CentOS:
	`sudo rpm -i mapr-patch-posix-client-basic-<new_patch_number>.rpm`
	• On SLES:
	`sudo zypper install mapr-patch-posix-client-basic-<new_patch_number>.rpm`
	• On Ubuntu:
	`sudo dpkg -i mapr-patch-posix-client-basic-<new_patch_number>.deb`

For the FUSE-based POSIX platinum service:	• On Red Hat / CentOS:
	`sudo rpm -i mapr-patch-posix-client-platinum-<new_patch_number>.rpm`
	• On SLES:
	`sudo zypper install mapr-patch-posix-client-platinum-<new_patch_number>.rpm`
	• On Ubuntu:
	`sudo dpkg -i mapr-patch-posix-client-platinum-<new_patch_number>.deb`

last-updated: Apr 11, 2022

5. Use one of the following commands to restart the POSIX client service:
   - For the `mapr-loopbacknfs` service:

     ```
     service mapr-loopbacknfs start
     ```

   - For the FUSE-based POSIX basic service:

     ```
     service mapr-posix-client-basic start
     ```

   - For the FUSE-based POSIX platinum service:

     ```
     service mapr-posix-client-platinum start
     ```

Related concepts

Applying a Patch on page 437
You can apply a patch by using the MapR Installer, by using the command line (a manual process), or by using an Installer Stanza.

Upgrading the MapR Client on page 326
Depending on which MapR client you want to update, you will either need to install and reconfigure or perform a package upgrade.

Upgrading the MapR POSIX loopbacknfs Client on page 328
Perform a package upgrade to get a newer version of the MapR POSIX loopbacknfs Client.

Packages and Dependencies for MapR Software on page 68
Package and dependency details for MapR 6.1 platform and ecosystem components.

6.1 MapR Data Platform

MapR Data Platform is the industry-leading data platform for AI and analytics that solves enterprise business needs.

The MapR Data Platform enables you to master critical data challenges, specifically:

- Speed up AI and analytics initiatives for more impact at production scale
- Accelerate time-to-value for hybrid cloud and multi-cloud strategies
- Create highly reliable, scalable data fabric
- Use data streams for real-time edge analytics
- Implement Kubernetes containerization more effectively

The MapR Data Platform allows you to address your critical data needs while providing industry-leading performance, data security, easy application development, and true scalability.

The MapR Data Platform enables you to solve critical business needs:

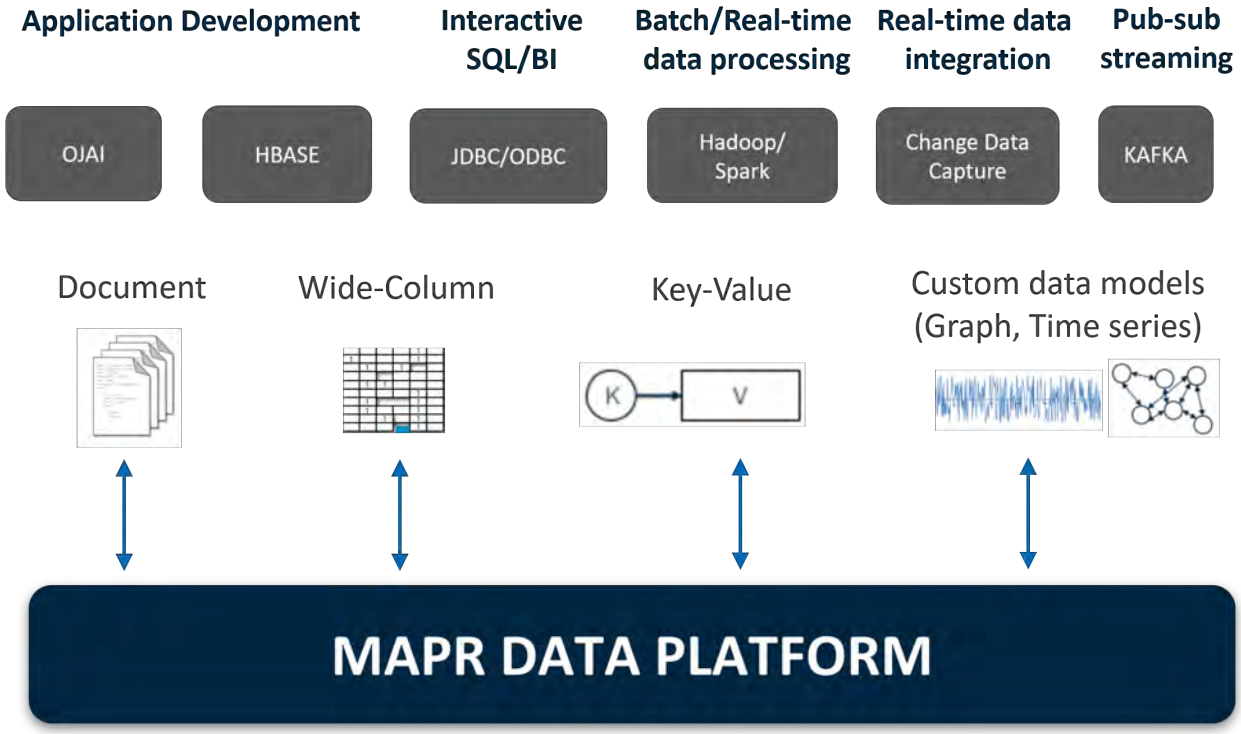
last-updated: Apr 11, 2022

Business Need	MapR Data Platform Provides....	Typical Use Cases...
AI and Analytics	A data platform approach for full range of AI, ML, Analytics with no silos, faster response and mission critical reliability at scale	• Contextual experiences • Recommendations • Churn detection • Real-time analytics • DWH offload • Operational data hub • Fraud detection • Security analytic
IOT and Edge Analytic	Seamless edge to on-prem or cloud data movement with analytic	• IoT Analytic • Edge to edge fabric • Anomaly detection • Preventative maintenance • Multi-cloud • Streaming analytic • Real-time response
Journey to Cloud	Easy data and application movement between on-prem and multiple clouds delivers lower TCO, higher flexibility	• Scale-out storage • Global repository, persistent data containers • High-performance filesystem • Multi-cloud choice • GDPR
Containers	Enable stateful applications in containers to use system-of-record data in a high-reliability platform	• Improve agility • Greater flexibility • Higher elasticity • Better utilization

last-updated: Apr 11, 2022

High-level View of the MapR Data Platform

The following diagram shows the basic components of the MapR Data Platform.



Get Started

To learn more about MapR Data Platform, see this course.

For planning information and the manual installation steps, see:

- Planning the Cluster on page 107
- Minimum Cluster Size on page 112
- Installing without the MapR Installer on page 141

You can jump right into using the MapR Data Platform:

- MapR Sandbox for Hadoop: Try out a single-node cluster that's ready to roll, right out of the box!
- MapR Installer: Use the MapR Installer to set up a production cluster, large or small. See MapR Installer on page 5395.

Learn More about the Architecture of the MapR Data Platform Components

This system overview contains architectural details about the components that run on the MapR Data Platform and the relationships between the components. See these topics to learn about each component.

Additional Resources

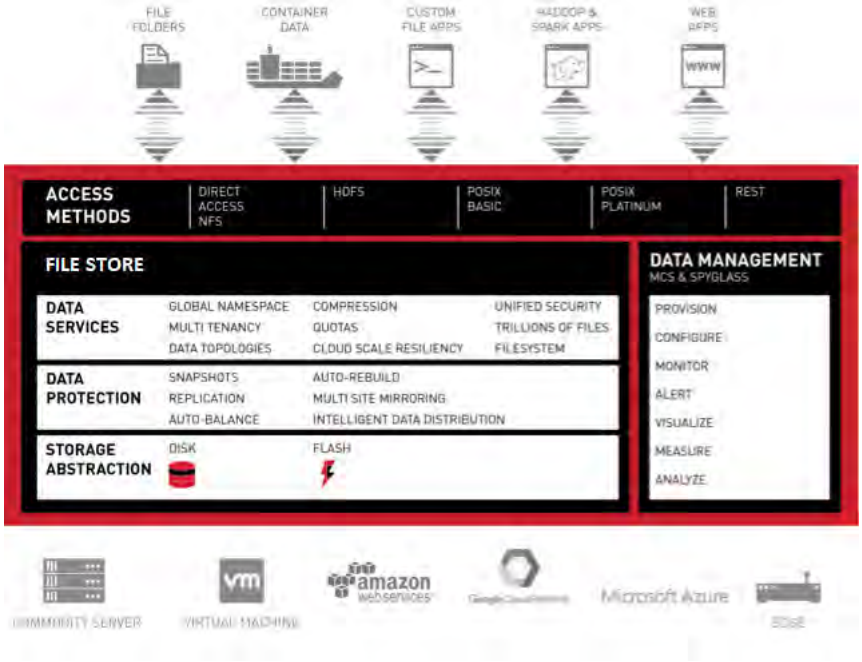
For an excellent introduction to how a data fabric can help enable a comprehensive data strategy, see this blog.

MapR XD Distributed File and Object Store

MapR XD Distributed File and Object Store is a distributed filesystem for data storage, data management, and data protection. MapR XD supports mounting and cluster access via NFS and FUSE-based POSIX clients (basic, platinum, or PACC) and also supports access and management via HDFS APIs.

MapR XD is the only cloud-scale data store that enables you to build a fabric of exabyte-scale. MapR XD supports trillions of files, 100s of 1000s of client nodes, and runs on edge clusters, on-prem data centers, and the public cloud.

You can manage your clusters from the Managed Control System (web console) and monitor them using HPE Ezmeral Data Fabric Monitoring (Spyglass initiative).



1. Direct Access NFS
2. HDFS
3. FUSE-based POSIX Clients
4. REST API
5. Storage Abstraction
6. Data Protection
7. MapR File System
8. MapR Control System
9. MapR Monitoring

How Do I Get Started?

Refer to the documentation specific to your role:

last-updated: Apr 11, 2022

Architect	Administrator/Dev Ops	Developer
Planning the Cluster	Installing the MapR XD	Managing data using maprcli
Planning for high availability	Setting up volumes, volume replication, snapshots, and mirroring schedules	Accessing MapR filesystem using HDFS APIs
Reviewing security capabilities and architecture	Configuring security with ACLs/ACEs and tickets	
	Mounting the cluster for access using NFS and POSIX clients	
	Managing the cluster using the Control System and monitoring the cluster using MapR Monitoring	

Additional Resources

See the following MapR page for more MapR XD information:

- HPE Ezmeral Product Page
- Install File Store
- Administer Files and Directories
- Administrator's Reference
- File Store APIs

File System

Discusses the features of the MapR distributed file system and compares it to the Hadoop Distributed File System (HDFS).

The MapR distributed file system provides a unified data solution for structured data (tables) and unstructured data (files).

The MapR file system is a random, read-write distributed file system that allows applications to concurrently read and write directly to disk. The Hadoop Distributed File System (HDFS), by contrast, has append-only writes and can only read from closed files. As HDFS is layered over the existing Linux file system, a large number of input/output (I/O) operations decrease the cluster's performance. The MapR distributed file system also eliminates the Namenode associated with cluster failure in other Hadoop distributions, and enables special features for data management, and high availability.

The storage system architecture used by the MapR distributed file system is written in C/C++ and prevents locking contention, eliminating performance impact from Java garbage collection.

The following table highlights some of the features of the :

Feature	Description
Storage pools	A group of disks to which the MapR file system writes data.
Containers	An abstract entity that stores files and directories in the MapR file system. A container always belongs to exactly one volume, and can hold namespace information, file chunks, or table chunks for that volume.
CLDB	A service that tracks the location of every container.
Volumes	A management entity that stores and organizes containers. Used to distribute metadata, set permissions on data in the cluster, and for data backup. A volume consists of a single name container, and a number of data containers.

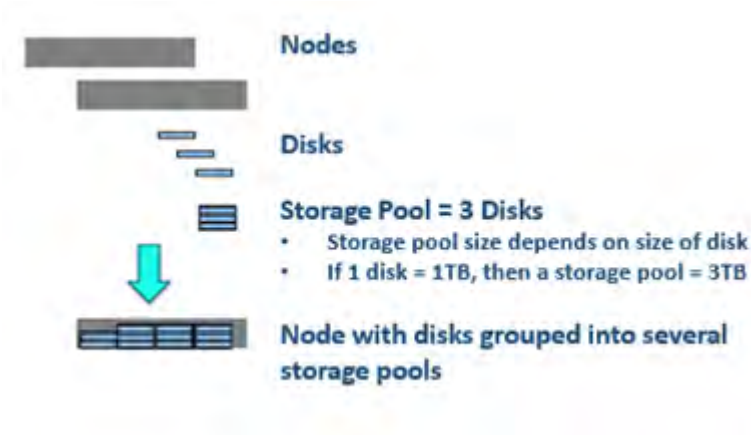
last-updated: Apr 11, 2022

Feature	Description
Direct Access NFS	Enables applications to read and write data directly on to the cluster.
POSIX Clients	The loopbacknfs, and FUSE-based POSIX clients connect to one or more MapR clusters, and allow app servers, web servers, and applications to write data directly, and securely to the MapR cluster.

Storage Pools
Describes what storage pools are.

The MapR File System storage architecture consists of multiple storage pools that reside on each node in a cluster. A storage pool is made up of one or more disks grouped by the MapR file system. The default number of disks in a storage pool is three. The containers that hold the MapR filesystem data are stored in, and replicated among the storage pools in the cluster.

The following image represents disks grouped together to create storage pools that reside on a node:



Write operations within a storage pool are striped across disks to improve write performance. Stripe width and depth are configurable with the disksetup script. As the MapR filesystem performs data replication, you do not need to configure RAID.

Containers and the CLDB
Describes what containers are, and the role of the Container Location Database (CLDB) in managing them.

The MapR filesystem stores data in abstract entities called containers that reside on storage pools. Each storage pool can store many containers. Blocks enable full read-write access to the MapR filesystem, with efficient snapshots.

An application can write, append, or update more than once in the MapR filesystem, and can also read a file as it is being written. In other Hadoop distributions, an application can only write once, and the application cannot read a file as it is written.

On average, a container size is 10-30 GB. The default container size is 32GB. Large number of containers allow for greater scaling and allocation of space in parallel, without bottlenecks.

Described from the physical layer:

- Files are divided into chunks.
- The chunks are assigned to containers.
- The containers are written to storage pools, which are made up of disks on the nodes in the cluster.

The following table compares the MapR filesystem storage architecture to the HDFS storage architecture:

last-updated: Apr 11, 2022

Storage Architecture	HDFS	MapR Filesystem
Management layers	Files, directories and blocks, managed by Namenode.	Volume, which holds files and directories, made up of containers, which manage disk blocks and replication.
Size of file shard	64MB block	256MB chunk
Unit of replication	64MB block	32GB container
Unit of file allocation	64MB block	8KB block

To preserve data, the MapR filesystem automatically replicates containers across various nodes on the cluster. Container replication creates multiple synchronized copies of the data across the cluster for failover. Container replication also helps localize operations, and ensures that read operations occur in parallel. When a disk or node failure brings a container's replication levels below a specified replication level, the MapR filesystem automatically re-replicates the container elsewhere in the cluster until the desired replication level is achieved. A container only occupies disk space when an application writes to it.

The CLDB (Container Location Database) maintains information about the location of every container in the cluster, defines the container precedence in the replication chain, and organizes container content updates across the replication chain. It runs as a system of independent servers, only one of which is a master at any time.

The MapR filesystem and other services (such as NFS Gateway and POSIX) send heartbeat (HB) messages to the master CLDB. The CLDB is registered with ZooKeeper, and the master CLDB to ZooKeeper connection is kept alive by sending a probe message every few seconds. The CLDB service tracks the location of every container, and uses these HB messages to determine the state of all containers on that node. The CLDB actively participates in the failover of a node in the event of a node failure.

Understanding Replication
Describes how replication works, and how to configure the replication factor.

Volumes are stored as pieces called containers that contain files, directories, and other data. By default, the maximum container size is 32 GB. The MapR administrator sets the maximum container size using the `cldb.container.sizemb` parameter (see the config commands). Containers are replicated to protect data. Normally, each container has three copies stored on separate nodes to provide uninterrupted access to all data, even if a node fails.

For each volume, you can specify a desired and minimum data replication factor, and a desired and minimum namespace (name container) replication factor.

When enabled, the CLDB manages the namespace container replication separate from the data container replication. Use this capability when you have low volume replication, but want to have higher namespace replication.

Note: The namespace container parameters, *nsreplication* or *nsminreplication*, must be the same or larger than the equivalent data replication parameter, *replication* or *minreplication*.

Data Replication

- The replication factor is the number of replicated copies that you need for normal operation and data protection. When the number of copies falls below the desired replication factor, but remains equal to or above the minimum replication factor, the CLDB actively creates additional copies of the container while trying to minimize the impact of making an additional copy of the container. Re-replication occurs after the timeout specified in the `cldb.fs.mark.rereplicate.sec` parameter (configurable using the configuration API). The minimum replication factor is 1 and the maximum is 6 (default: 3).
- The minimum value of the minimum replication factor is the smallest number of copies you need in order to adequately protect against data loss. When the replication factor falls below this minimum value, re-replication occurs aggressively if data is being actively written to the container. If the `enforceminreplicationforio` property is set to `true`, writes succeed only when the minimum replication factor requirements are met. If the `enforceminreplicationforio` property is set to `true` and the minimum number of copies are not available, the client is asked to retry. In the case of a:
  - Hard mount, the client might try for up to 10 minutes and then return an error
  - Soft mount, the client might return an error

  The minimum value of the minimum replication factor is 1 and the maximum value is 6 (default:2). In all cases, the minimum replication factor cannot be greater than the replication factor. When you increase the minimum replication factor, if the `enforceminreplicationforio` property (configurable at the volume level) is set to `true`, the requirement to maintain a minimum number of copies is not enforced during writes until new copies of all containers associated with the volume are created.

Name Container Replication

- The namespace replication factor is the number of namespace container replicated copies that you need for normal operation and data protection. When the number of copies falls below the desired replication factor, but remains equal to or above the minimum replication factor, the CLDB actively creates additional copies of the container while trying to minimize the impact of making an additional copy of the container. Re-replication occurs after the timeout specified in the `cldb.fs.mark.rereplicate.sec` parameter (configurable using the configuration API). The minimum replication factor is 1 and the maximum is 6 (default: 3).
- The minimum value of the minimum namespace replication factor is the minimum number of namespace container replicated copies you want in order to adequately protect against data

last-updated: Apr 11, 2022

loss. When the replication factor falls below this minimum value, re-replication occurs aggressively if data is being actively written to the container. If the `enforcemineplicationforio` property (configurable at the volume level) is set to `true`, writes succeed only when this minimum value of the minimum replication factor requirements are met. If this property is set to `true` and minimum number of copies are not available, the client is asked to retry. In the case of a:

- Hard mount, the client tries for up to 10 minutes and then return an error
- Soft mount, the client returns an error

The system does not wait for lost replicas to become available again. The minimum value of the muinimum replication factor is 1 and the maximum value is 6 (default: 2). In all cases, the minimum replication factor cannot be greater than the replication factor. When you increase the minimum replication factor, if the `enforceminreplicationforio` property is set to `true`, the presence of the minimum number of copies is not enforced during writes until new copies of all containers associated with the volume are created.

Note: The maximum replication setting of 6 does **not** apply for ***mapr.cldb.internal volume containers*** (CID-1). The number of CID-1 container replicas are always equivalent to the number of CLDB nodes in the cluster.

If any containers in the CLDB volume fall below the minimum value of the minimum replication factor, the cluster is inaccessible until aggressive re-replication restores the minimum level of replication. If a disk failure is detected, any data stored on the failed disk is re-replicated without regard to the timeout specified in the `cldb.fs.mark.rereplicate.sec` parameter.

If all copies of a container, which are neither under nor over replicated, are on the same rack, MapR automatically detects and distributes the copies, such that they are all not on the same rack, after 12 hours. If a container is under replicated and MapR is unable to find a different rack for the new copy, the creation of the copy is deferred. If another rack is unavailable for the new copy after 3 hours, MapR creates a copy of the container on the same rack and if this results in all copies of the container being on the same rack, MapR distributes the copies after 12 hours. Also, during replication, MapR tries to defer the scenarios where all copies end up on the same rack. As per deferring policy:

- If a container has copies less than the "minimum replication" but greater than 2 and if both copies end up on the same rack, then MapR tries to create the third copy on a different rack for up to 3 hours.
- If a container has copies more than the minimum but less than the desired and if all copies are on the same rack, then MapR tries to create the next copy on a different rack for up to 3 hours.

If you do not set the namespace (NS) replication and minimum namespace replication values explicitly, they assume the same values as (data) replication and minimum replication respectively. This means that all changes to (data) `replication` and `minreplication` parameters are also reflected in `nsreplication` and `nsminreplication`. If `nsreplication` or `nsminreplication` is modified or specified during creation, `nsreplication` and `nsminreplication` start assuming values different from `replication` and `minreplication`.

Table Replication vs Mirroring - Understanding the Differences

This section describes the advantages of both Table Replication and Mirroring, to let you determine the best option for your use case.

Advantages of Table Replication

1. Table replication replicates each table update instantaneously, in seconds (subject to compute and network resources). Mirroring has a much larger RTO (recovery time objective), in minutes.
2. Table replication also transmits lesser data because it just transmits the actual physical rows and nothing else.
3. In table replication, both the end points are READ-WRITE masters with the option of two-way multi-master replication.
4. Table replication proceeds from Source Table > Destination Gateway(s) > Destination Table, which provides reasonable isolation between the two end point clusters. The source table talks only to the Destination Gateway(s).

When using mirroring, avoid placing table replication sources in the mirror volume. Doing so, creates problems if the mirror is broken and promoted.

For tables and streams, table replication is usually the right choice. However, there are exemptions where mirroring is the best choice.

Advantages of Mirroring

1. Since a volume mirror represents a moment in time, there is a higher probability of recovering from a volume than from multiple tables.
2. You can retain old states of a mirror. If you have deleted a bunch of data in your tables and table replication has replicated those changes, then you can recover your data from a mirror.
3. Mirrors are helpful during development. Create a read-write mirror and use for development. Revert it to the last mirrored state and start over. The point is that you can revert the entire volume to a known state, as needed.
4. Use local mirror(s) to increase read throughput.
5. You can use mirrors to obtain traceability and reproducibility during data operations such as machine learning. You can have separate mirrors for different clusters, and operations on one mirror do not affect the other.

Understanding Topology

Provides an overview of how to define cluster topology.

The MapR software uses node topology to determine the location of replicated copies of data. Node topology describes the locations of nodes in a cluster. You can define the cluster topology by specifying a topology for each node in the cluster. Use topology to group nodes by rack or switch, to provide a hint as to how data should be replicated to protect against data loss or unavailability because of a switch or rack failure.

In a topology, MapR distributes container copies optimally among leaf nodes. For example, in a topology such as `europe/uk/london/DC2/room4/row22`, where `row22` contains multiple racks such as `row22/rack1`, `row22/rack2`, `row22/rack3`, and so on, MapR tries to ensure that all copies of the container do not end up on the same rack (for example, `rack1`). By setting each leaf value to correspond to a physical rack, you can ensure that replicated data is distributed across racks to improve fault tolerance.

Related concepts

Setting Up Node Topology on page 805
Define node topologies for every node in the cluster.

Setting Up Volume Topology on page 915
Specifies how to use volume topology to place volumes on specific racks, nodes, or groups of nodes.

Volumes, Snapshots, and Mirrors
Describes what Snapshots and Mirrors are, and the advantages of using them for replication.

Volumes are a management entity that logically organize a cluster's data. Since a container always belongs to exactly one volume, that container's replicas all belong to the same volume as well. Volumes do not have a fixed size and they do not occupy disk space until the MapR filesystem writes data to a container within the volume. A large volume may contain anywhere from 50-100 million containers.

The CLI and REST API provide functionality for volume management. Typical use cases include volumes for specific users, projects, development, and production environments. For example, if an administrator needs to organize data for a special project, the administrator can create a specific volume for the project. The MapR filesystem organizes all containers that store the project data within the project volume. A cluster can have many volumes.

The MapR filesystem creates a name container for each volume. The name container stores the volume's namespace and file chunk locations, along with inodes for the objects in the filesystem. The filesystem stores the metadata for files and directories in the name container, which is updated with each write operation.

The first 64KB of each file in a volume is written to the name container. Data beyond 64KB is written to data containers. Data containers are created only when the file or table data goes above 64KB. Each name or data container is associated with only one volume; volumes may have many associated data containers, but only one name container.

Local volumes are confined to one node, and are not replicated. Local volumes are part of the cluster's global namespace, and are accessible on the path `/var/mapr/local/<host>`.

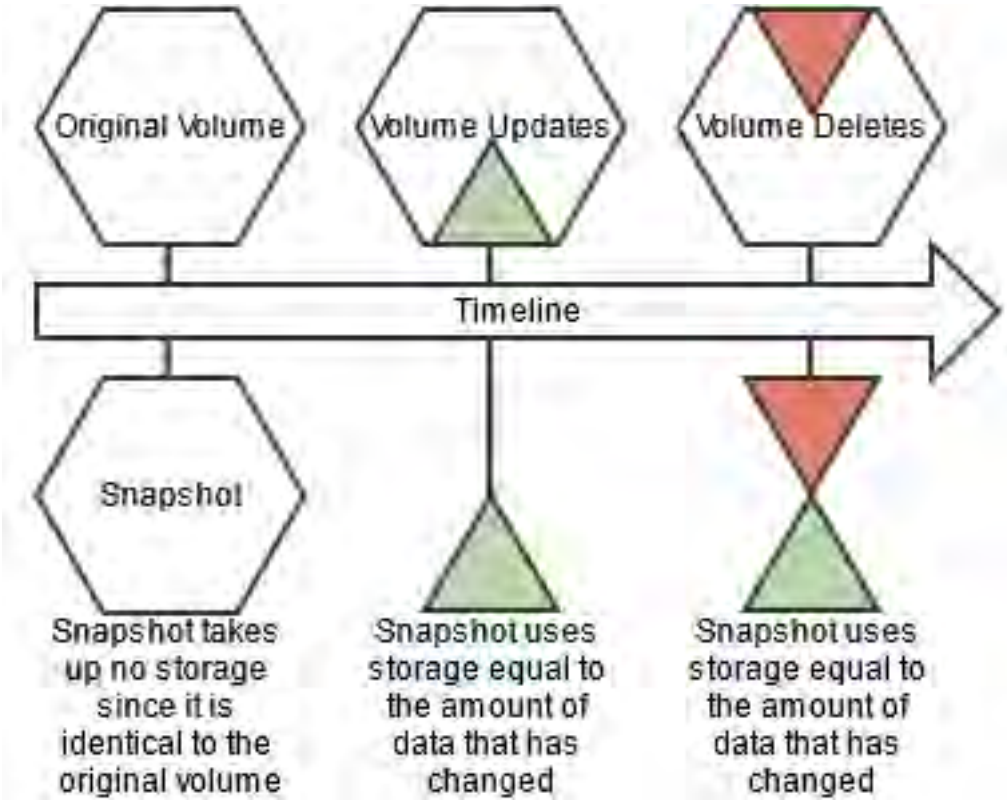
On a cluster with an Enterprise Edition or Enterprise Database Edition license, you can create a special type of volume called a mirror, a local or remote read-only copy of an entire volume. Mirrors are useful for load balancing or disaster recovery. You can also create a snapshot, an image of a volume at a specific point in time. Snapshots are useful for rollback to a known data set.

On a cluster, you can create a tenant share, or volume for tenant users. A tenant share is an isolated space where you can set different policies, quotas, and access privileges for specific users/hosts (referred to as tenants). This allows each tenant to own its own copy of storage space, users, data security, administration, and other such specifications. For more information, see Multitenancy on File System on page 488.

Snapshots

Snapshots enable you to roll back to a known good data set and recover data always in case of data corruption or accidental deletions, without the help of storage administrators. A snapshot is a read-only image of a volume that provides point-in-time recovery. Snapshots only store changes to the data present in the volume, and as a result make extremely efficient use of the cluster's disk resources. Snapshots preserve access to historical data, and protect the cluster from user and application errors. You can create a snapshot manually, or automate the process with a schedule. Snapshots are stored in the `.snapshots` directory. You can always view snapshots from this directory.

The following image represents a mirror volume, and a snapshot created from a source volume:



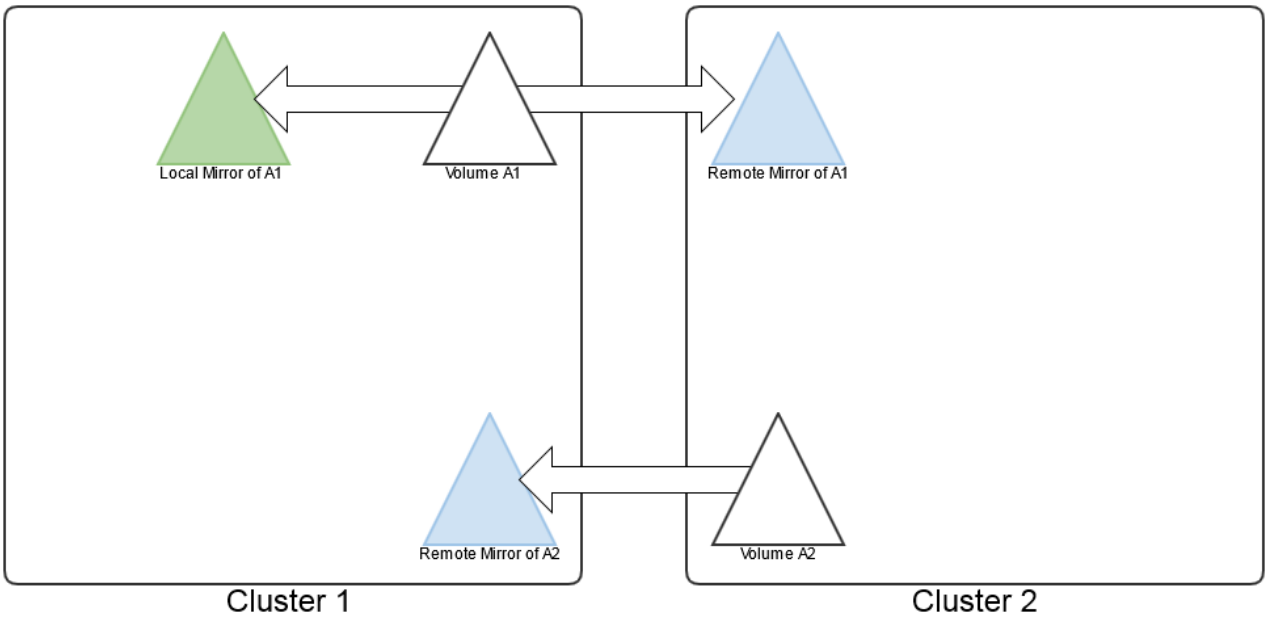
New write operations on a volume with a snapshot are redirected to preserve the original data. Snapshots only store the incremental changes in a volume’s data from the time the snapshot was created. The storage used by a volume's snapshots does not count against the volume's quota.

Mirror Volumes

MapR provides built-in mirroring to set recovery time objectives and to automatically mirror data for backup. You can create local or remote mirror volumes to mirror data between clusters, data centers, or between on-premise and public cloud infrastructures.

Mirror volumes are read-only copies of a source volume. You can control the schedule for mirror refreshes from the Control System or with the command-line tools. You can create local (on the same cluster) or remote (on a different cluster) mirror volumes from the Control System, or from the command line.

last-updated: Apr 11, 2022



When you create a mirror volume, the MapR filesystem creates a temporary snapshot of the source volume. The mirroring process reads content from the snapshot into the mirror volume. The source volume remains available for read and write operations during the mirroring process. The initial mirroring operation copies the entire source volume. Subsequent mirroring operations only update the differences between the source volume and the mirror volume.

Mirror volumes can be promoted to read-write volumes. The main use case for this feature is to support disaster-recovery scenarios in which a read-only mirror needs to be promoted to a read-write volume so that it can become the primary volume for data storage. In addition, read-write volumes that were mirrored to other volumes can be made into mirrors (to establish a mirroring relationship in the other direction). You can also convert read-write volumes back to read-only mirrors.

Related concepts

Understanding Replication on page 454
Describes how replication works, and how to configure the replication factor.

Types of Volumes

Lists the various types of volumes.

This glossary explains the different types of volumes.

Term	Definition

NC Standard Volume	A non-convertible (NC) standard volume is a volume with read-write capabilities, created *before* MapR version 4.0.2. These volumes cannot be converted to standard mirror volumes. If this volume type is designated as a source volume when a mirror volume is created, the mirror volume will be a NC mirror volume.
	A NC standard volume is designated as type 0 in the output of the `volume info` command. For example:
	`maprcli volume info -name oldrw`
	lists
	`"mirrortype":0`

last-updated: Apr 11, 2022

Term	Definition
Standard Volume	A standard volume is a read-write volume created as of MapR version 4.0.2. A standard volume can be converted from read-write to mirror (read-only). If a mirror is created from this type of volume, the mirror can be promoted to a read-write volume. A standard volume is designated as type `rw` on the command line. For example: `maprcli volume create -name volA -path /testvol -type rw`

NC Mirror Volume	A non-convertible read-only mirror volume is a volume created *before* MapR version 4.0.2. This volume type cannot be promoted to a read-write volume, and can only be created from a NC standard volume.
	A NC mirror volume is designated as type 1 in the output of the `volume info` command. For example:
	`maprcli volume info -name oldmirror`
	lists
	`"mirrortype":1`

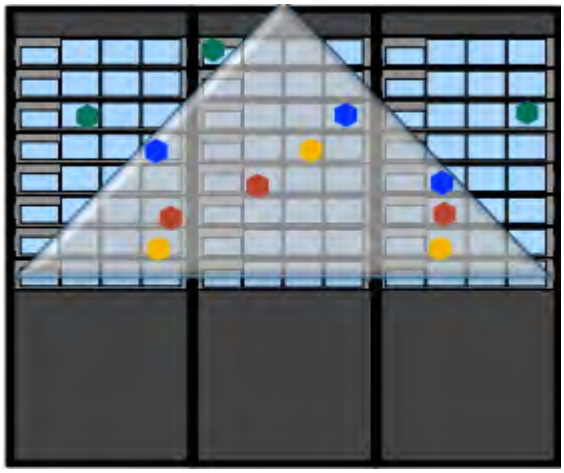
Standard Mirror	Standard mirror is a mirror volume that starts as a read-only volume, and can be promoted to a read-write volume. A standard mirror volume is designated as type `mirror` on the command line and can only use a standard volume as its source. For example: `maprcli volume create -name volB -path /mirvol -type mirror -source volA`

Volume Topology
Describes what volume topology is, and the topology of replicas and mirrors.

The topology describes the locations of nodes and racks in the cluster. Volume topology is based on node topology. You define volume topology after you define node topology. When you set up node topology, you can group nodes by rack or switch. MapR File System uses node topology to determine where to replicate data for continuous access to the data in the event of a rack or node failure.

A volume's topology defines which racks or nodes a volume includes.

The following image represents a volume that spans a cluster:



last-updated: Apr 11, 2022

Topology of Local Volume Replicas

The primary copies for containers of local volumes are placed on the local node. The nodes for the replica copies for containers of local volumes are chosen in the following order:

1. Use a topology that is explicitly specified for replicas during volume creation or modification.
2. Use the relative path for replicas of local volumes if the configuration parameter specifies such a path.
3. Use the default volume topology.

See Setting the Topology for Local Volume Replicas on page 916, Creating Replicas of Local Volumes in Custom Topology Using the CLI on page 917, and Setting Default Volume Topology Using the CLI on page 917.

Mirror Volume Topology

When the root volume on a cluster is mirrored, the source root volume contains a writable volume link, `.rw` that points to the read/write copies of all local volumes. In that case, the mount path `/` refers to one of the root volume's mirrors, and is read-only. The mount path `/.rw` refers to the source volume, and is read/write.

A mount path that consists entirely of mirrored volumes refers to a mirrored copy of the specified volume. When a mount path contains volumes that are not mirrored, the path refers to the target volume directly. In cases where a path refers to a mirrored copy, the `.rw` link is useful for navigating to the read/write source volume.

Sample Volume Topology with Mirrors

The following example shows a volume topology with mirrors:

For the four volumes `/`, `a`, `b`, and `c`, the following table indicates the volumes referred to by example mount paths for particular combinations of mirrored and not mirrored volumes in the path:

/	a	b	c	This Path	Refers To This Volume...	Which is...
Mirrored	Mirrored	Mirrored	Mirrored	`/a/b/c`	Mirror of c	Read-only
Mirrored	Mirrored	Mirrored	Mirrored	`/.rw/a/b/c`	c directly	Read/Write
Mirrored	Mirrored	*Not Mirrored*	Mirrored	`/a/b/c`	c directly	Read/Write
Mirrored	Mirrored	*Not Mirrored*	Mirrored	`/a`	Mirror of a	Read-only
Not Mirrored	Mirrored	Mirrored	Mirrored	`/a/b/c`	c directly	Read/Write

Authorization with Volumes: Intelligent Policy Management

Describes methods to manage volume permissions.

The MapR filesystem uses volumes as a unique management entity. A volume is a logical unit that you create to apply policies to a set of files, directories, tables, and sub-volumes. You can create volumes for each user, department, or project. Mirror volumes and volume snapshots provide data recovery and data protection functionality.

Volumes can enforce disk usage limits, set replication levels, establish ownership and control permissible actions, and measure the cost generated by different projects or departments. When you set policies on a volume, all files contained within the volume inherit the same policies set on the volume. Other Hadoop distributions require administrators to manage policies at the file level.

You can manage volume permissions through one of the following:

- Access Control Lists (ACLs) in the Control System or from the command line. ACLs can be used to control administrative access to volumes.
- Access Control Expressions (ACEs) in the Control System or from the command line. ACEs can be used to control data access using boolean expressions.

You can also set read, write, and execute permissions on a file or directory for users and groups with ACEs and standard UNIX commands, when that volume has been mounted through NFS, or using standard `hadoop fs` commands.

Mirror Volumes
Provides a synopsis of what mirror volumes are and the mirroring process.

Creating a mirror volume is similar to creating a normal read/write volume. However, when you create a mirror volume, you must specify a source volume from which the mirror retrieves content. This retrieval is called the mirroring operation. Like a normal volume, a mirror volume has a configurable replication factor. Only one copy of the data is transmitted from the source volume to the mirror volume. MapR volumes can only be mirrored and NOT replicated. However, the source and mirror volumes handle their own internal MapR filesystem replication (which is based on the replication factor) independently. MapR File System internally replicates source and mirror volumes independently of each other.

Note: Volume mirroring from a lower MapR version to higher MapR version is supported. For example, you can mirror volumes from a MapR 4.0.1 cluster to a MapR 5.2 cluster. However, you cannot mirror volumes from a MapR 5.2 cluster to a MapR 4.0.1 cluster.

Mirroring Process

The MapR system creates a temporary snapshot of the source volume at the start of a mirroring operation. The mirroring process reads content from the snapshot into the mirror volume. The source volume remains available for read and write operations during the mirroring process.

If the mirroring operation is schedule-based, the snapshot expires according to the value of the schedule's **Retain For** parameter. Snapshots created during manual mirroring persist until they are deleted manually.

The mirroring process transmits only the differences between the source volume and the mirror. The initial mirroring operation copies the entire source volume, but subsequent mirroring operations can be extremely fast. If the fastinodescan feature is enabled, mirroring will proceed significantly faster when there are large number of files and few changes since the last mirroring operation. The fastinodescan feature is enabled by default for all new installations, but must be manually enabled if you are upgrading from pre-5.2.x versions. See the Upgrade Guide for information on enabling this feature. To determine whether the fastinodescan feature is enabled, run the following command:

```
/opt/mapr/bin/maprcli config load -json | grep
mfs.feature.fastinodescan.support
```

To use the fastinodescan feature on converted or promoted volumes, mirroring must be restarted from the source volume after converting volume from mirror to read-write and vice versa.

The mirroring operation never consumes all available network bandwidth, and throttles back when other processes need more network bandwidth. The server sending mirror data continuously monitors the total round-trip time between the data transmission and arrival, and uses this information to restrict itself to 30% of the available bandwidth (continuously calculated). If the network or servers anywhere along the entire path need more bandwidth, the sending server throttles back automatically. If more bandwidth opens up, the sender automatically increases how fast it sends data. Mirror throttling can be disabled so that all available bandwidth is devoted to mirror operations. See Disabling Mirror Throttling for details.

During the copy process, the mirror is a fully-consistent image of the source volume. Mirrors are atomically updated at the mirror destination. The mirror does not change until all bits are transferred, at which point

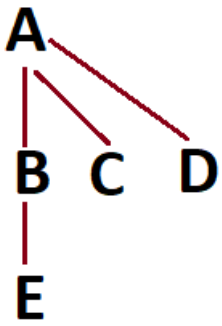
all the new files, directories, blocks, etc., are atomically moved into their new positions in the mirror-volume. The previous mirror is left behind as a snapshot, which can be accessed from the .snapshot directory. These old snapshots can be deleted on a schedule.

Mirroring is extremely resilient. In the case of a network partition, where some or all of the machines that host the source volume cannot communicate with the machines that host the mirror volume, the mirroring operation periodically retries the connection. Once the network is restored, the mirroring operation resumes.

Altering Mirror Relationships

You can use the volume modify on page 2005 command to change mirror relationships. You can change the mirror relationship to any of the volumes (either `rw` or `mirror`) that have the same mirror root, as the mirrored volume.

For example, consider the mirror tree:



Volume E mirrors volume B. However, if volume B becomes unavailable, you can set either volume A, or volume C, or volume D as the source of mirroring for volume E, since volume B, volume C, and volume D have the same mirror data source volume, which is volume A.

Mirror Types
Explains the available mirror types.

You can check the status of your volumes in terms of their mirror type. The `maprcli volume info` command returns the following `mirrortype` values:

mirrortype	Description	Volume upgrade required (to support promotability)
0	Old-format volume, created in an earlier release and present in the cluster after an upgrade to Version 5.0	Yes, if the volume is intended for use as a read-write mirror.
1	An old-format mirror volume whose source volume is a type 0 volume (in any MapR version). These mirror volumes cannot be upgraded.	No, not allowed. The `maprcli volume upgradeformat` command returns an error for these volumes.
2	New-format mirror volume that may be promoted to read-write (no upgrade command required).	No, not needed. These volumes are already in the new format and are promotable.
3	New-format standard volume: either created new in 5.0 or upgraded in 5.0 via the maprcli volume promote command.	No, not needed. These volumes are already in the new format and are promotable.

To check the mirror types for your volumes in Version 5.0, run the following command:

```
maprcli volume list -columns volumename,mirrortype -json
...
 {
   "volumename":"vol999",
   "mirrortype":0
  },
  {
   "volumename":"volume1",
   "mirrortype":3
  },
  {
   "volumename":"volume2",
   "mirrortype":3
  },
  {
   "volumename":"volume3",
   "mirrortype":2
  },
...
```

Local Mirroring
Describes the use of local mirror volumes. The local mirror volume and its source are present on the same cluster,

A *local mirror volume* is a mirror volume whose source is on the same cluster. Local mirror volumes are useful for load balancing or for providing a read-only copy of a data set.

You can locate your local mirror volumes in specific servers or on racks with particularly high bandwidth, mounted in a public directory separate from the source volume.

The most frequently accessed volumes in a cluster are likely to be the root volume and its immediate children. To load-balance read operations on these volumes, mirror the root volume (typically `mapr.cluster.root`, which is mounted at `/`). By mirroring these volumes, you can serve read requests from the mirrors, and distribute load across the nodes. Less-frequently accessed volumes that are lower in the hierarchy do not need mirror volumes. Since the mount paths for those volumes are not mirrored throughout, those volumes are writable.

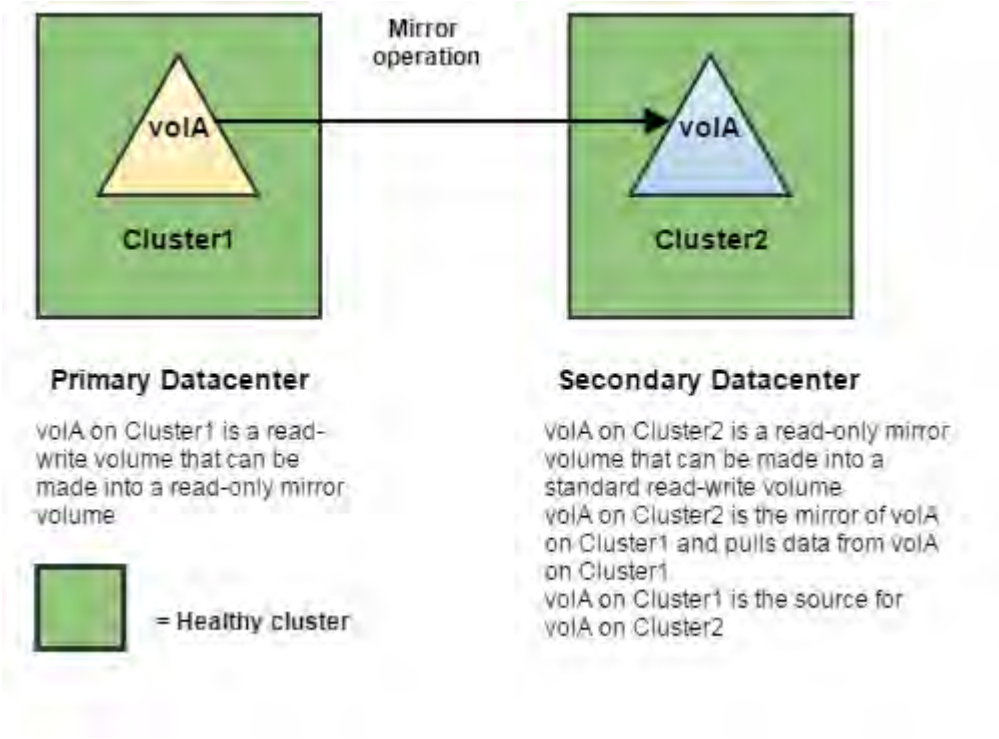
Remote Mirroring
Describes the concept and purpose of remote mirror volumes.

A remote mirror volume is a mirror volume with a source in another cluster. You can use remote mirrors for offsite backup, for data transfer to remote facilities, and for load and latency balancing for large websites. By mirroring the cluster's root volume and all other volumes in the cluster, you can create an entire mirrored cluster that keeps in sync with the source cluster.

Backup mirrors for disaster recovery can be located on physical media outside the cluster, or in a remote cluster. If disaster strikes the source cluster, you can check the time of last successful synchronization to determine the freshness of the backup (see Mirror Status).

Once data volumes are created in a primary data center, the MapR administrator creates mirror volumes in a remote secondary data center. The following diagram illustrates the mirror relationship between these two volumes:

last-updated: Apr 11, 2022



Note: When you use promotable mirrors, you must set up the volumes on the destination cluster in the same way as on the primary site. This means that volume names are the same and mount points are the same. If you use a hierarchical mounting structure (such as /A/B) on the primary site, you must recreate the same structure once you promote the mirror volumes at the secondary site.

Mirror Cascades
Describes what mirror cascades are, and their advantages.

In a cascade, one mirror synchronizes to the source volume, and each successive mirror uses a previous mirror as its source. Mirror cascades are useful for propagating data over a distance, then re-propagating the data locally instead of transferring the same data remotely again for each copy of the mirror. In the following example, the < character indicates a mirror's source:

```
/ < mirror1 < mirror2 < mirror3
```

A mirror cascade makes more efficient use of your cluster's network bandwidth, but synchronization can be slower to propagate through the chain. For cases where synchronization of mirrors is a higher priority than network bandwidth optimization, make each mirror read directly from the source volume:

```
mirror1 >   < mirror2
          /
mirror3 >   < mirror4
```

You can:

- Create a mirror cascade by setting the source volume of each mirror in the **Properties** tab of the Control System when creating a mirror volume.
- Break a mirror cascade made from existing mirror volumes by changing the source volume of each mirror in the **Properties** tab of the Control System when editing the mirror volume.

Promotable Mirrors
Explains the use of promotable mirrors for enhanced performance, data recovery, and business continuity.

In general, mirror volumes are created for the purpose of preventing or minimizing data loss. Data loss scenarios range from accidental overwrites to rack failures, to a disaster that destroys an entire data center. Mirror volumes are also used to improve performance or to make copies of data for use in other clusters without impacting production.

As of the 4.0.2 release, all new mirror volumes can be made into read-write volumes. In addition, read-write volumes that were mirrored to other volumes can be made into mirrors (to establish a mirroring relationship in the other direction). This functionality is useful in scenarios such as:

- Disaster recovery If a read-write volume with critical data goes down in a primary data center, a mirror volume in a remote data center can be made into a read-write volume in order to maintain business continuity. Later, if the primary data center comes back online, the original mirror relationship can be restored by making the new read-write volume back into a mirror volume.
- Running applications on a copy of production data
- Resynchronization (reestablishing a mirror relationship after it is broken)

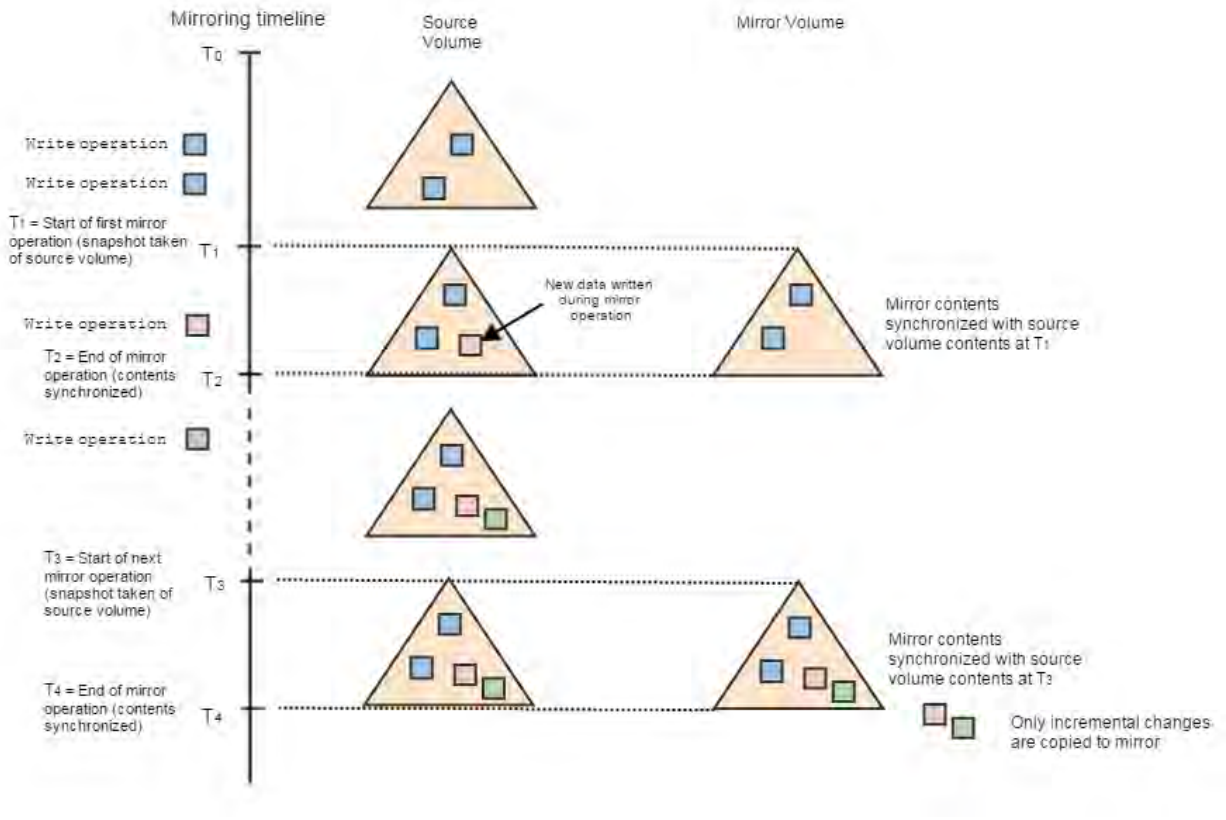
For a hands-on tutorial on promotable mirrors, check this blog.

Incorporating Mirror Volumes into a Disaster Recovery Plan
Lists the points to note when incorporating mirror volumes into a disaster recovery plan.

Mirroring critical data to a remote data center (with the ability to make mirror volumes into read-write volumes) addresses the following objectives:

- Recovery Point Objective (RPO) - the age of the files you need to recover (and how much data you can afford to lose)
- Recovery Time Objective (RTO) - how soon you need to have a working data center in order to maintain business continuity

In a typical scenario that employs remote mirrors, the contents of a source volume are mirrored to a mirror volume in a remote cluster at a frequency specified by the mirror schedule. At the start of each mirror operation, a snapshot is taken of the source volume's contents. The mirror operation takes some time to complete, and while the data is being copied from the snapshot to the mirror volume, more data is written to the source volume. This data will be captured during the next mirror operation. When each mirror operation completes, the contents of the mirror volume are identical to the contents of the source volume at the time of the snapshot. For subsequent mirror operations, only the incremental changes (additions and deletions) are copied to the mirror volume, which synchronizes its contents with the contents of the source volume at the time of the snapshot.



If the source cluster goes down, any data written to the source volume since the last successful mirror operation cannot be copied to the mirror. The amount of data lost depends on the number of write operations in the interval from the last successful mirror to the time the cluster goes down.

Factors that Affect RTO

Lists the factors that affect the Recovery Time Objective (RTO).

During a disaster, an administrator must first determine that the link between the primary data center and the secondary data center has failed. Next, the administrator begins the process of switching applications that were running on the primary data center over to the secondary data center. For write applications, the administrator begins converting mirror volumes to read-write volumes, starting with volumes that contain the most critical data. Note that read applications can run on read-only mirrors, but write applications can only run on read-write volumes.

To gauge how long it will take to switch applications from the primary data center to the secondary data center (and to set the RTO accordingly), consider these factors:

- Detection time (how long it takes to determine that the link is down between the two data centers)
- Switching time (how long it takes to switch applications from one data center to the other)
- Promotion time (how long it takes to change read-only mirror volumes to read-write volumes that can run write applications). Promotion time is based on the number of containers in a volume or across volumes.
- Whether mirror throttling is enabled (the default) or disabled (which speeds up the mirroring process)

Various factors affect the amount of data that can be recovered through the use of mirror volumes. To specify a realistic recovery point objective in your disaster recovery plan, take the following factors into account:

last-updated: Apr 11, 2022

- Mirror schedule (how often the mirror is synchronized with its source volume) - Note that the first mirror operation is a full synchronization between source and mirror volumes. Subsequent mirror operations are incremental - only the changes that occurred since the last mirror event need to be copied in order to synchronize the contents between the two volumes.
- Network link between the source volume and the mirror volume (consider the stability and quality of the link, as well as latency, throughput, and other activities across the link)

Data Tiering
Provides an overview of what tiering is, its various types, and its working.

MapR provides rule-based automated tiering functionality that allows you to seamlessly integrate with:

- Low-cost storage as an additional storage tier in the MapR cluster for storing file data that is less frequently accessed ("warm" data) in erasure-coded volume.
- 3rd party cloud object storage as an additional storage tier in the MapR cluster to store file data that is rarely accessed or archived ("cold" data).

In this way, valuable on-premise storage resources can be used for more active or "hot" file data and applications, while "warm" and/or "cold" file data can be retained at minimum cost for compliance, historical, or other business reasons. MapR provides consistent and simplified access to and management of the data.

See also: Working with Tiered Volumes on page 926

Where is data tiered?

For "warm" data, MapR allows you to offload data to specific nodes or low-cost hardware in a topology. MapR uses erasure coding to protect data on the low-cost hardware. Erasure coding also reduces the storage overhead in the range of 1.2x-1.5x. See Overview of Tiers on page 471 for more information on erasure coding.

For "cold" data, MapR allows you to easily offload your cluster data to public, private, and hybrid clouds. You can offload data to remote cloud from vendors such as Amazon AWS, Google Cloud Platform, Microsoft Azure, IBM Cleversafe, Hitachi HCP, and Minio. This allows you to tap into cloud-scale capacity.

Note: MapR supports tiering for only file and volume data; tiering of tables and streams is not supported.

MapR allows you to configure a volume at the time of volume creation for either warm or cold tier, but not both. If you do not know the type of tier to associate with the volume, you can still create a volume that is tiering-enabled and associate a specific tier later with the volume. However, volumes not enabled for tiering at the time of volume creation cannot be enabled for tiering after the volume is created. You cannot modify the type of tier associated with the volume after the volume is created.

When you create a volume and configure it for warm or cold tiering — associating a warm or cold tier, a storage policy (referred to as rule in the CLI), and an offload schedule — MapR automatically moves the data out of the volume and into the tier, and purges the data in the volume on the MapR cluster to release the disk space on the MapR cluster. However, for tiering-enabled volumes, the amount of hard quota you set is the total space allocated for the volume irrespective of the location (cluster or tier) of the volume data. Writes fail when volume disk space usage reaches the quota assigned for the volume whether or not volume data is local (on the cluster) or remote (on the tier). Also, if you want to recall volume data back to the MapR cluster, you must have the disk space in the volume equivalent to the amount of data being recalled from the tier. You can retrieve and view the disk space usage metric, including the amount of data offloaded to the tier, for a tiering-enabled volume using the Control System, the CLI, and REST API.

How frequently is data offloaded?

MapR automatically offloads data based on the criteria that you define in the storage policy for offloading data and at the frequency you specify in the schedule. MapR automatically offloads data in the volume at the frequency in the schedule only if data in the volume meets the criteria in the associated storage policy. If you do not specify a criteria, for volumes configured for:

- Erasure coding (warm tier), MapR applies a default criteria, which is a modification timestamp of 1 day, for offloading data.
- Remote archiving (cold tier), MapR does not associate a default criteria. You can use the Control System, CLI, and REST API to manually trigger an offload of volume data.

For more information, see Data Storage Policy on page 474. If you do not associate a schedule for offloading data, for volumes configured for:

- Erasure coding (warm tier), MapR automatically uses the default Automatic Tiering Scheduler, which uses internal policies to decide when to schedule the offload operation.
- Remote archiving (cold tier), MapR does not associate a default schedule. You can use the Control System, CLI, and REST API to manually trigger an offload of volume data.

Even when you manually trigger an offload, MapR offloads data only if the data meets the criteria defined in the storage policy. In addition, for warm-tier volumes, MapR offloads data only if the object (stripe) has data exceeding 90% of the object payload; if an object has data less than 90% of the object payload, the object is not offloaded and the metadata tables are not updated. For more information, see Data Offload and Purge on page 475.

What is the MAST Gateway?

The MapR automated storage tiering (MAST) Gateway acts as the centralized entry point for all the tiering operations. CLDB assigns tiering-enabled volumes to MAST Gateways for processing all tiering operations for the volume. For more information, see Overview of MAST Gateway on page 472.

How is compressed and encrypted data transferred and stored?

Data is encrypted during transfer to ensure security of data if the cluster is a secure cluster and if wire-level security is enabled for the volume. In addition, stored data is encrypted if:

- The warm-tier volume is enabled for data-at-rest encryption (`dare`).
- The cold-tier volume is enabled for tier encryption (`tierencryption`).

Data in the volume is transferred and stored as-is, compressed or uncompressed, on the tier. You can set up replication, snapshots, and mirror volumes for tiering-enabled volumes. See Data Replication, Snapshots, Mirroring, Auditing, and Metrics Collection on page 478 for more information.

How are reads, writes, and deletes handled?

When a client tries to read offloaded data, MapR processes the read request of the warm-tiered and cold-tiered standard and mirror volume data differently. Similarly, when a client writes to a tiered volume, MapR processes appends and overwrites differently. See Data Reads, Writes, and Recalls on page 481 for more information.

Data, once offloaded, is purged on the MapR cluster to release the disk space. When you delete an entire file, part of a file, or a snapshot, corresponding objects are removed from the tier also. See Data Compaction on page 485 for more information.

Enabling Tiering

To enable tiering, see Enabling Tiering on page 957

.

Overview of Tiers

Describes what warm and cold tiers are.

MapR considers data that is active and frequently accessed as "hot" data and data that is rarely accessed as “warm” or “cold” data. The mechanism used to store "hot" data is referred to as the hot-tier (or the MapR cluster), the mechanism used to store "warm" data is referred to as the EC-tier (or low-cost storage alternative on the MapR cluster), and the mechanism to store "cold" data is referred to as the cold tier (or low-cost storage alternative on the cloud). Hot, warm, and cold data is identified based on the rules and policies set by the administrator.

Data starts off as hot when it is first written to local storage (on the MapR cluster). It becomes warm or cold based on the rules and policies the administrator configures. Data can then be set up to be automatically offloaded using the MapR automated storage tiering (MAST) Gateway service to the erasure coded volume on the low-cost storage alternative on the MapR cluster (warm tier) or to the low-cost storage alternative on the 3rd party cloud object store (cold tier) like AWS S3.

Warm Tier	On the MapR cluster, every volume enabled for erasure coding (or warm tiering) acts as a "front-end" volume and has a corresponding hidden erasure coded (or EC) volume in the specified topology (of the low-cost storage alternative). Erasure coding (EC) is a data protection technique where data is broken into many fragments (or *m* pieces) and encoded with some extra redundant fragments (or *n* pieces) to guard against disk failures. That is, for volumes configured for erasure coding, file data in the volume is broken into many fragments (or *m* pieces) and encoded with pre-configured number of redundant fragments (or *n* pieces). In the event of disk failure, any *m* piece can be used to get back the original file. See Erasure Coding Scheme for Data Protection and Recovery on page 926 for more information. Although you write to and read from the front-end volumes, the front-end volume is akin to a staging area, where volume’s data is held on demand. Data written to a volume is periodically moved to the back end erasure coded volume, releasing the disk space for the front-end volume on the filesystem and providing the space savings of erasure coded volumes. Data in the front-end volume is moved to the corresponding erasure coded volume based on an offload schedule. The front-end volume holds only small amount of required data, and data is shuffled between the front-end volume and the corresponding erasure coded volume as required. See Data Reads, Writes, and Recalls on page 481 for more information. There is also a visible tier-volume on the MapR cluster for storing the metadata associated with the volume. When you create a warm tier, the tier volume named `mapr.internal.tier.<tiername>` is by default created in the `/var/mapr/tier` path. When you create a warm-tier volume using the `ecenable` parameter or the Control System, a warm tier is automatically created and the corresponding tier volume named `mapr.internal.tier.autoec.<volName>.<creationTime>` is, by default, created in the `/var/mapr/autoectier` path.

last-updated: Apr 11, 2022

	While three-way replicated regular volumes require 3 times the amount of disk space of the regular volume, erasure coded volumes reduce the storage overhead in the range of 1.2x-1.5x. On the MapR cluster, only the metadata of the volume in the namespace container is 3-way replicated. You can create one warm tier per volume using the Control System, the CLI, and REST API or create and associate multiple volumes with different erasure coding schemes with the same warm tier using the CLI and REST API (only). You cannot associate the same warm tier with multiple volumes using the Control System.
Cold Tier	On the MapR cluster, every cold tier (referred to as remote target in the Control System) has a bucket on the 3rd party cloud store where volume data is offloaded based on the policy configured by the administrator. Volume data in 64KB data chunks is packed into 8MB sized objects and offloaded to the bucket on the tier and the corresponding volume metadata is stored in a visible tier-volume as MapR Database tables on the MapR cluster. During writes and reads, volume data is recalled to the MapR cluster if necessary. Data written to the volume is periodically moved to the remote target, releasing the disk space on the filesystem. See Data Reads, Writes, and Recalls on page 481 for more information. Data stored on the MapR cluster requires 3 times the amount of disk space of the regular volume on premium hardware due to replication (default being 3). After offloading to the cloud, the space used by data (including data in the namespace container) in the volume on the MapR cluster is freed and only the metadata of the volume in the namespace container is 3-way replicated on the MapR cluster. There is also a visible tier-volume on the MapR cluster for storing the metadata associated with the volume. When you create a cold tier, the tier volume named `mapr.internal.tier.<tierName>` is by default created in the `/var/mapr/tier` path. A directory/folder for the volumes associated with the tier, identifiable by `volumeid`, is created under the path after the first offload of data from the volume to the tier. You can create one tier per volume or create and associate multiple volumes with the same tier using the Control System, the CLI, and REST API.

See also: Managing Tiers on page 957

Overview of MAST Gateway
Describes the role of the MAST Gateway for operations on tiered storage.

The MAST Gateway can be installed on specific hosts on the MapR cluster with access to the tier. The MAST Gateway acts as the centralized entry point for all the operations that need to be performed on the tiered storage including the following:

Warm Tier	For volumes configured for warm tiering, the MAST Gateway:

- Identifies files in the volume that are ready to be offloaded, fetches data corresponding to these files from MapR File System, and packs this data for offload. It:
  - Identifies and fetches the data to offload.

    It handles both compressed and uncompressed data. Compressed data from the file server is transferred and stored as-is on the warm tier.
  - Creates stripes based on the erasure coding scheme.

    For example, for an erasure coding scheme of 4+2, the stripe depth would be 6x4MB=24MB.
  - Manages statistics on the amount of data offloaded.
  - Prepares a corresponding metadata on the MapR cluster for the data.

    The MAST Gateway stores the metadata in MapR Database tables in a separate volume associated with the tier.
- Tracks invalid data and deletes stripelets that are completely invalid.
- Fetches data from the tier.
- Recalls whole volume from the tier to the MapR cluster.

Cold Tier

For volumes configured for cold tiering, the MAST Gateway:

- Identifies files in the volume that are ready to be offloaded, fetches data corresponding to these files from MapR File System, and packs this data for offload. It:
  - Identifies and fetches the data to offload and creates objects (including creating new buckets) in the storage tier for the data.
  - Manages statistics on the amount of data offloaded.
  - Updates metadata references for remote access.
- Tracks invalid data and deletes objects that are completely invalid.
- Fetches data from the tier. It:
  - Handles both compressed and uncompressed data. If data on file server is compressed, the compressed data is not uncompressed/re-compressed during offload or recall. Compressed data from the file server is transferred and stored as-is on the cold tier.

last-updated: Apr 11, 2022

    - Ensures that data is decrypted, if it is encrypted, before forwarding it to MapR File System.
- Recalls whole volume from the tier to the MapR cluster.

The MAST Gateway uses curl to transfer data to and from S3 cloud storage.

The MAST Gateway uses an exponential backoff retry mechanism. If curl fails to connect to the S3 destination even after a minute of trying, or if curl fails to fetch data from the S3 destination even after 5 minutes of being connected, the MAST Gateway declares a failure and reports it to the CLDB. The CLDB then reschedules the (vol) tasks after 30 minutes.

The MAST Gateway sends heartbeat messages to CLDB every 5 seconds. CLDB manages the discovery and a minimal global state of the MAST Gateway service. CLDB also manages the volumes and any policy configurations on the volumes. When a volume is assigned to a gateway, the volume remains assigned to the gateway across CLDB, Gateway, and cluster restarts. Volumes are assigned evenly to gateways and CLDB balances the gateway load. For more information, see Balancing Gateway Load on page 1264.

By default, the MAST Gateway uses 16 threads for volume and file offload and recall operations and another 16 threads for handling internal operations and other operations such as reads (which triggers automatic recall requests), writes, etc. Each thread processes uses the curl library to offload or recall a container (associated with a volume). Each MAST Gateway can process one or more volumes (and associated containers) simultaneously depending on the number of threads available for processing the containers associated with the volumes. Each volume is assigned to a MAST Gateway for a tiering operation irrespective of the number of containers associated with the volume.

When a MAST Gateway goes down during a volume-level offload, CLDB does not immediately reassign all the volumes assigned to that MAST Gateway to other gateways. CLDB waits for some time to allow the MAST Gateway to come back up and send heartbeat again; CLDB re-assigns volumes with pending tasks to other gateways if the MAST Gateway does not come back up again. All other volumes are redistributed when the gateway balancer runs again. On the other hand, if the MAST Gateway comes back up again, the volumes remain assigned to the MAST Gateway. The load on the MAST Gateways is rebalanced when the balancer runs again. See Balancing Gateway Load on page 1264 for more information. MAST Gateways use transactions to ensure that all the updates are consistent, and that any new gateway can pick up exactly from where the old gateway left.

If a MAST Gateway goes down during a file-level offload and if the offload was triggered using:

- The `hadoop` command, CLDB reassigns the volume to another MAST Gateway.
- The MapR CLI, REST API, or dot interface, CLDB does not reassign to another MAST Gateway.

See also: Managing the MAST Gateway on page 1259

Data Storage Policy
Provides an overview of creating storage policies and formulating rules to offload data.

You can configure a storage policy (or rules) for data at the volume level. The storage policy simplifies the lifecycle management of data in the volume including automated migration of files to low-cost storage alternatives. The policy can contain rules for files that have a well-defined lifecycle or for files you want to switch to different storage tiers during their lifecycle.

You can specify the rules, at the volume level, to selectively identify files to offload (such as file size, file owner, and file modification time), the schedule for offloading the data (for example, 2 months after file modification), and the settings for storing (such as the location and credentials for the tier) and recalling the offloaded data. You can configure one rule per volume using the CLI or REST API. You can also associate a schedule to automatically offload data at scheduled intervals based on the associated rules.

See Managing Storage Policies on page 970 for more information.

Data Offload and Purge
Describes the process of offloading data to warm and cold tiers, and purging data from storage pools.

The MAST Gateway service drives the offload process. On volumes configured for warm or cold tiering, the CLDB notifies the MAST Gateway service to start the offload based on either of the following:

- The schedule set at the volume level for offload.
- The request triggered by the client (through the Control System, the CLI, or REST API).

The MAST Gateway service then scans the files in the volume and starts the offload by picking the files that meet the criteria in the rule associated with the volume.

Offloading Data to the Warm Tier

On volumes configured for warm tiering, the MAST Gateway service detects the files that meet the criteria in the configured rules, collects data to offload from the read-write containers of the front-end volume on the MapR file system, and:

1. Creates objects based on the erasure coding scheme.

   For example, for an erasure coding scheme of 4 + 2 (6) and stripe depth of 4 MB, which is the default, the object size is 4 x 4 MB = 16 MB and the stripe length is 6 x 4 MB = 24 MB. When offloading a file, the file must contain data exceeding 90% of the object payload to qualify for offload. When offloading a volume, an object can contain multiple small files. Objects that fall below the threshold are not offloaded.

   Note: Data is broken into many fragments (or *m* pieces) and encoded with some extra redundant fragments (or *n* pieces) to guard against disk failures.

2. Prepares a corresponding metadata on the MapR cluster for the data.

   The MAST Gateway stores the metadata in MapR Database tables in a separate volume associated with the tier.

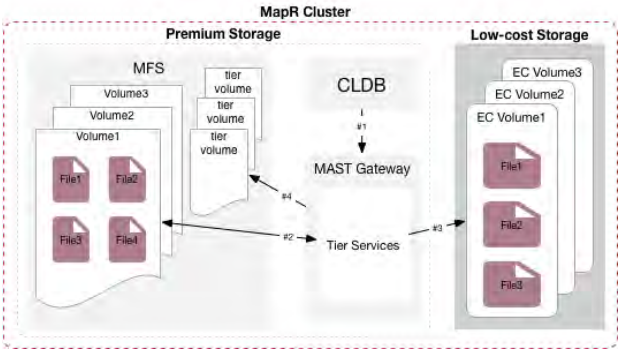
3. Offloads the objects to the tier.

   Note: If an object contains less than 90% of the object payload, the object is not offloaded and the metadata table is not updated; the volume might have local data. However, the MAST Gateway will report successful job completion.

   Data is offloaded to the tier in the same state, compressed or uncompressed, as was stored in the front-end volume. If data encryption is enabled on the front-end volume (using the `dare` parameter), data is encrypted during and after offload to the erasure-coded volume.

The following illustration shows the CLDB notifying the MAST Gateway service to start the offload (#1) and the MAST Gateway fetching data from the front-end volume (#2), offloading the data to the associated erasure-coded volume (#3), and then writing metadata to the tier volume associated with the front-end volume (#4).

last-updated: Apr 11, 2022



When offloading files, small files might not always qualify for offload because objects that contain less than 90% of the object payload do not qualify for an offload. For example, suppose the files in a volume enabled for warm-tier erasure coding scheme 4 + 2 are 2 MB each. If the object size for the volume is 16 MB (4 x 4 MB), an individual file of 2 MB is less than 90% of the object payload. In this case, a 2 MB file does not qualify for offload by itself. Similarly, portions of a large file might not qualify for offload. For example, suppose a file in the volume enabled for warm-tier erasure coding scheme 4 + 2 is 20 MB. The object size for the volume is 16 MB (4 x 4 MB), and the individual file of 20 MB exceeds the upper limit of the object payload. Portions of data in the file are offloaded, and up to 4 MB of file data might still be on the cluster.

Offloading Data to Cold Tier

On volumes configured for cold tiering, the MAST Gateway service detects the files that meet the criteria in the configured rules, collects data to offload from the read-write containers and snapshots for the volume on the MapR file system, and:

1. Packs 64 k data chunks into 8 MB-sized objects.
2. Creates the bucket on the tier (or remote target) if the specified bucket is already not present on the tier.
3. Prepares corresponding metadata on the MapR cluster for the data and creates the objects in the tier.

   The MAST Gateway stores the metadata in MapR Database tables in a separate volume associated with the tier.

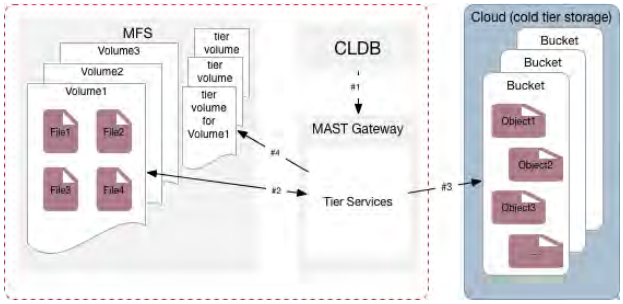
4. Offloads the data to the tier using libcurl.

   Note: Data is offloaded to the tier in the same state, compressed or uncompressed, it was stored on the MapR File System. If data encryption is enabled at the volume level (using the `tierencryption` parameter), data is encrypted during and after offload. See `volume create` or `volume modify` for more information about the parameter.

   For the offloaded data, the unique object IDs are generated using a combination of cluster ID, volume ID, and a unique sequence of numbers. For example, the names of the objects in S3 can look similar to the following:

   ```
   0.b258a07.86e.1
   0.b258a07.86a.1
   0.b258a07.86c.1
   ```

   The following illustration shows the CLDB notifying the MAST Gateway service to start the offload (#1) and the MAST Gateway fetching data from the volume (#2), offloading the data to the third-party storage alternative (#3), and then writing metadata to the tier volume associated with the volume (#4).



The MAST Gateway service notifies the CLDB when the offload operation completes successfully. Entire volumes can be moved from "hot" to "warm" tier or "hot" to "cold" tier and vice-versa on demand by using CLIs. For each offloaded volume, the MapR File System stores only the metadata for the offloaded data in a volume on the hot tier.

Note: If the offload fails, an alarm, VOLUME_ALARM_OFFLOAD_FAILURE, is raised. Check the log file for more information about the error. For log information, see Enabling Debug Logging for MAST Gateway on page 1266. For some errors, CLDB tries to offload the data again after a brief wait. For more information, see Retrying Failed Operation on page 939.

See also: Offloading a Volume to a Tier on page 933 and Offloading a File to a Tier Using the CLI and REST API on page 1012

Purging Data on MapR File System

While offloading, metadata is written to the MapR Database table in a separate volume associated with the tier, and the data blocks are removed from the storage pool in the hot-tier. An offload is considered successful only when data on all active replicas has been purged (or removed from the storage pool to release the disk space on the MapR file system) in the hot-tier. When you offload data at the file level, all data, including recalled data, is immediately purged from the hot-tier. For more information, see Data Compaction on page 485.

last-updated: Apr 11, 2022

Data Replication, Snapshots, Mirroring, Auditing, and Metrics Collection
Provides an overview of what Data Replication, Snapshots, Mirroring, Auditing, and Metrics Collection are.

Replication

Data from one of the replica containers is first offloaded and then the data in all the replica containers is purged. MapR File System only stores the metadata after data is offloaded. The offload is considered successful only when data on all active replicas have been purged (or removed from the storage pool to release the disk space on the MapR filesystem). If, during the offload, the node on which one of the replicas reside is down, the data on that container is purged once the node comes back up.

In the tiering architecture, although data is moved to the storage tier, the namespace of the volume continues to be 3-way replicated. So, the metadata related to namespace container has 3x cost.

The offloaded replica containers are recalled if/when the whole volume is recalled. When a replica is reinstituted to the cluster as a result of a recall operation, a re-synchronization happens to bring all the replicas up to date from the designated master container.

Note: The offload and recall settings on the master container are applicable to the replica containers as well.

Snapshots

You can associate a snapshot schedule with tiering-enabled volumes. When the data in the volume is offloaded, associated snapshots are also offloaded and MapR File System only stores the metadata. If the whole volume is recalled, the snapshots are also recalled to the MapR filesystem. When offloading recalled snapshots, the rules for data offload apply to snapshots as well.

Note: You may experience latencies when accessing snapshots associated with offloaded data.

Mirroring

You can create tiering-enabled source volumes and associate them with tiering-enabled mirror volumes. You cannot associate tiering-enabled mirror volumes with standard volumes that are not tiering-enabled and vice versa. Only homogeneous combination of mirror and standard volumes are supported; heterogeneous combination of mirror and standard volumes are **not** supported.

Note: Both mirror volume and source volume data can be set up to be offloaded to the same tier (that is the same cold tier) or different tiers (that is different cold tiers). MapR does not require the source and mirror volume to be configured to use the same tier or have the same tier settings. Warm tier enabled volumes can have the same tier settings; however, the volume's tier only stores the meta data and data in each volume is offloaded to an associated back-end volume.

When a synchronization of the tiering-enabled mirror volume with the (local or remote) tiering-enabled source volume is triggered (either manually or automatically based on a schedule), the mirror volume synchronizes with the source volume if source volume data is local (and not yet tiered). On the other hand, if the source volume data is tiered, the tiering-enabled mirror volume synchronizes with the tiered data fetched by the MAST Gateway that is assigned to the source volume. Incremental changes in the mirror volume are offloaded based on the offload rules associated with the tiering-enabled mirror volume.

Using Tiering-Enabled Mirror Volumes for Disaster Recovery	You can create a secondary, cost optimized disaster recovery cluster for a primary three-way replicated cluster. To do this, create two clusters — a primary tiering-enabled cluster with no active schedule to automatically offload data and an associated secondary cluster where primary cluster data is mirrored and then aggressively offloaded to the tier. While the primary or source cluster continues to be three-way replicated, if the the secondary, disaster recovery cluster data is:

last-updated: Apr 11, 2022

- Erasure coded (warm tier), it provides space savings in the range of 1.2x-1.5x.
- On a third-party cloud storage (cold tier), it can be three-way replicated on a low-cost storage alternative.

In case of a disaster, you can recall data from the tier to the MapR cluster.

Note: If you promote a tiering-enabled mirror volume during an offload or recall operation of the data associated with the mirror volume, the offload or recall operation is aborted and the mirror volume is converted to a read-write volume; the `tierjobstatus` command for the offload or recall job shows AbortedInternal status.

Auditing

The MapR audit feature lets you log audit records of cluster-administration operations and operations on the data in the volume. Scheduled (and automatically triggered) tiering operations such as offload and compaction are not audited. However, if auditing is enabled at the cluster level, the manually triggered volume-level tiering operations such as offload, recall, abort, etc. are audited in the CLDB audit logs. For example, you can see a record similar to the following in the `/opt/mapr/logs/cldbaudit.log.json` file for `volume offload` on page 2023 command:

```
{"timestamp":
{"$date":"2018-06-07T15:34:28.580Z"},"resource":"vol1","operation":"volumeOf
fload","uid":0,"clientip":"10.20.30.40","status":0}
```

If auditing is enabled for data in the tiering-enabled volume and files within, file-level tiering operations such as offload, recall, etc. triggered using the REST API, hadoop, and dot-interface are audited in the FS audit logs (`/var/mapr/local/<hostname>/audit/5661/FSAudit.log-<*>.json` file).See Auditing Data Access Operations on page 698 for the list of file-level tiering operations that are audited. You can selectively enable or disable auditing of these operations. See Selective Auditing of MapR File System, MapR Database Table, and MapR Event Store For Apache Kafka Operations Using the CLI on page 761 for more information. For example, you can see records similar to the following in the `/var/mapr/local/<hostname>/audit/5661/FSAudit.log-<*>.json` file for `file offload` on page 1659 command:

```
/mapr123/Cloudpool19//var/mapr/local/abc.sj.us/audit/5660/
FSAudit.log-2018-09-12-001.json:1:{"timestamp":
{"$date":"2018-09-12T05:47:04.199Z"},"operation":"FILE_OFFLOAD","uid":0,"ipA
ddress":"10.20.35.45","srcFid":"3184.32.131270","volumeId":16558233,"status"
:0}
```

Both the `tier rule list` on page 1892 and `tier list` on page 1880 commands are audited in the `/opt/mapr/logs/cldbaudit.log.json` file as well as the `/opt/mapr/mapr-cli-audit-log/audit.log.json` file. The record in the audit log might look something similar to the following:

```
{"timestamp":
{"$date":"2018-06-13T09:15:24.004Z"},"resource":"cluster","operation":"offlo
adRuleList","uid":0,"clientip":"10.10.81.14","status":0}
{"timestamp":
{"$date":"2018-06-13T09:14:42.304Z"},"resource":"cluster","operation":"tierL
ist","uid":0,"clientip":"10.10.81.14","status":0}
```

When auditing operations like `tierjobstatus` and `tierjobabort`, the coalesce interval set at the volume level is not honored. You may see multiple records of the same operation from the same client in the log.

Read requests processed using cache-volumes or erasure-coded volumes are not audited because when the file is accessed, the request first goes to the front-end volume and the operation is audited there. The audit record contains the ID of the front-end volume (volid) and primary file ID (fid). However, the write to the cache-volume for a volume-level recall of data is audited in the audit logs on the file server hosting the cache-volume with the primary file ID (fid). The write to the cache-volume for a file-level recall of data is not audited.

In addition, you can enable auditing of offload and/or recall events at both the volume and file levels by enabling auditing for `filetieroffloadevent` and `filetierrecallevent` at the volume level. By default, auditing is disabled for `filetieroffloadevent` and `filetierrecallevent`. If you enable auditing for `filetieroffloadevent` and `filetierrecallevent` using the `dataauditops` parameter with the `volume create` on page 1931 or `volume modify` on page 2005 command, the following are audited in the FS audit log:

- For `filetieroffloadevent`, files offloaded by running the `file offload` on page 1659 command or (only) files purged on MapR filesystem after running `volume offload` on page 2023 command.
- For `filetierrecallevent`, files recalled by running the `file recall` on page 1660 or `volume recall` on page 2025 command.

For example, you can see a record similar to the following in the `/var/mapr/local/<hostname>/audit/5661/FSAudit.log-<*>.json` file if auditing is enabled at the volume-level for `filetieroffloadevent`:

```
abc.sj.us/audit/5661/FSAudit.log-2018-06-07-001.json:{"timestamp":
{"$date":"2018-06-07T07:27:58.810Z"},"operation":"FILE_TIER_OFFLOAD_EVENT","
uid":2000,"ipAddress":"1}
```

For more information:

- Auditing in MapR on page 690
- Managing Auditing on page 757

Collecting Metrics

If volume metrics collection is enabled on the tiering-enabled volume, metrics for all read and write operations on the tiered volume are logged in the metrics log. For example, you can see a record similar to the following in the metrics log file:

```
{"ts":1534960230000,"vid":248672388,"RDT":0.0,"RDL":0.0,"RDO":0.0,"WRT":3636
22.7,"WRL":7209.0,"WRO":2580.0}
{"ts":1534960250000,"vid":248672388,"RDT":363686.7,"RDL":2856.0,"RDO":2847.0
,"WRT":0.0,"WRL":0.0,"WRO":0.0}
```

Tiering-related operations do not generate metrics records. That is, volume and file level offload, recall, and abort operations are not logged in the metrics log. However, the volumes created to support tiering (such as the cache-volume, the metadata volume, and the erasure-coded volume) have metrics collection enabled and the metrics records for these volumes are logged with the ID of the associated parent or front-end volume. That is, read operations on the the cache-volume are logged with the ID of the associated front-end volume. For example, you can see records similar to the following in the metrics log file for the volume:

```
{"ts":1534968850000,"vid":209801522,"RDT":6328.5,"RDL":161.0,"RDO":158.0,"WR
T":0.0,"WRL":0.0,"WRO":0.0}
{"ts":1534968860000,"vid":209801522,"RDT":234669.7,"RDL":5241.0,"RDO":5143.0
,"WRT":0.0,"WRL":0.0,"WRO":0.0}
```

See Enabling Volume Metric Collection on page 1301 and Collecting Volume Metrics on page 1300 for more information.

Data Reads, Writes, and Recalls
Provides a synopsis of how data is read and written to a warm or cold tier.

Once offloaded to the storage tier, data is considered to be warm or cold on the storage tier, but the data can still be accessed (read, written, and recalled).

Read of Tiered Data

Depending on whether the standard volume data is outside the MapR cluster and in the cloud (cold tiering) or on the MapR cluster (warm tiering), MapR processes the request to read standard volume data and mirror volume data.

Data Reads on Tiering-Enabled Standard Volumes

MapR processes client requests to read standard volume data on the warm tier and the cold tier differently.

Warm Tier

When a client attempts to read, the read request is first sent to the front-end volume and if the data exists in the front-end volume, the data is returned from the front-end volume. If data is not in the front-end volume, the data is returned from the erasure coded volume.

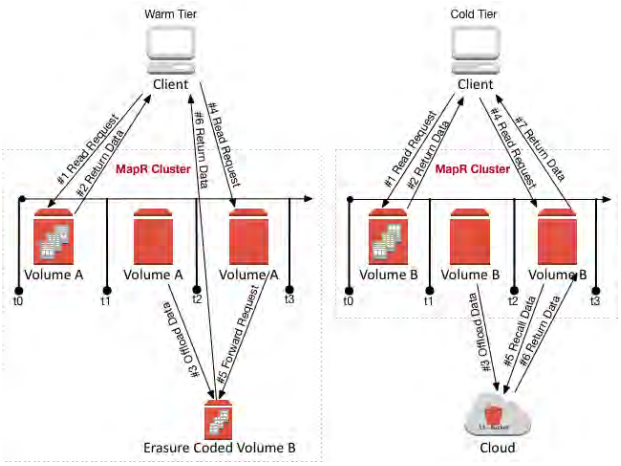
Cold Tier

When a client attempts to read, the read request is first sent to the volume on the MapR cluster and if the data exists in the volume on the cluster, the data is returned from the volume. On the other hand, if the data was offloaded, the MAST Gateway recalls the data from the cold-tier to process the read request. See Recall of Tiered Data on page 484 for more information on recalled data.

The following illustration shows a client sending the data read request first (#1) to the tiering-enabled volume and the response (#2) being served from the volume on the MapR cluster. Then (#3), data is offloaded to the back-end erasure coded volume (for Volume A) and to the cloud (for Volume B). When the client next sends a read request to the volume on the MapR cluster (#4), for:

- Volume A, the MAST Gateway forwards the request to the back-end erasure-coded volume (#5) from where data is returned (#6) to the client.

- Volume B, the MAST Gateway recalls the data (#5 and #6) from the cloud to the volume on the MapRcluster, from where data is returned (#7) to the client.



Data Reads on Tiering-Enabled Mirror Volumes

MapR processes client requests to read mirror volume data on the warm tier and the cold tier differently.

Warm Tier

When a client attempts to read, the read request is first sent to the front-end volume and if the data exists in the front-end mirror volume, the data is returned from the front-end volume. If data is not in the front-end volume, the data is returned from the erasure coded volume.

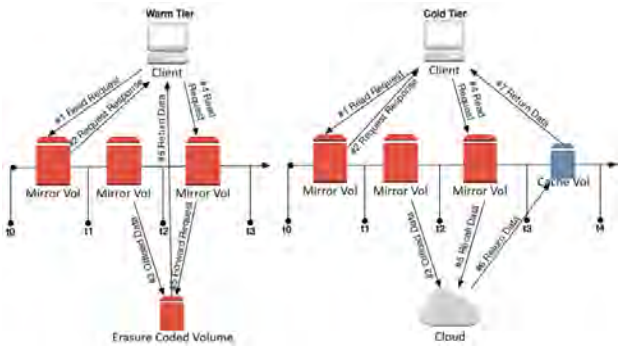
Cold Tier

When a client attempts to read, the read request is first sent to the volume on the MapR cluster and if the data exists in the volume on the cluster, the data is returned from the volume. On the other hand, if the data was offloaded, the MAST Gateway recalls or fetches a copy of the data (from the tier) into an associated cache-volume, from where data is returned to the client. See Recall of Tiered Data on page 484 for more information on recalled data.

The following illustration shows a client sending the read request first (#1) to the tiering-enabled mirror volume and the response (#2) being served from the volume on the MapR cluster. Then (#3), data is offloaded to the back-end erasure coded volume (for

warm tier) and to the cloud (for cold tier). When the client next sends a read request to the volume on the MapR cluster (#4), for:

- Warm tier, data is returned from the back end erasure-coded volume (#5).
- Cold tier, data is recalled in the associated cache-volume (#5 and #6), from where data is returned to the client (#7).



The cache-volume named `mapr.internal.cv.<volume-name>_<id>` is created when the tiering-enabled mirror volume is created. Although it may not hold any data initially, a copy of the tiered data is fetched into the cache-volume whenever there is a read of the cold-tiered mirror volume data or explicit recall of (cold or warm) tiered mirror volume data. You can use the `volume info` on page 1965 command on the tiering-enabled mirror volume to get the offload and recall statistics, which are from the cache-volume, for the mirror volume.

The cache-volume has the same replication factor as the mirror volume (at the time of volume creation); changes to the mirror volume replication factor do not trigger a change to the replication factor of the associated cache-volume.

If the tiering-enabled mirror volume is deleted, the cache-volume is also deleted. If the tiering-enabled mirror volume is promoted to a read-write volume, the associated cache-volume is deleted.

Write on Tiered Data

When writes happen, if the write is:

- An append, new data is offloaded when the data meets the criteria in the rule (associated with the volume) for offload.
- A change to existing data (overwrite), the data is recalled to the MapR filesystem to allow the write to succeed and then offloaded when the data meets the criteria in the rule (associated with the volume) for offload. See Recall of Tiered Data on page 484 for more information on recalled data.

Note: If cold data is accessed (read/written) frequently, I/O to that file may suffer large latencies. In such scenarios, recall the whole volume or the corresponding files.

Recall of Tiered Data

Offloaded data is automatically recalled when a client performs a read or overwrite on the data in the cold-tier, or when a client performs an overwrite on the data in the warm-tier. The MAST Gateway fetches a copy of the data to allow the operations to succeed. You can also manually trigger a recall of:

- All volume data using the `maprcli` command or REST API.

  See Recalling a Volume to MapR File System on page 936 for more information.

- File using the `hadoop` command, `maprcli` command, REST API, (loopbacknfs or FUSE-based) POSIX client, or the NFS client.

  See Recalling a File to MapR File System Using the CLI and REST API on page 1012 and Running Tiering Commands when maprcli and hadoop Commands are not Available on page 1014 for more information.

Based on the expiration time period set at the volume level for recalled data, recalled data is:

- Offloaded again based on the rules if there are changes to the data.
- Purged when the compactor runs if there are no changes to the data.

For a cold tiering volume, explicitly recall the volume before running any analytics jobs.

For mirror volumes, when you recall tiered data, data from the tier is recalled into an associated cache-volume, which is created at the time of the creation of the tiering-enabled mirror volume. For all explicit recall of warm-tiered data and explicit and automatic recalls of cold-tiered data, the MAST Gateway recalls data into the associated cache-volume. The data in the cache-volume is "hot" in the cluster, or available for reads, for the duration of the expiry-period. The recalled data is purged by the compactor when the expiration time that is set at the volume level is reached or has passed.

If the recall fails, CLDB retries the operation after some time. See Retrying Failed Operation on page 939 for more information.

Moving Data from Non-Tiered Volumes to Tier Enabled Volumes
Provides a synopsis of how to move data from non-tiered volumes to tier-enabled volumes.

Non-tiered volumes cannot be offloaded.

Use the following procedure to transfer data from a non-tiered volume to a fresh tiered volume, and then offload the data from the tiered volume.

If you are short on space, you can first break up large volumes into multiple small volumes.

You can then transfer one sub-volume at a time.

For example, assume that there are sub-volumes `/hugevolume/dir1`, `/hugevolume/dir2`, ..., `/hugevolume/dirN`. To transfer:

1. Create a new tiering enabled volume say `dir1`.
2. Mount it at `/tmp/dir1`.
3. Snapshot `/hugevolume/dir1`.
4. Use `distcp` to copy the snapshot to `/tmp/dir1`.
5. After the initial transfer, perhaps snapshot again and use rsync to sync the changes to `/tmp/dir1` to minimize downtime.
6. Delete `/hugevolume/dir1`.
7. Unmount the `dir1` volume and re-mount at `/hugevolume/dir1`.

8. Now `/hugevolume/dir1` will tier according to the schedule and rule specified when creating it in step 1.

9. Repeat the process for dir2 to dirN.

Data Compaction
Describes how data is purged from a cluster.

When you release the space allocated to a volume on the MapR cluster by deleting a file or snapshot, or by truncating a file, the MapR tier compactor can be set up to run automatically or manually to release the space on the tier associated with the volume by deleting the corresponding stripes or objects from the tier. In addition, when you recall data, the compactor automatically purges the recalled data on the MapR cluster if there are no changes to the data. By default, the compactor runs on an automatic internal schedule to determine if any deletion has happened on the MapR cluster since it last ran, and if necessary, remove the corresponding stripelets or objects from the tier.

MapR uses two settings (at the volume level) to determine when and how frequently to run the compactor:

- Overhead threshold — You can specify a percentage of offloaded data that must have been deleted to trigger the compaction operation. By default, the compactor performs the compaction operation only if at least 30% of the offloaded data is deleted.
- Compactor schedule — You can set up a custom schedule to run the compactor. By default, MapR uses the Internal Automatic Scheduler (ID is 4), which is based on internal parameters, to run the compactor.

The compactor runs only when there are no other tiering operations running for the volume. If there are other tiering operations, such as offload or recall, running for the volume, the compactor does not run until the tiering operation completes. If a tiering operation is triggered while the compactor is running, the tiering operation will fail. You cannot trigger a volume-level tiering job when another job is running for the volume. You can trigger a file-level offload or recall operation when a volume level job is running and vice-versa.

Purging Recalled Data

When you recall data to the MapR cluster explicitly (by running the `recall` command) or implicitly (by doing a read or an overwrite), the recalled data is purged by the compactor if there are no changes to the data and if the expiration time for recalled data has been reached or has passed. You can also manually run the compactor to force an immediate purge of recalled data. See Running the Compactor to Purge Recalled Data on the MapR Cluster on page 942 for more information.

Purging Stale Data

When you release space on the MapR cluster by deleting or modifying data, the compactor purges the data on the tier also. Depending on whether file data is completely deleted or partially deleted on the MapR cluster, the MapR compactor processes purging of data on the tier.

Warm Tier	For warm tiered volumes, the MapR compactor identifies corresponding objects (or stripes) on the tier and deletes entire objects first. After deleting entire objects, if there are partial objects to delete, the MapR compactor identifies the objects (with partially deleted data) that can be coalesced, fetches them, creates new objects with combined data, updates the metadata in the DB tables, deletes the old objects from the tier, and offloads the new objects to the tier. The compactor handles partial deletions only after deleting entire objects and only if the size of the remaining data to delete exceeds the compaction threshold.
Cold Tier	For cold tiered volumes in the S3 environment, while entire objects can be easily deleted, modifications and partial deletions are not supported. For example,

assume that data associated with a file is distributed across objects. When a file is deleted on the MapR cluster, corresponding data on the S3 tier can be easily removed if the object in the S3 tier only contains data associated with the deleted file. On the other hand, if the object also contains data from other files, the object cannot be deleted, and S3 does not support changes to the object or partial deletion of the object.

For partial deletions, the MapR compactor identifies the objects (with partially deleted data) that can be coalesced, fetches them, creates new objects with combined data, updates the metadata in the DB tables, deletes the old objects from the tier, and offloads the new objects to the S3 tier. The compactor handles partial deletions only after deleting entire objects and only if the size of the remaining data to delete exceeds the compaction threshold.

You can manually trigger the compactor to purge the stale data on the tier. See Running the Compactor to Purge Stale Data on the Tier on page 942 for more information.

Tuning Last Access Time
Provides an overview of the Last Access Time feature and its tuning.

What is Last Access Time?

Last Access Time (`atime`) is *file* metadata that is updated whenever a file is read. You can use `atime` for file management and governance decisions such as:

- Deleting files that have not been accessed for a while
- Tiering files (to warm or cold tier) that have not been accessed for a while
- Migrating files that have not been accessed frequently
- Purging files that have not been accessed for a time

Considerations When Enabling Last Access Time

- `atime` update can be enabled only on Standard/Erasure Coding/Object Tiering volumes. It cannot be enabled on mirrored volumes. If you convert the mirror volume to a Read/Write volume, `atime` is disabled by default. You can enable `atime` with the volume modify command.
- `atime` is applicable only for files. The `atime` of directories is NEVER updated.
- While the `read` operation is audited, the `atime` operation is not audited as it is an internal operation.
- The volume offload operation does not update `atime` but a file read from a backend/frontend volume updates `atime`.
- The file recall operation also updates `atime`.
- The file read operation on the EC/Tiered backend volume updates `atime` on the frontend volume.
- At the time of mirroring, the `atime` update frequency (`atimeUpdateInterval`) is propagated from the source volume to the mirror volume. However, any subsequent changes made to this frequency on the source volume, are not automatically propagated to the mirror volume.
- The time when you enabled `atime` updates (`atimeTrackingStartTime`) is updated to the current time in the following cases:

- Just started tracking `atime`, which means that the `atime` update frequency was previously zero
- If the value of `atime` update frequency is decreased
- If the value of `atime` update frequency is increased and `atime` has not been tracked for the duration of the new frequency value

Exceptions to Last Access Time Updates
`atime` is never updated when:

- Only the meta data of the file is being read
- The file is read from the client cache
- The file is read from a snapshot
- The `atimeUpdateInterval` has not been exceeded:

  For example, assume that for a volume the `atimeUpdateInterval` is set to 1 day. A file is created at 11AM and the file is read at 10:55AM the next day. If the read finishes at 10:58AM, `atime` will not be updated as the `atimeUpdateInterval` did not cross a day.

  For another example, assume that for a volume the `atimeUpdateInterval` is set to 1day. A file is created at 11AM and the file is read at 10:55AM the next day. If the read completes at 11:10AM, the `atime` will still not be updated though the read completed after 24 hours, because read was triggered at 10:55AM. `atime` will only be updated when the file is next read.

Upgrade Considerations

When a cluster is upgraded to MapR Data Platform 6.2, `atime` is not enabled on the old volumes. You need to enable `atime` manually using the volume modify command.

When a cluster along with a few clients are upgraded to MapR Data Platform 6.2, while the remaining clients are not upgraded, the older clients can not update `atime` on files. Only the upgraded clients can trigger an `atime` update. However, the older clients can see the updated `atime` value (updated by the upgraded clients).

Enabling the Last Access Time Feature

The Last Access Time feature is not automatically enabled irrespective of whether you perform a fresh installation or an upgrade. To enable and activate the Last Access Time feature, run:

```
maprcli cluster feature enable -name mfs.feature.update.atime
```

Enabling Last Access Time on Volumes

For performance reasons, the `atime` feature is disabled on volumes by default. You can enable `atime` updates at the volume level when creating or modifying volumes.

To set the frequency of `atime` updates, use the `atimeUpdateInterval` parameter when creating or modifying volumes. ***The value is in days***. The default value of `0` indicates that `atime` is never updated.

For example, a value of **2** indicates that the `atime` is updated *Once* every **2** days (48 hours) with the first read on the file. `atime` will not be updated on further reads on the file till the 48 hours have passed.

Viewing the Last Access Time Value

To view the `atime` value of a specific volume, use the volume info on page 1965 command.

last-updated: Apr 11, 2022

Last Access Time Example

The following command creates a volume and sets the `atime` to 2 days:

```
maprcli volume create -name stdvol1 -path /stdvol1 -atimeUpdateInterval 2d
```

To view the Last Access Time frequency, run:

```
maprcli volume info -name stdvol1 -json | grep atime
            "atimeUpdateInterval":"2",
            "atimeTrackingStartTime":"2021-03-14 22:45:25 GMT-0700",
```

Here, the frequency is set to 2 days. The time when `atime` was enabled on the volume is also displayed.

Related reference

volume create on page 1931
Creates a volume.

volume modify on page 2005
Modifies an existing volume. Permissions required: `m` or `fc` on the volume.

volume info on page 1965
Displays information about the specified volume. For JSON formatted output, use the `-json` option when running the command.

volume list on page 1979
Lists information about volumes specified by name, path, or filter.

tier rule create on page 1885
Creates a rule for offloading data to a tier.

Multitenancy on File System

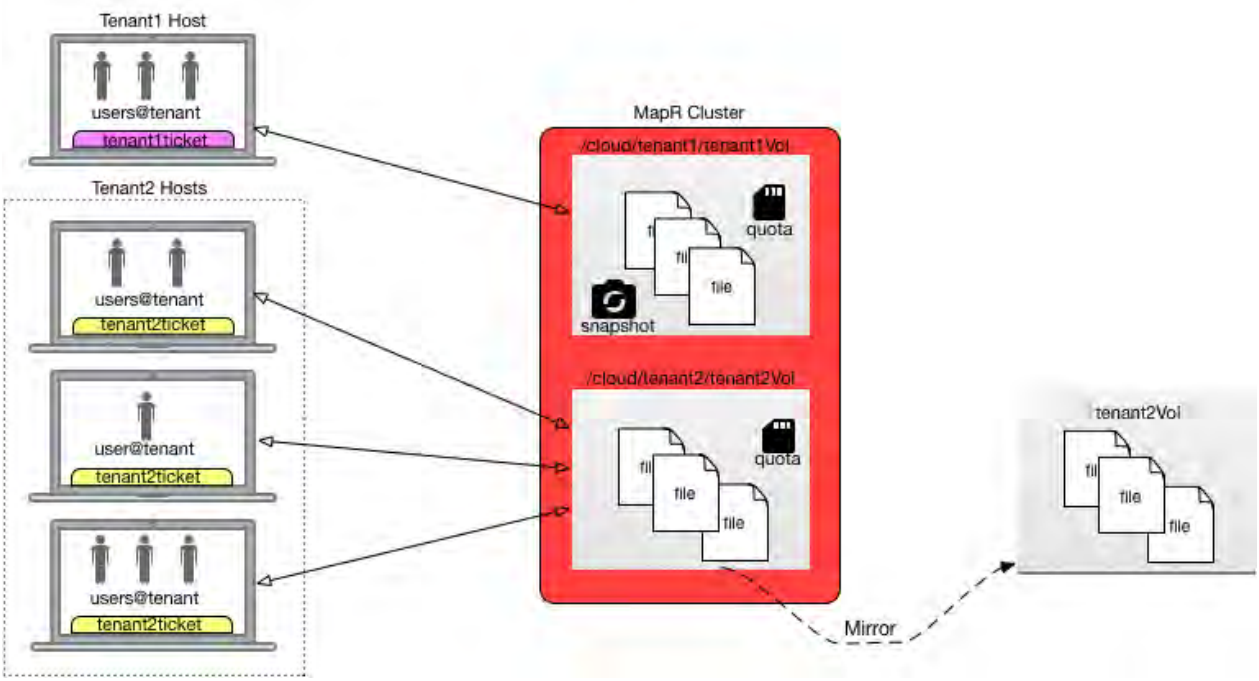
Describes what multitenancy is and how tenant data is kept private for each tenant.

Multitenancy architecture enables a single instance of a software to be provisioned for multiple customers or users, who are referred to as tenants. Each tenant, or group of users, has a specific share of the instance including access to its data, configuration, and access management. On the cloud, this enables a software-as-a-service (SaaS) provider to provision the software for multiple tenants.

The MapR File System multitenancy architecture enables you to create and restrict a MapR volume (referred to as a share) to a subset of client nodes. By doing this, you can isolate users or hosts (referred to as tenants). Isolation enables you to set policies, quotas, and access privileges for specific tenants. You can provision the MapR file system on the cloud to various tenants, with each tenant owning its own copy of storage space, users, data security, administration, and so on.

In a multitenant environment, tenants operate in their own provisioned spaces, unaware of other tenants on the cluster. Tenants have exclusive access to data in their environment only.

For example, the following diagram depicts a cluster provisioned on the cloud for two tenants, Tenant1 and Tenant2. The cluster has two separate volumes, mounted at directories `/cloud/tenant1`, and `/cloud/tenant2`. Each tenant volume contains file data created and managed by tenant users on the tenant host. Each tenant maps to a different volume and therefore, data in each volume can have different policies, disk-usage quotas, snapshot and mirroring schedules. By using appropriate tenant tickets, access to data in these volumes is restricted only to users on the appropriate tenant hosts, and eliminates the possibility of a user from Tenant2 accessing data on the Tenant1 volume, and vice versa.



You can access tenant shares using loopbacknfs and FUSE-based POSIX clients only. After you mount the tenant volume for access using (FUSE-based and loopbacknfs) POSIX clients, you can perform operations using standard Linux commands.

Setting Up a Tenant
Lists the process for setting up a tenant.

To set up a tenant:

1. On the server:
   a) Log in to the cluster as the administrator and create a user (for the tenant admin) on the cluster.
      The user (for the tenant) must exist on all the cluster nodes with the same UID and GID or all the cluster nodes must connect to the same LDAP server. See Managing Users and Groups on page 752 for more information.

      Note: The superuser for a tenant, referred to as tenant admin, must have a UID of `0` on the tenant host(s) to access the tenant volume (only) and all data in the tenant volume. Although the tenant admin has the same UID as the MapR superuser, the tenant admin does not have the same level of access and administration privileges as the MapR superuser because the tenant admin's access is based on the tenant ticket and is restricted to the tenant volume.

   b) Generate a tenant ticket for the user.
      For more information, see Generating a Ticket for a Tenant on page 1429.

   c) Copy the ticket to the tenant host and grant the tenant administrator read access to the ticket.

   d) Create a volume (or share) on the cluster for the tenant.
      For more information, see Creating a Volume for a Tenant on page 881.

2. On the tenant instance:
   a) Log in as tenant administrator (`root`).

last-updated: Apr 11, 2022

b) Mount the filesystem using loopbacknfs or the FUSE-based POSIX client.
For more information, see Mounting a Tenant Volume on page 904.

Note: While starting the POSIX client, use the tenant ticket configured in step 1.

c) As tenant admin, grant access to users by setting permissions to data using either file ACEs or mode bits.
For more information, see Enabling and Restricting Access to Tenant Volume and Data on page 925.

Provisioning File System for Multiple Tenants - Sample Workflow
Illustrates a sample workflow for provisioning the MapR file system to multiple clients.

For example, suppose there are two tenants Tenant1 and Tenant2. The following steps show the workflow for provisioning the two tenants:

1. The cluster administrator creates two users, Tenant1 and Tenant2, on the MapR cluster and creates volumes (or shares) on the cluster for the two tenants.
   For example, to create volumes on the cluster:

```
$ /opt/mapr/bin/maprcli volume create -name tenant1Vol -path /
tenant1Enoke -tenantuser Tenant1
$ /opt/mapr/bin/maprcli volume create -name tenant2Vol -path /
tenant2Enoke -tenantuser Tenant2
```

2. The cluster administrator generates tickets for the users, copies the tickets to the tenant servers (tenant1Host and tenant2Host), and grants the tenant admins (tenant1Admin and tenant2Admin) read access to the ticket.
   For example, to:
   - Generate ticket for the users:

```
$ maprlogin generateticket -type tenant -cluster myCluster -user
tenant1 -out /tmp/tenant_Tenant1_ticket.txt
$ maprlogin generateticket -type tenant -cluster myCluster -user
tenant2 -out /tmp/tenant_Tenant2_ticket.txt
```

   - Copy tickets to appropriate tenant hosts:

```
$ scp /tmp/tenant_Tenant1_ticket.txt
tenant1Admin@tenant1Host:~tenant1Admin/
$ scp /tmp/tenant_Tenant2_ticket.txt
tenant2Admin@tenant2Host:~tenant2Admin/
```

3. The tenant administrators log into their respective hosts and mount their shares by starting the client.
   For example, to start the:

FUSE-based POSIX client

a. Update the following parameters in the `fuse.conf` file:

`fuse.ticketfile.location`	For: • Tenant1, `tenant1Admi` `tenant_Tenant1_tic`

	• Tenant2, `tenant2Admi` `tenant_Tenant2_tic`
`fuse.mount.point`	For: • Tenant1, `/tenant1Eno` • Tenant2, `/tenant2Eno`
`fuse.export`	For: • Tenant1, `/tenant1Eno` `tenant1Vol` • Tenant2, `/tenant2Eno` `tenant2Vol`

b. Run the following command to start the service:

```
$ service mapr-posix-client-*
start
```

loopbacknfs POSIX client

a. Update the tenant ticket file location in `/etc/loopbacknfs/initscripts/mapr-loopbacknfs` file.

b. Run the following command to start the service:

```
$ service mapr-loopbacknfs start
```

4. The tenant administrators can grant access to users within their tenant namespace by modifying data access using *Access Control Expression (ACE)*s.

Direct Access NFS

Describes the MapR direct access file system.

The MapR direct access file system enables real-time read/write data flows using the Network File System (NFS) protocol. Standard applications and tools can directly access the MapR File System storage layer using NFS. Legacy systems can access data, and traditional file I/O operations work as expected on a conventional UNIX file system. A remote client can easily mount a MapR cluster over NFS to move data to and from the cluster. Application servers can write log files and other data directly to the MapR cluster's storage layer instead of caching the data on an external direct or network-attached storage.

You can mount a MapR cluster directly through a network file system (NFS) from a Linux or a Mac client. When you mount a MapR cluster, applications can read and write data directly into the cluster with standard tools, applications, and scripts. MapR enables direct file modification and multiple concurrent reads and writes with POSIX semantics. For example, you can run a MapReduce application that outputs to a CSV file, and then import the CSV file directly into SQL through NFS.

MapR exports each cluster as the directory `/mapr/<cluster name>`. If you create a mount point with the local path `/mapr`, Hadoop FS paths and NFS paths to the cluster will be the same. This makes it easy to work on the same files through NFS and Hadoop. In a multi-cluster setting, the clusters share a single namespace. You can see them all by mounting the top-level `/mapr` directory.

last-updated: Apr 11, 2022

POSIX Clients

Describes the usage of MapR POSIX clients.

The MapR filesystem supports direct and secure access to data using loopback NFS or FUSE-based POSIX clients.

The loopbacknfs POSIX client allows app servers, web servers, and other client nodes and apps to read and write data directly and securely to a MapR cluster, with transmitted data compressed in both directions. The MapR single-user mapr-loopbacknfs licenses gives secure access to one or more clusters, which allows native client applications to run securely on cluster data.

The FUSE-based POSIX basic and platinum clients run as a user space process to connect to one or more MapR clusters and allow app servers, web servers, and applications to read and write data directly and securely to the MapR clusters like a Linux filesystem. Each client implies a specific MapR filesystem throughput optimization of *n/G* per second.

Both loopbacknfs and FUSE-based POSIX clients can be installed on supported Linux and Ubuntu distributions and require direct network access to all MapR cluster nodes. They connect to the MapR cluster directly (no NFS gateway) to read and write data securely.

Related concepts

Managing MapR POSIX Clients on page 1229
Provides a brief synopsis of MapR POSIX clients.

MapR FUSE-Based POSIX Client on page 1238
Provides a brief description of the FUSE-based POSIX client.

Comparing the FUSE POSIX and Loopback NFS Plugins on page 670
This page compares the two types of MapR Container Storage Interface (CSI) Storage Plugins and describes when to use them.

MapR POSIX Clients on page 399
Describes how to install the POSIX loopback NFS, and the FUSE-based POSIX clients.

Managing the FUSE-Based POSIX Client on page 1255
Describes how to use the FUSE-based POSIX client.

Copying Data from Apache Hadoop to a MapR Cluster

Describes the procedure to copy data from an Apache Hadoop to a MapR cluster.

You can use the hdfs protocol, webhdfs protocol, or NFS to copy data from Apache Hadoop to a MapR cluster.

The following table describes these methods:

Method	Description
hdfs:// protocol	You can use the `hadoop distcp` command with the `hdfs://` protocol to copy data from a HDFS cluster into a MapR cluster if the HDFS cluster and the MapR cluster use the same RPC protocol version. For all other scenarios, use the `webhdfs://` protocol or NFS gateway to copy data to a MapR cluster.
webhdfs:// protocol	You can use the `hadoop distcp` command with the `webhdfs://` protocol to copy data from a HDFS cluster into a MapR cluster.
NFS	You can mount a MapR cluster to a HDFS cluster using NFS mount and then use the `hadoop distcp` command to copy data between the two clusters.

Refer to the following sections for information about how to copy data from Hadoop to a MapR cluster:

About the MapR Persistent Application Client Container (PACC)

This container gives you seamless access to MapR cluster services.

This topic introduces the MapR Persistent Application Client Container (PACC), including its function, benefits, components, and applications.

The MapR (PACC) is a Docker-based container image that includes a container-optimized MapR client. The PACC provides seamless access to MapR Converged Data Platform services, including MapR File System, MapR Database, and MapR Event Store For Apache Kafka. The PACC makes it fast and easy to run containerized applications that access data in MapR.

FUSE POSIX Client for File-Based Applications

To support persistent, file-based applications, the MapR PACC includes a FUSE-based MapR POSIX Client, optimized for containers, that allows app servers, web servers, and other applications to read and write data directly to the MapR filesystem. If your cluster has a MapR POSIX Client for Containers license, the PACC can connect with MapR 5.1 or later clusters.

Traditionally, all file data created by containers is lost when a container is terminated, which can happen during an application or hardware failure. By using the POSIX client within the PACC, applications can reliably persist file data directly to the MapR filesystem, where it can be re-attached to the container in the event of application or hardware failures.

Support for Microservice Applications

To support stateful microservice applications, the MapR PACC also contains a container-optimized version of the MapR client, which includes libraries for accessing MapR Database and MapR Event Store For Apache Kafka.

Secure Access

The MapR PACC is designed to provide access to a secure cluster for all MapR Converged Platform data services. Users can pass a MapR ticket file into the container at runtime. All data access, whether to MapR File System, MapR Database, or MapR Event Store For Apache Kafka, is authorized and audited according to the authenticated identity of the ticket file.

PACC Contents

The PACC includes the following components:

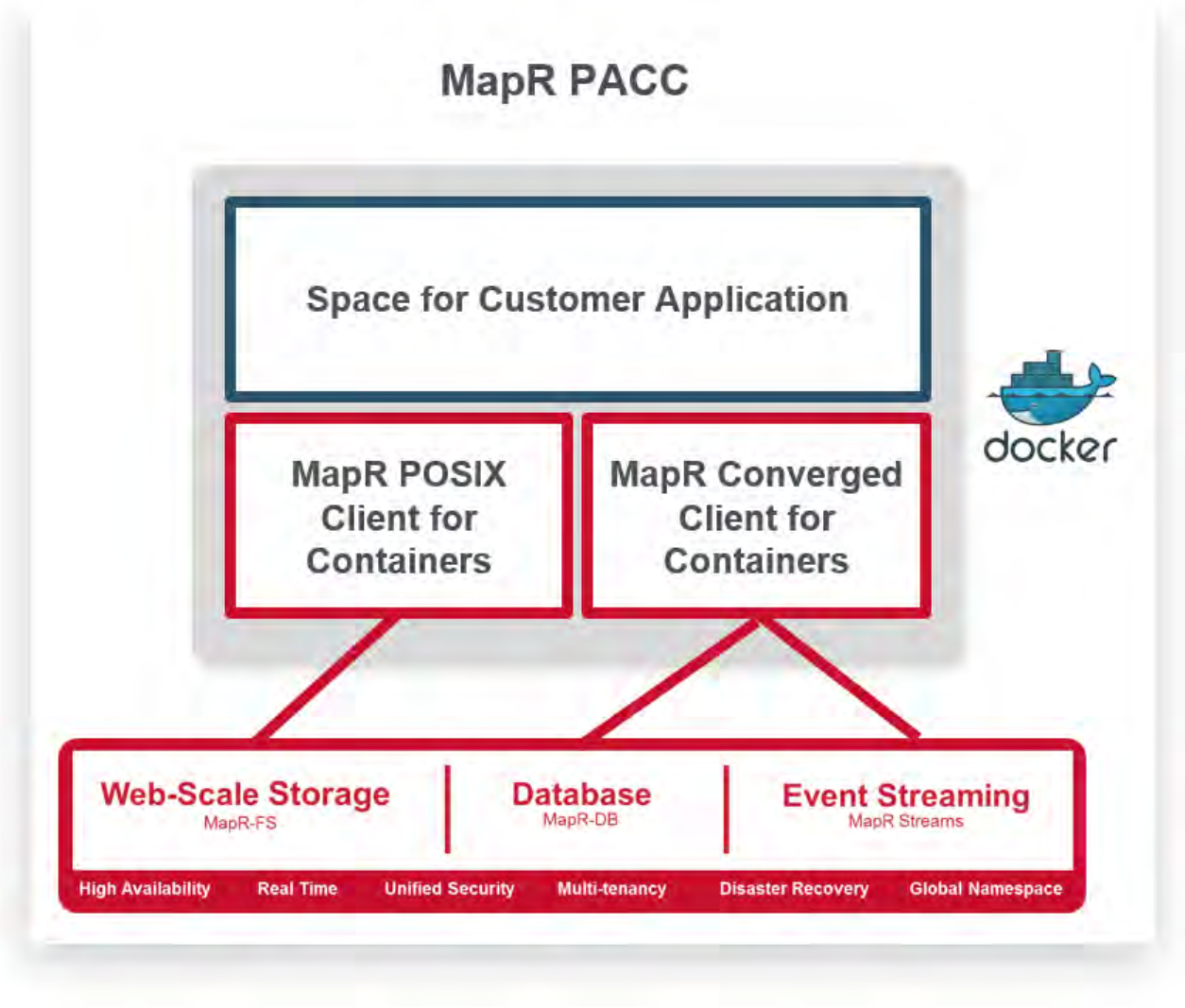
- MapR Database Client[1]
- MapR Event Store For Apache Kafka Client
- POSIX Client for Containers
- Hadoop Client with YARN[2]
- HBase Client[2]
- Hive Client[2]
- Pig Client[2]
- Python
- Java
- Curl, Wget, Openssl, NFS-common, etc

last-updated: Apr 11, 2022

[1]The MapR Database client includes support for MapR Database binary tables and MapR Database JSON tables.

[2]Included only if specified and only in MapR PACC images created using `mapr-setup.sh`.

The following diagram illustrates the contents of the PACC, and how it allows applications to access MapR Converged Data Platform services.



Pre-Built and User-Created Images

To get started with the MapR PACC, you can take advantage of pre-built Docker images or create your own images to include site-specific environmental parameters:

To . . .	See this topic
See a list of the MapR pre-built Docker images	Extending a MapR PACC on page 408
Create your own images containing MapR software	Creating a MapR PACC Image Using mapr-setup.sh on page 409

MapR Control System

Provides a brief description of the MapR Control System.

last-updated: Apr 11, 2022

The MapR Control System provides a graphical control panel for cluster administration with all the functionality of the command-line or REST APIs. The Control System provides job monitoring metrics and helps you troubleshoot issues, such as which jobs required the most memory in a given week, or which events caused job and task failures.

The Control System provides various views, which you can use to configure and monitor your cluster:

Overview	The Control System **Overview** page provides a summary of information about the cluster including a cluster heat map that displays the health of each node organized by service, an alarms summary, cluster utilization that shows the CPU, memory, and disk space usage, the number of available, unavailable, and under replicated volumes, and MapReduce applications.
Services	The Control System **Services** page provides a summary of the services running across the cluster.
Nodes	The Control System **Nodes** page provides a summary of information about the nodes on the cluster including a heat map that displays the health of each node, resource utilization that shows the CPU and memory usage, all active alarms, and a list of all the nodes on the cluster with links that provide shortcuts to more detailed information about the node.
Data	The Control System **Data** drop-down menu contains links to pages that provide summary of information about volumes, tables, and streams.
Admin	The Control System **Admin** drop-down menu contains links to pages for user and cluster management tasks such as setting up permissions, quotas, and email settings for users, enabling cluster-level and data auditing, configuring balancer settings, and adding licenses.

Note: During installation using the MapR Installer, you can configure metrics and logging using settings on the Monitoring page of the MapR Installer user interface. The metrics collection infrastructure must be installed because the Control System relies on these metrics to provide graphs and charts. If the metrics collection infrastructure is not installed, you cannot visualize the metrics in the panes on the Control System.

Related concepts

Setting Up the Control System on page 423
Describes how to configure and access the Control System.

Using MapR Data Platform Monitoring (Spyglass Initiative)

MapR Data Platform Monitoring (part of the Spyglass initiative) provides the ability to collect, store, and view metrics and logs for nodes, services, and jobs/applications.

Metric Monitoring

Administrators can monitor the current status of the cluster and anticipate future cluster requirements with dashboards. For example, you can use metrics dashboards to visualize the following:

Storage Utilization	Use metrics dashboards to monitor storage trends. For example, you can compare the volume of MapR File System usage at different times to the MapR File

last-updated: Apr 11, 2022

	System capacity and then allocate resources to the MapR File System accordingly.
Node Utilization	Use metrics dashboards to check for node overload. For example, if the CPU usage is high on a few nodes, you may want to distribute the load across more nodes for better performance and efficiency.
MapR Database Operational Trends	Use metrics dashboards to display historical trends for MapR Database operations. For example, if a user reports MapR Database slowness, the historical trends associated with row scans, get, and put operations can be used to identify the node(s) on which the performance degradation occurs.

Log Monitoring

Administrators can use dashboards to visualize, search, and review logs when troubleshooting issues. For example, you can use log dashboards to troubleshoot the following issues:

Service Failures	When metrics indicate that one or more services are down, use log dashboards to check the logs for each failed service and drill-down to each associated node.
Application Failures	When an application or job fails, use log dashboard to identify possible bottlenecks. For example, you can search the logs for a given application ID across all the nodes in the cluster.
MapR File System Performance	When users experience MapR File System or NFS slowness, use log dashboards to search the MapR filesystem logs for service errors or application issues.

Related Information

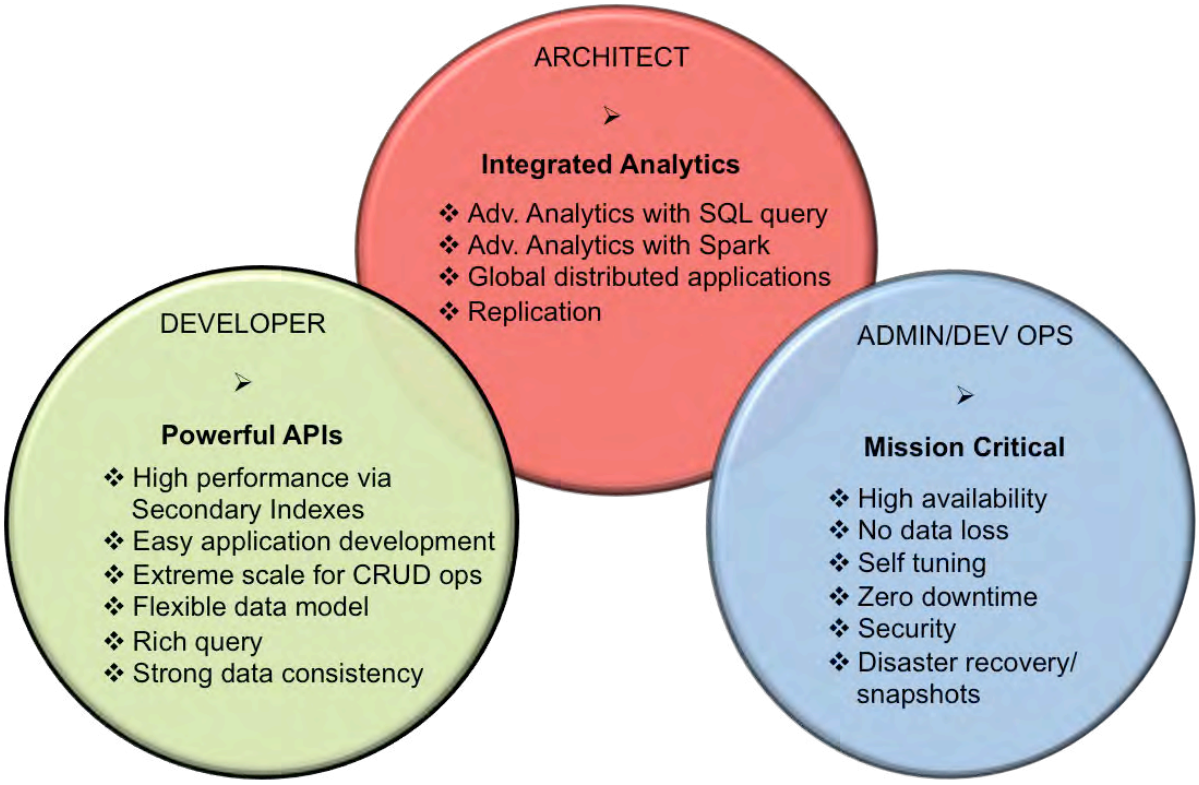
- Using MapR Data Platform Monitoring (Spyglass Initiative) on page 1330
- MapR Monitoring Storage Options on page 116
- Step 8: Install Metrics Monitoring on page 162
- Step 9: Install Log Monitoring on page 165

MapR Database

MapR Database is an enterprise-grade, high-performance, NoSQL database management system that you can use for real-time, operational analytics.

Why MapR Database?

MapR Database is built into the MapR Data Platform platform. It requires no additional process to manage, leverages the same architecture as the rest of the platform, and requires minimal additional management.



1. The MapR Database and Apps section provides information and examples on developing applications for MapR Database binary and JSON tables.
2. This section provides information on how to adminster tables, table regions, and column families. The tools for performing administration are the MCS (MapR Control System) user interface and the maprcli.
3. The MapR Database architecture covers topics associated with database design issues.

What databases does the MapR Database include?

The MapR Database includes two NoSQL databases:

- **Key-value database with HBase API**
  - Supports Apache HBase tables and databases.
  - Provides a native implementation of the HBase API for optimized performance on the MapR platform.
- **JSON document database based on the OJAI API**
  - Supports JSON documents as a native data store.
  - Stores JSON documents in MapR Database JSON tables.

How do I get started?

The following table provides links to useful resources for developers, architects, and administrators.

last-updated: Apr 11, 2022

Developer	Architect	Administrator/Dev Ops
MapR Database and Applications	Hbase and MapR Database: Designed for Distribution, Scale, and Speed	Installing MapR
Java App Examples for JSON Tables	Analytics with Drill	Administering MapR Database
C App Example for Binary Tables	Analytics with Spark	maprcli and REST API Syntax
How to Build Applications on a NoSQL Document Database and Perform Analytics in Place	Table Replication	Utilities for MapR Database JSON Tables
High Performance C APIs on MapR Database	Secondary Indexes	Utilities for MapR Database Binary Tables
Provisioning Secure Access Controls in MapR Database JSON		Security Overview

1. MapR Database and Applications
2. Java API Examples for MapR Database JSON Tables
3. C Application Example for Binary Tables
4. How to Build Applications on a NoSQL Document Database and Perform Analytics in Place
5. High Performance C APIs on MapR Database
6. Provisioning Secure Access Control in MapR Database JSON Tables
7. Hbase and MapR Database: Designed for Distribution, Scale, and Speed
8. Analytics with Drill
9. Analytics with Spark
10. Table Replication concepts
11. Installing MapR
12. Administering MapR Database
13. maprcli and REST API Syntax
14. Utilities for MapR Database JSON Tables
15. Utilities for MapR Database Binary Tables
16. Security Overview

last-updated: Apr 11, 2022

17. Secondary Indexes

Additional Resources

See the following MapR sites for more MapR Database information:

- MapR Database Product Page
- Blog: Real-Time User Profiles with Spark, Drill, and MapR Database
- Blog: How to Use a Table Load Tool to Batch Puts into HBase/MapR Database
- Blog: How to Persist Kafka Data as JSON in NoSQL Storage Using MapR Streams and MapR Database
- Blog: Provisioning Secure Access Controls in MapR Database

Architecture

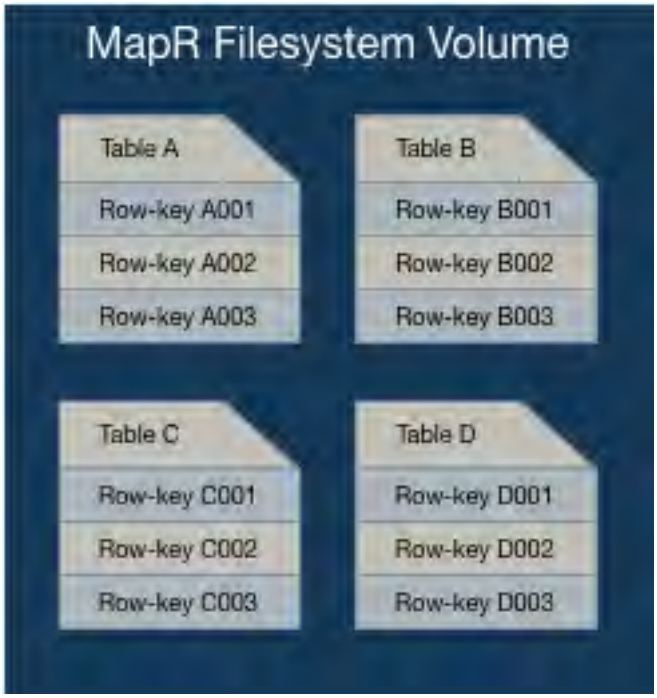
MapR Database is an enterprise-grade, high performance, NoSQL (“Not Only SQL”) database management system. You can use it to add realtime, operational analytics capabilities to big data applications. As a multi-model NoSQL database, it supports both JSON document models and key-value data models.

Why use MapR Database?

- **Integrated analytics with SQL**: MapR Database's integration with Drill for MapR provides a low latency, distributed, SQL query engine for large-scale datasets, including structured and semi-structured, nested data.
- **Operational analytics**: MapR Database can run in the same cluster as Apache™ Hadoop® and Apache Spark, letting you immediately analyze or process live, interactive data. This also enables you to eliminate data silos to speed the data-to-action cycle, providing a more efficient data architecture.
- **Global distribution of applications**: Application access to MapR Database tables is distributable on a global scale.
- **Flexible data model**: You can use MapR Database as both a document database and a column-oriented database. As a document database, MapR Database stores JSON documents in JSON tables. As a column-oriented database, it stores binary files in binary tables.

How is MapR Database Related to MapR XD Distributed File and Object Store?

MapR Database implements tables within the framework of the MapR filesystem. MapR Database creates tables (both binary and JSON tables) in logical units called *volumes*.



What are MapR Database's Architectural Advantages?

MapR Database's architecture has the following advantages:

- It reduces process overhead because it has no extra layers to pass through when performing operations on data.

  MapR Database, like several other NoSQL databases, is a log-based database. MapR Database runs inside of the MapR filesystem process, which enables it to read from and write to disks directly. In contrast, other NoSQL databases must communicate with a separate process to performs disk reads and writes. The approach taken by MapR Database eliminates extra process hops, duplicate caching, and needless abstractions, with the consequence of optimizing I/O operations on your data.

- It minimizes compaction delays because it avoids I/O storms when it merges logged operations with structures on disk.

  As a log-based database, MapR Database must write logged operations to disk. MapR Database stores table *regions* (also called *tablets*) and smaller structures within them partially as b-trees. Together with write-ahead logs (WAL), these b-trees comprise log-structured-merge trees. Write-ahead logs for the smaller structures within regions are periodically restructured by rolling merge operations on the b-trees. As MapR Database performs these merges at small scales, applications running against MapR Database see no significant effects on latency while the merges are taking place.

  Note: Apache HBase also uses the term *regions*.

What Design Factors are Important when Using MapR Database?

- **Rowkey Optimization**: The design of a table's rowkeys affects the speed at which client applications can access data. It also impacts database performance if hotspotting occurs. The better the design, the faster the data access. See Table Rowkey Design on page 542 for more information.

- **Column Family Optimization**: Column families enable you to group related sets of data and restrict queries to a defined subset, leading to better performance. When you design a column family, think about what kinds of queries you are going to use most often, and group your columns accordingly. See Column Families in JSON Tables on page 527 and Column Families in Binary Tables on page 541 for more information.
- **Replication Implementation**: The design of table replication (in addition to the automatic replication that occurs with table regions within a volume) depends on your desired outcome and the complexity of your environment. See Table Replication on page 610 for more information.
- **Security Implementation**: You can implement security at various levels including for table replication, JSON documents, and general access. Determining what level and where is part of the architectural design. See Security on JSON Tables on page 529, and Security on page 683.

MapR Database and MapR XD
Describes how MapR Database tables are implemented directly in the MapR filesystem, which allows MapR Database to leverage the same architecture as the rest of the platform and results in minimal additional management.

- MapR Database tables are created in logical units called *volumes*.
- MapR Database tables are sharded by implementing *table regions* (also called *tablets*)
- Table regions are stored in abstract entities called *data containers*.
- Data containers belong to MapR File System volumes.

Tables and Volumes

As volumes are a management entity that logically organize a cluster's data, they can be used to enforce disk usage limits, set replication levels, define snapshots and mirrors, and establish ownership and accountability.

Volumes do not have a fixed size and they do not occupy disk space until the filesystem writes data to a container within the volume. A large volume may contain anywhere from 50-100 million containers.

Tables are stored in containers and implemented in volumes, and provide the following capabilities:

- Multi-Tenancy
- Snapshots
- Mirroring and Replication

Table Regions and Containers

Each region of a table, along with its corresponding write-ahead log (WAL) files, b-trees, and other associated structures, is stored in one container. Each container (which can be from 16 to 32 GB in size) can store more than one region (which by default is 4096MB in size). The recommended practice is to use the default size for a region and allow it to be split automatically. Massive regions can affect synchronization of containers and load balancing across a cluster. Smaller regions spread data better across more nodes.

Note: Since a container always belongs to exactly one volume, that container's replicas all belong to the same volume as well.

The following are the key advantages to storing table regions in containers:

- Cluster Scalability
- High Data Availability

last-updated: Apr 11, 2022

For more information about containers, see Containers and the CLDB on page 453.

Cluster Scalability
Information about and location of tables (and files) is not tracked directly, but through MapR File System containers by the CLDB. As this architecture keeps the CLDB size small, it becomes practical to store 10s of exabytes in a MapR cluster, regardless of the number of tables and files.

The location of containers in a cluster is tracked by that cluster's container location database (CLDB). CLDBs are updated only when a container is moved, a node fails, or as a result of periodic block change reports. The update rate, even for very large clusters, is therefore relatively low. The MapR filesystem does not have to query the CLDB often, so it can cache container locations for very long times.

Moreover, CLDBs are very small in comparison to Apache Hadoop namenodes. Namenodes track metadata and block information for all files, and the locations for all blocks in every file as well. As blocks are typically 200 MB in size on an average, the total number of objects that a namenode tracks is very large. CLDBs, however, track containers, which are much larger objects, so the size of the location information can be 100 to 1000 times smaller than the location information in a namenode. CLDBs do not track information about tables and files.

High Availability
Due to the way updates to table regions (also called tablets) are applied and replicated, data in table regions are instantly available. Tables and table regions are part of abstract entities called *containers* that provide the automatic replication of table regions (with a default of three) across the nodes of a cluster.

Containers are replicated to a configurable number of copies. These copies are distributed to different nodes in the same cluster as the original or primary container. The cluster CLDB determines the order in which the replicas are updated. Together, the replicas form a replication chain that is updated transactionally. When an update is applied to a region (also called tablets) in the primary container (which is at the head of a replication chain), the update is applied serially to the replicas of that container in the chain. The update is complete only when all replicas in the chain are updated.

As a result of this architecture, when a hardware failure brings down a node, the regions served by that node are available instantly from one of the other nodes that have the replicated data.

MapR software can detect the exact point at which replicas diverge, even at a 2-GB-per-second update rate. The software randomly picks any one of the three copies as the new master, rolls back the other surviving replicas to the divergence point, and then rolls forward to converge with the chosen master. MapR software can do this on the fly with very little impact on normal operations.

Note: Since containers are contained in volumes, the automatic replication factor is set at the volume level.

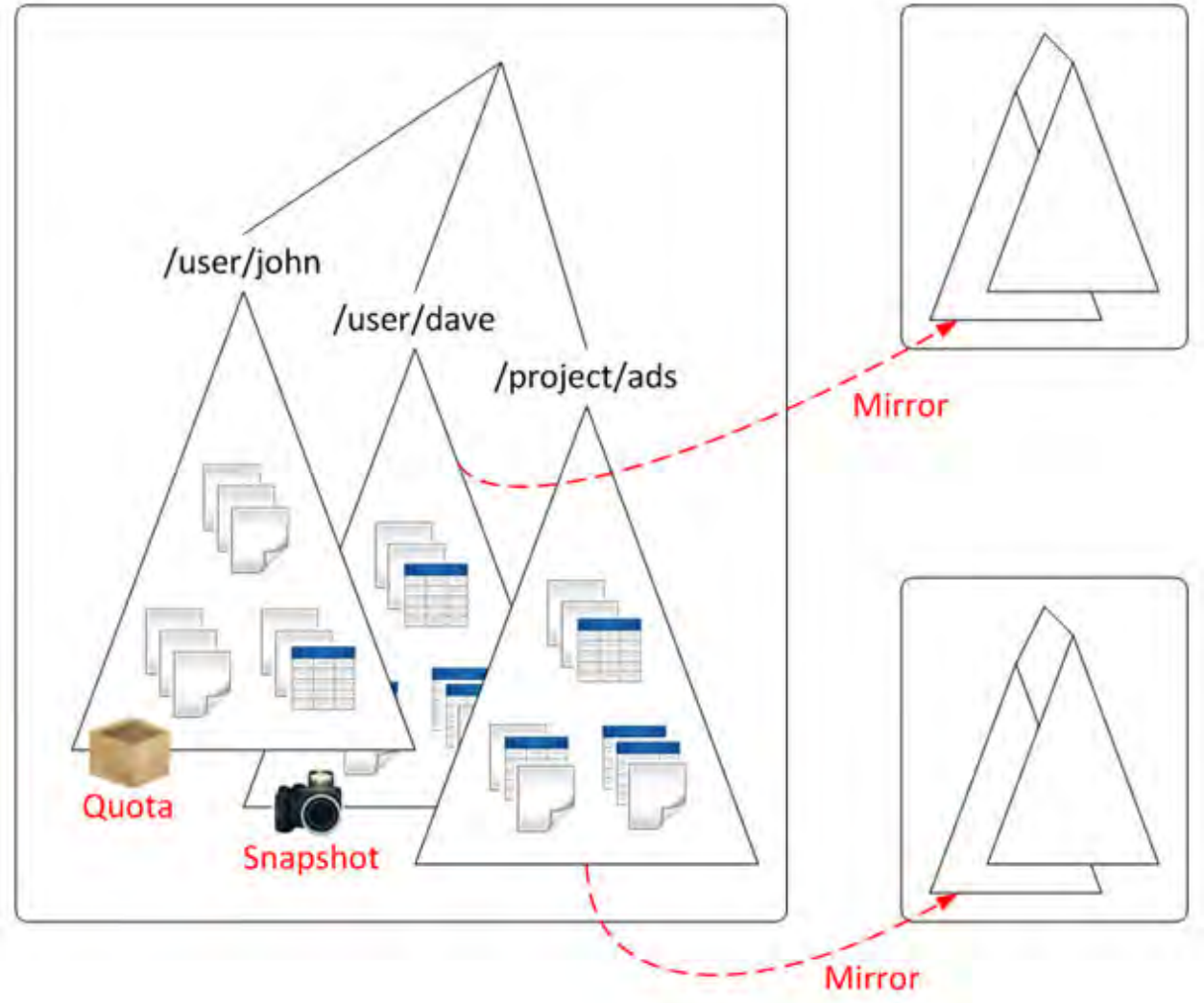
Multi-Tenancy
Since MapR Database tables are created in volumes, when you restrict the volume, you also restrict the table data. If a volume is restricted to a subset of a cluster's nodes, then it allows you to isolate sensitive data or applications, and even use heterogeneous hardware in the cluster for specific workloads.

For example, you can use data placement to keep personally identifiable information on nodes that have encrypted drives, or to keep MapR Database tables on nodes that have SSDs. You can also isolate work environments for different database users or applications and place MapR Database tables on specific hardware for better performance or load isolation.

Isolation of work environments for different database users or applications lets you set policies, quotas, and access privileges for specific users and volumes. You can run multiple jobs with different requirements without conflict.

As an example, the following diagram depicts a MapR cluster storing table and file data. The cluster has three separate volumes mounted at directories `/user/john`, `/user/dave`, and `/project/ads`. As shown, each directory contains both file data and table data, grouped together logically. Since each directory maps to a different volume, data in each directory can have a different policy. For example, `/`

`user/john` has a disk-usage quota, while `/user/dave` is on a snapshot schedule. Furthermore, two directories, `/user/john` and `/project/ads` are mirrored to locations outside the cluster, providing read-only access to high-traffic data, including the tables in those volumes.



Example: Restricting table storage with quotas and physical topology

This example creates a table with disk usage quota of 100GB restricted to certain data nodes in the cluster. First, create a volume named `project-tables-vol`, specifying the quota and restricting storage to nodes in the `/data/rack1` topology, and mounting it in the local namespace. Next, use the HBase shell to create a new table named `datastore`, specifying a path inside the `project-tables-vol` volume.

```
$ pwd
/mapr/cluster1/user/project

$ls
bin   src

$ maprcli volume create -name project-tables-vol -path /user/project/tables
\
    -quota 100G -topology /data/rack1

$ ls
bin   src   tables
```


last-updated: Apr 11, 2022

```
$ hbase shell
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
hbase(main):001:0> create '/user/project/tables/datastore', 'colfamily1'
0 row(s) in 0.5180 seconds
hbase(main):002:0> exit

$ ls -l tables
total 1
lrwxr-xr-x 1 mapr mapr 2 Oct 25 15:20 datastore ->
mapr::table::2252.32.16498
```

Snapshots

Since MapR Database tables are created in volumes, you can use a volume snapshot to capture the state of a volume's directories, MapR Database tables, and files at an exact point in time.

Use volume snapshots for rollbacks, hot backups, model training, and real-time data analysis management.

Rollback from errors	Application errors or inadvertent user errors can mistakenly delete data or modify data in an unexpected way. With volume snapshots, you can rollback your MapR Database tables to a known, well-defined state.
Hot backups	You can create backups of table data on the fly for auditing or governance compliance.
Model training	Machine-learning frameworks can use snapshots to enable a reproducible and auditable model training process. Snapshots allow the training process to work against a preserved image of the training data from a precise moment in time. In most cases, the use of snapshots requires no additional storage and snapshots are taken in less than one second.
Managing real-time data analysis	By using snapshots, query engines such as Apache Drill can produce precise synchronic summaries of data sources subject to constant updates, such as sensor data or social media streams. Using a snapshot of your MapR Database data for such analyses allows very precise comparisons to be done across multiple ever-changing data sources without the need to stop real-time data ingestion. See MapR Snapshots for more details.

Mirroring and Replication

Since MapR Database tables are created in volumes, mirroring of volumes lets you automatically replicate differential data in real-time across clusters. You might want mirror volumes to create disaster recovery solutions for databases or to provide read-only access to data from multiple locations.

As MapR Database does not require RegionServers to be reconstructed, databases can be brought up instantly on the mirrored site if the active site goes down.

Mirroring is a parallel operation, copying data directly from the nodes of one MapR cluster to the nodes in a remote MapR cluster. The contents of the volume are mirrored consistently, even if the files in the volume are being written to or deleted.

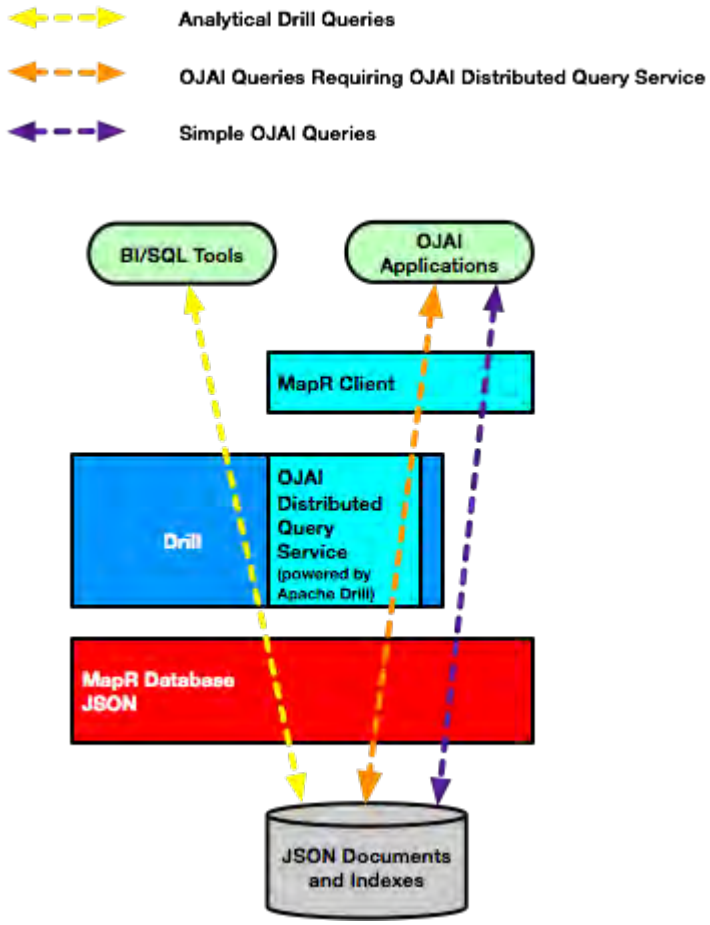
MapR captures only data that has changed at the file-block level since the last data transfer. After the data differential is identified, it is then compressed and transferred over the WAN to the recovery site, using very low network bandwidth. Finally, checksums are used to ensure data integrity across the two clusters. There is no performance penalty on the cluster because of mirroring.

See Mirror Volumes on page 463 for more details.

OJAI Distributed Query Service

OJAI queries either directly access MapR Database JSON or leverage the OJAI Distributed Query Service. The OJAI Distributed Query Service provides distributed query support for MapR Database JSON, powered by Apache Drill. The MapR client automatically determines whether OJAI queries benefit from using the OJAI Distributed Query Service, when the service is available. This section describes the architecture, including the code paths and components involved. It also discusses queries that originate from Drill SQL, which leverage the full functionality of MapR Drill.

The following diagram summarizes the different code paths and the components involved for processing MapR Database JSON queries.



MapR automatically chooses which code path to use.

The following table summarizes the functionality that each code path supports:

Simple OJAI Queries	OJAI Queries Requiring OJAI Distributed Query Service	Analytical Drill Queries
• Can use single secondary index • Configurable limit on sorts • Serial query execution	• Can use multiple secondary indexes • No imposed limit on sort size • Parallel query execution • Query optimization for operational queries	• Can use multiple secondary indexes • No imposed limit on sort size • Parallel query execution • Query optimization for analytical queries

last-updated: Apr 11, 2022

Simple OJAI Queries	The right path in the preceding image represents simple queries issued through an OJAI application. These queries can leverage a single index and operate on smaller data sets that do not benefit from parallel query execution. Queries in this code path do not use the OJAI Distributed Query Service. If the OJAI Distributed Query Service is not installed, all OJAI queries use this code path. When queries run through this code path, sorts on large data sets may fail if the result size exceeds the size of a configurable parameter. See Querying in OJAI Applications on page 2579 for more details.
OJAI Queries Requiring OJAI Distributed Query Service	The middle path represents more complex queries issued through an OJAI application. These queries use the enhanced functionality available in the OJAI Distributed Query Service. This includes sorting large data sets and distributed, parallel query execution. When the Distributed Query Service is installed, the MapR client decides whether a query benefits from the service and automatically redirects the query to the service. The OJAI Distributed Query Service is a lightweight subset of the full Drill query engine that is well suited for the typical queries issued by OJAI. It excludes the more advanced functionality needed by analytical queries. The OJAI Distributed Query Service also provides more sophisticated index selection, including leveraging multiple indexes in a single query. It uses a cost based optimizer to select the indexes that provide the best performance.
Analytical Drill Queries	The left path represents SQL queries issued by BI and SQL tools to Drill. This path leverages Drill's analytical capabilities. It can optimize and process complex queries on large data sets in parallel and provide interactive query response times. The optimizer used by Drill is a superset of the optimizer used by the OJAI Distributed Query Service. It is also a cost based optimizer, but includes more comprehensive optimization techniques needed by complex analytical queries.

For more information about how secondary index selection and execution works in MapR Database JSON, see Selection and Execution of Secondary Indexes on page 582.

Note: Prior to MapR version 6.0.1, the OJAI Distributed Query Service was named the OJAI Query Service.

Data Models

MapR Database can be used as both a document database and a column-oriented database. As a document database, JSON documents are stored in HPE Ezmeral Data Fabric Database JSON table. As a column-oriented database, binary files are in stored HPE Ezmeral Data Fabric Database binary tables.

MapR Database as a Document Database

JSON documents are stored in MapR Database JSON tables. When you create a table in a volume, the table type is specified as JSON. Only JSON-like documents can be stored in JSON tables. Typically, tables of the same type (in this case, JSON) are created in their own volume.

last-updated: Apr 11, 2022

Each document is stored in a table row and indexed by the "_id" field.

JSON Table
{“_id … }
{“_id … }
{“_id … }

MapR Database as a Column-Oriented Database

Data is stored as a collection of key-value pairs where the key serves as a unique identifier. Typically, tables of the same type (in this case, binary) are created in their own volume.

Binary Table			
Row Key	**Column A**	**Column B**	**Column C**
Key 001	Value for A	Value for B	Value for C
Key 002	Value for A	Value for B	Value for C
Key 003	Value for A	Value for B	Value for C

MapR Database as a Document Database

MapR Database supports JSON documents as a native data store. A JSON document is a tree of fields. These JSON documents are stored in MapR Database tables.

MapR Database as a document database, implements JSON documents in MapR Database JSON tables.

- MapR Database JSON tables use the OJAI data model and support the OJAI API.
- Documents are in JSON format; MapR Database stores them in an efficient binary encoding, rather than plain ASCII text.
- With JSON tables, each value has a unique key (`_id`). You identify fields in the document using field paths. For example, `address.street`:

```
{
    "_id": "ID001",
    "name" : "Bob",
    "address": {
        "house" : 123,
        "street": "Main",
        "phones": [
            { "mobile": "555-1234" },
            { "work": "+1-123-456-7890" }]},
    "hobbies": ["badminton", "chess", "beaches"]
}
```

Note: Each JSON document can have different fields.

last-updated: Apr 11, 2022

With JSON document support, you can:

- Store data that is hierarchical and nested, and evolves over time.
- Read and write individual document fields, subsets of fields, or whole documents from and to disk. To update individual fields or subsets of fields, there is no need to read entire documents, modify them, and then write the modified documents to disk.
- Build applications with the MapR Database JSON API library, which is an implementation of the Open JSON Application Interface (OJAI). This is an API library for easily managing complex, evolving, hierarchical data. You can use more data types than the standard types that JSON supports, create complex queries, and access JSON table documents without connection or configuration objects. This allows large-scale applications to manage JSON documents.
- Filter query results within MapR Database before results are returned to client applications.
- Run client applications on Linux, OS X, and Windows systems.
- Perform complex data analysis on your JSON data with Apache Drill or other analytical tools in real time without having to copy data to another cluster.
- Scale your data to span thousands of nodes.
- Control read and write access to single fields and subsets of fields within a JSON table by using access-control expressions (ACEs).
- Control the disk layout of single fields and subdocuments within JSON tables.
- Use Secondary Indexes on page 544 to improve query performance.

JSON Documents
A JSON document is a tree of fields. Each field has a name, type, and value. In the case of an array type, the array field name and array index identify individual elements in the array. Field names are strings. The root of each document is a map. The advantages of JSON documents include the data types it supports and its schema flexibility. MapR Database provides tools that enable you to operate on JSON documents.

An online retailer of sports equipment might have this JSON document for storing data about a set of bicycle pedals:

```
{
    "_id" : "2DT3201",
    "product_ID" : "2DT3201",
    "name" : " Allegro SPD-SL 6800",
    "brand" : "Careen",
    "category" : "Pedals",
    "type" : "Components",
    "price" : 112.99,
    "features" : [
        "Low-profile design",
        "Floating SH11 cleats included"
    ],
    "specifications" : {
        "weight_per_pair" : "260g",
        "color" : "black"
    }
}
```


last-updated: Apr 11, 2022

Data Types

MapR Database JSON documents support a richer set of data types beyond what JSON supports. JSON documents can have scalar data, nested documents, and arrays. MapR Database JSON stores the data in a format that maintains the types. To access JSON documents, you can use the OJAI API. The API exposes data types in a manner specific to the programming language of the API. JSON Document Data Types on page 511 describes each category of types in relation to the sample JSON document shown earlier.

Comparing and Sorting Data Types

Comparisons and sorts of data types differ depending on whether the types are comparable or not. See Using Comparable JSON Document Data Types in Comparisons and Sorts on page 513 and Using Non-comparable JSON Document Data Types in Comparisons and Sorts on page 514 to learn which types fall into each category and to understand their behavior.

Schema Flexibility

The structure of each document, called the document's *schema*, is easy to change. Simply add new fields. For example, if the online retailer wants to allow customers to review products, it is simple to add the reviews to any document for a product.

In this example, highlighted in bold, the `comments` are added as an array with two nested documents:

```
{
    "_id" : "2DT3201",
    "product_ID" : "2DT3201",
    "name" : " Allegro SPD-SL 6800",
    "brand" : "Careen",
    "category" : "Pedals",
    "type" : "Components",
    "price" : 112.99,
    "features" : [
        "Low-profile design",
        "Floating SH11 cleats included"
    ],
    "specifications" : {
        "weight_per_pair" : "260g",
        "color" : "black"
    },
    "comments" : [
        {
            "username" : "hlmencken",
            "comment" : "Best money I ever spent!"
        },
        {
            "username" : "vwoolf",
            "comment" : "What hlmencken said!"
        }
    ]
}
```

Identifying Document Fields

To learn about how to access JSON document fields, see JSON Document Field Paths on page 515. The material includes examples that use the JSON document shown earlier.

Querying Document Fields

MapR Database allows you to specify query conditions in a JSON format using syntax supported by the OJAI API. See OJAI Query Condition Syntax on page 2606 for details.

last-updated: Apr 11, 2022

JSON Document Size

The default maximum size of a JSON document is 32 MB. This size includes the field values in the document, as well as the names, types, and other field metadata. You can configure this size by running the command described at Configuring Maximum Row Sizes Using the CLI on page 1025.

Tools for Working with JSON Documents

These are the tools you can use to create, read, update, and delete JSON documents in MapR Database:

MapR Database Shell

This shell is a light-weight tool for administering, manipulating, and querying JSON tables and documents. Learn more about it at MapR Database Shell (JSON Tables) on page 5286.

OJAI API

The OJAI API provides an interface for creating, reading, updating, and deleting JSON documents.

MapR Database JSON supports the OJAI API in the following languages:

- Java
- Node.js
- Python
- C#
- Go

To learn about how to create, update, and delete JSON documents, see Managing JSON Documents on page 2542. To learn about how to query JSON documents, see Querying in OJAI Applications on page 2579.

For information that is specific to each language, see the following:

Java

See Java OJAI Client API for the complete API.

Node.js

See Using the Node.js OJAI Client on page 2673 for an introduction to this client.

See Node.js OJAI Client API for the complete API.

Python

See Using the Python OJAI Client on page 2678 for an introduction to this client.

See Python OJAI Client API for the complete API.

C#

See Using the C# OJAI Client on page 2688 for an introduction to this client.

See C# OJAI Client API for the complete API.

Go	See Using the Go OJAI Client on page 2692 for an introduction to this client. See Go OJAI Client API for the complete API.
MapR Database JSON REST API	The REST API enables you to use HTTP calls to perform basic operations on MapR Database JSON tables. Learn more about it at Using the MapR Database JSON REST API on page 2696.

JSON Document Data Types

MapR Database JSON documents support a richer set of data types beyond what JSON supports. JSON documents can have scalar data, nested documents, and arrays.

Scalar Data

Scalar data fields can contain strings or numbers. The scalar fields in the sample document are highlighted in bold as follows:

```
{
    "_id" : "2DT3201",
    "product_ID" : "2DT3201",
    "name" : " Allegro SPD-SL 6800",
    "brand" : "Careen",
    "category" : "Pedals",
    "type" : "Components",
    "price" : 112.99,
    "features" : [
        "Low-profile design",
        "Floating SH11 cleats included"
    ],
    "specifications" : {
        "weight_per_pair" : "260g",
        "color" : "black"
    },
    "comments" : [
        {
            "username" : "hlmencken",
            "comment" : "Best money I ever spent!"
        },
        {
            "username" : "vwoolf",
            "comment" : "What hlmencken said!"
        }
    ]
}
```

Scalar fields can contain the following data types:

Data Type	Description
Binary	An uninterpreted sequence of bytes
Boolean	A data type of two possible values that are typically denoted by `true` and `false`
Byte	A 8-bit signed integer that ranges in value from -128 to 127
Date	A 32-bit integer that represents the number of DAYS since epoch, that is. January 1, 1970 00:00:00 UTC. The value is absolute and is time-zone independent.

last-updated: Apr 11, 2022

Data Type	Description
Double	A double-precision 64-bit floating-point number
Float	A single-precision 32-bit floating-point number
Int	A 32-bit signed integer that ranges in value from −2,147,483,648 to 2,147,483,647
Long	A 64-bit signed integer that ranges in value from -(2^{63}) to 2^{63} - 1
Short	A 16-bit signed integer that ranges in value from −32,768 to 32,767
String	A sequence of characters
Time	A 32-bit integer that represents time of the day in milliseconds. The value is absolute and is time-zone independent.
Timestamp	A 64-bit integer that represents the number of milliseconds since epoch, that is, January 1, 1970 00:00:00 UTC. Negative values represent dates before epoch.

Nested Documents

Nested document fields can contain documents that themselves contain scalar data, nested documents, arrays, or a combination of any of these types. The nested documents in the sample document are highlighted in bold as follows:

```
{
    "_id" : "2DT3201",
    "product_ID" : "2DT3201",
    "name" : " Allegro SPD-SL 6800",
    "brand" : "Careen",
    "category" : "Pedals",
    "type" : "Components",
    "price" : 112.99,
    "features" : [
        "Low-profile design",
        "Floating SH11 cleats included"
    ],
    "specifications" : {
        "weight_per_pair" : "260g",
        "color" : "black"
    },
    "comments" : [
        {
            "username" : "hlmencken",
            "comment" : "Best money I ever spent!"
        },
        {
            "username" : "vwoolf",
            "comment" : "What hlmencken said!"
        }
    ]
}
```

Note: Nested documents can also be referred to as *maps*.

A nested document can include subfields that are themselves nested documents. In the following example, `location` is a nested document that has two nested document subfields, `address` and `geoCoordinates`:

```
{
   "_id": "001",
   "location": {
      "address": {
```

```
         "number": 100,
         "street": "Main St.",
         "city": "San Francisco",
         "state": "CA",
         "zipCode": "90210"
      },
      "geoCoordinates": {
         "latitude": 37.7817529521,
         "longitude": -122.39612197
      }
   }
}
```

There is no limit on the number of nestings in a nested document. However, you should consider the extra complexity that additional nestings may add to your applications.

Arrays

Array fields contain lists of values that are accessible by means of index numbers. The values can be scalar, nested documents, arrays, or a combination of any of these types. For example, the following document has two arrays, both highlighted in bold:

- `features`: An array with two scalar strings
- `comments`: An array with two nested documents

```
{
    "_id" : "2DT3201",
    "product_ID" : "2DT3201",
    "name" : " Allegro SPD-SL 6800",
    "brand" : "Careen",
    "category" : "Pedals",
    "type" : "Components",
    "price" : 112.99,
    "features" : [
        "Low-profile design",
        "Floating SH11 cleats included"
    ],
    "specifications" : {
        "weight_per_pair" : "260g",
        "color" : "black"
    },
    "comments" : [
        {
            "username" : "hlmencken",
            "comment" : "Best money I ever spent!"
        },
        {
            "username" : "vwoolf",
            "comment" : "What hlmencken said!"
        }
    ]
}
```

Note: Arrays can also be referred to as *lists*.

Using Comparable JSON Document Data Types in Comparisons and Sorts
: Defines comparable data types and their usage.

  Data types that have a well defined order amongst the types are comparable data types. In a filter condition, if a document's field value and the comparison value are of comparable types, the document

last-updated: Apr 11, 2022

qualifies if the condition returns true. This applies regardless of whether you have created secondary indexes on the comparison fields.

Based on the preceding definition, numeric types are comparable. This includes the following types:

- INT
- SHORT
- LONG
- FLOAT
- DOUBLE

Note: FLOAT and DOUBLE are approximate representations of decimal values. They may not return `true` in equality comparisons against their equivalent decimal values.

Example

Consider the following example where you have four documents, each with a field, `AccountBalance`. The types of the field differ, as noted in the table, but they are all comparable numeric types:

Document Name	`AccountBalance` Field Value	`AccountBalance` Field Type
DOCUMENT1	1900.12	FLOAT
DOCUMENT2	10000	INT
DOCUMENT3	10	LONG
DOCUMENT4	27.88	DOUBLE

If you specify a sort on the field `AccountBalance`, MapR Database sorts the field in the following order:

Document Name	`AccountBalance` Field Value
DOCUMENT3	10
DOCUMENT4	27.88
DOCUMENT1	1900.12
DOCUMENT2	10000

Secondary indexes sort and store data based on the values of the indexed fields. When reading through the index, MapR Database returns the documents in the order of index.

For example, suppose you have an index where `AccountBalance` is the indexed field. A query with the condition, `"AccountBalance > 20"`, returns the documents in the following order if MapR Database processes the query using the index:

- DOCUMENT4
- DOCUMENT1
- DOCUMENT2

Using Non-comparable JSON Document Data Types in Comparisons and Sorts

Defines non-comparable data types and their usage.

Non-comparable data types are data types that do not follow a well-defined order. In contrast to comparisons between comparable types, comparisons between fields and values of non-comparable types

last-updated: Apr 11, 2022

do not qualify even if you perceive a match in values. This is true whether you have indexed the field you are comparing or not.

Arrays and nested documents also fall into the non-comparable category. Since these entities do not have a defined ordering, only equality comparisons on these types are meaningful. For arrays, the order of the array elements must match; for nested documents, all fields in the nested document must match, but the order of the fields is not relevant.

You cannot order on Container Field Paths on page 518. For example, you cannot order on the field `a[].b`, even if the subfield `b` has scalar data.

Example

Consider the following example. If your field, `docField`, has string values and you compare it against a numeric value, none of the string values in the field match the numeric. Likewise, if your field has numeric values and you compare it against a string value, none of the numeric values in the field match the string. Both field and comparison values must be strings or integers to match:

Document Name	Value of Field `docField`	Type of Field `docField`	Filter Condition	Field Value Qualifies Filter Condition?
DOCUMENT1a	23	STRING	`docField = 23`	No
DOCUMENT1b	23	INT	`docField = 23`	Yes
DOCUMENT2a	45	INT	`docField = '45'`	No
DOCUMENT2b	45	STRING	`docField = '45'`	Yes
DOCUMENT3		No type due to missing value	`docField = 23`	No
DOCUMENT4	NULL	No type due to NULL value	`docField = '45'`	No

MapR Database does not define a fixed ordering across non-comparable types. It sorts the values within comparable types and within each non-comparable type, but not across both.

In the previous example, when sorting on `docField`, you could obtain the following for a sort in ascending order:

Document Name	Value of Field `docField`	Type of Field `docField`
DOCUMENT1b	23	INT
DOCUMENT2a	45	INT
DOCUMENT1a	23	STRING
DOCUMENT2b	45	STRING
DOCUMENT4	NULL	No type due to NULL value
DOCUMENT3		No type due to missing value

Note the independent ordering of the integer and string values. Also note that for the rows with NULL and missing field values, the row with NULL appears before the row with a missing value.

JSON Document Field Paths

To access fields in a JSON document, you use a *field path*. The syntax for a field path can vary, depending on the data type you are accessing: nested documents, arrays, nested documents within arrays, and multidimensional arrays.

last-updated: Apr 11, 2022

The examples in this topic reference the following sample JSON document:

```
{
    "_id" : "2DT3201",
    "product_ID" : "2DT3201",
    "name" : " Allegro SPD-SL 6800",
    "brand" : "Careen",
    "category" : "Pedals",
    "type" : "Components",
    "price" : 112.99,
    "features" : [
        "Low-profile design",
        "Floating SH11 cleats included"
    ],
    "specifications" : {
        "weight_per_pair" : "260g",
        "color" : "black"
    },
    "comments" : [
        {
            "username" : "hlmencken",
            "comment" : "Best money I ever spent!"
        },
        {
            "username" : "vwoolf",
            "comment" : "What hlmencken said!"
        }
    ]
}
```

In the simplest case, the field path is the name of the field and refers to the entire field.

Nested Documents

If a field is a nested document, specifying the nested document identifies the entire nested document.

To identify individual fields in a nested document, you use a *dot notation* to specify their paths. A field path is a sequence of field names that leads to the particular field that you are interested in. The names are separated by dots.

The following shows a document with multiple levels of nested documents:

```
{
   "a" : {
       "b" : {
            "c" : {
                "d" : "value_for_d"
            }
        }
    }
}
```

The field path for field d using dot notation is `a.b.c.d`.

The following table shows examples of field paths using dot notation for the sample JSON document:

Field Path	Value Returned
`specifications`	`{ "specifications": {"color":"black","weight_per_pair":"260g"} }` The entire nested document field `specifications`
`specifications.weight_per_pair`	`{"specifications":{"weight_per_pair":"260g"}}` The `weight_per_pair` subfield in `specifications`
`specifications.color`	`{"specifications":{"color":"black"}}` The `color` subfield in `specifications`

Arrays

If the field is an array, specifying the array's field name identifies the entire array.

To access an element in an array, specify the position of the element in the array, starting at offset zero.

The following table shows examples of field paths that reference arrays for the sample JSON document:

Field Path	Value Returned
`features`	`{ "features":[ "Low-profile design", "Floating SH11 cleats included" ] }` The entire `features` array
`features[0]`	`{"features":["Low-profile design"]}` The first element of the `features` array
`features[1]`	`{ "features": [null,"Floating SH11 cleats included"] }` The second element of the `features` array **Note:** `null` is shown in the first element of the array to signify that the element returned is the second entry from the array.
`comments[0]`	`{ "comments": [{"comment":"Best money I ever spent!","username":"hlmencken"}] }` The first element of the `comments` array, which is a nested document

last-updated: Apr 11, 2022

Container Field Paths

Starting in MapR 6.1, MapR Database introduces the notion of a *container field path*. Using a container field path, you can access a field that is either a single value or an arbitrary array element.

If you have a field that has a single value (rather than an array of values), when using a container field path, MapR Database treats the single value as an array with one element. This enables you to use a container field path to access a field that has both array elements and scalar values. The array elements and scalar values can be of any type.

To specify a container field path, place square brackets after the field name:

```
fieldName[]
```

A container field path is useful if you want to perform one of the following scenarios:

- Perform comparisons on a field path that is either a single value or an arbitrary array element
- Access subfields in a nested document, where the nested document is either an arbitrary array element or a single nested document
- Access arbitrary elements in an array

See OJAI Query Conditions Using Container Field Paths on page 2615 for more details about the first scenario. The next two sections describes the second and third scenarios.

Nested Documents Within Arrays

Array elements can be nested documents. You can reference individual subfields within these nested documents with container field paths, starting in MapR 6.1. If you have a field that has a single value (rather than an array of values), if you use a container field path, MapR Database treats the single value as an array with one element. This enables you to use a container field path to access a field that has both array elements and scalar values.

For example, suppose you have the following two JSON documents in a MapR Database table, and `addresses` has an array of nested documents in the first document and a nested document in the second document:

```
{
  "_id":"1",
  "addresses":[
    {"state":"CA","city":"SJ"},
    {"state":"CA","city":"SC"},
    {"state":"WA","street":"NE 39th"}
  ]
}
{
  "_id":"2",
  "addresses":{"state":"CA","city":"SJ"}
}
```

You can use `addresses[].state` to reference the `state` subfield across all nested documents in both documents.

The following table describes the field paths supported and what each field path returns:

Field Path	Value Returned *(Number in Description Corresponds to Document ID)*

`addresses`	`{`
	`  "addresses": [`
	`  {"city":"SJ","state":"CA"},`
	`  {"city":"SC","state":"CA"},`
	`  {"state":"WA","street":"NE 39th"}]`
	`}`
	`{`
	`  "addresses":{"city":"SJ","state":"CA"}`
	`}`
	1. The array containing three nested documents
	2. The single nested document

`addresses.city`	`{}`
	`{"addresses":{"city":"SJ"}}`
	1. Empty because `addresses` is not a nested document
	2. The `city` subfield in the nested document

`addresses[0]`	`{`
	`  "addresses":`
	`  [{"city":"SJ","state":"CA"}]`
	`}`
	`{}`
	1. The first element in the `addresses` array
	2. Empty because `addresses` is not an array

`addresses[2].state`	`{`
	`  "addresses":`
	`  [null,null,{"state":"WA"}]`
	`}`
	`{}`
	1. The `state` subfield from the nested document in the third element of the `addresses` array
	Note: `null` is shown in the first two elements of the array to signify that the element returned is the third entry from the array
	2. Empty because `addresses` is not an array

`addresses[0].state, addresses[0].city`	`{"addresses":[{"city":"SJ","state":"CA"}]}`
	`{}`
	1. The `city` and `state` subfields from the nested document
	2. Empty because `addresses` is not an array

last-updated: Apr 11, 2022

Field Path	Value Returned *(Number in Description Corresponds to Document ID)*
`addresses[].city` **Note:** Supported starting in MapR 6.1	`{ "addresses": [ {"city":"SJ"}, {"city":"SC"}, {} ] } { "addresses": {"city":"SJ"} }` **1.** An array of nested documents with a `city` subfield; the third array element is empty because the third nested document does not have a `city` subfield **2.** A single nested document with a `city` subfield
`addresses[].state, addresses[].city` **Note:** Supported starting in MapR 6.1	`{ "addresses": [ {"city":"SJ","state":"CA"}, {"city":"SC","state":"CA"}, {"state":"WA"} ] } { "addresses": {"city":"SJ","state":"CA"} }` **1.** An array of nested documents with `city` and `state` subfields; the third array element has only a `state` subfield because the third nested document does not have a `city` subfield **2.** A single nested document with `city` and `state` subfields

Container Field Paths Across Multiple Levels of Nested Documents

You can use container field paths at any level of a nested document.

For example, suppose you have the following document:

```
{
   "_id": "account001",
   "projects": [
      {
         "id": "proj001",
         "manager": { "name": "Guy Bones", "email": "gbones@pro.com" },
         "customer": {
            "name": "My Company",
            "contacts": [
               {
               "id": "user_jdoe",
               "emails": [
                  { "type": "work", "value": "jdoe@comp.com" },
```

```
                  { "type": "personal", "value": "jdoe@gmail.com" }
               ],
               "addresses": [
                  {
                     "type": "work",
                     "value": {"street":"21 King Av","city": "Redwood",
"zip": 94065, "state": "CA" }
                  }
               ],
               "phones": [
                  { "type": "cell", "value": "+16505556764" },
                  { "type": "office", "value": "+14075556764" }],
               "role": "CEO"
               },
               {
               "id": "user_simsom",
               "emails": [
                  { "type": "work", "value": "simson@comp.com" },
                  { "type": "personal", "value": "simson@gmail.com" }
               ],
               "addresses": [
                  {
                  "type": "work",
                  "value": {"street":"21 King Av.","city":"Redwood","zip":
94065,"state": "CA" }
                  }
               ],
               "phones": [
                  { "type": "cell", "value": "+16505556777" },
                  { "type": "office", "value": "+1407555444" }],
               "role": "PM"
               }
            ]
         }
      },
      {
         "id": "proj002",
         // ...
      }
   ]
}
```

The following table shows field paths that use the container field paths across multiple nested documents and the values returned:

Field Path	Value Returned

`projects[].customer.contacts[].emails[].value`	<pre>{
	  "projects":[
	    {
	    "customer":{
	      "contacts":[
	        {
	          "emails":[
	            {"value":"jdoe@comp.com"},
	            {"value":"jdoe@gmail.com"}
	          ]
	        },
	        {
	          "emails":[
	
	{"value":"simson@comp.com"},
	
	{"value":"simson@gmail.com"}
	          ]
	        }
	      ]
	    },
	    { // data for proj002 }
	  ]
	}</pre>

`projects[].customer.contacts[].role`	<pre>{
	  "projects":[
	    {
	      "customer":
	        {
	          "contacts":[
	            {"role":"CEO"},
	            {"role":"PM"}
	          ]
	        }
	    },
	    { // data for proj002 }
	  ]
	}</pre>

Multidimensional Arrays

Arrays can have more than one dimension.

For example, suppose you want to store the high and low temperatures by week. The following document contains the high and low temperatures in Fahrenheit for the seven days beginning on April 29th, 2018. The document uses a two-dimensional array to store the high and low temperatures for each day. The first element of each nested array element is the high temperature for a day, and the second element is the low:

```
{
   "_id" : "001",
   "temps" : [[61,49],[74,51],[75,51],[74,52],[78,54],[75,53],[75,54]],
   "weekOf" : "4/29/2018"
}
```

To access individual high or low temperatures by day, you specify a two-dimensional array element with the desired array indexes. To access a pair of high and low temperatures, you specify a single array index.

last-updated: Apr 11, 2022

Field Path	Value Returned
`temps[0]`	`{"temps":[[61,49]]}`
`temps[5][1]`	`{"temps":[null,null,null,null,null,[null,53]]}` **Note:** `null` is shown for all array elements preceding the desired element

There is no limit on the number of dimensions in an array.

Container Field Paths with Multidimensional Arrays

Starting in MapR 6.1, a container field path can refer to arbitrary array elements across multiple array dimensions. To reference arbitrary elements in the two-dimensional `temps` array shown earlier, you specify:

```
temps[][]
```

Extending the convention by which a container field path with one set of square brackets treats a scalar value as an array with one element, a container field path with two square brackets treats a one-dimensional array as a two-dimensional array with a single element, where the element is that one-dimensional array.

For example, in the following document, although `temps` has only a single array, you can use `temps[][]` to refer to either the high or low temperature in the array:

```
{
   "_id" : "002",
   "temps" : [81,60],
   "weekOf" : "5/12/2018"
}
```

The same convention applies across *N* dimensions. A container field path with *N* square brackets treats an *(N-1)*-dimensional array as the only element in an *N*-dimensional array.

You can also use the container field paths for a subset of dimensions, provided a dimension that specifies container field path does not precede a dimension that specifies an explicit element. The following table illustrates this:

Field Path	Value Returned

`temps[0][]`	`{"temps":[[61,49]]}`
	`{"temps":[81,60]}`
	The temperatures on the first day of the week

`temps[2][]`	`{"temps":[null,null,[75,51]]}`
	`{"temps":[]}`
	The temperatures on the third day of the week

`temps[]`	`{"temps":[[61,49],[74,51],[75,51],[74,52],[78,54],[75,53],[75,54]]}`
	`{"temps":[81,60]}`
	The temperature pairs across all days

last-updated: Apr 11, 2022

Field Path	Value Returned
`temps[][0]`	Disallowed because the container field path in the first dimension precedes element 0 in the second dimension

MapR Database JSON Tables

JSON documents are stored in MapR Database JSON tables. MapR Database supports schema flexibility in the documents and provides the tools to efficiently access them. It optimizes the storage of the JSON documents, providing high performance.

When a JSON document is added to a JSON table, it is put in a table row. The table row is part of one column family (although you can create more, as described in Column Families in JSON Tables on page 527). The value in the row is a single JSON document that is stored in a binary format. The binary format allows MapR Database to make a number of optimizations to the document's layout to make data access fast and efficient. MapR Database also maintains the data types associated with fields in a JSON document.

The JSON documents in a table need not have identical structures. It is possible to include in a table any number of JSON documents that have no common fields or share only a subset of fields.

For example, an online retailer might have the following three documents in a single JSON table. Only a subset of fields is common to all three documents. These are key differences:

- Each document has a different nested document in a field named `specifications`.
- Only two of the documents have arrays in the field `features`.
- The `retailers` field has different types in the first and third documents.

Document 1

```
{
    "_id" : "ID1",
    "product_ID" : "4GGC859",
    "name" : "Thresher 1000",
    "brand" : "Careen",
    "category" : "Bicycle",
    "type" : "Road bicycle",
    "price" : 2949.99,

    "specifications" : {
        "size" : "55cm",
        "wheel_size" : "700c",
        "frameset" : {
            "frame" : "Carbon Enduro",
            "fork" : "Gabel 2"
        },
        "groupset" : {
            "chainset" : "Kette 230",
            "brake" : "Bremse
FullStop"
      },
        "wheelset" : {
            "wheels" : "Rad Schnell
10",
            "tyres" : "Reifen Pro"
        }
    },

    "retailers": {
        "name" : "Eden Bicycles",
        "location" : {
```


last-updated: Apr 11, 2022

```
            "city" : "Castro Valley",
            "state" : "CA"
        }
    }
}
```

Document 2

```
{
    "_id" : "ID2",
    "product_ID" : "2DT3201",
    "name" : " Allegro SPD-SL 6800",
    "brand" : "Careen",
    "category" : "Pedals",
    "type" : "Components",
    "price" : 112.99,

    "features" : [
        "Low-profile design",
        "Floating SH11 cleats
included"
    ],

    "specifications" : {
        "weight_per_pair" : "260g",
        "color" : "black"
    }
}
```

Document 3

```
{
    "_id" : "ID3",
    "product_ID" : "3ML6758",
    "name" : "Trikot 24-LK",
    "brand" : "Careen",
    "category" : "Jersey",
    "type" : "Clothing",
    "price" : 76.99,

    "features" : [
        "Wicks away moisture.",
        "SPF-30",
        "Reflects light at night."
    ],

    "specifications" : {
        "sizes" :
["S","M","L","XL","XXL"],
        "colors" : [
            "white",
            "navy",
            "green"
        ]
    },

    "retailers" : [
        {
            "name" : "Bespoke Cycles",
            "city": "San Francisco",
            "state" : "CA"
        },
        {
            "name" : "Trek Bicycle",
```

```
            "city" : "New York",
            "state" : "NY"
        }
    ]
}
```

Container Syntax

Starting in MapR Database 6.1, even though the `retailers` field is an array of nested documents in document 1 and a nested document in document 3, you can reference subfields of the nested documents in both documents using the following container syntax:

```
retailers[].name
```

Specifying that field reference returns the following for the three documents:

```
{
    "retailers":{"name":"Eden Bicycles"}
}
{}
{
    "retailers":[
        {"name":"Bespoke Cycles"},
        {"name":"Trek Bicycle"}]
}
```

Note: An empty document is returned for the second document because that document does not have a `retailers` field.

See Container Field Paths on page 518 for more information.

Table Paths

Tables are stored in the MapR filesystem. When providing the path to a table in MapR tools and APIs, use these conventions:

- For a path on the local cluster, start the path at the volume mount point. For example, for a table named `test` under a volume with a mount point at `/volume1`, specify the following path: `/volume1/test`
- For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customer` in `volume1` in the `sanfrancisco` cluster, specify the following path: `/mapr/sanfrancisco/volume1/customer`

Tools for Creating and Administering JSON Tables

These are the tools available for creating and administering JSON tables in MapR Database:

MapR Database Shell	This shell is a light-weight tool for manipulating JSON tables and documents. Learn more about it at MapR Database Shell (JSON Tables) on page 5286.
MapR Database JSON Client API	This API allows you to manage MapR Database JSON tables. The API includes methods to create, alter, and drop tables and column families. Learn more about these APIs at Managing JSON Tables on page 2522.
Python OJAI Client	This API allows you to create and drop MapR Database JSON tables in Python. Learn more about it at Using the Python OJAI Client on page 2678.

MapR Database JSON REST API	The REST API allows you to create and drop MapR Database JSON tables using HTTP calls. Learn more about it at Using the MapR Database JSON REST API on page 2696.
MapR Database JSON utilities	MapR Database JSON supports several utilities for loading tables. Learn more about these utilities at Loading Documents into JSON Tables on page 1036.
maprcli commands	The `maprcli table` commands fully support JSON tables. See table.

Note: For a list of tools available to query and manage documents in MapR Database JSON tables, see Tools for Working with JSON Documents on page 510.

Column Families in JSON Tables

JSON tables store data in column families. A column family is a collection of fields that are stored together on disk. You can use column families to improve the performance of your queries.

Each table has a default column family, which is default storage for all fields in the documents of a table. You can create additional column families to store data for a collection of fields in a separate location on disk. Queries and other operations that only run on the data stored in a column family are more efficient and better performing than queries on the same data when that data is stored with other data in a table. You can also cache values from a column family in memory.

Default Column Families

Suppose you have three JSON documents in a table and all three documents have the field a.

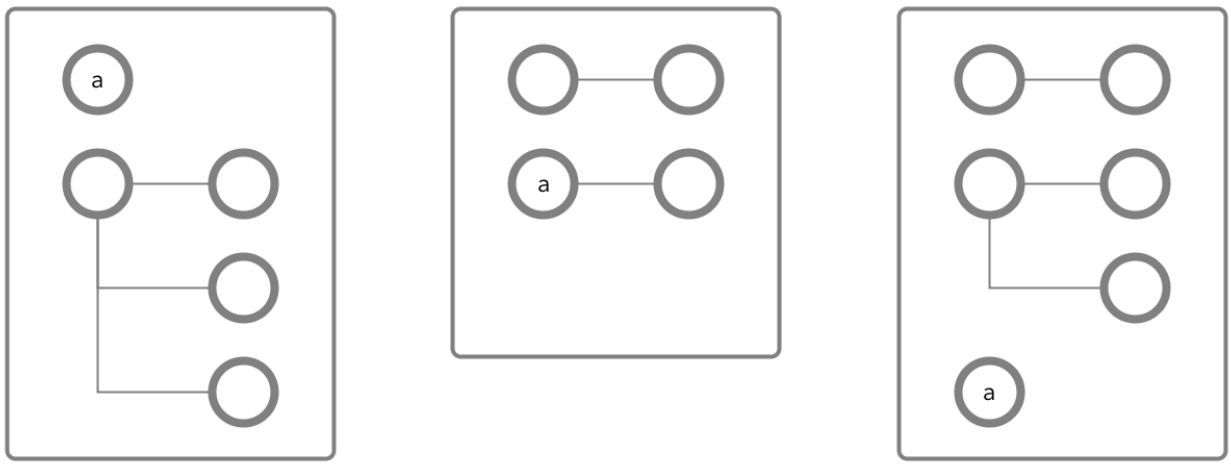


Figure 1: Schematic diagrams of three JSON documents, showing fields but not values, each document with a field named a

At this point, you have not created any non-default column families. So, all of the data in the table resides in the *default* column family. Each JSON table is created with a *default* column family.

Using Column Families to Optimize Data Access

To optimize data access for your applications, you plan to place some data that will be heavily queried in a new column family at path `a.b`, where `b` is a field that does not exist yet. Fields do not have to exist before you create column families on them.

last-updated: Apr 11, 2022

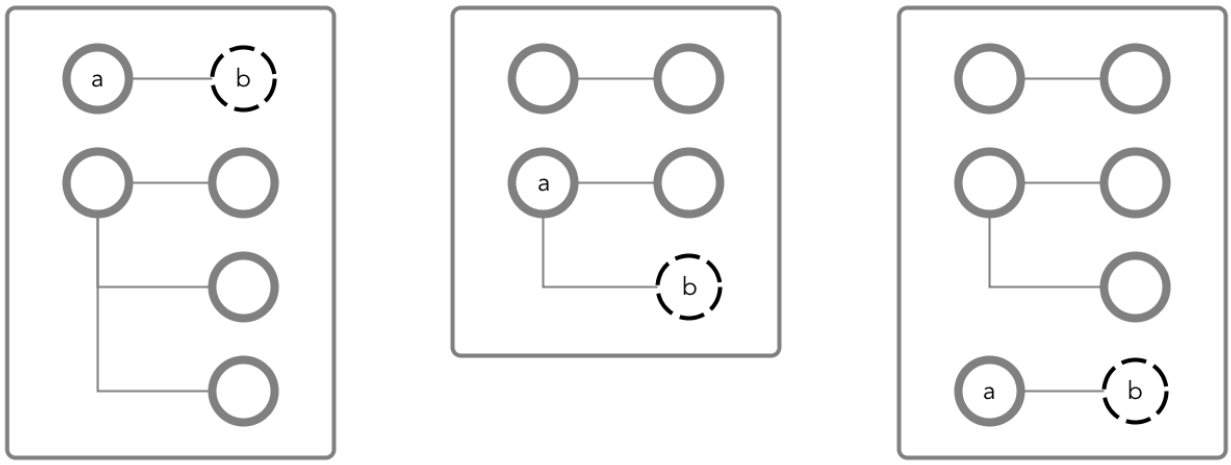


Figure 2: The same three JSON documents, showing where the new column family will be created

You create a column family at the path `a.b` with the name `CF1`.

When you create field `b`, it will belong to the column family `CF1`. All values of `b`, as well as the values of all fields that might be created after `b`, will be stored together on disk. Applications can read data directly from this column family and avoid reading the rest of the document at the same time, making queries faster and more efficient.

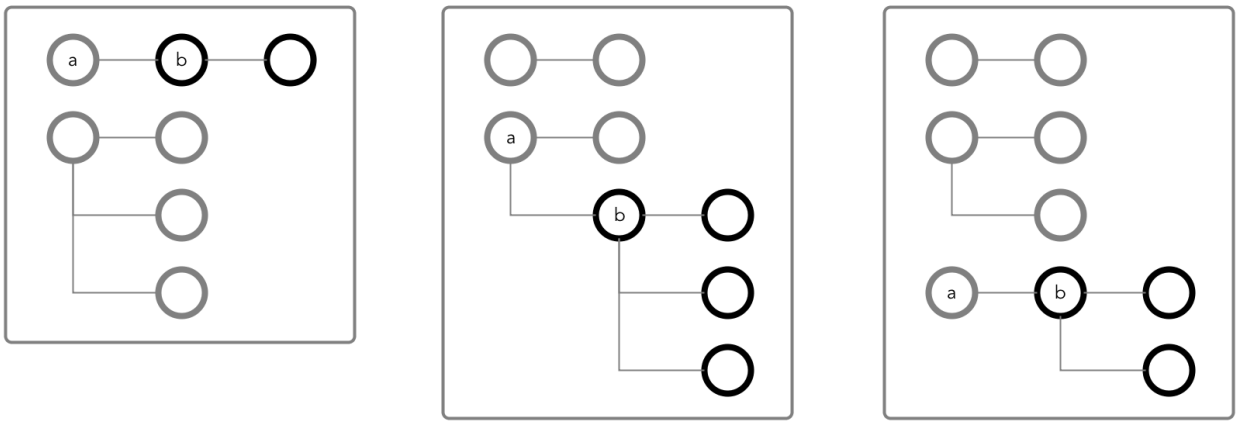


Figure 3: The three JSON documents with column family `CF1` in black

Creating Multiple Column Families

You can create up to 64 column families in a JSON table. The column families can be at any location in your documents. For example, these two documents both use the same non-default column families at the paths `a.b`, `a.b.c`, and `d`.

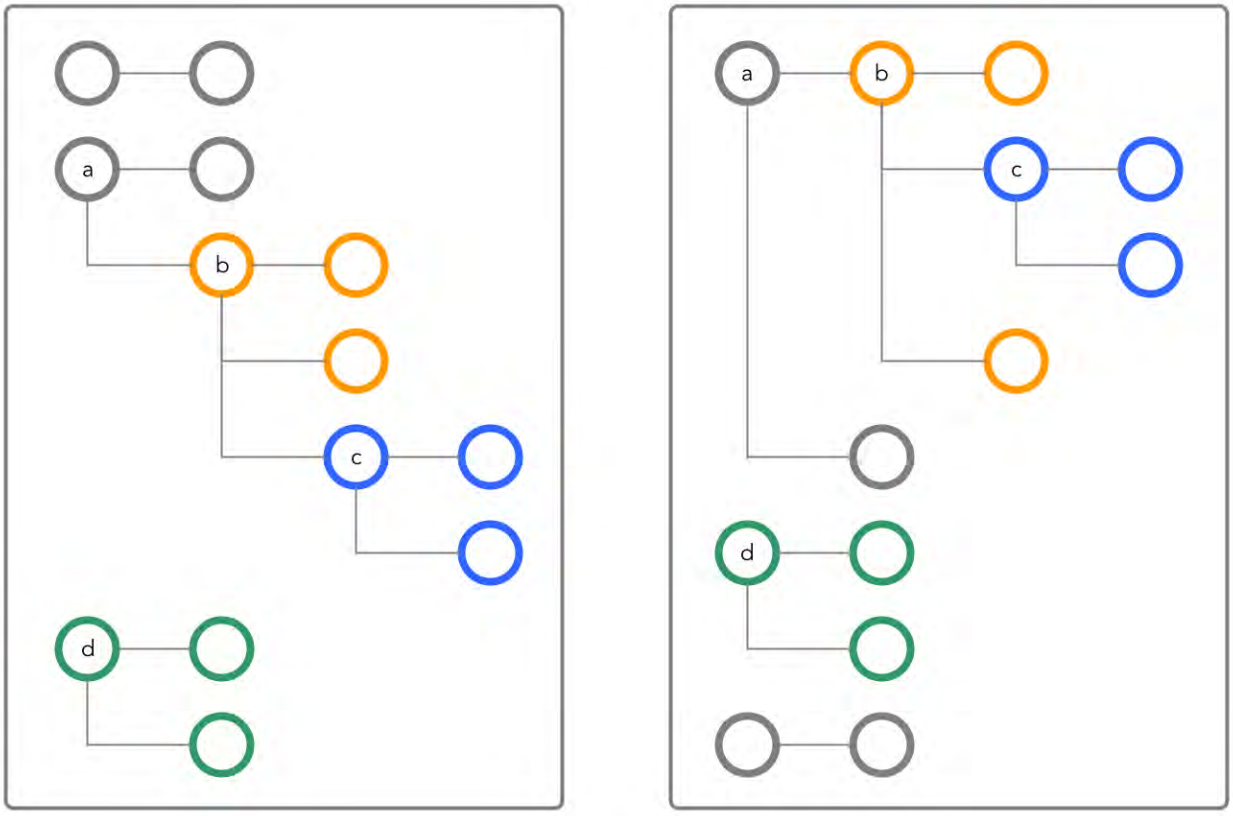


Figure 4: Two JSON documents that use the same non-default column families are highlighted in orange, blue, and green

Column Family Best Practices

If the path at which you want to create a column family already exists, it is recommended that the path and any fields under it contain no data. After the conversion of the path to a column family, it is possible that data existing in the path before the conversion could become inaccessible.

Applications and Column Families

Applications do not need to be aware of the existence of column families. They perform CRUD operations using the paths of fields in a document. For example, to update any of the fields under `a.c`, an application does not need to be aware that the field is in the column family at the path `a.c`. The application simply moves through the document along the path to the field.

Column Family Limitation

You cannot define column families across array type fields, for example:

```
maprcli table cf create -path /tbl-mcf -cfname abc -force true -jsonpath
a.b[0]
ERROR (22) -  Malformed path "a.b[0]", valid format is like "a.b.c".
```

For information about array fields, see JSON Document Field Paths.

Security on JSON Tables

By using access control expressions (ACEs), you can grant or deny access to fields and column families that are in JSON tables.

For an explanation of the syntax of ACEs, see ACE Syntax on page 1448.

last-updated: Apr 11, 2022

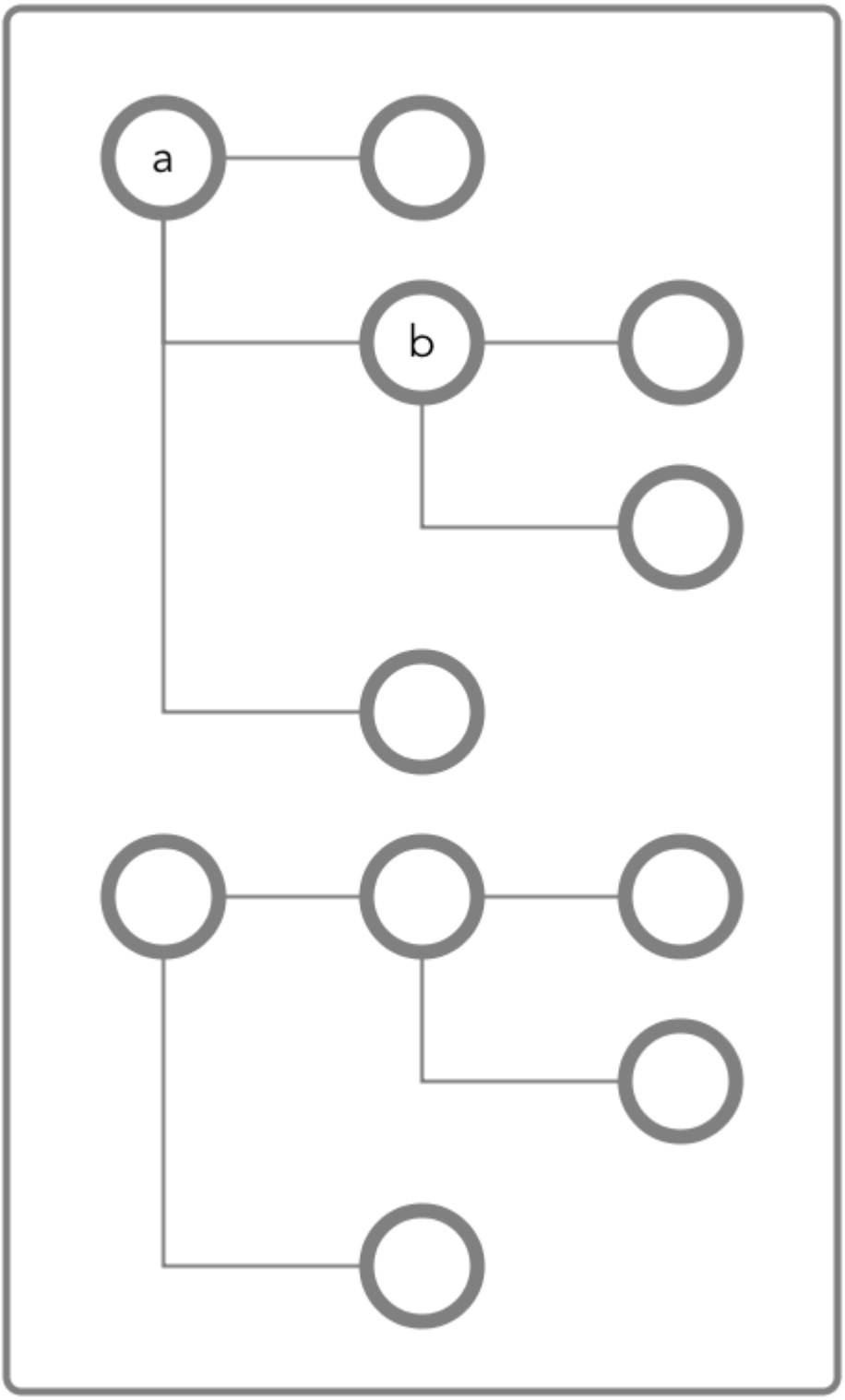
There are three types of permission:

- Traverse (`traverseperm`)
- Read (`readperm`)
- Write (`writeperm`)

Traverse (`traverseperm`)

This permission allows the grantee to descend a hierarchy of fields to access the fields to which the grantee has write or read permission.

For example, suppose that a user has read and write access only on field b in this document.

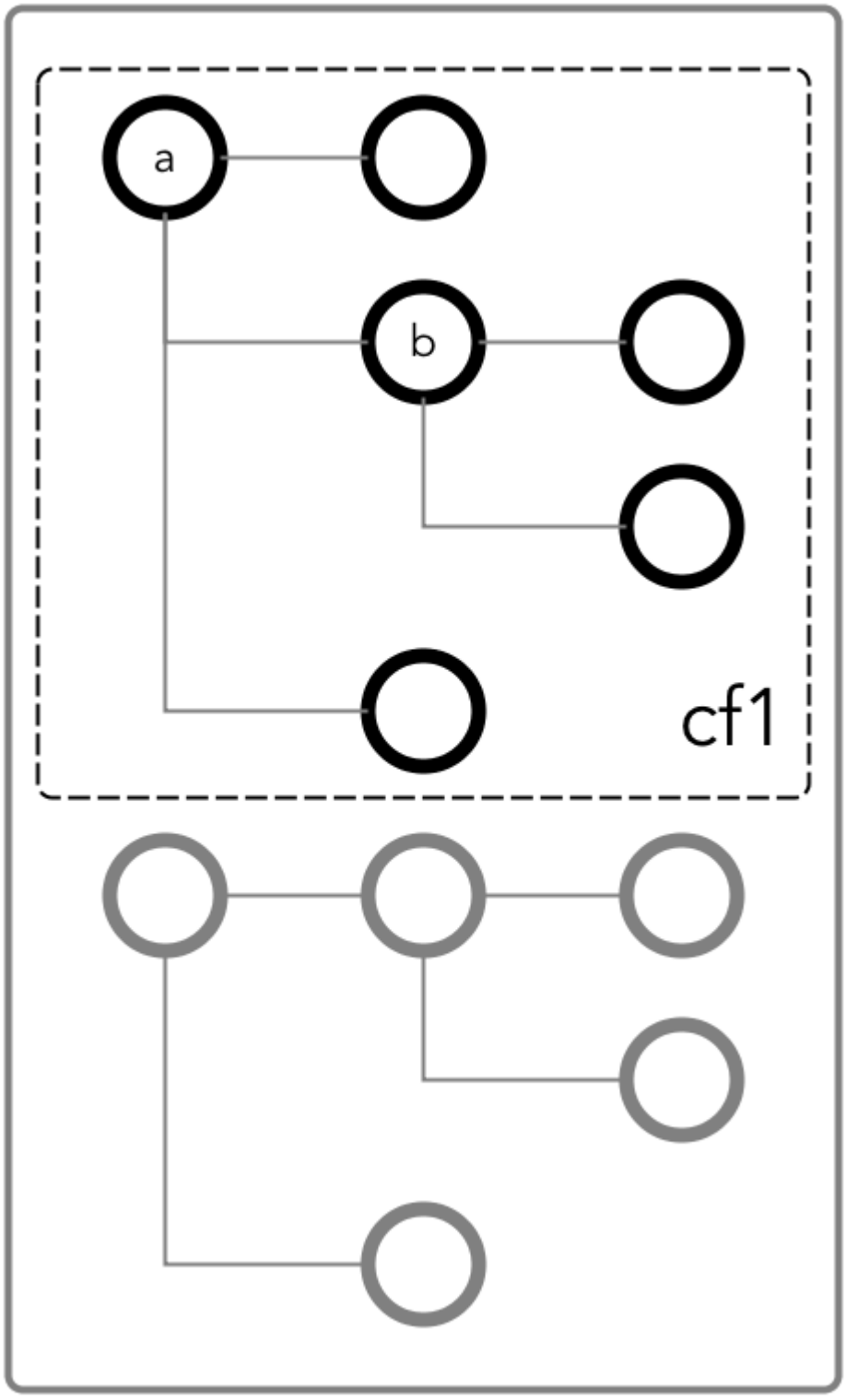


To access field b, the user would need to be able to traverse (pass through) field a. In this case, as the entire document is in the default column family, the user could be granted traverse permission on the default column family. Field a would inherit the traverse permission.

If the user is denied the traverse permission on the default column family, the user cannot access field b. Granting traverse permission on field a in this case has no effect.

last-updated: Apr 11, 2022

In the next example, field a is a column family named `cf1`.



To be able to read and write at field b, the user could be granted the traverse permission on the column family.

Read (`readperm`)

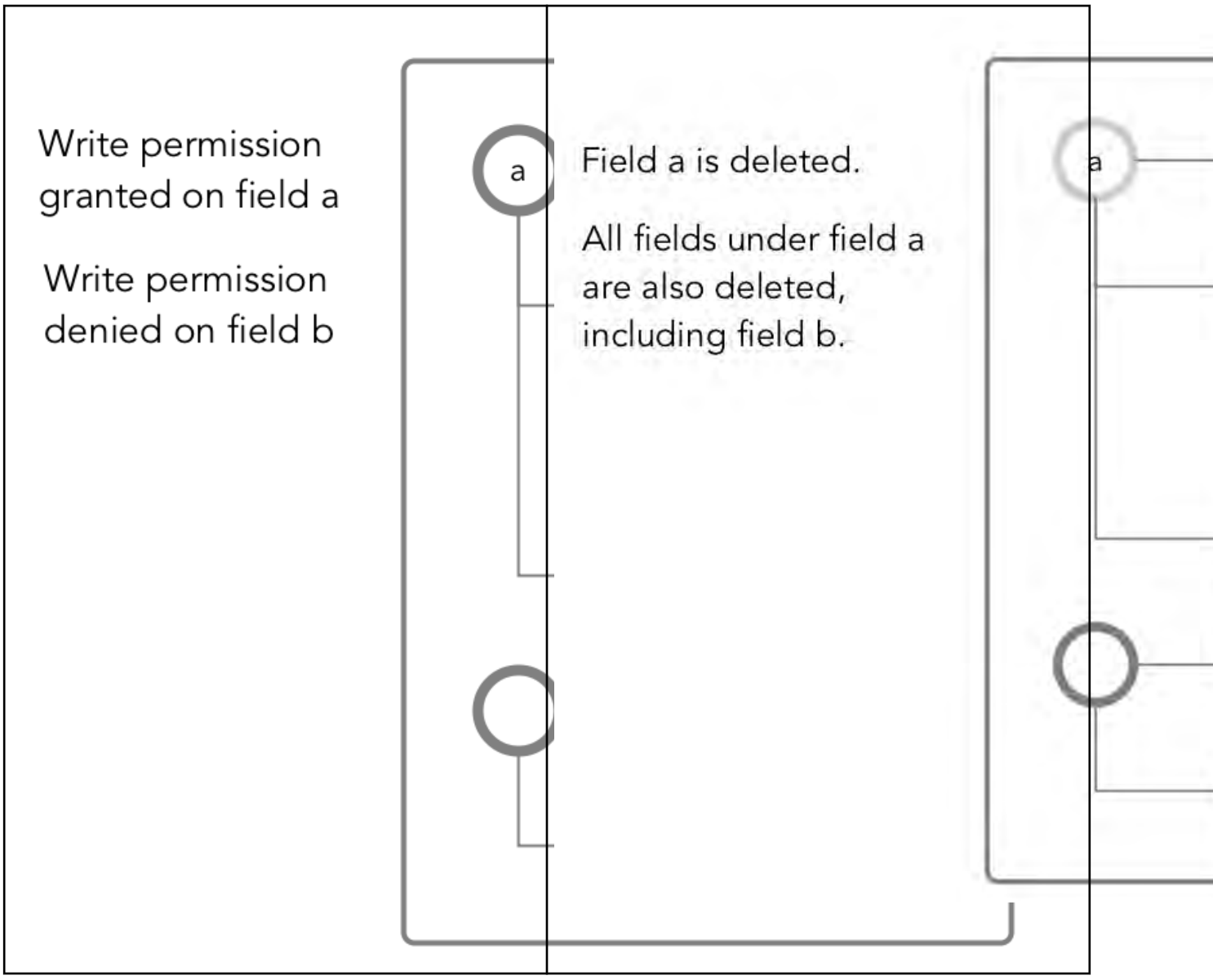
This permission allows the grantee to read from a field.

This permission extends to fields that are nested below the field that was granted permission. However, grantees can be explicitly denied the permission on any of the nested fields.

Write (`writeperm`)

This permission allows the grantee to delete a field, insert a value into a field, or overwrite a field's value.

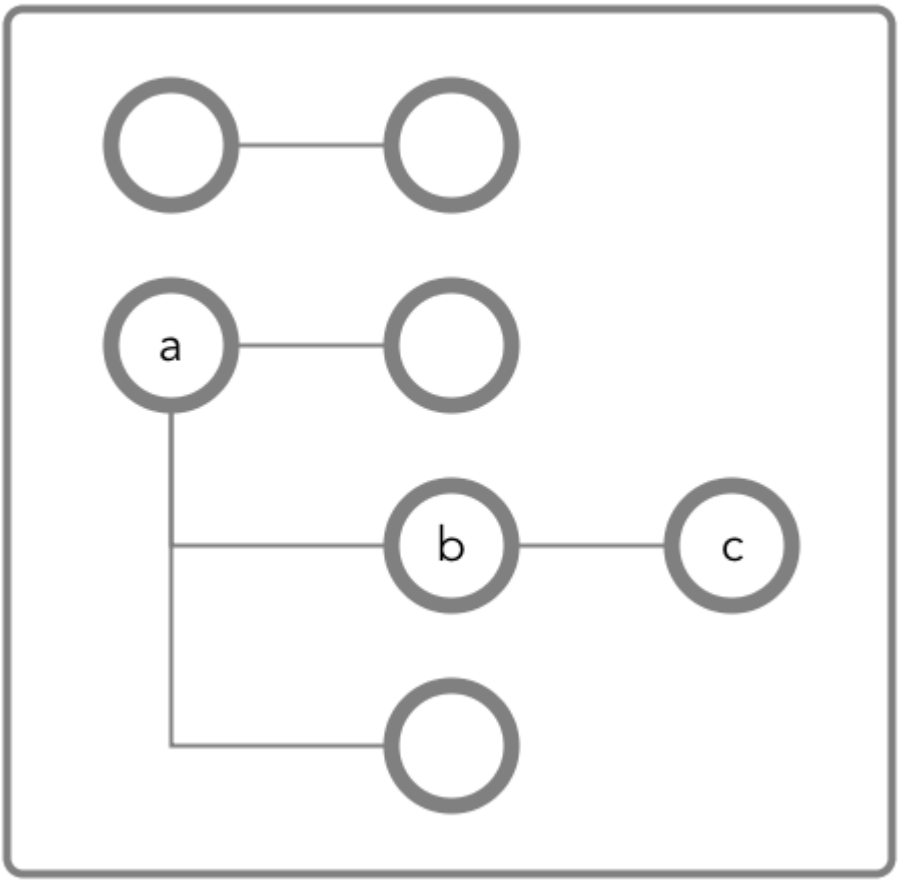
As illustrated in the following two diagrams, deleting a field also deletes all fields that are nested within that field, even those fields on which the write permission is explicitly denied.



Permissions on the Default Column Family
If a JSON document field is in the MapR Database JSON default column family, you must have `readperm` and `writeperm` permissions to perform read and write operations on the field. You either receive the permissions from the default column family, inherit them from the field's parent field, or have the permissions from an explicit grant on the field.

The following diagram shows a JSON document where all fields are in the default column family.

last-updated: Apr 11, 2022



Granting Read and Write Permissions on Field c

To perform both read and write operations on field c, when it is in the default column family, you must have both `readperm` and `writeperm` access on field c:

- If you have `readperm` and `writeperm` permissions on the default column family, then you have access to field c.
- If you have `readperm` and `writeperm` permissions on field b, then you have access to field c. You do not need any further permissions. Field c inherits your `readperm` and `writeperm` permissions from field b.
- If you have `readperm` and `writeperm` permissions on the default column family *but* either field a or b denied you permissions:
  - You must have `traverseperm` permission granted to you on the field that denied you access (field a or b).
  - You must have `readperm` and `writeperm` permissions explicitly granted to you on field c.
- If you do *not* have `readperm` and `writeperm` permissions on the default column family:
  - You must have `traverseperm` permission granted to you on either the default column family or field b.
  - You must have `readperm` and `writeperm` permissions explicitly granted to you on field c.

The following are examples of commands that grant these permissions:

```
/opt/mapr/bin/maprcli table cf colperm set
  -path <path to JSON table >
  -cfname default
  -name a.b
  -traverseperm u:<user ID> | <existing ACE for this field>
```

```
/opt/mapr/bin/maprcli table cf colperm set
  -path <path to JSON table >
  -cfname default
  -name a.b.c
  -readperm u:<user ID> | <existing ACE for this field>
  -writeperm u:<user ID> | <existing ACE for this field>
```

```
/opt/mapr/bin/maprcli table cf edit
  -path <path to JSON table >
  -cfname default
  -traverseperm u:<user ID> | <existing ACE for this field>
```

```
/opt/mapr/bin/maprcli table cf colperm set
  -path <path to JSON table >
  -cfname default
  -name a.b.c
  -readperm u:<user ID> | <existing ACE for this field>
  -writeperm u:<user ID> | <existing ACE for this field>
```

Granting Read or Write Permission on Field c

To perform either read or write operations on field c, when it is in the default column family, you must have either `readperm` or `writeperm` access on field c:

- If you have the same permission (`readperm` or `writeperm`) on the default column family, then you have access to field c.
- If you have the same permission (`readperm` or `writeperm`) on field b, then you have access to field c. You do not need any further permissions. Field c inherits your `readperm` or `writeperm` permission from field b.
- If you have the same permission (`readperm` or `writeperm`) on the default column family *but* either field a or b denied you permission:
  - You must have `traverseperm` permission granted to you on the field that denied you access (field a or b).
  - You must have `readperm` or `writeperm` permission explicitly granted to you on field c.
- If you do *not* have the same permission (`readperm` or `writeperm`) on the default column family:
  - You must have the `traverseperm` permission granted to you on either the default column family or field b.
  - You must have `readperm` or `writeperm` permission explicitly granted to you on field c.

The following example grants `traverseperm` permission:

```
/opt/mapr/bin/maprcli table cf colperm set
  -path <path to JSON table>
```


last-updated: Apr 11, 2022

```
   -cfname default
   -name a.b
   -traverseperm u:<user ID> | <existing ACE for this field>
```

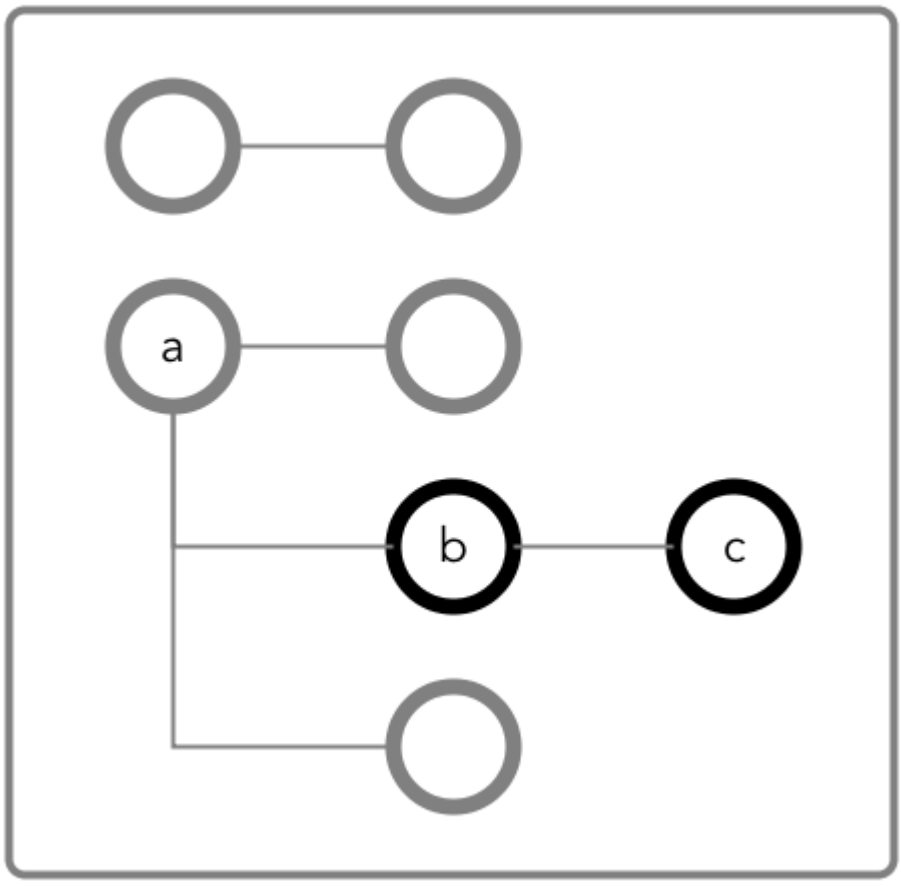
The following example grants `readperm` permission:

```
/opt/mapr/bin/maprcli table cf colperm set
   -path <path to JSON table>
   -cfname default
   -name a.b.c
   -readperm u:<user ID> | <existing ACE for this field>
```

Permissions on Non-default Column Families
If a JSON document field is not in the MapR Database JSON default column family, you must have `readperm` and `writeperm` permissions to perform read and write operations on the field. You either receive the permissions from the column family, inherit them from the field's parent field, or have the permissions from an explicit grant on the field.

Note: Non-default column families are an advanced feature of MapR Database's native JSON support. For more information, see Managing Column Families on page 2538.

The following diagram shows a JSON document where fields b and c are in a column family `cf1` that is defined at field b with the path `a.b`.



Granting Read and Write Permissions on Field c

To perform both read and write operations on field c, when it is in column family `cf1`, you must have both `readperm` and `writeperm` access on field c:

- If you have `readperm` and `writeperm` permissions on `cf1`, then you have access to field c.
- If you have `readperm` and `writeperm` permissions on field b, then you have access to field c. You do not need any further permissions. Field c inherits your `readperm` and `writeperm` permissions from field b.
- If you have `readperm` and `writeperm` permissions on `cf1` *but* either field a or b denied you permissions:
  - You must have `traverseperm` permission granted to you on the field that denied you access (field a or b).
  - You must have `readperm` and `writeperm` permissions explicitly granted to you on field c.
- If you do *not* have `readperm` and `writeperm` permissions on `cf1`:
  - You must have `traverseperm` permission granted to you on either `cf1` or field b.
  - You must have `readperm` and `writeperm` permissions explicitly granted to you on field c.

The following are examples of commands that grant these permissions:

```
/opt/mapr/bin/maprcli table cf colperm set
  -path <path to JSON table >
  -cfname cf1
  -name a.b
  -traverseperm u:<user ID> | <existing ACE for this field>
```

```
/opt/mapr/bin/maprcli table cf colperm set
  -path <path to JSON table >
  -cfname cf1
  -name a.b.c
  -readperm u:<user ID> | <existing ACE for this field>
  -writeperm u:<user ID> | <existing ACE for this field>
```

```
/opt/mapr/bin/maprcli table cf edit
  -path <path to JSON table >
  -cfname cf1
  -traverseperm u:<user ID> | <existing ACE for this field>
```

```
/opt/mapr/bin/maprcli table cf colperm set
  -path <path to JSON table >
  -cfname cf1
  -name a.b.c
  -readperm u:<user ID> | <existing ACE for this field>
  -writeperm u:<user ID> | <existing ACE for this field>
```

Granting Read or Write Permission on Field c

To perform either read or write operations on field c, when it is in column family `cf1`, you must have either `readperm` or `writeperm` access on field c:

- If you have the same permission (`readperm` or `writeperm`) on `cf1`, then you have access to field c.

- If you have the same permission (`readperm` or `writeperm`) on field `b`, then you have access to field `c`. You do not need any further permissions. Field `c` inherits your `readperm` or `writeperm` permission from field `b`.
- If you have the same permission (`readperm` or `writeperm`) on `cf1` *but* either field `a` or `b` denied you permission:
  - You must have `traverseperm` permission granted to you on the field that denied you access (field `a` or `b`).
  - You must have `readperm` or `writeperm` permission explicitly granted to you on field `c`.
- If you do *not* have the same permission (`readperm` or `writeperm`) on `cf1`:
  - You must have the `traverseperm` permission granted to you on either `cf1` or field `b`.
  - You must have `readperm` or `writeperm` permission explicitly granted to you on field `c`.

The following example grants `traverseperm` permission:

```
/opt/mapr/bin/maprcli table cf colperm set
  -path <path to JSON table>
  -cfname cf1
  -name a.b
  -traverseperm u:<user ID> | <existing ACE for this field>
```

The following example grants `readperm` permission:

```
/opt/mapr/bin/maprcli table cf colperm set
  -path <path to JSON table>
  -cfname cf1
  -name a.b.c
  -readperm u:<user ID> | <existing ACE for this field>
```

Permissions on Arrays
When granting permissions on a field, if the field contains array data, you must grant the permission on the array field. This grants access not only to array data in the field, but also nested documents and scalar data. It is also possible to set permissions on subfields within nested documents that are stored in an array.

Note: This topic describes the behavior of permissions in MapR Database version 6.1 and later, regardless of the MapR version you used to grant the permissions. To understand how permissions on arrays behave in earlier releases, see Operational Changes (MapR 6.1.0) - Permissions on Arrays in MapR Database JSON.

Granting Permissions on Array Elements

Suppose you have the following documents where `person` is:

- An array of nested documents in document `id001`
- A single nested document in document `id002`
- A scalar value in document `id003`

```
{
    "_id" : "id001",
    "person" : [
        {"name" : {"last" : "Smith", "first" : "John"}},
        {"name" : {"last" : "Subramanium", "first" : "Ananya"}}
    ]
```

```
}
{
    "_id" : "id002",
    "person" : {"name" : {"last" : "Doe", "first" : "Jane"}}
}
{
    "_id" : "id003",
    "person" : "Unknown"
}
```

If you grant a user read permission on the array `person[]`, that user can read every field in every nested document within the array in document `id001`. The permission also enables the user to read the `person` field in documents `id002` and `id003`.

If you receive an error when trying to grant permission on `person[]` because you previously granted permission on `person`, then you (or an administrator with the appropriate permissions) must first remove the existing permission on `person`. If you expect the schema of the `person` field to evolve to include non-array and array data, then you should grant the permission on `person[]` rather than `person`, to avoid having to remove the conflicting `person` permission.

You cannot grant permissions on individual elements in an array; for example: `person[1]`. Granting permission on an array enables access to the entire array.

Granting Permissions on Nested Document Fields in an Array

If you want to restrict read access to only specific fields in `person`, whether the field is an array of nested documents or a single nested document, perform the following steps:

1. Deny the user read permission on the array `person[]`.
2. Grant the user traverse permission on the array `person[]`.
3. Grant the user read permission on the specific fields.

For example, to grant the user read permission on only the first names in the nested documents, for the third step, grant read permission on `person[].name.first`. The permission enables the user to read the field in all nested documents in documents `id001` and `id002`.

If permissions already exist on `person.name.first`, then all attempts to define permissions on `person[].name.first` fails. You (or an administrator with the appropriate permissions) must first remove the existing permission on `person.name.first`. Similar to the scenario described in the previous section, if you expect the schema of the `person` field to evolve to include individual nested documents as well as arrays of nested documents, then you should grant the permission on `person[].name.first` to avoid having to remove the conflicting permission.

If you already have permissions on `person[].name.first`, then attempting to define permissions on `person.name.first` fails. There is no need to add this permission.

MapR Database as a Column-Oriented Database

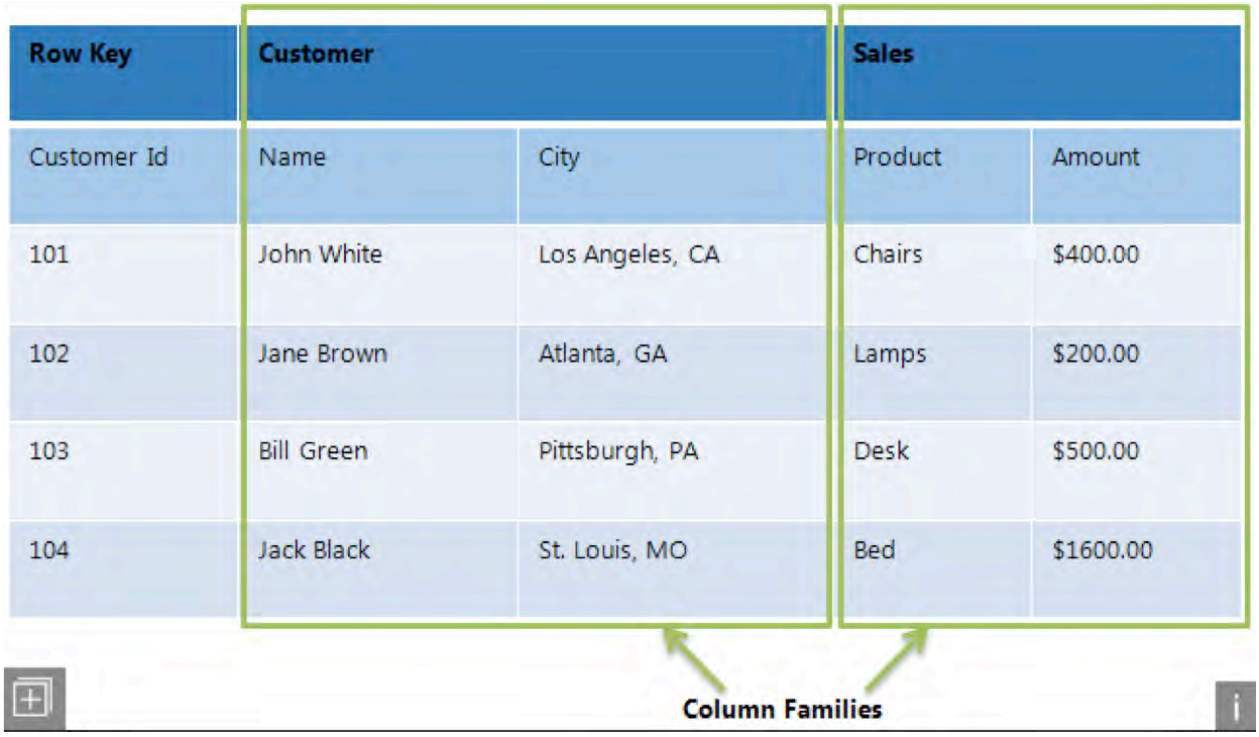
MapR Database supports column-oriented databases as a native data store. Column-oriented database tables in MapR Database are conceptually identical to tables in Apache HBase.

As a column-oriented database, MapR Database stores data in binary format. MapR Database supports the Apache HBase API and provides a native implementation of the HBase API. HBase applications can use MapR Database tables without modifying any code.

- MapR Database tables use the HBase data model.
- Allows for large-scale applications managing columnar data.

last-updated: Apr 11, 2022

- Binary compatibility with applications using standard HBase application APIs.
- With the binary tables, rows are indexed by key, columns identify data elements in each row, and column families are made up of columns.



MapR Database Binary Tables
MapR Database stores data as a nested series of maps. Each map consists of a set of key-value pairs, where the value can be the key in another map.

MapR Database stores structured data as a nested series of maps. Each map consists of a set of key-value pairs, where the value can be the key in another map. Keys are kept in strict lexicographical order: 1, 10, and 113 come before 2, 20, and 213.

In descending order of granularity, the elements of a binary table are:

- **Key**: Keys identify the rows in a table. In MapR Database, the maximum supported size of a row key is 64 KB. However, the recommended practice is to keep it lower than a few hundred bytes.
- **Row:** Rows span one or more column families and columns. In MapR Database, the maximum supported size of a row is 2 GB. However, the recommended practice is to keep the size under 2 MB. In general, MapR Database performs better with many small rows, rather than with fewer very large rows.
- **Column family**: A column family is a key associated with a set of columns. Specify this association according to your individual use case, creating sets of columns. A column family can contain an arbitrary number of columns. MapR Database binary tables support up to 64 column families.
- **Column**: Columns are keys that are associated with a series of timestamps that define when the value in that column was updated.
- **Timestamp**: The timestamp in a column specifies specifies when the data was written to that column.
- **Value**: The data written to that column at the specific timestamp.

This structure results in values with versions that you can access flexibly and quickly. Since MapR Database binary tables are *sparse*, any of the column values for a given key can be null.

last-updated: Apr 11, 2022

Example Table

This example uses JSON notation for representational clarity. In this example, timestamps are arbitrarily assigned.

Queries return the most recent timestamp, by default. For example, a query for the value in "arbitrarySecondKey"/"secondColumnFamily:firstColumn" returns `valueThree`. Specifying a timestamp with a query for "arbitrarySecondKey"/"secondColumnFamily:firstColumn"/11 returns `valueSeven`.

```
{
 "arbitraryFirstKey" : {
     "firstColumnFamily" : {
          "firstColumn" : {
               10 : "valueFive",
               7 : "valueThree",
               4 : "valueOne",
               }
          "secondColumn" : {
               16 : "valueEight",
               1 : "valueSeven",
               }
          }
     "secondColumnFamily" : {
          "firstColumn" : {
               37 : "valueFive",
               23 : "valueThree",
               11 : "valueSeven",
               4 : "valueOne",
               }
          "secondColumn" : {
               15 : "valueEight",
               }
          }
     }
 "arbitrarySecondKey" : {
     "firstColumnFamily" : {
          "firstColumn" : {
               10 : "valueFive",
               4 : "valueOne",
               }
          "secondColumn" : {
               16 : "valueEight",
               7 : "valueThree",
               1 : "valueSeven",
               }
          }
     "secondColumnFamily" : {
          "firstColumn" : {
               23 : "valueThree",
               11 : "valueSeven",
               }
          }
     }
}
```

Column Families in Binary Tables

Scanning an entire table for matches can be very performance-intensive. *Column families* enable you to group related sets of data and restrict queries to a defined subset, leading to better performance. When you design a column family, think about what kinds of queries are going to be used the most often, and group your columns accordingly.

You can specify compression settings for individual column families, which lets you choose the settings that prioritize speed of access or efficient use of disk space, according to your needs.

Be aware of the approximate number of rows in your column families. This property is called the column family's *cardinality*. When column families in the same table have very disparate cardinalities, the sparser table's data can be spread out across multiple nodes, due to the denser table requiring more splits. Scans on the sparser column family can take longer due to this effect. For example, consider a table that lists products across a small range of *model* numbers, but with a row for the unique serial numbers for each individual product manufactured within a given model. Such a table will have a very large difference in cardinality between a column family that relates to the model number compared to a column family that relates to the serial number. Scans on the model-number column family will have to range across the cluster, since the frequent splits required by the comparatively large numbers of serial-number rows will spread the model-number rows out across many regions on many nodes.

For a list of the properties that you can set when you create a column family, see the documentation for the `maprcli` command `table cf create` .

Note: When replicating a specific column family or column from a binary source table and a row is deleted, the destination table will show only a deletion for the specific column family or column. When replicating a specific column from a binary source table and its column family is deleted, the destination table will show only a deletion for the specific column.

Column Design

MapR Database tables split at the row level, not the column level. For this reason, extremely wide tables with very large numbers of columns can sometimes reach the recommended size for a table split at a comparatively small number of rows.

Warning: In general, design your schema to prioritize more rows and fewer columns.

As the MapR Database tables are *sparse*, you can add columns to a table at any time. Null columns for a given row do not take up any storage space.

Table Rowkey Design

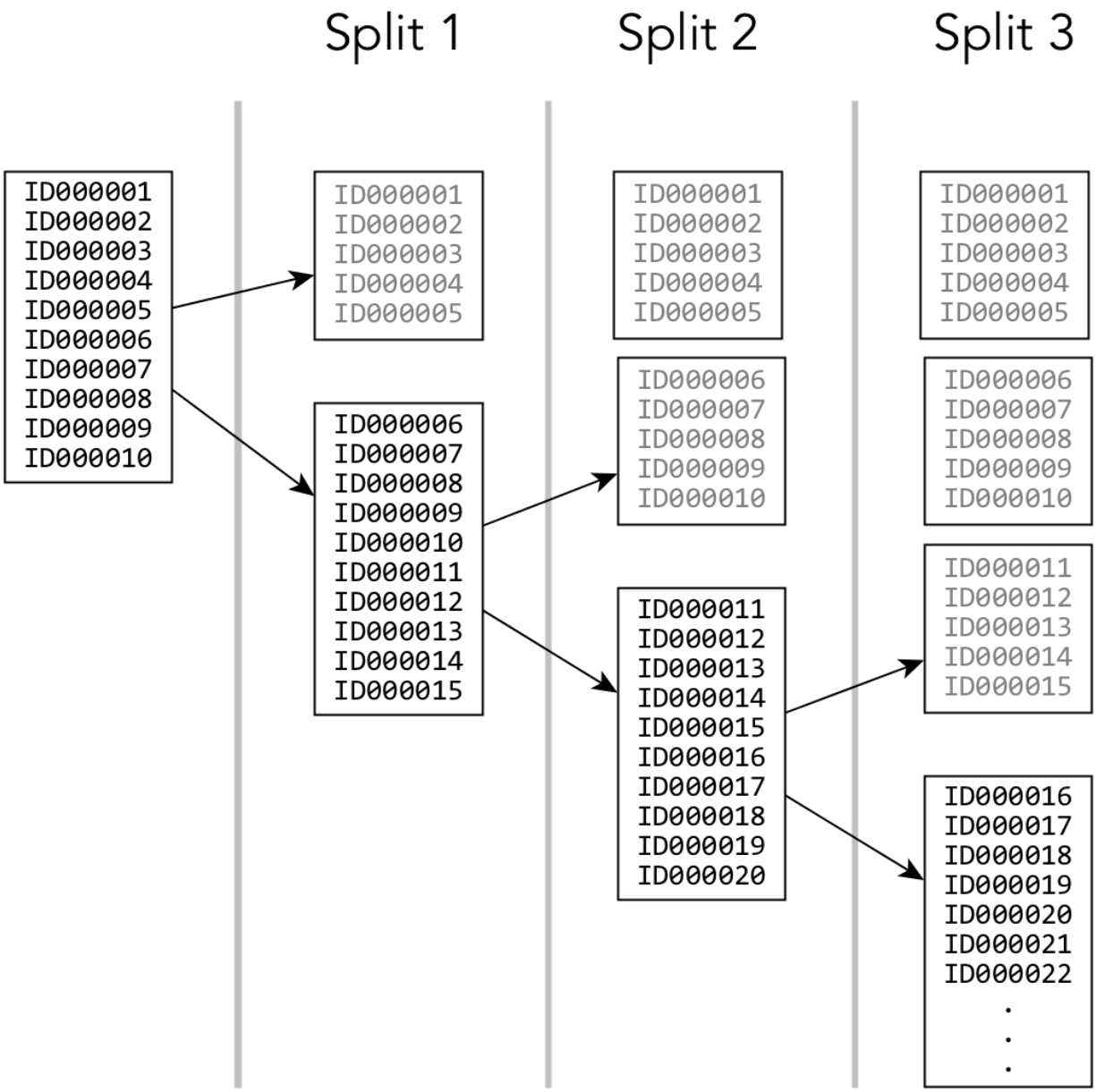
The design of a table's rowkeys affects the speed at which client applications can access data and the database performance if hotspotting occurs. The better the design, the faster the data access.

What is a Row Key?

For binary tables:	A row key identifies a row in a MapR Database binary table.
For JSON tables:	A row key identifies a row in a MapR Database JSON table. You specify row keys in the _id field in JSON documents. For example, if the value of the _id field in a JSON document is user000001, that value is also the rowkey for the row in which the JSON document is stored in a JSON table.

Avoiding Hotspotting

Because records in tables are stored in lexicographical order of their rowkeys, using a sequential generation method for rowkeys can lead to a hot-spot problem, as illustrated in this diagram.



A table region reaches a predetermined size and then splits into two regions. Because the rowkeys for new records are being created sequentially, new rows are added to only one of the new regions. The other region is not written to and remains at half of its maximum size. The problem is repeated with each subsequent split.

With MapR Database tables, the cluster handles sequential keys and table splits to keep potential hotspots moving across nodes, decreasing the intensity and performance impact of hot spots. However, hotspotting can still hamper database performance.

There are two strategies that you can use to avoid hotspotting:

Hashing keys	To spread write and insert activity across the cluster, you can randomize sequentially generated keys by hashing the keys, inverting the byte order. Note that these strategies come with trade-offs. Hashing keys, for example, makes table scans for key subranges inefficient, since the subrange is spread across the cluster.

last-updated: Apr 11, 2022

Salting keys	Instead of hashing the key, you can salt the key by prepending a few bytes of the hash of the key to the actual key. For a key based on a timestamp, for instance, a timestamp value of `1364248490` has an MD5 hash that ends with `ffe5`. By making the key for that row `ffe51364248490`, you avoid hotspotting. Because you know that the first four digits are a hash salt, you can derive the original timestamp by dropping those digits.

Composite Keys

Each row in a table can have only a single key. You can create composite keys to approximate multiple keys in a table. A composite key contains several individual IDs joined together, for example userID and applicationID. You can then scan for the specific segments of the composite row key that represent the original, individual ID.

Because rows are stored in sorted order, you can affect the results of the sort by changing the ordering of the fields that make up the composite row key. For example, if your application IDs are generated sequentially but your user IDs are not, using a composite key of userID+applicationID will store all rows with the same user ID closely together. If you know the userID for which you want to retrieve rows, you can specify the first userID row and the first userID+1 row as the start and stop rows for your scan, then retrieve the rows you're interested in without scanning the entire table.

When designing a composite key, consider how the data will be queried during production use. Place the fields that will be queried the most often towards the front of the composite key, bearing in mind that sequential keys will generate hotspotting.

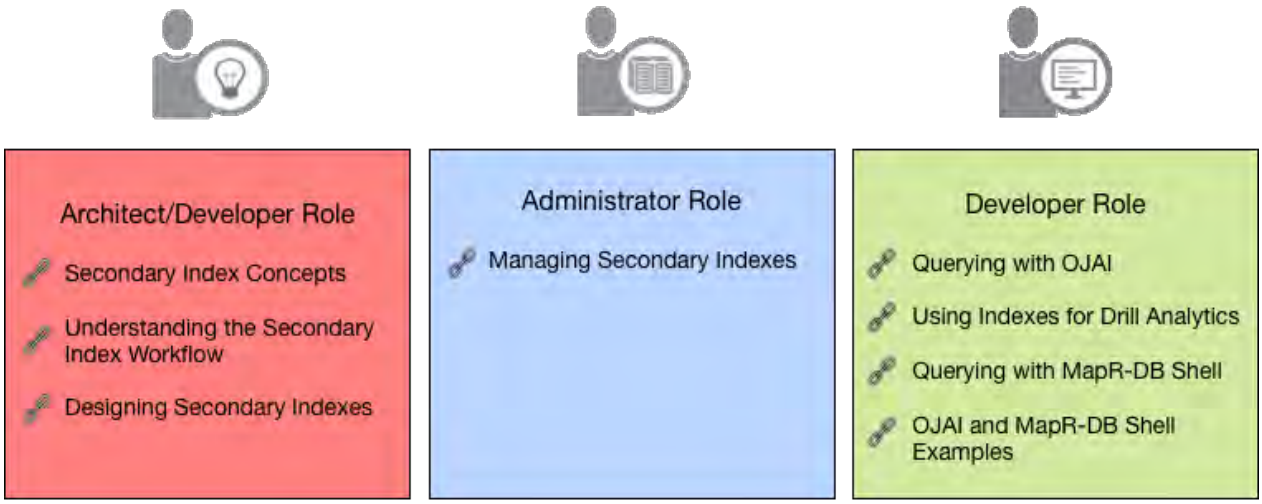
For binary tables:	You must create your own custom logic for working with composite keys in applications that use the HBase Java API. This API does not have built-in support for composite keys.
For JSON tables:	You must create your own custom logic for working with composite keys in applications that use the MapR Database OJAI Java API library. This API library does not have built-in support for composite keys.

Secondary Indexes

Beginning with MapR 6.0, MapR Database JSON natively supports secondary indexes on fields in JSON tables. Indexes provide you with flexible, high performance access to data stored in MapR Database.

How Do I Get Started?

The following diagram provides links to topics that you need to understand and use Secondary Indexes. Topics include conceptual information about indexes, how to decide what indexes to create, how to set up and use indexes, the `maprcli` commands used to create and maintain indexes, and how to query your data to leverage indexes. The information is organized based on roles.



1. Describes secondary index concepts, including use cases, types of indexes, types of queries that benefit from indexes, and how indexes are implemented
2. Describes the overall workflow for using secondary indexes. This includes the roles of different users and the workflow steps involved.
3. Describes how to design secondary indexes to provide the most benefit to MapR Database JSON queries
4. Describes how to manage secondary indexes including creating, deleting, and listing indexes, setting up your cluster for querying, and troubleshooting
5. Describes how to use the OJAI API library to query JSON tables, including special considerations related to secondary indexes
6. Describes how to leverage indexes when issuing SQL queries with Drill
7. Describes how to use the MapR Database Shell to query JSON tables
8. Contains samples of OJAI programs and MapR Database Shell commands that query JSON tables

What are Secondary Indexes?

A *secondary index* (also sometimes referred to in this documentation as an *index*) is a special table that stores a subset of document fields from a JSON table. The index orders its data on a set of fields, defined as the *indexed fields*. This is in contrast to the JSON table that orders its data on the table primary key (rowId or rowKey). If you have administrator privileges, you can create one or more indexes on each JSON table. After the indexes are created, applications can leverage them to accelerate query response times. Secondary indexes can also contain additional fields known as *included fields* (or sometimes *covered fields*) beyond those being indexed, so that many queries can be satisfied with a single read.

Secondary indexes provide efficient access to a wider range of queries on data in MapR Database. They allow queries to efficiently query data through fields other than the primary key. This capability results in MapR Database supporting a broader set of use cases. Applications that benefit include rich, interactive business applications and user-facing analytic applications. Secondary indexes also enable Business Intelligence tools and ad-hoc queries on operational datasets. See Uses for Secondary Indexes on page 548 for more information.

Important: Secondary indexes can be created only on MapR Database JSON tables.

last-updated: Apr 11, 2022

Why Use Secondary Indexes?

With the ever increasing amount of data stored in MapR Database JSON, indexing that data becomes critical. Without indexes, queries unnecessarily scan large amounts of data from the underlying JSON table. Queries could potentially scan every document in the table, even if they contain conditions that limit the documents to select. Query performance suffers and resource bottlenecks are inevitable when you use this data model.

Without indexes, applications and query layers resort to limited interactivity to avoid performance concerns. Using indexes solves this limitation in application scale, by reducing the number of documents client applications read, even when querying large data sets. This reduces I/O and CPU costs, resulting in improved performance.

The functionality and benefits of indexing available in MapR Database are similar to that of indexes in relational databases. The difference is that MapR Database indexes provide performance benefits at high scale, in combination with JSON flexibility on the query side and simplicity on the management side.

How Can I Use Secondary Indexes?

You can leverage MapR Database secondary indexes by using either the OJAI API, the MapR Database JSON REST API, or MapR Drill.

OJAI is the business application development interface on MapR Database. Typically, business applications are characterized by ultra low latency and extremely high throughput. When you build an application using OJAI, filtering and sorting through the API can leverage secondary indexes to accelerate query response times.

The MapR Database JSON REST API enables you to use HTTP calls to perform basic operations on MapR Database JSON tables, including querying.

Drill is the analytics SQL interface on MapR Database. Drill is a distributed SQL query engine that provides interactive response time for operational analytics, Business Intelligence (BI) tools such as Tableau, and ad-hoc queries on MapR Database. With MapR Drill, SQL queries can also leverage secondary indexes to accelerate query response times.

Regardless of whether queries originate from OJAI or Drill SQL, each interface seamlessly selects the optimal indexes to use. You do not need to write explicit code or provide directives on which indexes to use. If an appropriate index exists for a query, MapR Database leverages the index.

For more information about the OJAI API, see the following API links:

- Java OJAI Client API
- Node.js OJAI Client API
- Python OJAI Client API
- C# OJAI Client API
- Go OJAI Client API

For information about the MapR Database JSON REST API, see Using the MapR Database JSON REST API on page 2696.

For information about MapR Drill, see Apache Drill on MapR.

Related information

OJAI source code on github

https://drill.apache.org/

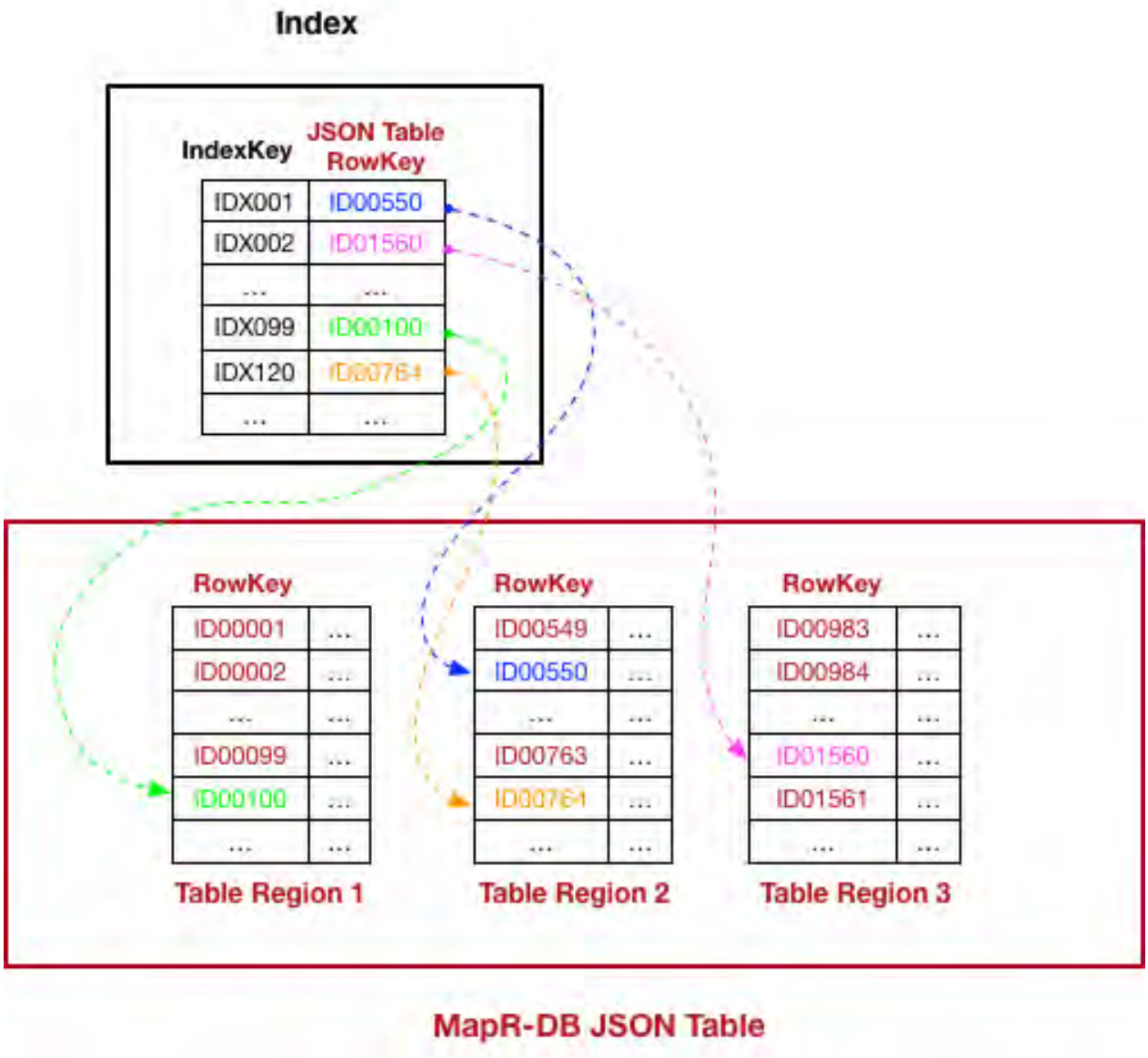
Secondary Index Concepts
Describes secondary index concepts, including use cases, types of indexes, types of queries that benefit from indexes, and how indexes are implemented.

Indexes created on regularly queried JSON table fields provide MapR Database quick access to data. Indexes primarily benefit queries with filters in the WHERE clause, queries with an ORDER BY clause for sorting, and queries where all fields projected in the query are included in the index. They provide the most benefit when an index contains all fields referenced in a query. For filters, indexes reduce the amount of data read. MapR Database implements indexes using JSON tables. Like JSON tables, an index stores data in sort order. Reading data through the index eliminates the need to sort the data if the index and query sort orders match.

Each JSON table in MapR Database has a unique field that serves as the rowkey. A secondary index contains indexed and included fields. The indexed fields, also referred to as *index keys*, define the sort order of the index. The index stores the values of the index keys along with the rowkey corresponding to each key value. The rowkey links the index to the JSON table. MapR Database can perform a range scan on the index and then use the corresponding rowkeys to quickly locate data in the JSON table. Additional fields can be included in the index so that queries that only need these included (or covered) fields can get all the data they need from the index and therefore will not require access to the base table.

The following diagram illustrates the mapping. Each index entry consists of the index key value followed by the rowkey of the corresponding JSON document. The color coding highlights the matching index and JSON table entries.

last-updated: Apr 11, 2022



Important: Secondary indexes can only be created on MapR Database JSON tables.

Uses for Secondary Indexes

Describes typical use cases that can benefit from secondary indexes.

Operational Analytics	Operational analytics require highly scalable, highly responsive, interactive, user-facing applications. Application developers can use OJAI API to build richer and more interactive applications. This enables users to retrieve data on a variety of columns in MapR Database JSON tables in a flexible way. In addition to processing queries, OJAI also enables them to sort on columns and paginate or restrict the results. The applications can be operational applications or operational analytical applications. For both categories, the level of user interactivity and query complexity is high. Sample applications include Customer 360, expense reporting systems, game management, product catalogs, and a variety of domain-specific analytics as service applications.

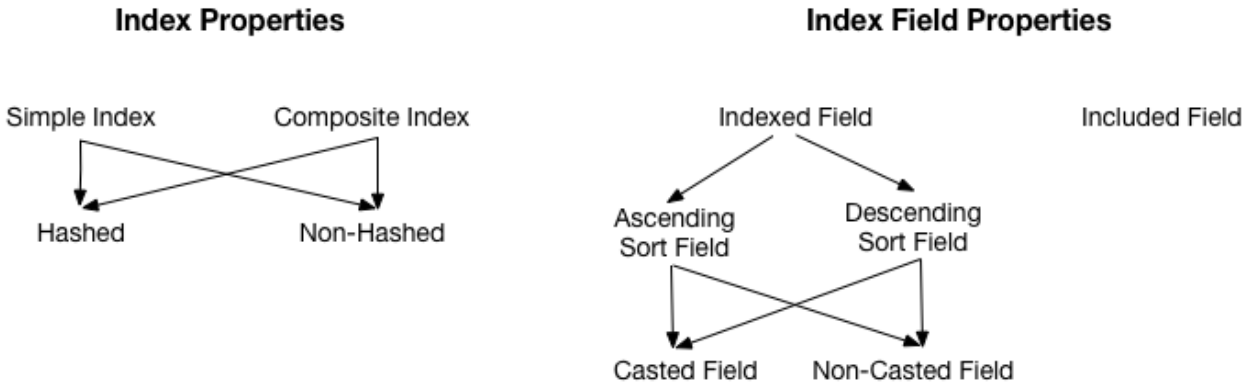
last-updated: Apr 11, 2022

Operational BI and Dashboards	Typical analytical workloads query snapshots of read-only data. Querying against MapR Database using Drill SQL enables these applications to get insights into the latest, changing data. Moreover, operational analytical queries usually access only a subset of fields from a table, often aggregating the data on a variety of dimensions and time ranges. Secondary indexes are extremely useful in these use cases. They improve the performance of well known query patterns.
Self-service Data Exploration	Users want to use their favorite BI reporting tool to issue ad-hoc queries against MapR Database JSON tables. They can achieve a simple, high performing data exploration experience using MapR Database and Drill SQL, while leveraging the capabilities of both. Using the MapR Database document model provides end to end JSON flexibility at the data storage level. Using Drill SQL provides dynamic schema discovery.

Types of Secondary Indexes

MapR Database JSON supports several index types, including simple indexes, composite indexes, hashed indexes, and indexes with casting. This section describes the properties of these indexes and the situations where each provides value.

The following diagram illustrates the different properties of indexes and index fields. Lines connecting properties represent properties that can be used in combination with one another. Click on the text in the diagram for a description of each property.



1. Simple vs Composite Indexes
2. Simple vs Composite Indexes
3. Hashed vs Non-Hashed Indexes
4. Hashed vs Non-Hashed Indexes
5. Indexed vs Included Fields
6. Indexed vs Included Fields
7. Indexed Field Sort Order
8. Indexed Field Sort Order
9. Casting
10. Casting

last-updated: Apr 11, 2022

Indexed vs Included Fields

An index consists of indexed and included fields. Indexed fields are also referred to as *index keys*. The following lists describe the characteristics of each type of field:

Indexed Fields	• Determine the sort order of the index and the order of the query result when used • Allow filter conditions and ORDER BY conditions defined on these fields to be optimized
Included Fields (sometimes referred to as *covered fields*)	• Do not affect the sort order of the index or the order of the query result • Can avoid the need to read the base table if all required fields are included in the index

In general, you should define indexed fields on fields you filter and order on, and included fields on fields you reference but do not filter and order.

The following example illustrates when you would define an indexed vs an included field in your index. Assume you have a MapR Database JSON table with the following sample data that contains customer information.

```
{
   "_id": "10000",
   "FullName": {
      "LastName": "Smith",
      "FirstName": "John"
   },
   "Address": {
      "Street": "123 SE 22nd St.",
      "City": "Oakland",
      "State": "CA",
      "Zipcode": "94601-1001"
   },
   "Gender": "M",
   "AccountBalance": 999.99,
   "Email": "john.smith@company.com",
   "Phones": [
        {"Type": "Home", "Number": "555-555-1234"},
        {"Type": "Mobile", "Number": "555-555-5678"},
        {"Type": "Work", "Number": "555-555-9012"}
   ],
   "Hobbies": ["Baseball", "Cooking", "Reading"],
   "DateOfBirth": "10/1/1985"
}
```

Your query does the following:

1. Filters on `Address.Zipcode`
2. Selects `FullName.FirstName` and `FullName.LastName`

Since your query filters on `Address.Zipcode`, you should include that field as an indexed field. Also, because this query only needs the fields `FullName.FirstName` and `FullName.LastName`, you can set `FullName` as an included field. The result is that this query will only need to read from the index and will

not need to look at the original table. Other queries that, for example, need to read the phone numbers or address would still need to go back to the base table.

```
maprcli table index add -path /customerInfo -index zipCodeIdx \
   -indexedfields Address.Zipcode \
   -includedfields FullName
```

There are additional differences in how indexed and included fields behave. The following table summarizes these differences:

Indexed Field	Included Field
There are some restrictions in the data types of indexed fields. See Data Types and Secondary Index Fields on page 561 for the complete list of types.	Data types of included fields can be any type. There is no data type restriction.
The collective size of all indexed fields is a maximum of 32KB.	Included fields do not affect the size limit of an index.
Adding indexed fields increases the cost of key comparisons when scanning the index, due to the increase in the index key size.	Adding included fields does not impact the index scan cost.

Included fields influence whether an index is a *covering index* for a query. See Covering Indexes on page 560 for more information about this concept.

Indexed Field Sort Order

You can define each field in your index key to sort in either ascending or descending order. The default is ascending. Typically, you define the sort order to match the ORDER BY clause in your query. This allows the MapR Database to avoid performing an explicit sort. For example, if you issue queries where you return `AccountBalance` in descending order, create the following index.

```
maprcli table index add -path /customerInfo -index BalanceIdx \
   -indexedfields AccountBalance:-1
```

Simple vs Composite Indexes

Simple indexes are indexes with a single indexed field (or key). Composite indexes have more than one key. In both cases, you can define zero or more included fields. See Simple Indexes on page 552 and Composite Indexes on page 553 for additional details.

Hashed vs Non-Hashed Indexes

By default, indexes are stored in sort order across the index key values. This can lead to hotspots if the sort order of the index keys match the order data that is inserted into the JSON table. For example, if the indexed field has monotonically increasing timestamp values, such as the date a document is created, the tail end of the index becomes a hotspot. Hashed indexes avoid hotspotting by evenly distributing index writes across a number of logical partitions.

The following example creates a hashed index named `idx` on table, `tab`, with a single key, `idxKeyCol`.

```
maprcli table index add -path /tab -index idx -indexedfields idxKeyCol \
    -hashed true
```

See Hashed Indexes on page 555 for further details.

last-updated: Apr 11, 2022

Casting

You can CAST individual indexed fields to a specific data type. This is applicable when Drill SQL queries contain CAST expressions. The following example creates an index that casts the `age` field to an INT type and the `height` field to a FLOAT type.

```
maprcli table index add -path /castTable -index castIdx \
   -indexedfields '$CAST(age@INT)','$CAST(height@FLOAT)'
```

See Using Casts in Secondary Indexes on page 557 for further details.

Note: This feature only applies for queries issued through the Drill SQL interface. The OJAI API does not have CAST support.

Simple Indexes
A *simple index* is a secondary index that has only one indexed field and zero or more included fields. Simple indexes enable you to optimize queries that filter and sort on a single field. If all fields referenced in a query are either indexed or included fields in a simple index, then you can process the query by reading only the index.

Sort Order

MapR Database sorts simple indexes on the single indexed field. MapR Database sorts the indexed field values in ascending order by default, although you can specify a descending order when you create the index. Sorting indexes benefits your ORDER BY queries because the index eliminates the need for a SORT operator in the query plan.

Simple Index Examples

The following CLI commands demonstrate how you can create various types of simple indexes. For these examples, assume that you have a MapR Database JSON table with the following sample data:

```
{
   "_id": "10000",
   "FullName": {
      "LastName": "Smith",
      "FirstName": "John"
   },
   "Address": {
      "Street": "123 SE 22nd St.",
      "City": "Oakland",
      "State": "CA",
      "Zipcode": "94601-1001"
   },
   "Gender": "M",
   "AccountBalance": 999.99,
   "Email": "john.smith@company.com",
   "Phones": [
       {"Type": "Home", "Number": "555-555-1234"},
       {"Type": "Mobile", "Number": "555-555-5678"},
       {"Type": "Work", "Number": "555-555-9012"}
   ],
   "Hobbies": ["Baseball", "Cooking", "Reading"],
   "DateOfBirth": "10/1/1985"
}
```

CLI Command	Description

`maprcli table index add -path /people \`
`-index emailIdx \`
`-indexedfields Email`
	• Creates a simple index on the `Email` field, with no included fields
	• Enables you to filter on the `Email` field

`maprcli table index add -path /people \`
`-index dobIdx \`
`-indexedfields DateOfBirth \`
`-includedfields FullName`
	• Creates a simple index, with an included field
	• Enables you to filter on `DateOfBirth` and project on `FullName`

`maprcli table index add -path /people \`
`-index LastNameIdx \`
`-indexedfields FullName.LastName:-1`
	• Creates a simple index on the `FullName.LastName` subfield, as a descending sort key
	• Allows you to filter on `FullName.LastName` and sort on the subfield in descending order

`maprcli table index add -path /people \`
`-index fullNameIdx \`
`-indexedfields FullName`
	• Creates a simple index on the nested document field `FullName`
	• Allows you to perform equality lookups on both the `LastName` and `FirstName` subfields of `FullName`

`maprcli table index add -path /people \`
`-index hobbiesIdx \`
`-indexedfields Hobbies`
	• Creates a simple index on the `Hobbies` array field
	• Allows you to filter for a specific list of hobbies

`maprcli table index add -path /people \`
`-index hobbyIdx \`
`-indexedfields Hobbies[]`
	• Creates a simple index using the container field path `Hobbies[]`
	• Allows you to filter for a specific hobby

`maprcli table index add -path /people \`
`-index phoneNumberIdx \`
`-indexedfields Phones[].Number`
	• Creates a simple index on the container field path `Phones[].Number`
	• Allows you to filter for a phone number, regardless of whether it is a home, work, or cell phone

Composite Indexes
A *composite index* is an index that has more than one indexed field and zero or more included fields. Composite indexes enable you to optimize queries that filter and sort on multiple fields. If all fields referenced in a query are either indexed or included fields in a composite index, then you can process the query by reading only the index.

Sort Order

MapR Database sorts the composite index in the order in which you have defined the indexed fields. For example, if you have an index on `Field1` and `Field2`, MapR Database sorts on `Field1` as the primary sort key and `Field2` as the secondary.

Each component in a composite index can have its own ordering. For example, you can specify an ascending sort order for one field and a descending sort order for another.

Composite Indexes and Container Field Paths

The indexed fields in a composite index can be Container Field Paths on page 518. However, if you specify more than one container field path in your indexed fields, the prefixes of the container field paths must be the same. This allows MapR Database to store index values that originate from the same array element in a single index row.

Examples of Supported Composite Indexes

Indexed Fields Allowed for Composite Index	Why Allowed?
`a[].b, x.z`	The indexed fields have only one container field path.
`a[].b, a[].c`	The indexed fields have a common container prefix, `a[]`.
`a[].b, a[].c[]`	The indexed fields have a common container prefix, `a[]`.
`a[].b[].c, a[].b[].d`	The indexed fields have a common container prefix, `a[].b[]`.

Examples of Unsupported Composite Indexes

If a composite index includes the same subfield in multiple indexed fields, the implied types of the subfields must also be consistent. The third and fourth rows in the following table show examples of this restriction:

Indexed Fields Disallowed for Composite Index	Why Disallowed?
`a[].b, x[].y`	`a[]` and `x[]` are different container prefixes.
`a[].b[], a[].c[]`	Although both indexed fields have a common container prefix of `a[]`, `b[]` and `c[]` are different container field paths.
`a.b[].c, a.b.d`	In the first indexed field, subfield `b` is an array of nested documents. In the second, `b` is a single nested document. This results in a type conflict.
`a, a[]`	The first indexed field `a` is a scalar type while the second is an array. This also results in a type conflict.

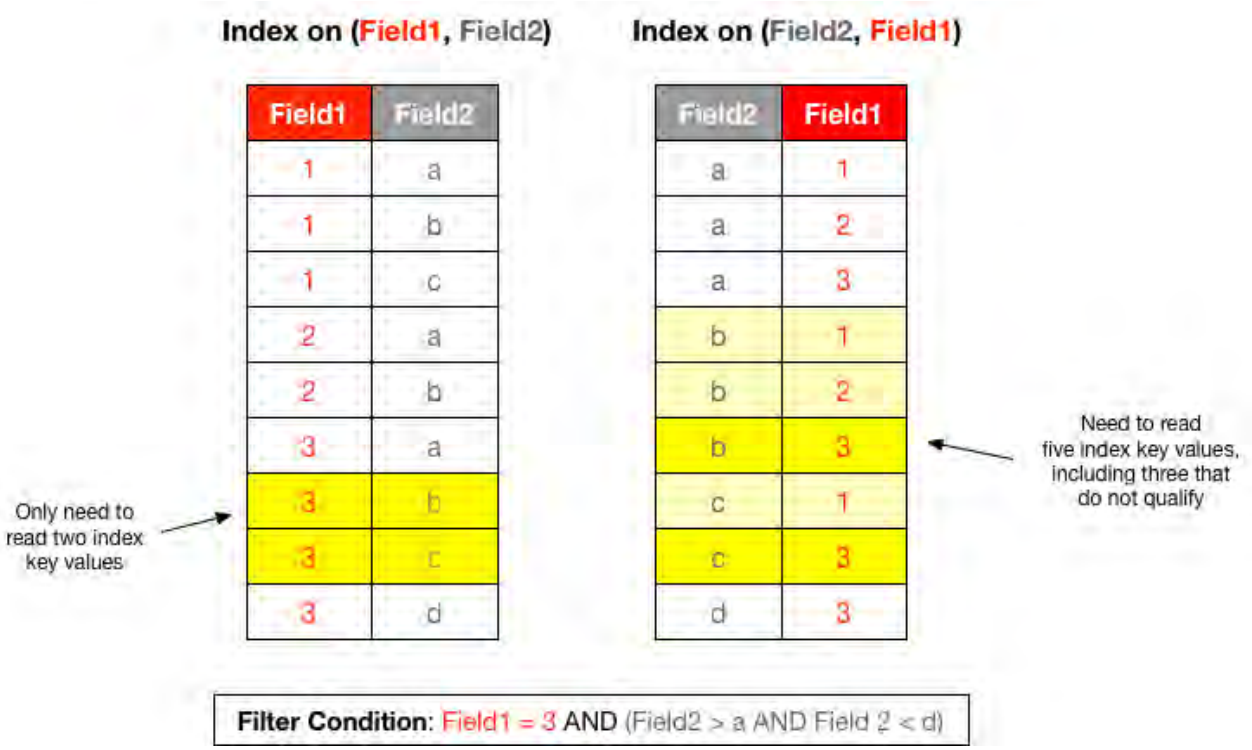
Query Conditions Using Composite Indexes and Container Field Paths

When you have a composite index defined on container field paths, your query condition must use the elementAnd operator to use all keys of the index. With the `and` operator, the conditions do not have to match the same array element; as a result, the matching indexed field values might span different index rows, preventing use of the composite index. See OJAI Query Conditions Using elementAnd on page 2619 for more details.

Composite Index Example

Consider the following example, which illustrates the impact of key order in a composite index:

last-updated: Apr 11, 2022



In the index on the left, the prefix key is `Field1`. MapR Database sorts the index first on `Field1`, and then on `Field2`. This aligns with the equality condition, `Field1 = 3`. MapR Database reads the least number of entries from the index on the left, due to this equality condition and the range condition on `Field2`.

In the index on the right, MapR Database sorts the index on `Field2`, followed by `Field1`. In this case, the matching key values are not contiguous in the index, as highlighted by the entries in a lighter shade of yellow. MapR Database uses the filter conditions on `Field2` to start the search in the index. It applies the filter condition on `Field1` while reading the index. This is less efficient because MapR Database must read those extra non-matching key values.

Related concepts

Restrictions on Secondary Indexes on page 565
This topic lists and describes the restrictions on secondary indexes. It is important for you to understand the type, size, field definition, option, and index use restrictions when defining and using secondary indexes.

Hashed Indexes

A *hashed index* is a secondary index that distributes keys across logical partitions to avoid creating hot spots when MapR Database updates the index with new keys from the JSON table.

Hot spots occur when data inserted into an indexed field has monotonically increasing values, or when a burst of write activity occurs. The former occurs with timestamp values. The latter occurs when you have a burst of updates on an indexed field over a small range of values. Hashed indexes enable MapR Database to evenly distribute new writes on an index and avoid hot spots.

Note: Hashed indexes do not resolve hot spots on the JSON table. For information about how to design rowkeys and avoid hot spots in the JSON table, see Table Rowkey Design.

Hashed indexes support the same conditional queries as non-hashed indexes, except that hashed indexes do not have a guaranteed sort order. Hashed indexes do not support ORDER BY queries due to the distribution of data across logical partitions. Consequently, sorting is performed by the query layer, which can increase the CPU costs and negatively impact performance.

By default, MapR Database creates ten partitions for a hashed index. You can modify this value when you create a hashed index using the maprcli table index add command or through the Control System. When a hashed index exists, MapR Database distributes table updates to the index across the logical partitions, which reside on different nodes. MapR Database orders the keys within each partition instead of ordering them across the entire index.

Note: Once you create an index with hashing enabled, you cannot disable hashing. You can remove the hashed index and then create a non-hashed (default) index on the field. See Removing Indexes and Adding Indexes.

Guidelines on Creating Hashed Indexes

- Create a hashed index on fields with monotonically increasing values, such as timestamp values.
- Create a hashed index on fields that MapR Database updates in bursts of write activity, for example when MapR Database updates a small range of possible values for the indexed field.
- Do not create hashed indexes for ORDER BY queries.
- Use the `maprcli table index list` command or the Control System to determine if an index is hashed. See maprcli table index list or Listing Indexes.
- After you create an index with hashing enabled, you cannot disable hashing.

Example Comparison of a Non-Hashed Index and Hashed Index

The following images depict a non-hashed (default) index and a hashed index. For the purpose of this example, assume that an index was created on the `DateCreated` field of a JSON table in MapR Database. Yellow highlighted areas indicate updates to the index.

Non-Hashed (Default) Index

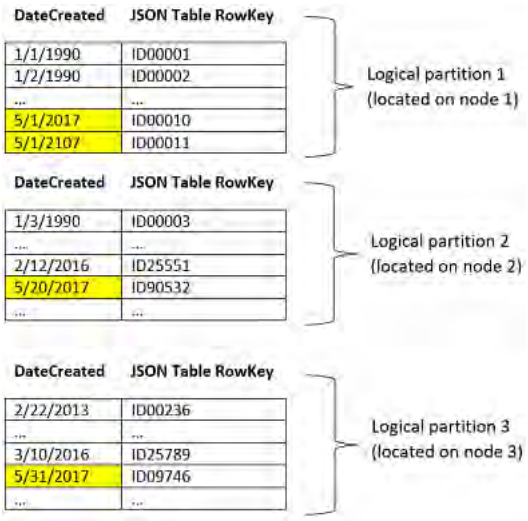
The non-hashed index propagates `DateCreated` field updates from the JSON table to the index. Notice that the dates are sorted within the index and no partitions exist. Depending on the size of the index, the index may exist on one or multiple nodes

DateCreated	JSON Table RowKey
1/1/1990	ID00001
1/2/1990	ID00002
...	...
...	...
5/1/2017	ID00010
5/1/2017	ID00011
...	...
5/20/2017	ID90532
...	...
5/31/2017	ID09746

.

Hashed Index

The hashed index propagates `DateCreated` field updates across the index partitions which reside on different nodes. Notice that dates are sorted within each partition and each partition resides on a different node.



Using Casts in Secondary Indexes
Defining an index that specifies index keys with CAST functions provides fast access for queries that contain CAST functions. The index converts the indexed field to the type specified by the CAST function and stores the result.

Create indexes using CAST functions if you want to CAST fields to specific data types in your queries. To define an index with the CAST function applied to a field, specify a CAST when defining the index key. The following example creates an index that casts the `age` field to an INT type and the `height` field to a FLOAT type.

```
maprcli table index add -path /castTable -index castIdx \
   -indexedfields '$CAST(age@INT)','$CAST(height@FLOAT)'
```

When issuing Drill queries through Business Intelligence (BI) tools, you can include CAST functions in your queries to create Drill views. Including CAST functions provides the metadata needed to optimally process the queries. For more information about using the CAST function with Drill, see Data Type Conversion.

Casting from NULL in Drill

You can cast from null to any data type supported by the indexes with the CAST function. However, null can be a valid JSON value for the string data type, for example:

```
{ "name": null }
```

Null can also represent the absence of an actual value, for example:

```
{ "_id":1,"name":"Annie" }
{ "_id":2 }  (name does not exist)
```

When you cast on columns with missing values, Drill does not return null for the missing values. Drill only returns null in cases where an actual value of null exists.

For example, if you have the following data stored in a JSON table named t1:

a1	b1
1	'abc'
2	null
3	'null'

And you issue the following query against the table:

```
SELECT a1, b1(cast b1 as varchar(20))) from t1;
```

Drill returns the following data:

a1	b1
1	abc
2	
3	null

Drill does not return null where null represents a missing value. Drill only returns null in the instance where null is stored as a string value.

Guidelines for Using Casts in Indexes

The following rules apply to CAST functions used in secondary indexes:

- You can include the CAST function only on indexed fields.
- You do not have to cast between comparable data types.
- Indexes support casting to the following data types:
  - Boolean
  - String
  - Int
  - Long
  - Float
  - Double
  - Date
  - Time
  - Timestamp

  Note: MapR Database does not support casting from any data type to `byte`, `short`, `decimal`, `binary`, or `interval`.

  Note: Queries that use the CAST function on fields with `timestamp` and `binary` data types are not supported.

- When casting to a `string` type, you can optionally specify a length. If you do not specify a length, it defaults to the maximum length of 255.
- When casting a `float` or `double` type to a `string` type, you cannot control the precision of the digits in the resulting string value. `Float` and `double` are approximate representations of decimal values.
- When casting from a `binary` type, MapR Database assumes that the binary value is a UTF-8 formatted string representation of the resulting data type.

- If MapR Database cannot cast a value, MapR Database indexes the row with an encoding error that specifies a CAST issue.
- You cannot cast all data types to the supported data types. See the Casting Matrix below for supported and unsupported combinations.

Casting Matrix

The following matrix displays supported and unsupported casting, from the data type shown in the column to the data type shown in the row. **Y** indicates a supported casting; **N** indicates an unsupported casting. Hyphen (**-**) indicates that casting is unnecessary, because the data types are comparable.

	int	**long**	**float**	**double**	**string**	**boolean**	**date**	**time**	**timestamp**
byte	Y	Y	Y	Y	Y	Y	N	N	N
short	Y	Y	Y	Y	Y	Y	N	N	N
int	-	Y	Y	Y	Y	Y	Y[1]	Y[2]	Y
long	Y	-	Y	Y	Y	Y	Y	Y	Y
float	Y	Y	-	Y	Y	Y	N	N	N
double	Y	Y	Y	-	Y	Y	N	N	N
string	Y	Y	Y	Y	-	Y[3]	Y	Y	Y
boolean	Y	Y	N	N	Y	-	N	N	N
date	N	N	N	N	Y	N	-	Y	Y
time	N	N	N	N	Y	N	N	-	N
timestamp	N	N	N	N	Y[4]	N	Y	Y	-
binary	Y	Y	Y	Y	Y	N	N	N	N
array	N	N	N	N	N	N	N	N	N
nested document	N	N	N	N	N	N	N	N	N

[1] When casting int/long to a date type, the date value is constructed based on the int/long value being the number of milliseconds since epoch.

[2] When casting int/long to a time type, the time value is constructed based on the int/long value being the time of day in milliseconds.

[3] MapR Database casts the strings `true`, `yes`, `on`, `y`, `t`, and `1` to boolean `true`. MapR Database casts the strings `false`, `no`, `off`, `n`, `f`, and `0` to boolean `false`.

[4] The string represents the time in UTC timezone.

Example Using Cast Function in an Index

This example shows you how to create an index with the CAST function.

The following statement queries a table named `lineitem` and casts the `L_LINENUMBER` and `L_ORDERKEY` fields to the `int` data type:

```
SELECT L_LINESTATUS, L_QUANTITY FROM lineitem WHERE CAST(L_LINENUMBER as
int) = 1 AND CAST(L_ORDERKEY as int) = 550;
```

You can create an index on the `L_LINENUMBER` and `L_ORDERKEY` fields and indicate the use of the CAST function and data type for each field, as follows:

```
maprcli table index add -path /drill/testdata/qa/sf1/maprdb/json/
lineitem -index l_cast_comp_1 \
   -indexedfields '$CAST(L_LINENUMBER@INT)','$CAST(L_ORDERKEY@INT)' \
   -includedfields L_LINESTATUS,L_QUANTITY
```

The index stores the values of the `L_LINENUMBER` and `L_ORDERKEY` fields as the `int` data type. MapR Database can use the index for any subsequent queries that use the CAST function to retrieve these fields as the `int` type, instead of accessing data in the primary table and converting the values to `int`.

Note: If you created an index on the `L_LINENUMBER` and `L_ORDERKEY` fields without the CAST function, the query used in this example would not benefit from the index.

Covering Indexes
A *covering index* is an index that allows MapR Database to process a query using secondary indexes without reading the JSON table. Using a *covering index* makes a query more efficient by avoiding the I/O overhead of fetching data from the JSON table.

If all fields referenced in a query are either indexed or included fields in a secondary index, then the secondary index is a covering index for that query. MapR Database determines whether an index is covering for a query.

A query that uses a covering index can reference only indexed fields from the index or a combination of indexed and included fields. While adding included fields to an index enables it to become a covering index, note that each field you add to an index increases its storage requirement. As the storage size increases, the cost of reading the index also increases; likewise, for the cost of adding and updating documents. Consider the impact on storage and updates when adding included fields to an index.

In contrast to a covering index, a noncovering index is an index that does not store all fields referenced by a query. In this case, lookups occur on the JSON table to retrieve the referenced fields that are not available in the index itself.

Whether an index is covering or noncovering depends on the query that uses the index. In the example at Types of Secondary Indexes on page 549, `zipCodeIdx` is a covering index for the noted query. If the query also selects the `Gender` field, `zipCodeIdx` is no longer a covering index for the query, but it still optimizes the filter condition.

Covering Indexes and Container Field Paths

When a query uses an index in which the indexed fields are container field paths, MapR Database cannot rely on only the indexed fields to treat the index as covering. This is due to the way MapR Database stores data in an index for container field paths. As described in Using Container Field Paths as Indexed Fields on page 562, MapR Database stores one row in the index for each array element. Thus, reading only the rows corresponding to matching array elements might not retrieve the other elements of the array.

To allow an index to be covering in this scenario, the referenced field must be an included field in the index.

For example, using the example at Types of Secondary Indexes on page 549, suppose you want to run the following query:

- Filter where `Hobbies[]` contains "Baseball"
- Select the `FullName` and all `Hobbies`

For an index to be covering for this query, you must define the index with following fields:

- Indexed Fields: `Hobbies[]`
- Included Fields: `Hobbies`, `FullName`

Note: MapR Database does not permit you to specify the same field as both an indexed and included field, unless the indexed field is a container field path.

Data Types and Secondary Index Fields

Secondary indexes support a specific set of data types. This section describes how indexed and included fields in secondary indexes behave for various categories of data types.

Data Types of Indexed Fields

Prior to MapR 6.1, the indexed fields in a secondary index had to contain scalar data. For each scalar data value, MapR Database stored a row in the index. See the table in the **Scalar Data** section of JSON Document Data Types for a list of scalar types.

Beginning with MapR 6.1, indexed fields can also be nested documents or arrays, but not array elements. As with scalar data values, MapR Database stores a row in the index for each nested document and array. The index improves equality filters on the entire nested document or array.

MapR 6.1 also supports using container field paths as indexed fields.

The following table summarizes what MapR Database supports, depending on the characteristics of the indexed field:

Characteristics of Indexed Field	Pre-6.1 Behavior	6.1 Behavior
Field contains scalar data	Supported	Supported
Field contains nested document data	Not supported	Supported
Field contains array data	Not supported	Supported
Field path is a nested document subfield	Supported only if the subfield contains scalar data	Supported for any data type
Field is an individual array element	Not supported	Not supported
Field uses a container field path	Not applicable	Supported

To understand what MapR Database stores for an indexed field defined on different data types, consider an example in which you have the following documents:

```
{"_id":"0", "field":0}
{"_id":"1", "field":[0,1,2]}
{"_id":"2", "field":{"subField":1}}
{"_id":"3", "field":{"subField":[1,2,3]}}
{"_id":"4", "field":[{"subField":1},{"subField":2}]}
{"_id":"5", "field":[{"subField":[1,2,3]},{"subField":[4, 5]}]}
```

The following table shows what an index defined on `field` stores and an OJAI query condition that matches the value stored in the index:

Document ID	Value Stored in Index Defined on `field`	Matching OJAI Query Condition
0	`0`	`{"$lt":{"field":1}}`
1	`[0,1,2]`	`{"$eq":{"field":[0,1,2]}}`
2	`{"subField":1}`	`{"$eq":{"field":{"subField":1}}}`

last-updated: Apr 11, 2022

Document ID	Value Stored in Index Defined on `field`	Matching OJAI Query Condition
3	`{"subField":[1,2,3]}`	`{"$eq":{"field":{"subField":[1,2,3]}}}`
4	`[{"subField":1},{"subField":2}]`	`{"$eq":{"field":[{"subField":1},{"subField":2}]}}`
5	`[{"subField":[1,2,3]},{"subField":[4, 5]}]`	`{"$eq":{"field":[{"subField":[1,2,3]},{"subField":[4,5]}]}}`

The following table shows what an index defined on `field.subField` stores and an OJAI query condition that matches the value stored in the index:

Document ID	Value Stored in Index Defined on `field.subField`	Matching OJAI Query Condition
0	*Missing* [1]	N/A
1	*Missing* [1]	N/A
2	`1`	`{"$lt":{"field.subField":5}}`
3	`[1,2,3]`	`{"$eq":{"field.subField":[1,2,3]}}`
4	*Missing* [2]	N/A
5	*Missing* [2]	N/A

Note:

[1] The index entry for documents 0 and 1 are missing because `field` is not a nested document in these documents.

[2] The index entries for documents 3 and 4 are missing because `field` is an array in those documents.

These indexes enable MapR Database to quickly look up values stored in the index. As shown in the table, these values can be scalars, arrays, or nested documents. In the case of the latter two types, MapR Database can only use the index for equality conditions.

Data Types of Included Fields

There are no type restrictions on the included fields in an index.

Using Container Field Paths as Indexed Fields

Starting in MapR 6.1, indexed fields in an index can be Container Field Paths on page 518. When you use a container field path as your indexed field and the field contains an array, then the index contains one row per array element. Therefore, the size of your index is proportional to the number of elements in the array.

Important: Consider the storage implications of your index if you decide to use a container field path as an indexed field. Also consider the performance impact from index updates. Updating an indexed array field in a single JSON document may require updating multiple index rows.

last-updated: Apr 11, 2022

When an indexed field is not a container field path, the index contains one row per field value.

For example, suppose you have the same set of documents shown earlier:

```
{"_id":"0", "field":0}
{"_id":"1", "field":[0,1,2]}
{"_id":"2", "field":{"subField":1}}
{"_id":"3", "field":{"subField":[1,2,3]}}
{"_id":"4", "field":[{"subField":1},{"subField":2}]}
{"_id":"5", "field":[{"subField":[1,2,3]},{"subField":[4, 5]}]}
```

The following table shows what each index stores if you define the index on the following container field paths:

- `field[]`
- `field[].subField`
- `field.subField[]`
- `field[].subField[]`

Each entry in the table represents a row in the index.

Document ID	Indexed Field Path			
	field[]	**field[].subField**	**field.subField[]**	**field[].subField[]**
0	`0`	*Missing* [1]	*Missing* [1]	*Missing* [1]
1	`0`	*Missing* [1]	*Missing* [1]	*Missing* [1]
	`1`			
	`2`			
2	`{"subField":1}`	`1`	`1`	`1`
3	`{"subField": [1,2,3]}`	`[1,2,3]`	`1`	`1`
			`2`	`2`
			`3`	`3`
4	`{"subField":1}`	`1`	*Missing* [2]	`1`
	`{"subField":2}`	`2`		`2`

last-updated: Apr 11, 2022

Document ID	Indexed Field Path			
	`field[]`	`field[].subField`	`field.subField[]`	`field[].subField[]`
5	`{"subField": [1,2,3]}`	`[1,2,3]`	*Missing* [2]	`1`
				`2`
	`{"subField":[4, 5]}`	`[4, 5]`		`3`
				`4`
				`5`

Note:

[1] The index entries for documents 0 and 1 are missing in all indexes except the index on `field[]` because `field` is not a nested document.

[2] The index entries for documents 3 and 4 are missing in the index on `field.subField[]` because `field` is an array in those documents.

To use these indexes, your query condition must use container field paths that correspond to the indexed fields. The following are sample OJAI query conditions that you might use with each index:

Indexed Field Path	Sample OJAI Query Condition	Matching Document(s)
`field[]`	`{"$eq":{"field[]":0}`	0, 1
	`{"$eq":{"field[]":{"subField": [1,2,3]}}}`	3, 5
`field[].subField`	`{"$eq":{"field[].subField":1}}`	2, 4
`field.subField[]`	`{"$gt":{"field.subField[]":2}}`	3
`field[].subField[]`	`{"$eq":{"field[].subField[]":2}}`	3, 4, 5

See OJAI Query Conditions Using Container Field Paths on page 2615 for further details about how these types of conditions behave.

Defining an Index With and Without a Container Field Path

As shown in these examples, defining an index on a container field path is different from defining an index on an entire array field. For example, an index on `field[]` can filter on individual array elements, whereas the index on `field` can filter only the entire value. Similarly, defining an index on `field[].subField[]`

last-updated: Apr 11, 2022

provides the most generality. It allows you to filter on any elements in `subField`, regardless of the data types in *both* `field` and `subField`. However, you also incur the overhead of storing more data in your index and the performance impact of updating the index.

Using Container Field Paths in Covering and Composite Indexes

With a container field path, you may need to add included fields in your index to make the index covering. See Covering Indexes and Container Field Paths on page 560 for details.

There are also limitations in the composite indexes you can define. See Composite Indexes and Container Field Paths on page 554 for details.

Comparisons and Sorts on Indexed Fields

Comparisons and sorts across data types differ depending on whether the types are comparable or noncomparable. This is not specific to secondary indexes. However, it impacts comparisons when using secondary indexes and the order MapR Database stores data in an index. See Using Comparable JSON Document Data Types in Comparisons and Sorts on page 513 and Using Non-comparable JSON Document Data Types in Comparisons and Sorts on page 514 to learn which types fall into each category and to understand their behavior.

Related concepts

Restrictions on Secondary Indexes on page 565
This topic lists and describes the restrictions on secondary indexes. It is important for you to understand the type, size, field definition, option, and index use restrictions when defining and using secondary indexes.

Restrictions on Secondary Indexes

This topic lists and describes the restrictions on secondary indexes. It is important for you to understand the type, size, field definition, option, and index use restrictions when defining and using secondary indexes.

Name Restrictions

You cannot use the following characters in the index name and in the indexed fields:

```
< > ? % \
```

To use the following characters in the index name and in the indexed fields, enclose them either in single or double quotes:

```
; | ( ) /
```

For example:

```
maprcli table index
add -path /volume1/MYTABLE -index
"MYTABLE1_ANALYSIS_1♦=#;{}&()/" \
  -indexedfields "●timestamp":desc,"●
♥,"LOTNo" -includedfields \
  "♥,"♦=#;{}&()/" (or)

maprcli table index
add -path /volume1/MYTABLE -index
'MYTABLE1_ANALYSIS_1♦=#;{}&()/' \
  -indexedfields "●timestamp":desc,"●
♥,"LOTNo" -includedfields \
  '♦,'♦=#;{}&()/'
```

To use either the **'** or the **"** character in the index name and in the indexed fields, enclose:

last-updated: Apr 11, 2022

- the ' character within double quotes (")
- the " character within single quote (')

For example:

```
maprcli table index
add -path /volume1/MYTABLE -index
"'MYTABLE1_ANALYSIS_1♦=#;{}&()/" \
  -indexedfields "●timestamp":desc,"♦●
♥,"LOTNo" -includedfields \
  "♦","♦=#;{}&()/" (or)

maprcli table index
add -path /volume1/MYTABLE -index
'"MYTABLE1_ANALYSIS_1♦=#;{}&()/' \
  -indexedfields "'●timestamp":desc,"●
♥,"LOTNo" -includedfields \
  '♥',"♦=#;{}&()/"
```

Type Restrictions

- If a composite index includes the same subfield in multiple indexed fields, the implied types of the subfields must be consistent.

  For example, you cannot create an index with the following indexed fields:

  ```
  a.b[].c, a.b.d
  ```

  Although subfield b appears in both indexed fields, in the first, it is an array and in the second, it is a nested document.

  See Composite Indexes and Container Field Paths on page 554 for more details.

Size Restrictions

- The maximum size of all indexed fields in an index is 32 KB.

  If the collective size exceeds 32 KB, then an insert of the corresponding document results in an encoding error (`INDEX_ROW_KEY_ENCODER_ERROR_ENCODING_IS_TOO_LONG`).

- The maximum number of indexes that you can create on a JSON table is 32.

Field Definition Restrictions

- You cannot specify individual array elements as indexed fields.
- You cannot specify a table's `_id` field as an indexed field.
- If a field contains an array of nested documents and you want to index on subfields in the nested documents, then you must define the indexed field using a container field path.
- You can include a specific field only once as either an indexed or included field, with the following two exceptions:

- The indexed field is a container field path:

```
maprcli table index add -path /
people \
   -index phoneNumberIdx \
   -indexedfields
Phones[].Number \
   -includedfields
Phones[].Number
```

- The field specifies a cast to another type.

  You can create an index in which the `score` field is an indexed field cast as a `double` type, and `score` is also an included field. The included field retains the original data type of the `score` field:

```
maprcli table index add -path /
castTable \
   -index castIdx1 \
   -indexedfields
'$CAST(score@DOUBLE)' \
   -includedFields score
```

  You can create an index in which the `score` field is an indexed field, cast as a `double` type, and the `score` field is also another indexed field, cast as a `long` type:

```
maprcli table index add -path /
castTable \
   -index castIdx2 \
   -indexedfields
'$CAST(score@DOUBLE)','$CAST(scor
e@LONG)'
```

- You cannot use casts with included fields.

- You cannot specify a field as either an indexed or included field if the field is also specified as a column family JSON path name.

  For example, suppose you have the following JSON table:

```
{
     "_id" : "ID",
     "a" :
          {
               "b" :
                    {
                         "c" :
"value",
                         "d" :
"value"
                    },
               "e" : "value"
          }
}
```

  If you create a column family at field `c` in the JSON path `a.b.c`, you cannot define field `a.b.c` as either an indexed or included field. You can define the fields `a`, `a.b`, and `a.b.d` as either indexed or included fields.

- You cannot specify an included field in which the data in the field spans more than one column family.

  In the following example, the included field `sl1.sl2` spans column families, `cf2` and `cf3`:

```
maprcli table cf list -path /cftab
compressionperm  readperm
traverseperm  jsonfamilypath
writeperm  minversions
maxversions  compression
ttl         inmemory  cfname
memoryperm
u:root           u:root
u:root
u:root     0
1            lz4
2147483647  false     default
u:root
u:root           u:root
u:root         sl1
u:root     0
1            lz4
2147483647  false     cf1
u:root
u:root           u:root
u:root         sl1.sl2.sl3
u:root     0
1            lz4
2147483647  false     cf2
u:root
u:root           u:root
u:root         sl1.sl2.sl3.sl4
u:root     0
1            lz4
2147483647  false     cf3
u:root

maprcli table index add -path /
cftab -index i1 -indexedfields
sl1.sl2.sl3.sl4.l4a,
sl1.l1a -includedfields
sl1.sl2,sl1.sl2.sl3.sl4.sl5.l5b -js
on
{
        "timestamp":1507419777919,
        "timeofday":"2017-10-07
04:42:57.919 GMT-0700 PM",
        "status":"ERROR",
        "errors":[
                {
                        "id":22,

"desc":"Data for included field
sl1.sl2 may not span more than one
column family."
                }
         ]
}
```

	• You cannot specify a composite index with more than one container field path as your indexed fields, unless the prefixes of the container field paths are the same. See Composite Indexes and Container Field Paths on page 554 for more details. • You cannot specify a composite index with an indexed field that is a subfield of another indexed field. For example, you cannot create an index with the following indexed fields: `a, a.b` The indexed field `a.b` is a subfield of the indexed field `a`.
Option Restrictions	• As indexes are automatically split, you cannot disable splits when you create your index.
Index Use Restrictions	• Indexes do not optimize non-existence filter conditions.

Queries that Benefit from Secondary Indexes

Secondary indexes benefit queries with filter conditions, ORDER BY clause, and projections.

They benefit these query elements in the following ways:

Filter Conditions	Eliminates full table scans, reducing the number of documents that MapR Database reads, if the filtering fields are keys in an index.
ORDER BY Clause	Eliminates the need to sort the data after scanning the index, if the index's sort order matches the query's sort order.
Projections	Eliminates the need to read the MapR Database JSON table, if all fields referenced in the query are fields in an index.

Note: The projections optimization is not supported for OJAI queries that execute through the OJAI Distributed Query Service, and SQL queries issued to MapR Drill. See OJAI Distributed Query Service on page 505 for details about the types of OJAI queries that use the service.

The following topics describe the specific query types and provide examples.

Using Indexes to Optimize Equality Conditions

Using indexes can help you improve the performance of queries that have equality conditions. You can define indexes that optimize equality conditions on scalar data fields, nested document and array fields, and container field paths.

If the index has a single key, the condition limits the index search to only the keys matching the scalar value. If the index has more than one key and there are equality conditions on all keys, the conditions limit the search to the combined matching values. If there are conditions on a subset of fields and the most significant keys have equality conditions, MapR Database limits the search to those scalar values.

last-updated: Apr 11, 2022

Assume that you have a MapR Database JSON table with documents in the following format:

```
{
   "_id": "10000",
   "FullName": {
      "LastName": "Smith",
      "FirstName": "John"
   },
   "Address": {
      "Street": "123 SE 22nd St.",
      "City": "Oakland",
      "State": "CA",
      "Zipcode": "94601-1001"
   },
   "Gender": "M",
   "AccountBalance": 999.99,
   "Email": "john.smith@company.com",
   "Phones": [
        {"Type": "Home", "Number": "555-555-1234"},
        {"Type": "Mobile", "Number": "555-555-5678"},
        {"Type": "Work", "Number": "555-555-9012"}
   ],
   "Hobbies": ["Baseball", "Cooking", "Reading"],
   "DateOfBirth": "10/1/1985"
}
```

The examples in the following sections reference this sample JSON document.

Indexes on Scalar Data Fields in Equality Conditions

The following table provides examples where MapR Database can and cannot use the index with equality conditions on scalar data. The last entry in the table illustrates the case where you can use index to optimize an equality condition in combination with a range condition.

Note: This example assumes that a composite index exists on fields `Address.State` and `Address.City`.

Query Condition	How MapR Database Uses the Index
`{ "$and":[ {"$eq":{"Address.State":"CA"}}, {"$eq":{"Address.City":"Oakland"}} ] }`	Performs a lookup on the specified state and city values, and reads the index until the conditions no longer match.
`{"$eq":{"Address.State":"CA"}}`	Performs a prefix lookup to find matching state values. The value of the `Address.City` field is not relevant. Continues reading from the index until the state field no longer matches `"CA"`.
`{ "$and":[ {"$in":{"Address.State": ["CA","NY","MA"]}}, {"$eq":{"Address.City":"Springfield"}} ] }`	Performs the following three lookups in the index: • `Address.State = "CA" and Address.City = "Springfield"` • `Address.State = "NY" and Address.City = "Springfield"` • `Address.State = "MA" and Address.City = "Springfield"`

last-updated: Apr 11, 2022

Query Condition	How MapR Database Uses the Index
`{"$eq":{"Address.City":"Oakland"}}`	Even if the query references the field `Address.State`, MapR Database cannot use the index unless there is also an equality condition on the leading key of the index, `Address.State`.
`{"$in":{"Address.State":["CA","NY","MA"]}}`	Performs three prefix lookups, one for each of the values in the `IN` clause.
`{ "$and":[ {"$eq":{"Address.State":"CA"}}, {"$ge":{"Address.City":"Oak"}} ] }`	Reads from the index starting at the condition `Address.State = "CA"` and `Address.City = "Oak"`. Continues reading the index until the condition `Address.State = "CA"` no longer qualifies.

Indexes on Nested Document Fields in Equality Conditions

Starting in MapR 6.1, you can define an index on fields that contain nested documents. These indexes benefit only equality conditions. The query condition must specify all subfields from the nested document. They must match the subfields of nested documents stored in your MapR Database JSON table. The order of the subfields is not relevant.

For example, if you define an index on the `Addresses` field, and specify the following query condition:

```
{
  "$eq":{
    "Addresses":{
      "Street":"123 SE 22nd St.",
      "City":"Oakland",
      "State":"CA",
      "Zipcode":"94601-1001"
    }
  }
}
```

MapR Database can use the index to locate the sample document shown earlier.

On the other hand, if you specify the following condition instead:

```
{
  "$eq":{
    "Addresses":{
      "City": "Oakland",
      "State": "CA"
    }
  }
}
```

When MapR Database reads using the index and applies this query condition, it does not match the sample document. The condition is missing the `Street` and `Zipcode` subfields. If you want to match on only the `City` and `State` subfields, you can define a composite index on those subfields as described in the previous section.

Indexes on Array Fields in Equality Conditions

Starting in MapR 6.1, you can define an index on fields that contain array data. These indexes benefit only equality conditions. The array elements and their order specified in your query condition must match the content and order stored in your MapR Database JSON table.

For example, if you define an index on the `Hobbies` field, and specify the following query condition:

```
{"$eq":{"Hobbies":["Baseball", "Cooking", "Reading"]}}
```

MapR Database can use the index to locate the sample document shown earlier.

On the other hand, if you specify the following condition instead:

```
{"$eq":{"Hobbies":["Cooking", "Baseball", "Reading"]}}
```

When MapR Database reads using the index and applies this query condition, it does not match the sample document. Although the individual array elements match, the order does not.

If `Hobbies` also has scalar data, MapR Database can use the index to locate documents with the following condition:

```
{"$eq":{"Hobbies":"Baseball"}}
```

If your MapR Database JSON table has a document where the `Hobbies` field has a single value "Baseball", MapR Database can use the index to locate the matching document.

Indexes on Container Field Paths in Equality Conditions

Starting in MapR 6.1, you can define an index using a container field path as the indexed field.

For example, suppose you want to search for individual hobbies within the `Hobbies` array field, rather than matching the entire array field. You can define an index on the following field:

```
Hobbies[]
```

The following examples show equality conditions that benefit from this index:

Query Condition	Description
`{"$eq":{"Hobbies[]":"Baseball"}}`	Finds documents that contain Baseball as a hobby
`{"$in":{"Hobbies[]":["Baseball","Cooking"]}}`	Finds documents that contain either Baseball or Cooking as a hobby

<pre>{
  "$and":[
  {"$eq":{"Hobbies[]":"Baseball"}},
  {"$eq":{"Hobbies[]":"Cooking"}}
  ]
}</pre>
	Finds documents that contain both Baseball and Cooking as hobbies

When using the `Hobbies[]` container field path in the query condition, the condition matches both array elements and individual scalar values.

last-updated: Apr 11, 2022

For another example, suppose you want to filter on phone types. You can define an index on the following field:

```
Phones[].Type
```

The following examples show equality conditions that benefit from this index:

Query Condition	Description
`{"$eq":{"Phones[].Type":"Mobile"}}`	Finds documents that have a mobile phone number
`{"$in":{"Phones[].Type":["Mobile","Work"]}}`	Finds documents that contain either a mobile or work phone number
`{ "$and":[ {"$eq":{"Phones[].Type":"Mobile"}}, {"$eq":{"Phones[].Type":"Work"}} ] }`	Finds documents that contain both mobile and work phone numbers

When using the `Phones[].Type` container field path in the query condition, the condition matches instances where `Phones` is an array of nested documents as well as a single nested document.

Important: To use an index defined on a container field path, the container field paths in the query condition and indexed fields must match.

The following table shows examples of conditions that **do not** benefit from the index shown:

Indexed Field	Query Conditions that *do not* Benefit
`Hobbies`	`{"$eq":{"Hobbies[]":"Baseball"}}` This condition requires an index defined on `Hobbies[]`.
`Hobbies[]`	`{"$eq":{"Hobbies":["Baseball", "Cooking"]}}` This condition requires an index defined on `Hobbies`.
`Phones[].Type`	`{"$eq":{"Phones[0].Type":"Mobile"}}` This condition cannot be used with indexes because you cannot define an index on array elements.
`temps[][]`	`{"$ge":{"temps[][1]":60}}` This condition cannot be used with indexes because you cannot define an index on array elements..
	`{"$eq":{"temps[]":[78,54]}}` This condition requires an index defined on `temps[]`.

last-updated: Apr 11, 2022

Indexed Field	Query Conditions that *do not* Benefit
`temps[]`	`{"$ge":{"temps[][]":60}}` This condition requires an index defined on `temps[][]`.

Related concepts

OJAI Query Condition Syntax on page 2606
OJAI defines a syntax for specifying query conditions that allows you to express query conditions in a JSON format. This topic describes the supported operators and provides examples of these query conditions.

Using Indexes to Optimize Range Conditions
Indexes can improve the performance of queries that have range conditions. The range conditions can appear in combination with equality conditions when the most significant index keys have equality conditions. You can define indexes that optimize range conditions on scalar data fields and container field paths.

The following range condition operators benefit from indexes:

- Less than (or equal to)
- Greater than (or equal to)
- Pattern matching operator LIKE, provided the pattern in the condition does not start with a wildcard character

Assume you have a MapR Database JSON table with documents in the following format:

```
{
   "_id": "10000",
   "FullName": {
      "LastName": "Smith",
      "FirstName": "John"
   },
   "Address": {
      "Street": "123 SE 22nd St.",
      "City": "Oakland",
      "State": "CA",
      "Zipcode": "94601-1001"
   },
   "Gender": "M",
   "AccountBalance": 999.99,
   "Email": "john.smith@company.com",
   "Phones": [
        {"Type": "Home", "Number": "555-555-1234"},
        {"Type": "Mobile", "Number": "555-555-5678"},
        {"Type": "Work", "Number": "555-555-9012"}
   ],
   "Hobbies": ["Baseball", "Cooking", "Reading"],
   "DateOfBirth": "10/1/1985"
}
```

The examples in the following sections reference this sample JSON document.

Indexes on Scalar Data Fields in Range Conditions

The following table provides examples of when MapR Database can and cannot use the index with range conditions on scalar data. Assume that a composite index exists on the `Address.State` and

last-updated: Apr 11, 2022

`Address.City` fields. To use both indexed fields in the composite index, you must have an equality condition on `Address.State`.

Filter Condition	How MapR Database Uses the Index
`{"$le":{"Address.State":"CA"}}`	Reads from the beginning of the index up to and including the condition `Address.State <= "CA"`.
`{"$gt":{"Address.State":"CA"}}`	Reads from the index starting at the condition `Address.State > "CA"` through the end of the index.
`{"$like":{"Address.State":"C%"}}`	Performs a simple prefix match starting at the condition `Address.State >= "C"`. Continues reading the index until the filter no longer qualifies.
`{ "$and":[ {"$eq":{"Address.State":"CA"}}, {"$ge":{"Address.City":"Oak"}} ] }`	Reads from the index starting at the condition `Address.State = "CA" and Address.City >= "Oak"`. Continues reading from the index until the condition `Address.State = "CA"` no longer qualifies.
`{ "$and":[ {"$in":{"Address.State": ["CA","NY","MA"]}}, {"$gt":{"Address.City":"Spring"}} ] }`	Performs these three lookups and reads through the index: • `Address.State = "CA" and Address.City > "Spring"` • `Address.State = "NY" and Address.City > "Spring"` • `Address.State = "MA" and Address.City > "Spring"`
`{ "$and":[ {"$gt":{"Address.State":"C"}}, {"$gt":{"Address.City":"Oak"}} ] }`	Reads from the index starting at the condition `Address.State > "C"` through the end of the index. Although `Address.City` is part of the index key, MapR Database does not use `Address.City > "Oak"` when initiating the index search. Applies that filter while reading the index.
`{"$le":{"Address.City":"Oak"}}`	Even if the query references the field `Address.State`, MapR Database cannot use the index unless there is also an equality condition on the prefix key of the index, `Address.State`.

Indexes on Container Field Paths in Range Conditions

Starting in MapR 6.1, you can define an index using a container field path as the indexed field.

For example, suppose you want to apply range conditions on individual hobbies within the `Hobbies` array field. You can define an index on the following field:

```
Hobbies[]
```

The following examples show range conditions that benefit from this index:

last-updated: Apr 11, 2022

Query Condition	Description
`{"$like":{"Hobbies[]":"B%"}}`	Finds documents that contain hobbies that start with the letter "B"
`{"$gt":{"Hobbies[]":"Read"}}`	Finds documents that contain hobbies with a value greater than "Read"
`{ "$elementAnd":{ "Hobbies[]":[ {"$ge":{"$":"D"}}, {"$lt":{"$":"J"}} ] } }`	Finds documents that contain hobbies that start with letters between "D" and "I", inclusive

When using the `Hobbies[]` container field path in the query condition, the condition matches both array elements and individual scalar values.

For another example, suppose you want to apply range filters on phone numbers. You can define an index on the following field:

```
Phones[].Number
```

The following examples show range conditions that benefit from this index:

Query Condition	Description
`{"$like":{"Phones[].Number":"555%"}}`	Finds documents with phone numbers that have a 555 prefix
`{ "$elementAnd":{ "Phones[]":[ {"$gt": {"Number":"408-555-1234"}}, {"$lt":{"Number":"408-555-9999"} ] } }`	Finds documents that contain phone numbers in the specified range

When using the `Phones[].Number` container field path in the query condition, the condition matches instances where `Phones` is an array of nested documents as well as a single document.

Related concepts

OJAI Query Condition Syntax on page 2606
OJAI defines a syntax for specifying query conditions that allows you to express query conditions in a JSON format. This topic describes the supported operators and provides examples of these query conditions.

Using Indexes to Optimize ORDER BY Queries
Using indexes can help you improve the performance of queries that have an ORDER BY clause. This includes ORDER BY clauses with either ascending or descending sorts, as well as more than one ordering field. The same index can optimize both filter conditions and the ORDER BY clause.

last-updated: Apr 11, 2022

To use the index for an ORDER BY query, the index's key list order and sort order must match the orderings specified in the query. If the index's keys also match filter conditions in the query, using the index also reduces the amount of data read from the index.

Index Key List Order and Sort Order Examples

The following table provides examples of when MapR Database can and cannot use an index for ordering, based on the index key list ordering and sort ordering specified. Assume that you have a table that has a composite index on fields `Address.State` and `FullName.LastName`. You have defined both keys in ascending order. Further assume that the query has an ORDER BY on the fields `Address.State` and `FullName.LastName`, both in ascending order:

Ordering in Query	Use of Index for Ordering
`Address.State:ASC`	Yes
`Address.State:DESC`	No Sort direction does not match.
`Address.State:ASC,` `FullName.LastName ASC`	Yes
`FullName.LastName:ASC`	No `Address.State` must be included as a prefix in the ordering.
`FullName.LastName:ASC,` `Address.State:ASC`	No Sort directions match, but the order of fields does not match.

Filtering and ORDER BY Query Examples

Assume that you have a composite index defined with the following two indexed fields:

- `Address.State:ASC`
- `FullName.LastName:ASC`

The following table shows examples for different filtering and ORDER BY scenarios using this composite index:

Query Condition	Ordering in Query	Index Use
`{"$eq":{"Address.State":"CA"}}`	`FullName.LastName:ASC`	Both filtering and ordering
`{"$gt":{"Address.State":"CA"}}`	`Address.State:ASC`	Both filtering and ordering
`{"$gt":{"Address.State":"CA"}}`	`Address.State:DESC`	Only filtering, because the sort direction does not match

last-updated: Apr 11, 2022

Query Condition	Ordering in Query	Index Use
`{ "$and":[ {"$eq":{"Address.State":"CA"}}, {"$ge": {"FullName.LastName":"Smith"}} ] }`	`FullName.LastName:ASC`	Both filtering and ordering
`{"$gt":{"Address.State":"CA"}}`	`Address.State:ASC, FullName.LastName:ASC`	Both filtering and ordering
`{"$gt":{"Address.State":"CA"}}`	`FullName.LastName:ASC`	Only filtering
`{"$in":{"Address.State":["CA","TX"]}}`	`FullName.LastName:ASC`	Only filtering
`{"$ge":{"FullName.LastName":"Smith"}}`	`Address.State:ASC, FullName.LastName:ASC`	Only ordering, because `FullName.LastName` is not a prefix in the filter lookup

Index Sort Order for Complex Types

Although you can define indexes on complex data types, there are limitations in the behavior.

Arrays and Nested Documents	Indexes defined on arrays and nested documents do not have a meaningful ordering because these types do not have a defined ordering.
Container Field Paths	You cannot order on a container field path. For example, you can define an index on the field `a[].b`, but you cannot order on it.

Partial Sorts with Non-Covering Indexes

MapR Database updates secondary indexes asynchronously, which can result in updates to the index lagging the parent JSON table. You can avoid this behavior in your OJAI application by setting a query option in your application. See Avoiding Partial Sorts with Secondary Indexes in OJAI on page 2589 for details about how to do this.

One consequence of this index update lag is the impact on queries that use non-covering indexes to provide the ordering of a query. Since the index is not fully synchronized with its parent JSON data, data read through the index might be out of date.

The following example illustrates this behavior.

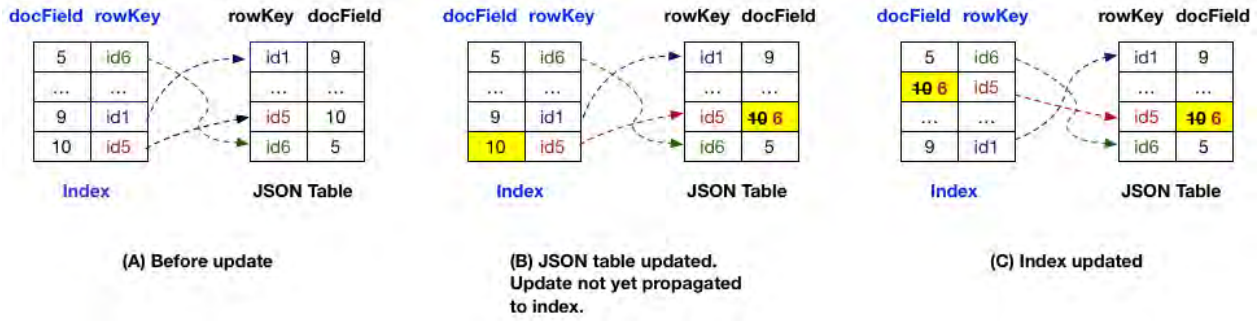
- Suppose you have a query with the following criteria:
  - Selects `docField`
  - Filter condition where `docField >= 5`
  - Order by on `docField`

last-updated: Apr 11, 2022

- You have an index where `docField` is an indexed field. The index optimizes both the filter condition and order by clause. The query also selects other fields, so the index is a non-covering index for the query.
- When reading through the index, MapR Database reads a document in which the `docField` value is 9. The data for that field in the JSON table is also 9. The data is consistent.
- The next entry in the index has `docField` set to 10. This value is in the proper sort order relative to the previous value of 9, but the data in the JSON table has changed from 10 to 6. The update is not yet reflected in the index.
- MapR Database returns the value 6 (not 10), which is out of order, relative to data previously read from the index.

The following table and diagram illustrates this example:

Update State	Query Result in docField Sort Order
Before update	5, …, 9, 10
JSON table updated, but not index	5, …, 9, 6
Index updated	5, 6, …, 9



Note: This behavior does not occur with covering indexes. MapR Database only reads from a single data source, the index, when using covering indexes.

See Asynchronous Secondary Index Updates on page 587 for a more detailed discussion of asynchronous index updates.

Using Indexes to Optimize Projections in Queries

OJAI queries that do not use the OJAI Distributed Query Service can use indexes even when there are no filter conditions referencing the fields of an index. This requires a full scan of the index. However, in cases where all fields referenced in the query are fields in an index, the need to read the MapR Database JSON table is eliminated. The referenced fields can be either indexed fields or included fields

The following table provides examples where MapR Database can and cannot use the index for projections. Assume you have an index with the following fields:

- Indexed fields - `IdxField1`, `IdxField2`
- Included fields - `Field3`, `Field4`, `Field5`

Further assume that the fields referenced in the index are a small subset of the total fields in the MapR Database JSON table. With these assumptions, avoiding reads on the JSON table is beneficial.

OJAI Query Elements	Use of Index for Projections
• SELECT `IdxField1`, `Field4`	Yes All fields referenced are fields in index.
• SELECT `Field3`, `Field4`	Yes All fields referenced are fields in index.
• SELECT `IdxField1`, `Field6`	No `Field6` not included in index.
• SELECT `IdxField1`, `Field3` • WHERE condition on `IdxField2`	No All fields referenced are fields in the index, but the index cannot be used with the WHERE condition.
• SELECT `IdxField1`, `Field3` • WHERE condition on `Field4`	No All fields referenced are fields in the index, but the index cannot be used with the WHERE condition.
• SELECT `IdxField1`, `Field4` • ORDER BY on `IdxField2`	No All fields referenced are fields in the index, but the index cannot optimize the ORDER BY. The query needs the OJAI Distributed Query Service to sort large data sets.

Projections on Container Field Paths

When your query projects a container field path and the container field path is an included field in an index, then MapR Database can use the index for the projection. It is not enough for the container field path to be an indexed field. See Covering Indexes and Container Field Paths on page 560 for details.

Using Multiple Indexes to Optimize Query Conditions
Indexes benefit queries that have multiple filter conditions. The OJAI Distributed Query Service can optimize these queries by creating query plans that scan multiple indexes and take the intersection of the matching documents.

Scanning multiple indexes is an alternative to using Composite Indexes on page 553. The following example illustrates how the OJAI Distributed Query Service does this.

Suppose you have a JSON table with an index on the `Address.State` field, another index on the `Address.City` field, and the query has the condition:

```
{
  "$and":[
    {"$lt":{"Address.State":"D"}},
    {"$gt":{"Address.City":"Oak"}}
  ]
}
```

The OJAI Distributed Query Service creates a query plan that uses the indexes as follows:

- Performs a scan on the first index using the `Address.State < "D"` condition.
- Performs a scan on the second index using the `Address.City > "Oak"` condition.
- Takes the intersection of the document IDs that match both conditions.

If you do not apply conditions on both `Address.State` and `Address.City` in most of your queries, defining separate indexes instead of a single composite field index may be more desirable. With a

last-updated: Apr 11, 2022

composite index on fields `Address.State` and `Address.City`, the query service does not choose the index unless there is a condition on field `Address.State`. If there is a condition on `Address.State`, the query service can choose the composite index. However, in order to restrict the search on both fields, there must be an equality condition on `Address.State`. See Using Indexes to Optimize Equality Conditions on page 570 for further details.

You can define separate single key indexes as well as a composite index, but this requires more storage and impacts performance throughput. See the sections on storage and throughput considerations in Designing Secondary Indexes on page 591 for further guidance.

The following table illustrates the differences between using a composite index versus multiple indexes. The second column shows the behavior when you have a composite index defined on `(Address.State, Address.City)`. The third column shows the behavior when you have separate simple indexes defined on each field.

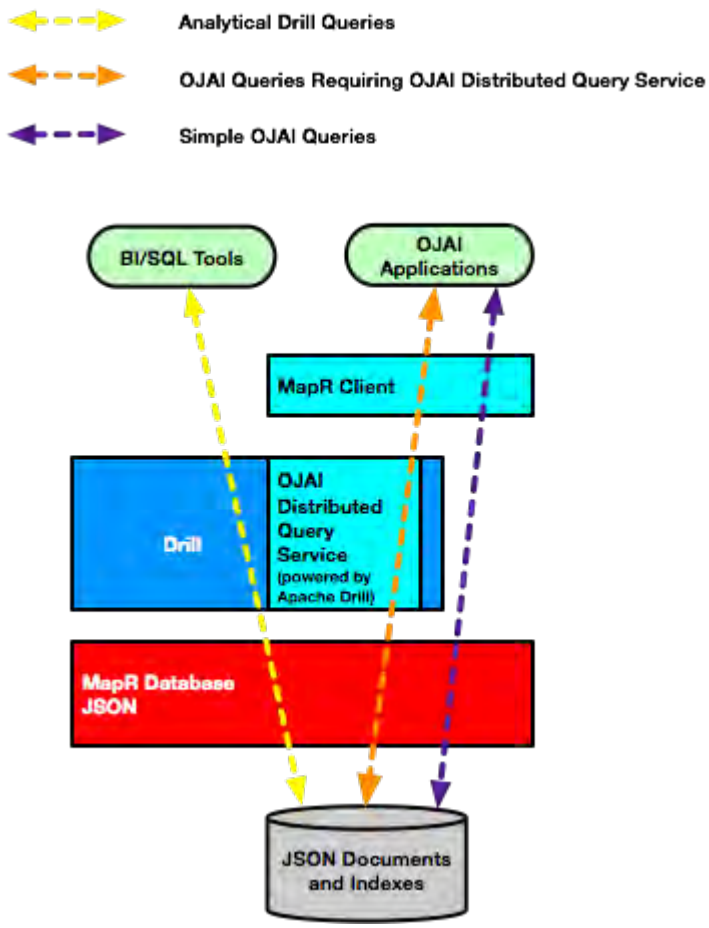
Filter Condition	Composite Index	Two Simple Indexes
`{ "$and":[ {"$eq":{"Address.State":"CA"}}, {"$eq": {"Address.City":"Oakland"}} ] }`	Index searched using both conditions	Separate index searches using each filter condition. Results intersected.
`{ "$and":[ {"$eq":{"Address.State":"CA"}}, {"$ge":{"Address.City":"Oak"}} ] }`	Index searched using both conditions	Separate index searches using each filter condition. Results intersected.
`{"$eq":{"Address.State":"CA"}}`	Index searched using single filter condition	Same as composite index case. However, since the simple index has only a single indexed field, it is smaller and more efficient to read.
`{ "$and":[ {"$ge":{"Address.State":"CA"}}, {"$eq": {"Address.City":"Oakland"}} ] }`	Index search initiated using only the `Address.State` condition. `Address.City` filter applied while reading the index.	Separate index searches using each filter condition. Results intersected.
`{"$eq":{"Address.City":"Oakland"}}`	Cannot use index	Index searched using single filter condition

Selection and Execution of Secondary Indexes

This section provides an overview of secondary index selection and execution in MapR Database JSON. It describes the variations in functionality, depending on the components you are using.

The following diagram summarizes the code paths and the components involved when using secondary indexes in MapR Database JSON. See OJAI Distributed Query Service on page 505 for more information about the components and code paths.

last-updated: Apr 11, 2022



Index Selection

All three code paths use a cost-based approach to select an optimal query plan. Cost based optimization chooses between alternatives where it may not be obvious which is the better index to use. Assume that you have the following two indexing options:

- Index 1 can be used to filter condition A in a query but cannot satisfy the sort criteria.
- Index 2 can be used to filter condition B in a query and also satisfy the sort criteria.

If filter condition A is more selective than filter condition B, although using index 1 requires reading less data, it requires a sort of the data. In contrast, using index 2 requires reading more data but does not incur the cost of sorting the data. A cost based optimizer estimates the cost of both options and chooses the one with the lower cost. It also estimates the cost of a full table scan. It may choose the full table scan if the index-based plans do not use selective filters.

The Simple OJAI Query code path can use indexes even when a query does not have filter or ORDER BY conditions that match the fields of an index. See Using Indexes to Optimize Projections in Queries on page 580 for details.

The Drill query optimizer and the optimizer used by the OJAI Distributed Query Service can select multiple indexes to process a query. The OJAI Distributed Query Service scans the indexes and takes the intersection of the matching documents from each index. The MapR client invokes scans of only a single index.

The rest of this section generally discusses the optimizer flow. Except where noted, the discussion applies to the optimizer used in all three code paths.

MapR Database gives the optimizer a list of indexes associated with the JSON table referenced in the query. The optimizer enumerates through the possible index choices using the following steps:

1. Identifies the set of indexes whose keys match filter conditions and possibly also the ORDER BY specification.
2. Estimates the cost of using each index.
3. Considers combinations of indexes and estimates the cost of these combinations. (Applies to the Drill and OJAI Distributed Query Service optimizers only.)

Using the cost estimates, the optimizer selects the index (or indexes) with the lowest cost, or if appropriate, a full table scan. The cost is a function of the index properties, table size, and selectivity of the filter conditions applied. Each of these factors contribute to the estimated cost in the following ways:

Index Properties

MapR Database provides the Drill optimizer with index properties. Index properties include the fields that comprise the index, whether the field is an indexed or included field, and the sort direction of each indexed field.

Table Size

MapR Database maintains information about table regions, including table size. The optimizer uses table size when calculating the cost of the query plan.

If JSON tables are small and fit into a single region, the overhead of using indexes on the table may not provide enough performance benefits to justify an index-based plan. In such a scenario, the optimizer could calculate a full table scan as cheaper to perform than an index scan, rendering any index on the table unnecessary. Even if you apply selective filters on small tables, the overhead of using indexes may not provide performance benefits.

Filter Selectivity

Filter Selectivity is the estimated number of rows based on the selectivity of each conditional expression in the WHERE clause. Filter selectivity is calculated as:

```
(output row count)/(total table row count)
```

For example, if you have 100 documents and 25 documents qualify the filter condition, the selectivity is .25.

Filter selectivity ranges between 0 and 1. The closer to 0, the more selective the filter. The more selective a filter, the lower the cost. High filter selectivity results in better query performance. If filter conditions are not selective enough for the optimizer to choose the index, remove the index to free up storage.

For example, defining an index on a field like `gender`, which has only two possible values, does not result in selective filtering. Consider adding other fields to define a composite index to make filtering with that index more selective. In general, define indexes on high cardinality fields unless your queries also sort on those fields.

For covering queries, Drill selects an index plan if the number of rows selected is less than or equal to .75 of the total number of rows in the JSON table. If the number of rows selected is greater than .75 of the total

last-updated: Apr 11, 2022

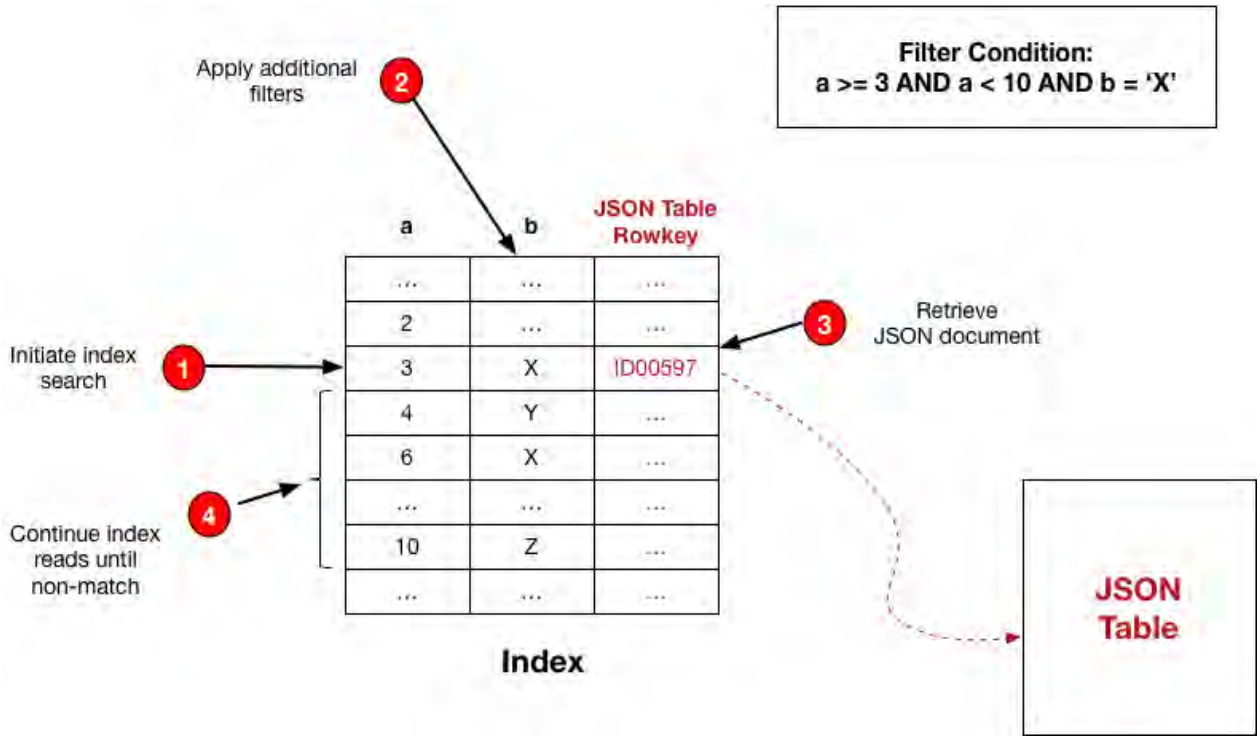
number of rows in the JSON table, Drill performs a full table scan.

For non-covering queries, the threshold is .025.

Note: In the **Simple OJAI Queries** code path, if you are using the OJAI API, you can force the MapR client to use a particular index, regardless of cost considerations. See Forcing Secondary Index Usage in OJAI on page 2589 for details.

Index Execution

After either the MapR client or Drill select an optimal query plan, MapR Database has the index (or list of indexes, in the case of a plan generated by Drill) from which to read. It reads the index to retrieve the corresponding documents from the JSON table. The following diagram and table illustrate the flow for a read from a composite index created on fields a and b.



Step #	Description	Details
1	Initiates index search	MapR Database searches the index, starting at the condition a >= 3.
2	Applies additional filters	MapR Database applies the filter condition on field b. It either moves to step 3 or 4, depending on whether the condition b = 'X' matches. For example, when b contains the value 'X', it proceeds to step 3. When b contains the value 'Y', it skips to step 4.
3	Retrieves JSON document	Using the rowkey in the entry, MapR Database reads the corresponding JSON document.

last-updated: Apr 11, 2022

Step #	Description	Details
4	Continues index reads until non-match	MapR Database reads the subsequent index keys provided they match the filter condition a `< 10`. If the condition matches, it goes back to step 2. Otherwise, MapR Database stops the search. For example, the reads stop when MapR Database reads the value 10 from field a.

Note: When a covering index satisfies the query, MapR Database skips reading the JSON table. TThis read is not required because the index provides all selected fields. In the preceding example, the MapR Database skips step 3.

Implementation of Secondary Indexes
This topic describes how MapR Database implements secondary indexes. It provides an overview of basic architectural concepts and the rationale behind design choices.

Global Indexes

MapR Database implements secondary indexes as *global indexes* rather than *local indexes*. With local indexes, each JSON table's regions (also called tablets) has a corresponding index tablet. The JSON table's and index's tablets are co-located. In contrast, with a global index, the index is a single, separate table with its own tablets and split points. Unlike JSON tables, indexes are always auto-split. There is no option to disable auto-splitting. When splitting index tablets, indexes are range partitioned by default. An alternative to range partitioning is to use hash partitioning to avoid creating hot spots.

Global indexes have the following advantages:

- They provide an ordering across all values in the indexed fields. A scan through the index can generate the sort required by ORDER BY clauses.
- They avoid having to read all partitions. When the data is range partitioned, MapR Database can direct index scans to the subset of partitions that qualify the desired key range.
- They require less data processing by minimizing the partitions that need to be read.

In summary, global indexes are well suited for scalable, read intensive use cases.

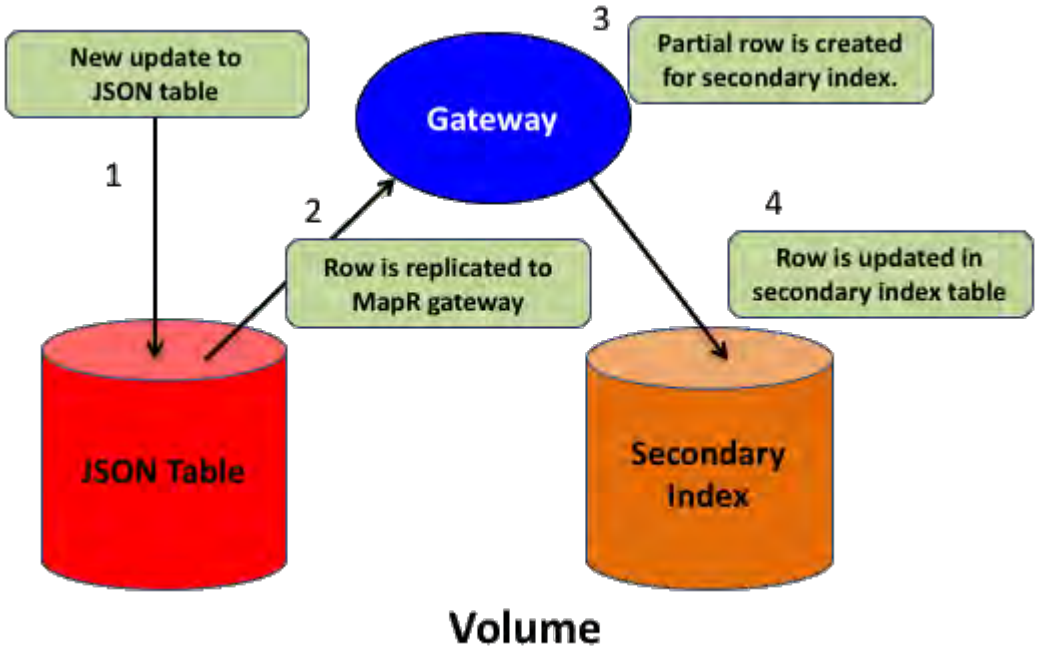
While global indexes are optimized for read intensive use cases, maintaining a global index incurs more overhead. Updates are more expensive if the JSON table and its indexes are on different nodes in the cluster.

Index Placement

Each secondary index shares the same volume and topology as its JSON table. Users cannot specify the path of an index. This simplifies the behavior of snapshot and mirroring.

Data Flow

After a secondary index has been created on a JSON table, the following occurs when an update (put operation) is made to the table:



1. A document in the JSON table is updated.
2. The row with the change is replicated to an internal MapR gateway.
3. The MapR gateway determines whether a secondary index is impacted and, if so, creates a partial row that contains the secondary index's indexed and included fields.
4. The secondary index row is updated.

When a secondary index is added, data that is already present in the JSON table is propagated to the index using a scan of the JSON table that retrieves indexed and included fields. This replaces step 2. Steps 3 and 4 are executed to populate the index.

Asynchronous Secondary Index Updates
Secondary indexes are updated asynchronously. The asynchronous approach favors performance and scalability over synchronous, transactional updates. However, this also means that indexed data can be stale compared to data in the JSON table, even though the data eventually becomes consistent with the JSON table data.

Impact of Asynchronous Indexes

By updating the index asynchronously, this avoids delaying updates to the JSON table.

From a user point of view, secondary indexes updates are complete when the MapR Database table data appears in the index. This occurs without application developers having to write any explicit code. Because indexes are asynchronously updated relative to the JSON table, there is a lag in updates appearing in the index. For a reasonably sized cluster, secondary index updates will typically occur within a few seconds of the update on the JSON table. When the JSON table and its secondary indexes are on separate nodes, the updates to the index are more expensive. The lag is potentially higher.

The following example illustrates how the lag in updates impacts queries that use indexes.

Suppose you have a JSON table that has a document with `_id=DOC1`. An update occurs on the indexed field, `a.b.c`, changing the value from v1 to v2. For queries that use a covering index, any of the following values might be returned for the `(_id, a.b.c)` pair:

- Only (DOC1,v1) - This occurs if the new value v2 has not yet been indexed.

- Only (DOC1,v2) - This occurs if the new value v2 is indexed and the old value v1 is deleted.
- Both (DOC1,v1) and (DOC1,v2) - This occurs if the new value v2 is indexed and the old value v1 is not yet deleted.
- Neither (DOC1,v1) nor (DOC1,v2) - This occurs if the value v1 is not indexed. The newer value v2 is not yet indexed, because value v1 is always indexed first.

For queries that use non-covering indexes, MapR Database re-reads the indexed and included fields when reading additional fields from the JSON table. This ensures that the query results are consistent in spite of update lags.

In the case where a non-covering index provides the ordering for the ORDER BY specification of a query, index lag can result in a partial sort of the result. See Partial Sorts with Non-Covering Indexes on page 579 for further details.

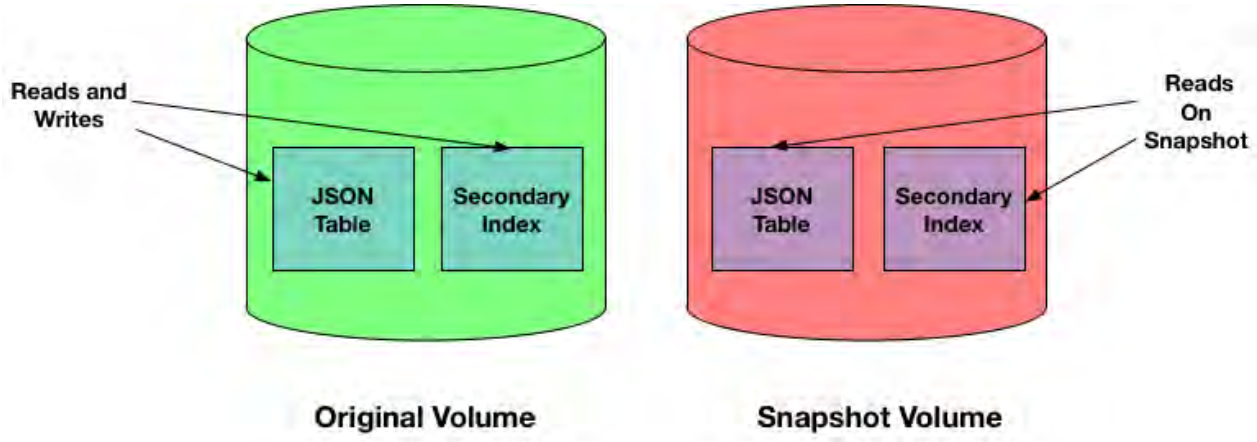
See Troubleshooting Secondary Indexes on page 1092 for information about how to determine if an index is lagging its JSON table.

Snapshots

Queries against snapshots containing tables with secondary indexes can return inconsistent results. This occurs if the data queried is actively changing at the time of snapshot creation. When creating a snapshot, if a secondary index on a JSON table does not have current data due to asynchronous updates of the index, the snapshot retains the lag in updates. The lag leads to the following behavior, which is similar to the behavior discussed in the previous section.

- For a query using a covering index, if the indexed data is out of sync, the query could return data that is current, old, or both.
- For a query using a non-covering index, if the indexed data is out of sync, the query returns the most recent data records.

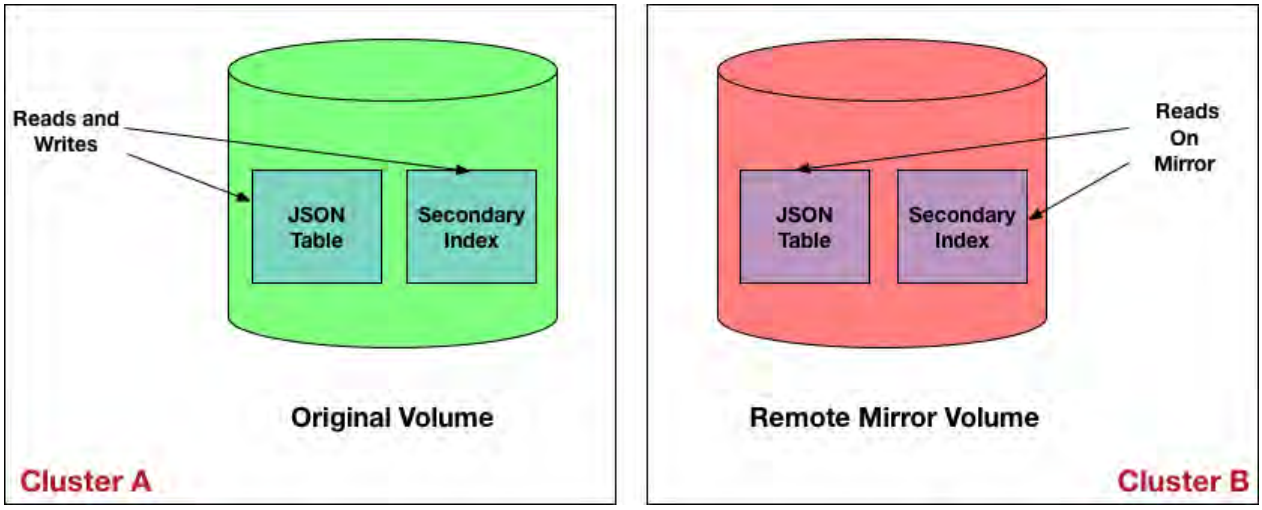
Important: Unlike data in the original volume, with snapshots, any lag between a JSON table and its secondary index will never get resolved. The snapshot data is read-only and never updated..



Mirroring

Queries against mirror volumes behave like queries against snapshots. Lags in the source volume carry over into the mirror volume. Upon refreshing a mirror volume, the lag can resolve itself.

last-updated: Apr 11, 2022



Reading Your Own Write Operations

Certain classes of applications require users to immediately see the data they have written. In these cases, getting stale data can confuse users. Think about an expense report application example where the user enters his expenses and wants to immediately see the entries. The asynchronous nature of indexes could be an issue in such a case. To avoid the possibility of reading stale data due to asynchronous indexes, the Java OJAI API Library provides functionality that enables you to read the result of your own write operation. See Reading Your Own Writes in Java OJAI on page 2668 to learn about how to use this feature in your application.

Replication and Security
Describes how secondary indexes are impacted by replication and security.

Replication

Secondary indexes are not replicated when tables are replicated by table replication (using the replication gateway). Only the JSON table data is replicated.

If you intend to query destination tables and use indexes, you must explicitly add an index to the replica table. Replicating tables and adding indexes are independent of each other.

Security

Secondary indexes reflect the access permissions of the underlying JSON table. *ACE* permissions are required on all indexed and included fields of an index before a query can use an index.

To add a secondary index on a JSON table, you need the `indexperm` permission. The table owner automatically has permission, but any other user must be assigned `indexperm` permission.

See table create on page 1788 for information about *ACE*, and table edit on page 1822 for information about table permissions.

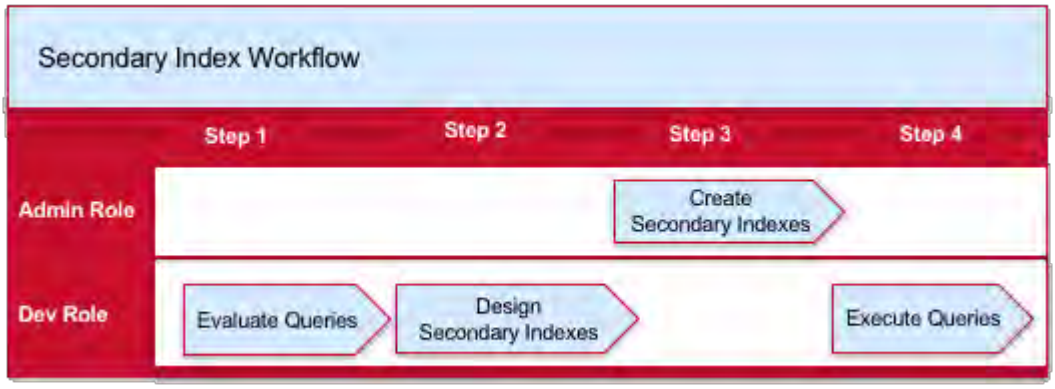
Understanding the Secondary Index Workflow

Describes the overall workflow for using secondary indexes. This includes the roles of different users and the workflow steps involved.

Before deploying secondary indexes, it is assumed that you have installed and configured MapR Database and MapR-Drill to use secondary indexes, and have created and populated your MapR Database JSON tables. Implementing secondary indexes on JSON tables in MapR Database requires that you understand indexing concepts, know which administrative tasks to perform, and design your indexes to provide the most benefits for your queries.

last-updated: Apr 11, 2022

The following diagram depicts the workflow and identifies the roles and order of tasks. Each step contains a link to a section in this page with further details.



1. How to Evaluate Queries that Benefit from Secondary Indexes
2. How to Design Secondary Indexes
3. How to Create Secondary Indexes
4. How to Query MapR Database JSON Tables

The following is a brief summary of each step:

1. Evaluate your queries to identify those that can benefit from indexes.
2. Design your indexes by determining which fields need to be indexed.
3. Create your indexes using either the Control System or `maprcli`.
4. Execute your queries.

How to Evaluate Queries that Benefit from Indexes

MapR Database JSON supports indexes with various properties. Each property benefits a certain class of queries. As part of deciding which of your queries will benefit from indexes, it is important to have a general understanding of these concepts. See Types of Secondary Indexes on page 549 and Queries that Benefit from Secondary Indexes on page 570 for more information.

How to Design Secondary Indexes

After you decide which queries can benefit from indexes, determine the set of indexes that provide the maximum benefits. See Designing Secondary Indexes on page 591 for more information.

How to Create Secondary Indexes

You can create secondary indexes using either the Control System or the maprcli table index command.

For example, to create a secondary index on the `name` field, use the following `maprcli` command:

```
maprcli table index add -path /Data/business -index newIndex -indexedfields
name
```

See Managing Secondary Indexes on page 1089 for other commands to manage secondary indexes.

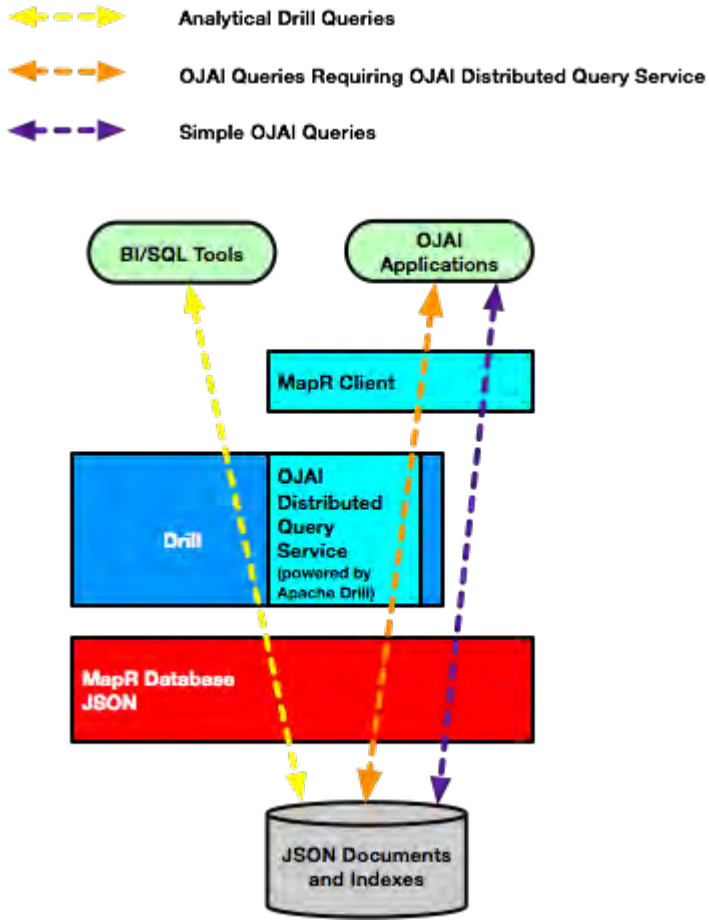
How to Query MapR Database JSON Tables

Depending on your use case, applications can access data in MapR Database through the following client interfaces:

OJAI Client API	Use for user-facing applications that need very high concurrency and ultra-low latency. The API is available in Java, Node.js, and Python.
MapR Database JSON REST API	Use for applications in which you want to access MapR Database JSON with HTTP calls.
MapR Drill SQL	Use for performing operational analytics or Business Intelligence (BI) for medium-to-high complexity queries that require low-to-medium concurrency and interactive response times.

These APIs seamlessly select the optimal indexes to use. You do not need to write explicit code or provide directives on which indexes to use.

The following diagram summarizes the components involved in the different scenarios.



For OJAI applications, the MapR client chooses the more appropriate of two possible execution paths, without user interaction. One of the paths leverages the OJAI Distributed Query Service on page 505, which supports more advanced index selection and parallel query execution. It also supports sorting large data sets. For example, if the sort order specified in your OJAI query does not match the sort order of an index, the MapR client automatically invokes the OJAI Distributed Query Service to perform the sort.

Designing Secondary Indexes

It is important that you create secondary indexes that provide the most benefit to your MapR Database JSON queries. This topic describes a general design approach that includes identifying query patterns,

last-updated: Apr 11, 2022

using common query patterns involving filters and ordering to determine which indexes to create, weighing the benefits of indexes against their update and storage costs, and taking into consideration index limitations.

The following diagram summarizes the general approach to designing your indexes:

1. Identify common query patterns
2. Determine potential indexes to create based on your query patterns
3. Evaluate the impact of index synchronization
4. Evaluate the storage requirements of indexes
5. Consider index restrictions
6. Evaluate tradeoffs and limitations

Before designing your secondary indexes in relationship to your queries, make sure you understand the index feature, how to set up and use indexing, the commands used to perform tasks, and how to query the data through your application. The following cover these topics:

- Secondary Index Concepts on page 547
- Understanding the Secondary Index Workflow on page 589

Identify Query Patterns

Query patterns, such as queries with filter conditions and ORDER BY clauses, indicate where indexes can improve performance. If a query does not contain selective filters, the overhead of using an index may cost more than a full table scan. You should also define your indexes so a single index benefits either multiple queries or individual queries that you run most often.

See Selection and Execution of Secondary Indexes on page 582 to understand how MapR Database chooses which secondary indexes to use and how they improve performance.

Determine Potential Indexes Based on Query Patterns

Based on your query patterns, the following table describes the types and characteristics of indexes you might want to create:

Identified Query Pattern	Potential Indexes to Create
Compares individual fields with selective filter conditions	Define single field indexes on the fields that you compare against. Verify that the fields contain supported data types.

last-updated: Apr 11, 2022

Filters against specific combinations of fields	Define composite field indexes instead of single field indexes. Specify the sequence of the index keys so fields that appear in equality conditions are the prefixes in the keys. See Using Multiple Indexes to Optimize Query Conditions on page 581 for additional guidance on defining composite vs single field indexes.
Accesses a subset of fields in a document, but does not filter or sort on these fields	Add those fields as included fields in indexes.
Filters on a subfield in a nested document	Define the index key on the subfield.
Filters on subfields in nested documents that are array elements	Define the index key using a container field path: for example, `arrayField[].subField`.
Filters and projects using a container field path	Define the container field path as both an indexed field and included field. See Covering Indexes and Container Field Paths on page 560 for more details.
Filters on individual elements of an array, which can appear in any position in the array	Define an index using a container field path: for example, `arrayField[]`.
Issues Drill SQL queries with filter conditions that contain CAST expressions	Specify the CAST function when defining the index key.
Sorts on fields	Define the sequence and order direction of the index keys to match the sequence and order direction of the fields your query sorts. If the sort order of the index keys matches the insertion order of documents, define hashed indexes.
Sorts on one set of fields and filters on another set using equality conditions	Define a composite index so that fields using equality conditions are the prefixes in the index keys, followed by the sort fields.

Evaluate Tradeoffs and Limitations

Synchronizing Indexes

When you design your indexes, remember that MapR Database must synchronize each index when you insert and update documents in the corresponding JSON table. This impacts the throughput performance of inserts and updates because MapR Database must perform additional writes. The impact increases with each additional index.

MapR Database performs the synchronization operation asynchronously, which minimizes throughput overhead. The consequence is that an index may be inconsistent relative to its JSON table. If your application cannot tolerate lag time between the update to the JSON table and the update to the index, you should take that into consideration when deciding whether to index specific fields.

See Asynchronous Secondary Index Updates on page 587 for more details about this feature.

Index Storage Requirements

Indexes increase your storage requirements. The storage size depends on the number of indexed and included fields in the index and the size of values stored in those fields. As the size of the index increases, the cost of reading the index also increases.

Consider the storage costs when creating indexes and deciding on the fields to add to the index.

Index Restrictions

When designing your indexes, make sure MapR Database indexes support the functionality you need. For example, it may not be possible to create an index on a particular field path.

See Restrictions on Secondary Indexes on page 565 for a complete list.

last-updated: Apr 11, 2022

Related concepts

Types of Secondary Indexes on page 549
MapR Database JSON supports several index types, including simple indexes, composite indexes, hashed indexes, and indexes with casting. This section describes the properties of these indexes and the situations where each provides value.

Queries that Benefit from Secondary Indexes on page 570
Secondary indexes benefit queries with filter conditions, ORDER BY clause, and projections.

Examples of Designing Secondary Indexes

These examples illustrate the concepts behind designing your secondary indexes. Although the examples focus on query patterns and do not account for sizing, storage, and updates, you should always weigh the benefits of indexes against these other requirements.

Assume that you have a MapR Database JSON table with the following customer data:

```
{
   "_id": "10000",
   "FullName": {
      "LastName": "Smith",
      "FirstName": "John"
   },
   "Address": {
      "Street": "123 SE 22nd St.",
      "City": "Oakland",
      "State": "CA",
      "Zipcode": "94601-1001"
   },
   "Gender": "M",
   "AccountBalance": 999.99,
   "Email": "john.smith@company.com",
   "Phones": [
      {"Type": "Home", "Number": "555-555-1234"},
      {"Type": "Mobile", "Number": "555-555-5678"},
      {"Type": "Work", "Number": "555-555-9012"}
   ],
   "Hobbies": ["Baseball", "Cooking", "Reading"],
   "DateOfBirth": "10/1/1985"
}
```

The following table contains fields in the document that are candidates for indexing based on the sample queries:

Query #	Query	Candidate Fields for Indexing
1	Find all customers who were born in the 1970s.	• `DateOfBirth`
2	Find all customers who have an account balance greater than $10K. Order the information in descending order of balance.	• `AccountBalance`
3	List customers who live in California, ordering the list by `LastName`, `FirstName`.	• `Address.State` • `FullName.LastName` • `FullName.FirstName`
4	Find the ids and emails of customers who live in a specific zip code.	• `Address.Zip`

last-updated: Apr 11, 2022

Query #	Query	Candidate Fields for Indexing
5	Find customers who live in a specific set of states and have an account balance less than a specific value.	• `Address.State` • `AccountBalance`
6	Find male customers having a last name starting with the letter "S."	• `Gender` • `FullName.LastName`
7	Find all customers who have "Reading" as a hobby.	• `Hobbies[]`
8	Find all customers who have a mobile phone number with a prefix of "650".	• `Phones[].Type` • `Phones[].Number`

The following table contains indexes you can create to optimize the queries listed in the previous table and the rationale for doing so:

Index	Rationale
Simple index on `DateOfBirth`	Optimizes the range condition on `DateOfBirth` in Query 1. You need not create a hashed index, because it is unlikely that the order of `DateOfBirth` correlates with the insert order of new data.
Simple index on `AccountBalance`, specified as a descending key	• Optimizes the range condition on `AccountBalance` in Query 2. • Descending order of key meets the ordering criteria in Query 2. • Also optimizes the range condition on `AccountBalance` in Query 5 in combination with the index on `Address.State`.
Composite index on: • `Address.State` • `FullName.LastName` • `FullName.FirstName`	• Optimizes both the equality condition on `Address.State` and ordering in Query 3. • Inclusion of the name fields in the index meets Query 3 ordering. • Also optimizes the IN condition in Query 5 when used in combination with the index on `AccountBalance`.

Simple index with:
• Indexed field on `Address.Zip`
• Included fields on:
  • `Id`
  • `Email`
	• Optimizes the equality condition on `Address.Zip` in Query 4.
	• Adding the included fields avoids reading the JSON table in Query 4.

Composite index on: • `Gender` • `FullName.LastName`	• Optimizes equality condition on `Gender` and pattern matching condition on `FullName.LastName` for Query 6. • Specifying `Gender` as the leading key in combination with `FullName.LastName` results in more selective index lookups for Query 6.

last-updated: Apr 11, 2022

Index	Rationale
Simple index on `Hobbies[]`	Optimizes the equality condition on array elements of `Hobbies` in Query 7: `{"$eq":{"Hobbies[]":"Reading"}}`
Composite index on: • `Phones[].Type` • `Phones[].Number`	Optimizes the following two conditions in Query 8: • Equality condition on the `Type` subfield in nested documents in the `Phones` array. • Pattern matching condition on the `Number` subfield in nested documents in the `Phones` array.

Example with Multiple Container Field Paths

The following example references documents that store the high and low temperatures for each day in a week. They use an array to store the data, where each element in the array corresponds to a day of the week. For each day of the week, there is a two-element array of nested documents. The nested documents indicate whether the temperature corresponds to the high or low for that day. Typically, the outermost array has seven elements, one for each day of the week. But in cases where data is unavailable, the document has only the available days.

```
{
   "_id": "001",
   "temps": [{"hiLo": [{"type": "hi", "temp": 61}, {"type": "lo", "temp":
49}], "dow": "Sun"},
            {"hiLo": [{"type": "hi", "temp": 74}, {"type": "lo", "temp":
51}], "dow": "Mon"},
            {"hiLo": [{"type": "hi", "temp": 75}, {"type": "lo", "temp":
51}], "dow": "Tue"},
            {"hiLo": [{"type": "hi", "temp": 74}, {"type": "lo", "temp":
52}], "dow": "Wed"},
            {"hiLo": [{"type": "hi", "temp": 78}, {"type": "lo", "temp":
54}], "dow": "Thu"},
            {"hiLo": [{"type": "hi", "temp": 75}, {"type": "lo", "temp":
53}], "dow": "Fri"},
            {"hiLo": [{"type": "hi", "temp": 75}, {"type": "lo", "temp":
54}], "dow": "Sat"}],
   "weekOf": "4/29/2018"
}
{
   "_id": "002",
   "temps": {"hiLo": [{"type": "hi","temp": 81}, {"type": "lo", "temp":
60}], "dow":"Sat"},
   "weekOf": "5/12/2018"
}
{
   "_id": "003",
   "temps": [{"hiLo": [{"type": "hi", "temp": 80}, {"type": "lo", "temp":
55}], "dow": "Sun"},
            {"hiLo": [{"type": "hi", "temp": 78}, {"type": "lo", "temp":
54}], "dow": "Mon"},
            {"hiLo": [{"type": "hi", "temp": 79}, {"type": "lo", "temp":
54}], "dow": "Tue"},
            {"hiLo": [{"type": "hi", "temp": 77}, {"type": "lo", "temp":
53}], "dow": "Wed"},
            {"hiLo": [{"type": "hi", "temp": 79}, {"type": "lo", "temp":
54}], "dow": "Thu"},
            {"hiLo": [{"type": "hi", "temp": 77}, {"type": "lo", "temp":
```

```
54}], "dow": "Fri"},
             {"hiLo": [{"type": "hi", "temp": 78}, {"type": "lo", "temp":
54}], "dow": "Sat"}],
   "weekOf": "5/13/2018"
}
```

Suppose you frequently run the following queries:

Query	Description	Documents Returned
`find /apps/hiLoTemps --f weekOf --c '{"$eq": {"temps[].hiLo[].temp":60}}'`	Find weeks where any day has either a high or low temperature of 60.	002
`find /apps/hiLoTemps --f weekOf,temps[].hiLo[].type,temps[].hiLo[].temp --c '{ "$elementAnd":{ "temps[].hiLo[]":[ {"$eq": {"type":"hi"}}, {"$ge":{"temp":80}} ] } }'`	Find weeks and the high/low temperatures for all days on those weeks where any day of the week has a high temperature of at least 80.	002, 003

To optimize these queries, you should define an index with the following fields:

- Indexed fields: `temps[].hiLo[].temp`, `temps[].hiLo[].type`
- Included fields: `weekOf`, `temps[].hiLo`

By defining the composite index with `temps[].hiLo[].temp` as the first indexed field, the index can optimize both queries.

By adding `weekOf` as an included field, the index is a covering index for the first query. By adding `temps[].hiLo`, the index becomes a covering index for the second query as well. Note that you must add this included field even though the sub-fields are also indexed fields. This is due to how indexes with container field paths store data. For more details, see Covering Indexes and Container Field Paths on page 560.

Change Data Capture

The Change Data Capture (CDC) system allows you to capture changes made to data records in MapR Database tables (JSON or binary) and propagate them to a MapR Event Store For Apache Kafka topic.

These data changes are the result of inserts, updates, and deletions and are called change data records. Once the change data records are propagated to a topic, a MapR Event Store For Apache Kafka/Kafka consumer application is used to read and process them.

Note: The order of the records in the topic-partition is the same as the order of the changes made to the table. The order is retained because change data records for the same key are propagated to the same topic-partition.

last-updated: Apr 11, 2022

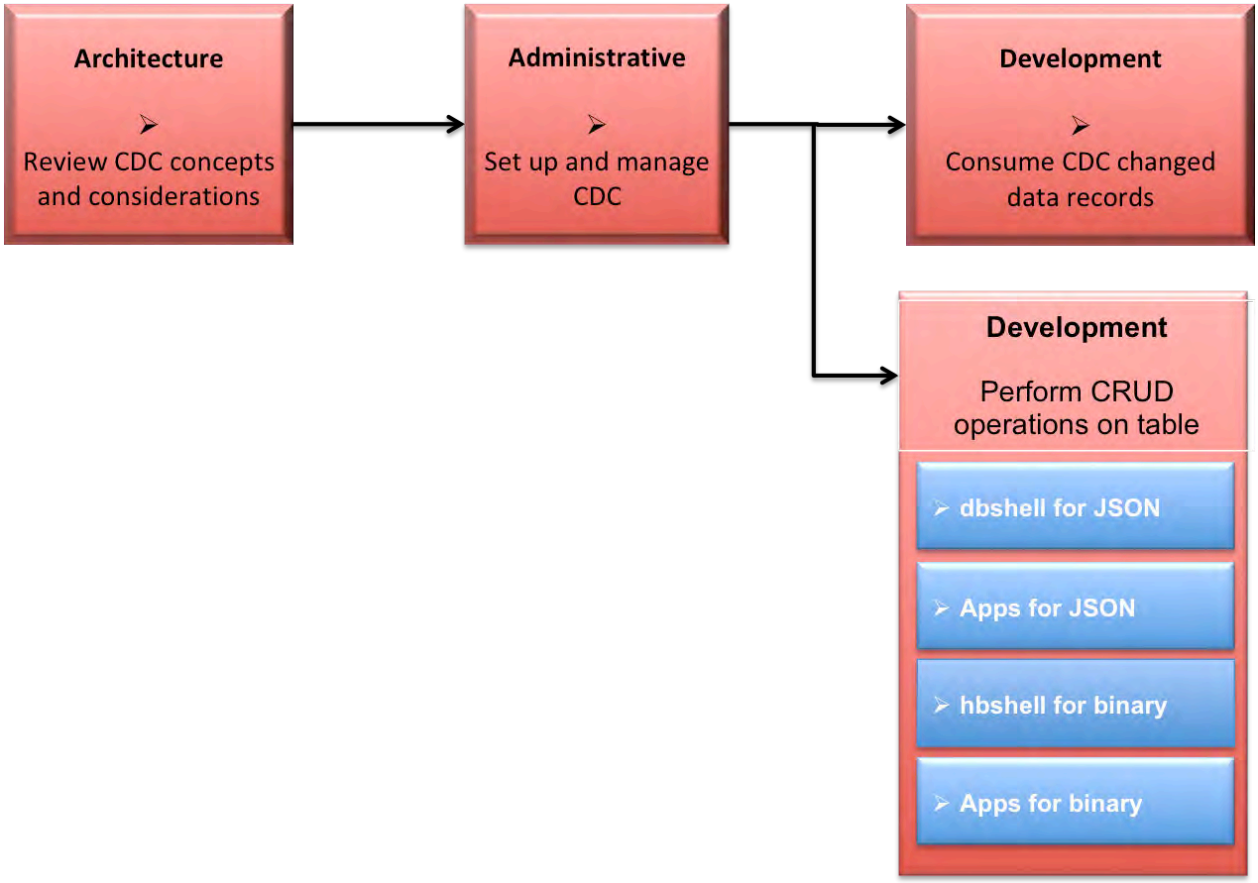
Why Use Change Data Capture?

CDC can be used in many ways, including the following:

- To track changes occurring in a MapR Database table and perform real-time processing on the data.
- To keep caches for search indexes (such as Elastic Search, Solr), materialized views, synchronization between data warehouses or data marts with data stored in MapR Database in real time.
- To manage separate MapR Database instances for transactional and reporting purposes and to keep them in sync in real time for real time analytics.
- To provide arbitrary external systems the ability to globally consume MapR Database table changes.

How Do I Get Started?

The following topics provide information you need to understand the CDC feature, to setup and use CDC, the maprcli commands used to perform tasks, and to consume the data via your application.



1. Learning about CDC
2. Administering Change Data Capture
3. Consuming CDC changed data records
4. Using dbshell to perform CRUD operations on MapR Database JSON tables
5. Developing client applications for MapR Database JSON tables.
6. Using hbshell to perform CRUD operations on MapR Database binary tables.

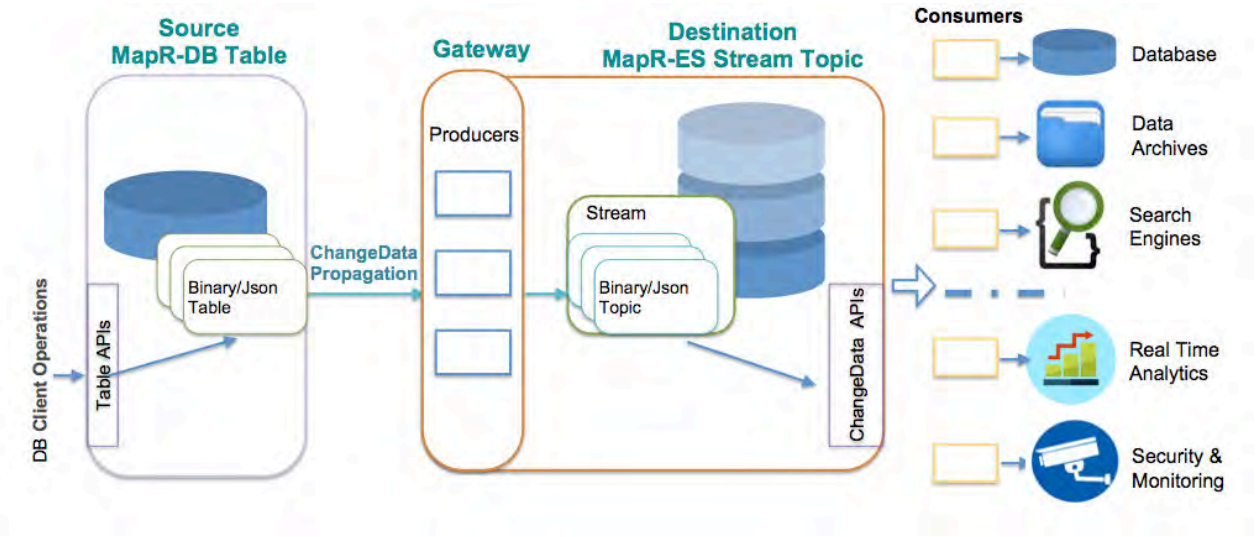
last-updated: Apr 11, 2022

7. Developing client applications for MapR Database binary tables.

Architecture and CDC

This section provides an overview of how CDC works.

CDC uses a log-based data capture for the changed data records, propagates the data (from the source table) using replication remote procedural calls (RPCs) through an internal MapR gateway and produces the data to a MapR Event Store For Apache Kafka destination stream topic(s). Once data is received by the topic, the changed data records can be consumed by external applications. The consumer application registers the CDC Deserializer as its record value deserializer and pulls the topic data by using a Kafka API. The data changes can be read from the ChangeDataRecord through the OJAI ChangeData APIs. Consumers could be databases, data archives, search engines, or applications that perform real-time analytics, security, or monitoring.



How are the Change Data Records Propagated?

The propagation is accomplished by setting up a change log that establishes a relationship between the source table and the destination stream. The change log can be setup by using the Control System, maprcli, or REST. Each change log can be paused, resumed, and removed. See Administering Change Data Capture on page 1106 and the `maprcli` table changelog on page 1813 command for more information.

As data is changed on the source table (through CRUD operations), each changed data record is propagated (replicated) to an internal MapR gateway. The order of when the data is produced to the stream topic is the same order of when the changed data records are replicated to the gateway. The data flow is one way, meaning, the flow is from a MapR Database source table to a MapR Event Store For Apache Kafka destination stream topic(s).

Note: When an array value is updated, the changed data record is the full array record rather the specific data change.

What is the Impact of using Columns/Column Families?

When propagating a specific column family or column from a binary source table and a row is deleted, the destination stream topic shows only a deletion event for the specific column family or column. When propagating a specific column from a binary source table with its entire column family deleted, the destination stream topic shows only a deletion event for the specific column.

In the scenario where you have a binary source table with fam0, fam1, and fam2 and you set up the change log *without* columns or column families:

last-updated: Apr 11, 2022

- If you delete fam0, fam1, and fam2, the change data event will be "delete fam0", "delete fam1" and "delete fam2".
- If you delete the row, the change data event will be "delete row".

In the scenario where you have a binary source table with fam0, fam1, and fam2 and you set up the change log *with* a column setup as fam1:col1, fam2.

- If you delete fam0, fam1, and fam2, the change data event will be "delete fam1:col1", "delete fam2".
- If you delete the row, the change data event will be "delete fam1:col1", "delete fam2".

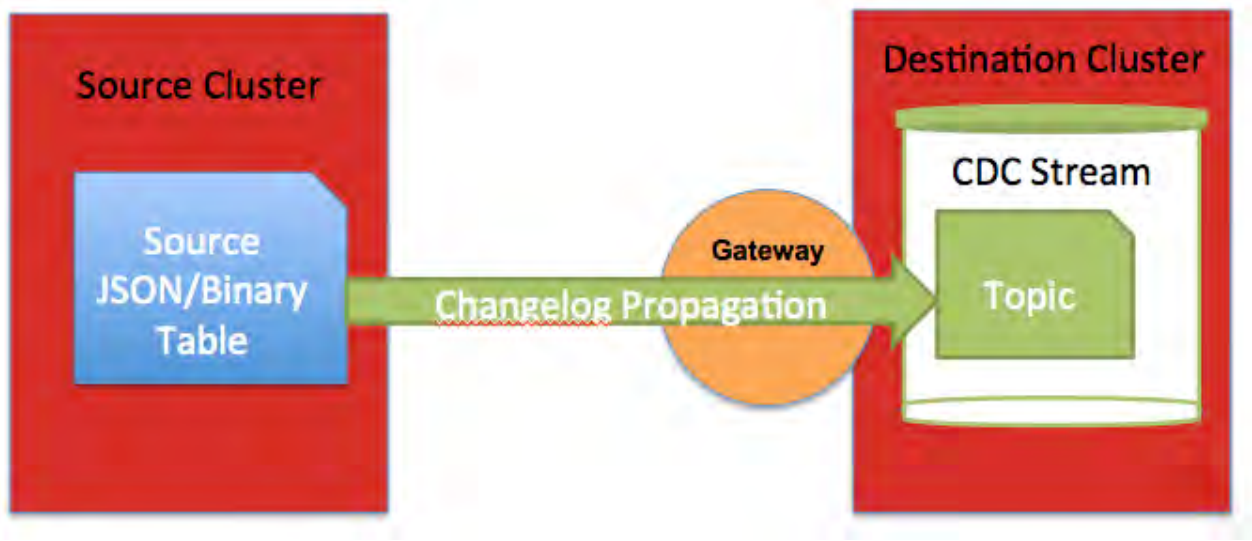
Where is the Destination Stream Setup?

The destination MapR Event Store For Apache Kafka stream can either be on the same cluster as the MapR Database source table or on a remote MapR cluster. Where and how destination streams are setup depends on the purpose for using CDC.

If you are propagating changed data from a source table on a source cluster to a destination stream topic on a remote destination cluster, you must setup a gateway. Gateways are setup by installing the gateway on the destination cluster and specifying the gateway node(s) on the source cluster. See Administering MapR Gateways on page 1150 and Configuring Gateways for Table and Stream Replication on page 1152.

The following diagram shows a simple CDC data model, with one source table to one destination topic on one stream. Since this scenario has the destination stream topic on a remote destination cluster, you must setup and configure a gateway.

Note: More complex CDC scenarios can be implemented and multiple gateways can be setup.



Important: If you have a secure cluster, you must setup secure configuration. See Configuring Secure Clusters for Cross-Cluster Mirroring and Replication on page 1486.

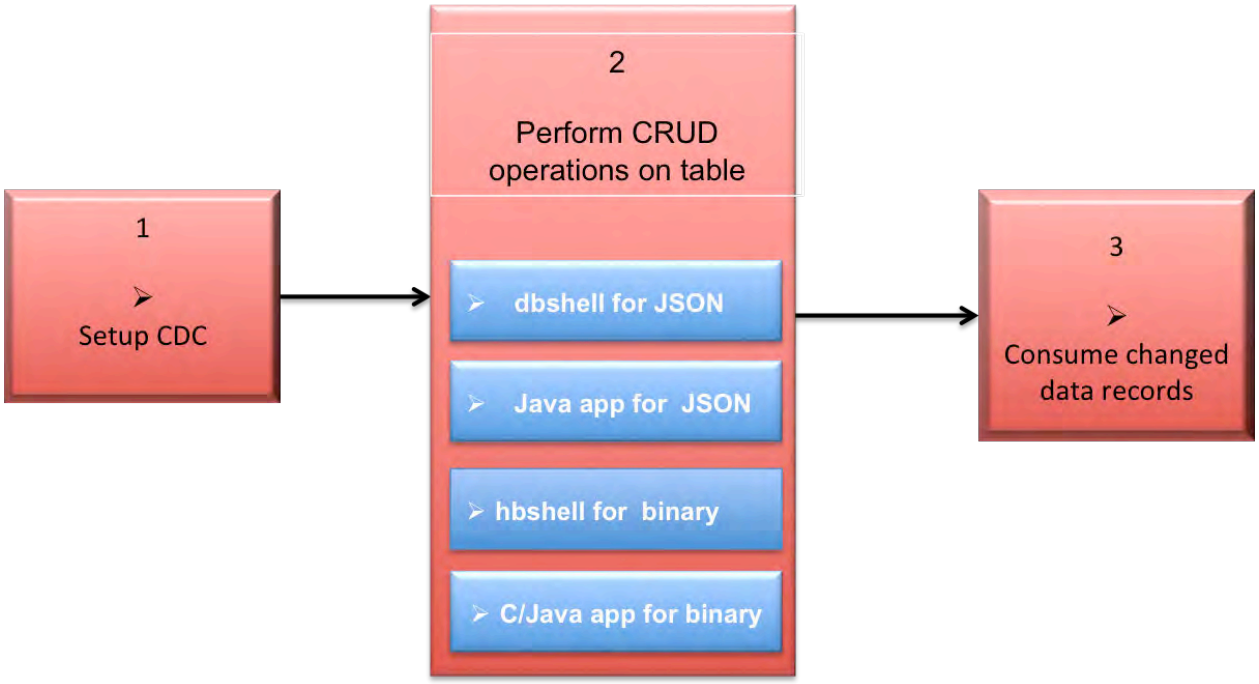
Getting Started with CDC

Describes an end-to-end flow of how to establish and use Change Data Capture (CDC). It assumes that a new table and dataset will be created, although an existing table with data can also be used.

End-to-End Workflow

The following diagram shows an end-to-end workflow of the Change Data Capture (CDC) feature.

Note: Steps 2 and 3 are interchangable. You may decide to start the consumer application for CDC changed data records *before* performing CRUD operations on the table.



1. Setup the CDC Environment
2. Using dbshell to perform CRUD operations on MapR Database JSON tables
3. Developing client applications for MapR Database JSON tables.
4. Using HBase to perform CRUD operations on MapR Database binary tables.
5. Developing client applications for MapR Database binary tables.
6. Consuming CDC changed data records

1. Setup the CDC environment.
   a. If you are propagating changed data from a source table on a source cluster to a destination stream topic on a *remote* destination cluster, you must setup a gateway. Gateways are setup by installing the gateway on the destination cluster and specifying the gateway node(s) on the source cluster. See Administering MapR Gateways on page 1150 and Configuring Gateways for Table and Stream Replication on page 1152.

   b. If you have a secure cluster, you must set up secure configuration. See Configuring Secure Clusters for Cross-Cluster Mirroring and Replication on page 1486.

   c. Establish a MapR Database table (JSON or binary) with data. You can create a new table and add data, or use an existing table with data. See maprcli table create for creating a new table or use the Control System. If you are using an existing table with data, skip to the next step.

   d. Create a MapR Event Store For Apache Kafka stream for the propagated changed data records using the `maprcli stream create -ischangelog` parameter. See maprcli stream create or use the Control System.

last-updated: Apr 11, 2022

e. Create a MapR Event Store For Apache Kafka stream topic for the changed data records. You can use the maprcli stream topic create command, the maprcli table changelog add command (this command creates a changelog relationsip between the source table and the destination stream topic), or the Control System when creating either a stream topic or a table changelog.

f. Create a changelog relationship between the source table and the destination stream topic with the maprcli table changelog add command or use the Control System. By creating a changelog relationship, you are creating an environment that propagates changed data records from a source table to a MapR Event Store For Apache Kafka topic.

Note: Propagation of existing table data is enabled by default. If you do *not* want to propagate existing source table data, set the `-propagateexistingdata` parameter to **false**. The default is `true`.

Note: Propagation is enabled as soon as you add the table changelog relationship. If you do *not* want propagation to begin, set the `-pause` parameter to **true**. The change data records are stored in a bucket until you resume the changelog relationship; at this point, the stored change data records are propagated to the stream topic. See table changelog resume on page 1820 for more information.

g. Verify that the changelog exists. See table changelog list on page 1816 for information about your changelogs.

2. Perform CRUD operations (inserts, updates, and deletes) on the source table. The following utility and application can be used:

- mapr dbshell for MapR Database JSON documents
- hbshell for MapR Database binary data
- Java applications for MapR Database JSON
- C or Java applications for MapR Database binary data

3. Write a consumer with the Apache Kafka and OJAI API libraries that subscribes to the topic and consumes the change data records. There are multiple interfaces that are used for writing a CDC consumer. See Consuming CDC Records on page 2723 for a list of interfaces. See Building Consumers for CDC on page 2726 for an example.

Use Cases

Scenario	Setup Task	Notes
You want a CDC stream topic to contain all of the table data as changed data records.	You would setup CDC in the following manner before performing operations on the source table and consuming the change data records. **1.** Create an **empty** source table. **2.** Create the changelog stream. **3.** Create the changelog stream topic. **4.** Add the table changelog relationship. In this case, it does not matter if the `-propagateexistingdata` is set to true or false because you are starting with an empty source table. **5.** Verify that the changelog exists and that `replicaState` is REPLICA_STATE_REPLICATING. See table changelog list on page 1816 for more information.	In this case, all table data is propagated to the stream topic as change data records and the operation type is identified on each individual data record.
You want a CDC stream topic to contain all of the existing table data and changed data records.	You would setup CDC in the following manner before performing operations on the source table and consuming the change data records. **1.** Create a source table and add **data**, or alternatively, use an existing table that contains data. **2.** Create the changelog stream. **3.** Create the changelog stream topic. **4.** Add the table changelog relationship. Be sure that the `-propagateexistingdata` parameter is set to **true**. If you are using the command line to add the changelog, then you do not need to specify this parameter because the default is `true`. **5.** Verify that the changelog exists and no error is reported in the changelog list. When all the existing data in the table is delivered to the changelog, the replicaState becomes REPLICA_STATE_REPLICATING. See table changelog list on page 1816 for more information.	In this case, the existing table data is propagated to the stream topic and that data's operation type is identified as a SET operation. Subsequently, operations on the source table are propagated as changed data records and the operation type is identified on each individual data record. You can consume data at any time, however, there may be a delay before all of the existing table data is completely propagated, expecially if you have a large dataset. Be sure to check the `copyTableCompletionPercentage` field.

Scenario	Setup Task	Notes
You want a CDC stream topic to *not* contain any original table data and to capture only subsequent changed data records	You would setup CDC in the following manner before performing operations on the source table and consuming the change data records. **1.** Create a source table and **add** data, or alternatively, use an existing table that contains data. **2.** Create the changelog stream. **3.** Create the changelog stream topic. **4.** Add the table changelog relationship. Be sure that the `-propagateexistingdata` parameter is set to **false**. **5.** All new data operations applied to a source table after the replicaState becomes REPLICA_STATE_REPLICATING is *not* treated as original data and is delivered to the changelog. See table changelog list on page 1816 for more information.	In this case, the existing table data is not propagated to the stream topic and the operation type is identified on each individual data record.

Data Modeling and CDC

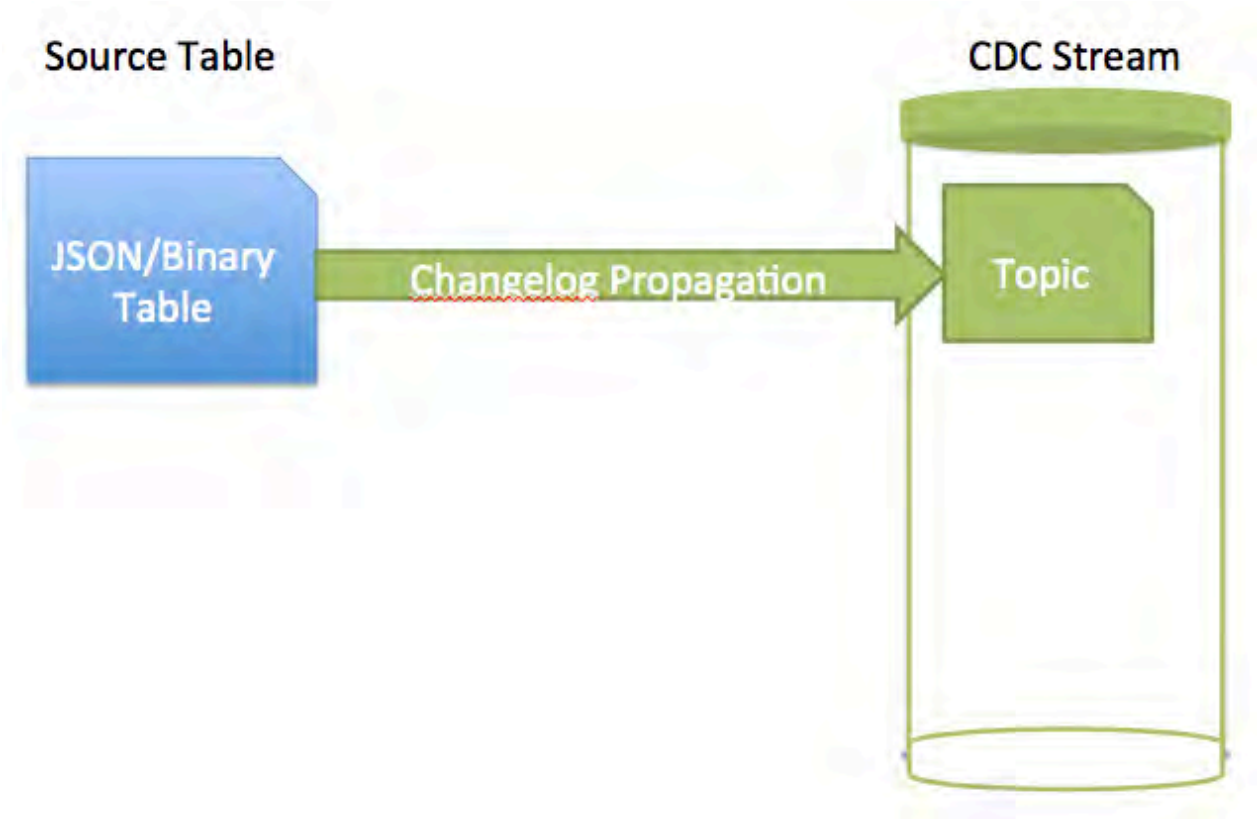
Change Data Capture (CDC) changed data records propagate in one direction - from a source table to a topic in a changelog stream. One stream with one topic can be created for the changed data records or multiple streams with multiple topics can be created.

Note: Propagation from multiple source tables to one stream topic is not supported.

One source to one destination topic on one stream

You might use this scenario if there are a large number of changed data records being propagated, and you want the topic on a separate or isolated volume, so that resources are dedicated to these particular changed data records.

The following graphic shows a source table's change data records being propagated to one topic on one stream.



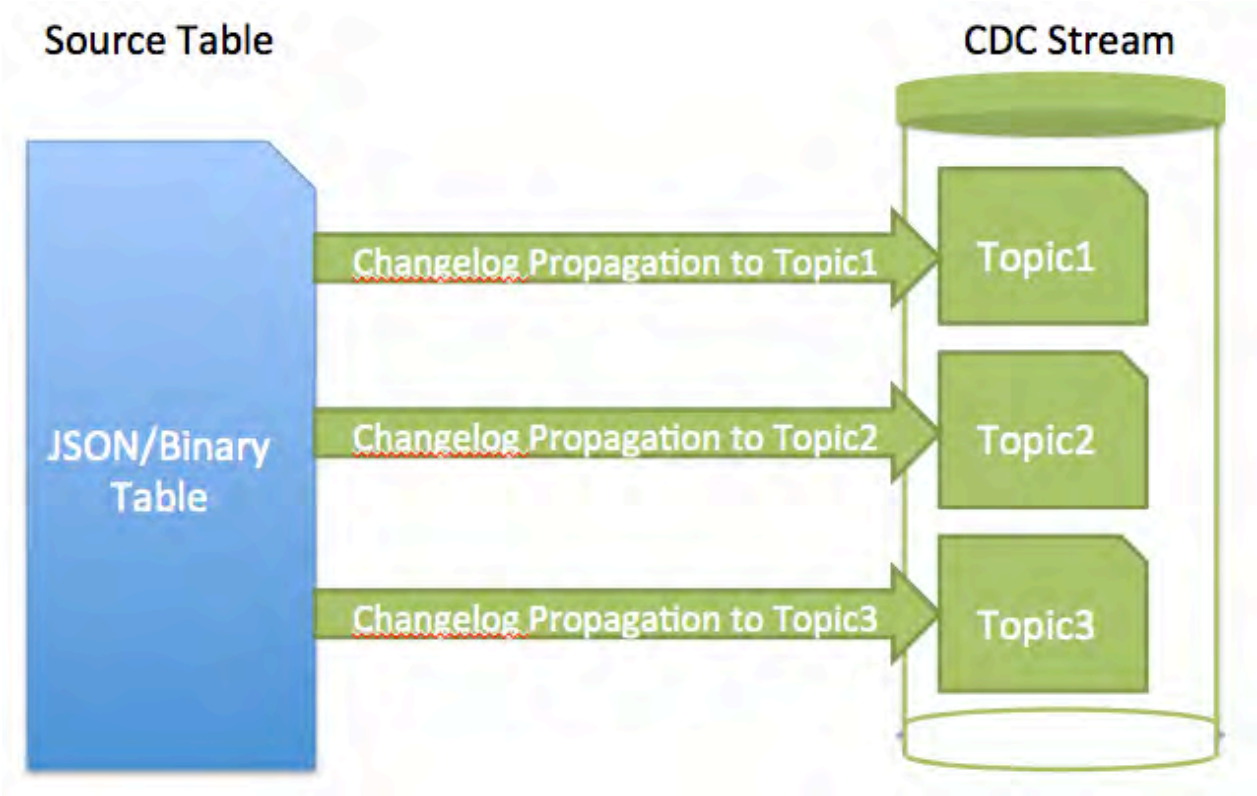
One source to multiple destination topics on one stream

You might use this scenario if you want to propagate specific changed data records from one source table to different topics.

When you set up a table changelog for data propagation, you can specify the column parameter to propagate a specific field or column family. Default: All fields are propagated. See table changelog add on page 1813 for information about adding a table changelog.

The following graphic shows a source table's change data records being propagated to multiple topics on a stream.

last-updated: Apr 11, 2022

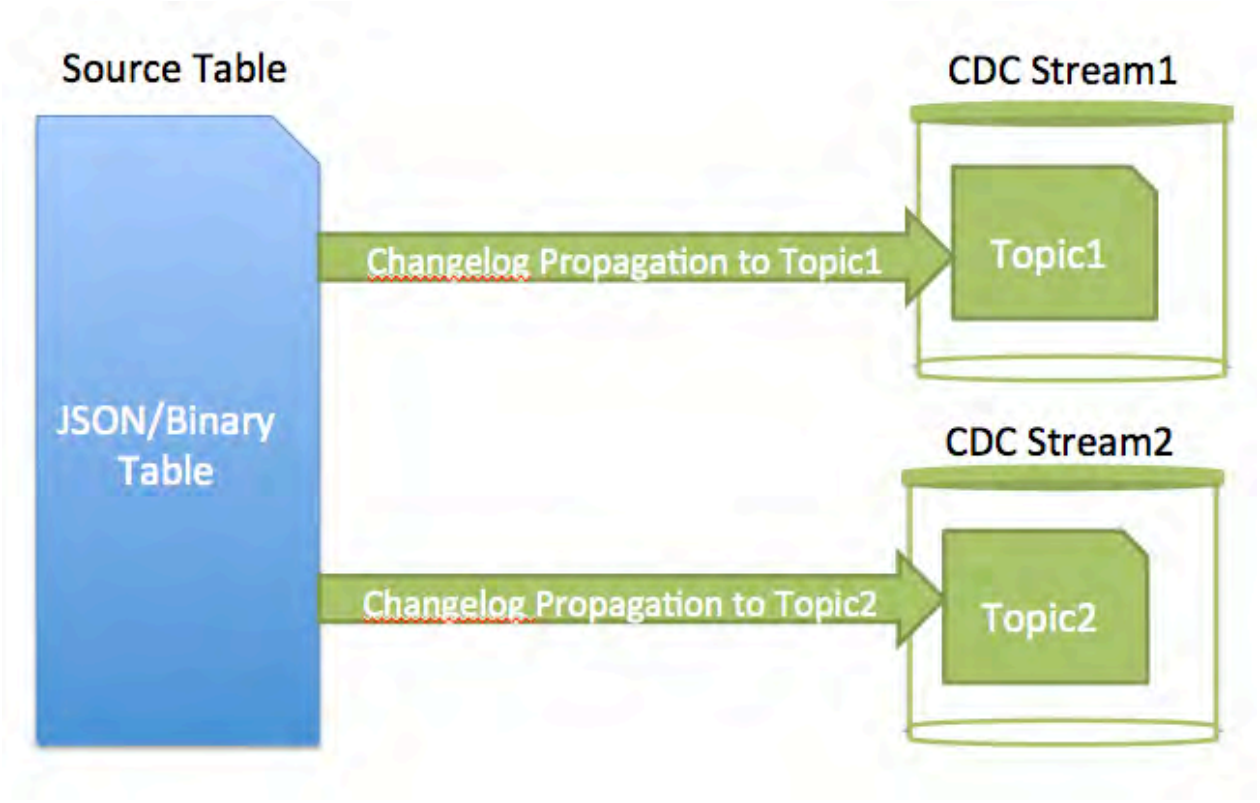


One source to multiple destination topics on multiple streams

You might use this scenario if the change data records are important and you want to have an extra copy for backup purposes.

The following graphic shows a source table's change data records being propagated to topics on multiple streams.

last-updated: Apr 11, 2022

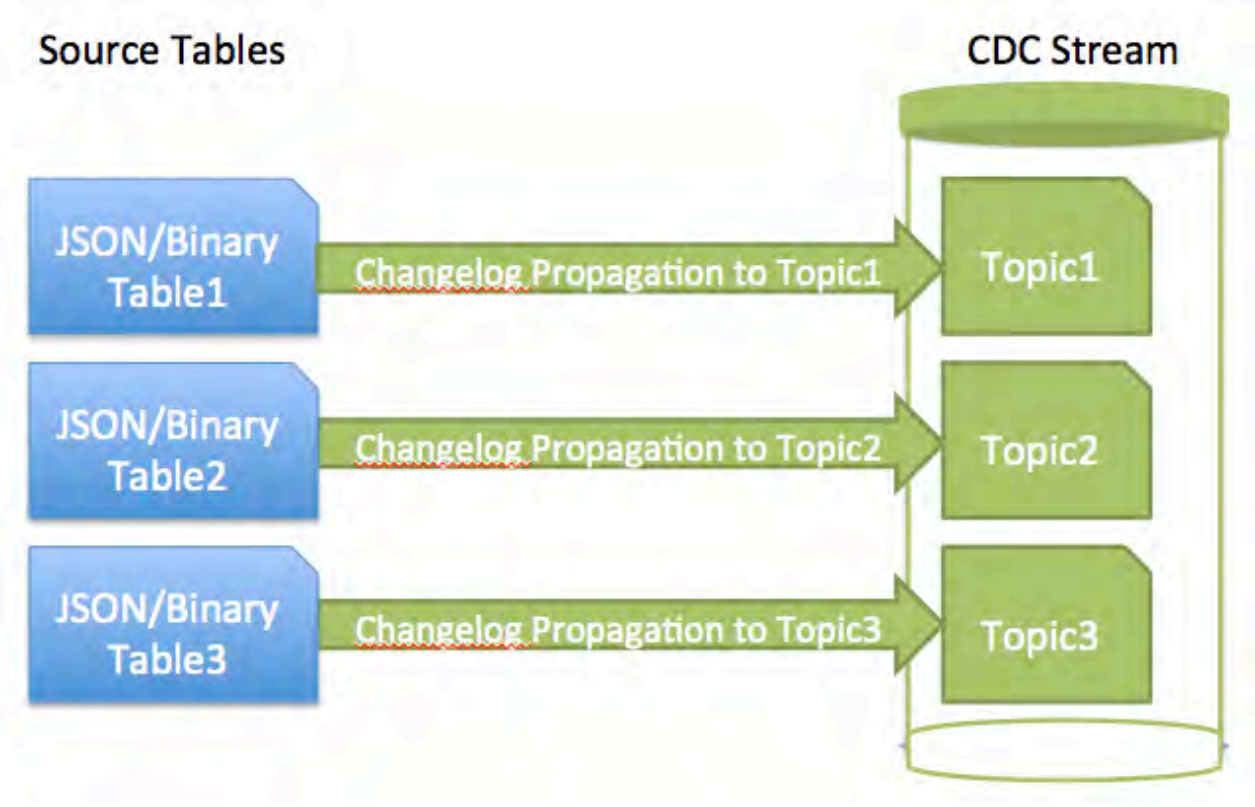


Multiple sources to multiple destination topics on one stream

You might use this scenario if you want to set up permissions to one stream so that a team has access to all the topics that they want to access. For example, if table1 and table2 has change data records that a team wants to monitor, then on the stream, you would grant permission to the monitoring team.

The following graphic shows three source tables' change data records being propagated to three topics on the same stream.

last-updated: Apr 11, 2022

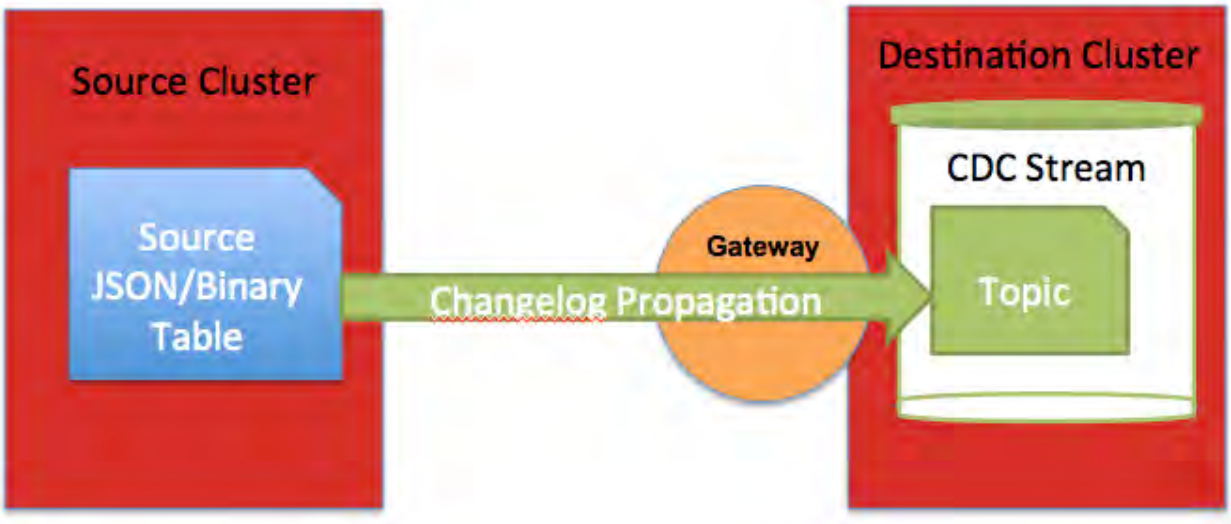


Source Cluster to Destination Cluster

If you are propagating changed data from a source table on a source cluster to a destination stream topic on a remote destination cluster, you must setup a gateway. Gateways are setup by installing the gateway on the destination cluster and specifying the gateway node(s) on the source cluster. See Administering MapR Gateways on page 1150 and Configuring Gateways for Table and Stream Replication on page 1152.

The following diagram shows a simple CDC data model, with one source table to one destination topic on one stream. Since this scenario has the destination stream topic on a remote destination cluster, you must setup and configure a gateway.

Note: More complex CDC scenarios can be implemented, and multiple gateways can be setup.



last-updated: Apr 11, 2022

Important: If you have a secure cluster, you must setup secure configuration. See Configuring Secure Clusters for Cross-Cluster Mirroring and Replication on page 1486.

Security and CDC
Security for CDC is applied through Access Control Expressions (ACEs). In addition, if a secure cluster configuration is implemented, then additional setup may be needed depending on the configuration.

Access Control Expressions (ACEs)

Since Change Data Capture (CDC) changed data records are propagated from a MapR Database source table to a MapR Event Store For Apache Kafka stream topic, use the access control expressions (ACEs) on the source table and destination stream for establishing permissions.

Once a MapR Event Store For Apache Kafka stream is created for purposes of receiving change data records, it is dedicated for that sole purpose. For example, a producer application should not perform CRUD operations on the topics in the stream.

The following permissions are applicable depending on the scenario:

- If you are a normal user and you want to create a changelog from a source table and to a destination stream topic, the following permissions are required:
  - `replperm` on the source table in the source cluster
  - `topicperm` on the destination stream in the destination cluster
- If you are a normal user and want to create a changelog between your own MapR Database table and someone else's stream topic, you must be granted `topicperm` permissions on the destination stream.
- If you are a normal user and want to receive or read the data in a stream topic, you must be granted `consumeperm` permission on the destination topic.

For more information about ACEs, see Managing Access Control Expressions on page 1448

Secure Clusters

The destination MapR Event Store For Apache Kafka stream could be in same cluster as the MapR Database source table or it could be on a remote MapR cluster. The configuration setup depends on the purpose for using CDC.

- If your destination stream is on the same cluster as the source table and the cluster is secure, then additional configurations are *not* required.
- If your destination stream is on a remote secure cluster, then a gateway and secure configuration must be setup. See Table Replication on page 610, Administering MapR Gateways on page 1150, and Configuring Secure Clusters for Cross-Cluster Mirroring and Replication on page 1486

Restrictions for CDC
Lists the limitations for Change Data Capture.

The limitations for Change Data Capture are as follows:

- Non-CDC data cannot be propagated to changelog stream topics.
- Metadata or policy-driven operations are not propagated to changelog stream topics; only the changed data is propagated. For example, since column family and time-to-live (TTL) is metadata, they are not propagated to changelog stream topics. If metadata is changed in the source table, that information is unknown in relation to the destination stream topic data.
- Propagation of MapR Database JSON arrays is expensive because the full array is propagated to the changelog stream topic.

last-updated: Apr 11, 2022

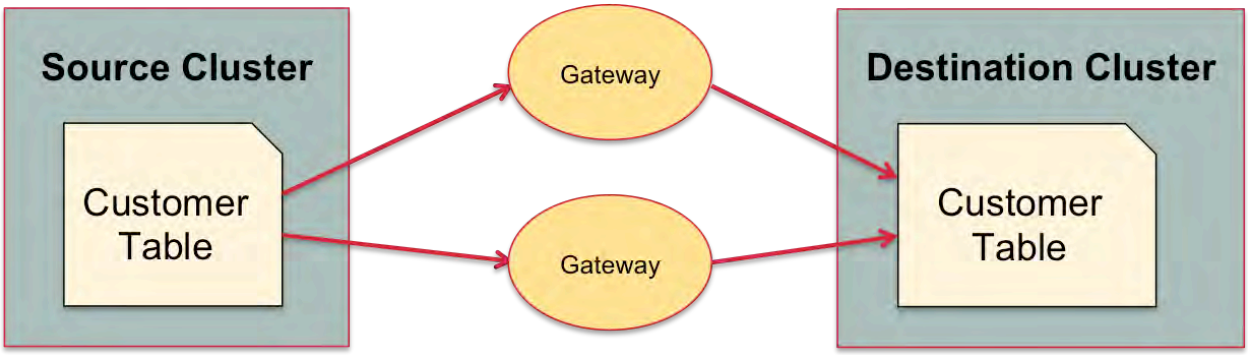
Table Replication

You can replicate data in one table to another table that is in the same cluster or in a separate cluster. This type of replication is in addition to the automatic replication that occurs with table regions within a volume.

You can replicate changes (puts and deletes), entire tables, specific column families, and specific columns.

MapR binary tables can only be replicated to binary tables; MapR JSON tables can only be replicated to JSON tables.

- Tables from which data is replicated are called source tables. Tables to which the data is replicated are called replicas.
- Clusters from which data is replicated are called source clusters. Clusters to which data is replicated are called destination clusters. A single cluster can be both a source cluster and a destination cluster, depending on the replication configuration in which the cluster participates.
- Replication takes place between source and destination clusters. However, source clusters do not send data to nodes in the destination cluster directly. The replication stream (the data being pushed to the replicas) is consumed by one or more MapR gateways in the destination cluster. The gateways receive the updates from the source cluster, batch them, and apply them to the replica tables. Multiple gateways serve the purpose of both load balancing and failover.



For more information about gateways, see Administering MapR Gateways on page 1150.

The maximum number of replicas that a source table can replicate to is 64. The maximum number of source tables from which a replica can accept updates is 64.

Modes of replication

Describes the asynchronous and synchronous modes of table replication.

You can replicate table data in one of two replication modes. You specify the mode per source-replica pair.

Asynchronous replication

In this replication mode, MapR Database confirms to client applications that operations are complete after the operations are performed on source tables. Updates are replicated in the background. Therefore, the latency of updates from client applications is not affected by the time required for the network round trip between the source cluster and the destination cluster.

This type of replication is well-suited for clusters that are geographically separated in wide-area networks.

MapR Database can throttle the replication stream to minimize the impact of the replication process on incoming operations during periods of heavy load. Throttling distributes disk reads and CPU usage more evenly over time, so that incoming operations on a source table can be completed faster. Throttling is disabled by default.

Asynchronous replication is the default replication mode.

Synchronous replication

In this replication mode, MapR Database confirms to client applications that changes have been applied to a source table only when these two conditions are true:

- The change was sent to all of the container copies in the local cluster.
- The change was sent to a gateway in the destination cluster. This operation takes place only after the first. Puts are not sent to gateways until after they are sent to all container copies in the cluster where the source table is located.

If a gateway fails, the source detects this and resends operations to the gateway when it is restarted or a new gateway is brought online.

Due to the confirmations that MapR Database receives on source clusters, synchronous replication is especially well-suited for creating a backup of your data for disaster recovery.

When the latency of a replication stream is high, MapR Database switches to asynchronous replication temporarily so that client applications are not blocked indefinitely. After the latency is sufficiently reduced, MapR Database switches back to synchronous replication. The same switching occurs when a gateway fails, and MapR Database does not resume synchronous replication until a new gateway is established or the failed gateway is restarted.

Supported replication topologies
Lists the `primary-secondary` and `multi-master` replication technologies.

There are two types of basic topologies that you can use for your replication scenarios: `primary-secondary` replication, with which you can construct several different types of more complicated topologies, and `multi-master` replication.

Primary-Secondary Replication
In this topology, you replicate one way from source tables to replicas. The replicas can be in a remote cluster or in the cluster where the source tables are located.

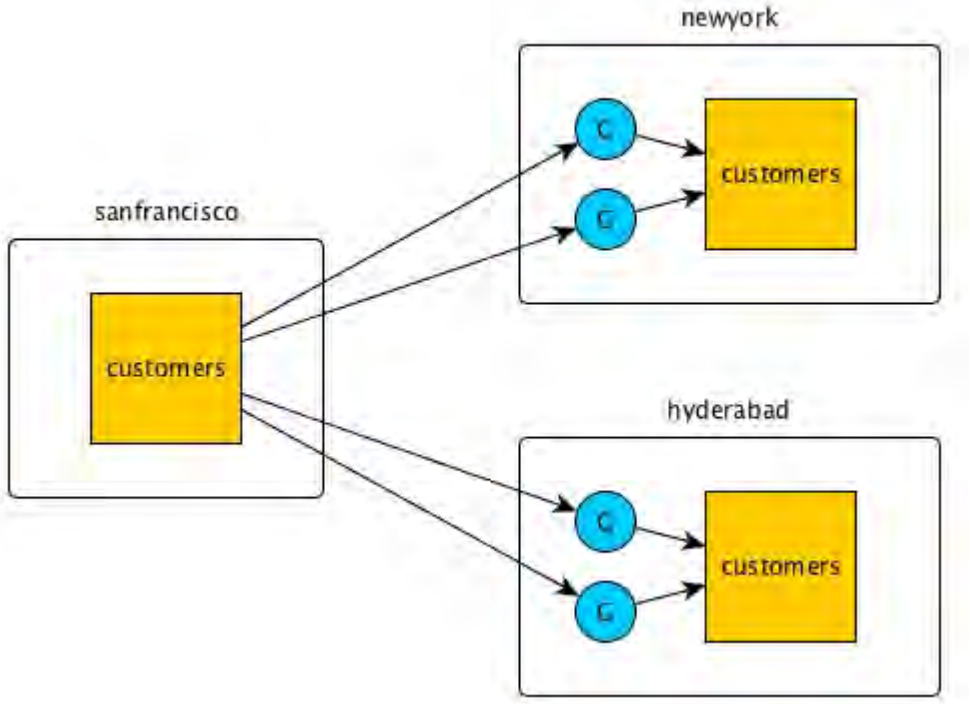
Several topologies are possible for primary-secondary replication:

Replication from one source table to one or more replica tables

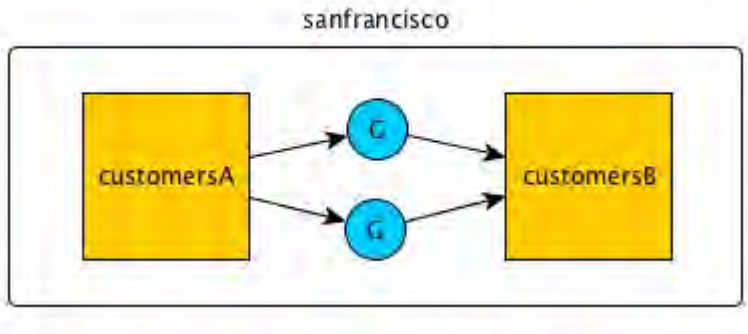
In this topology, updates on a source table are replicated to one or more replicas, but updates to the replicas are not replicated back to the source table.

For example, in this diagram, updates to the `customers` table in the cluster `sanfrancisco` are being replicated to the `newyork` and `hyderabad` clusters. The circles marked G each represent a MapR gateway.

However, changes to the table in the `newyork` and `hyderabad` clusters are not replicated back to the table in the `sanfrancisco` cluster.

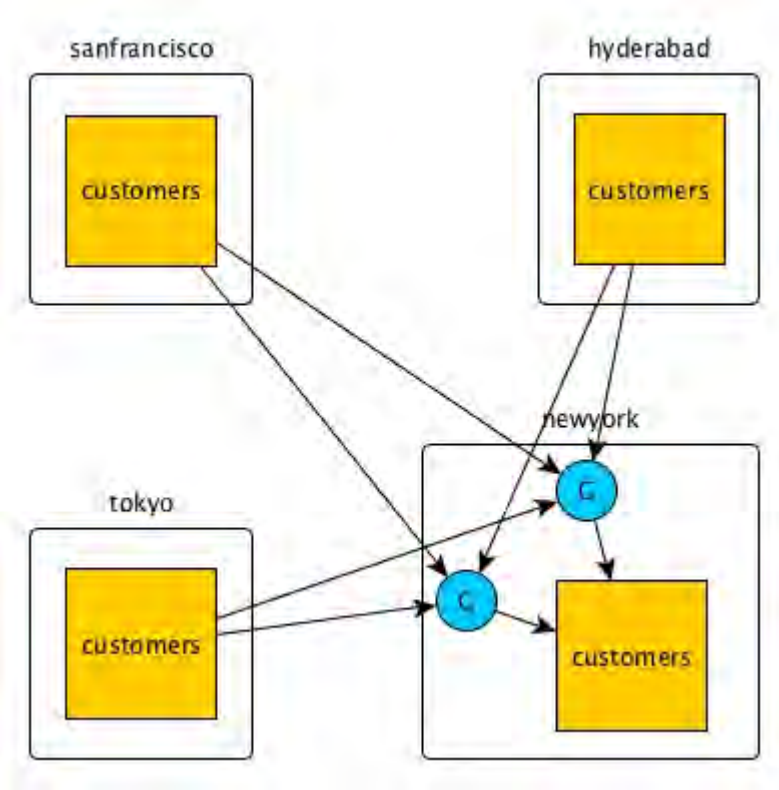


You can also replicate within a single cluster. In this example, the cluster `sanfrancisco` contains both the source table and the replica.



Many-to-one replication

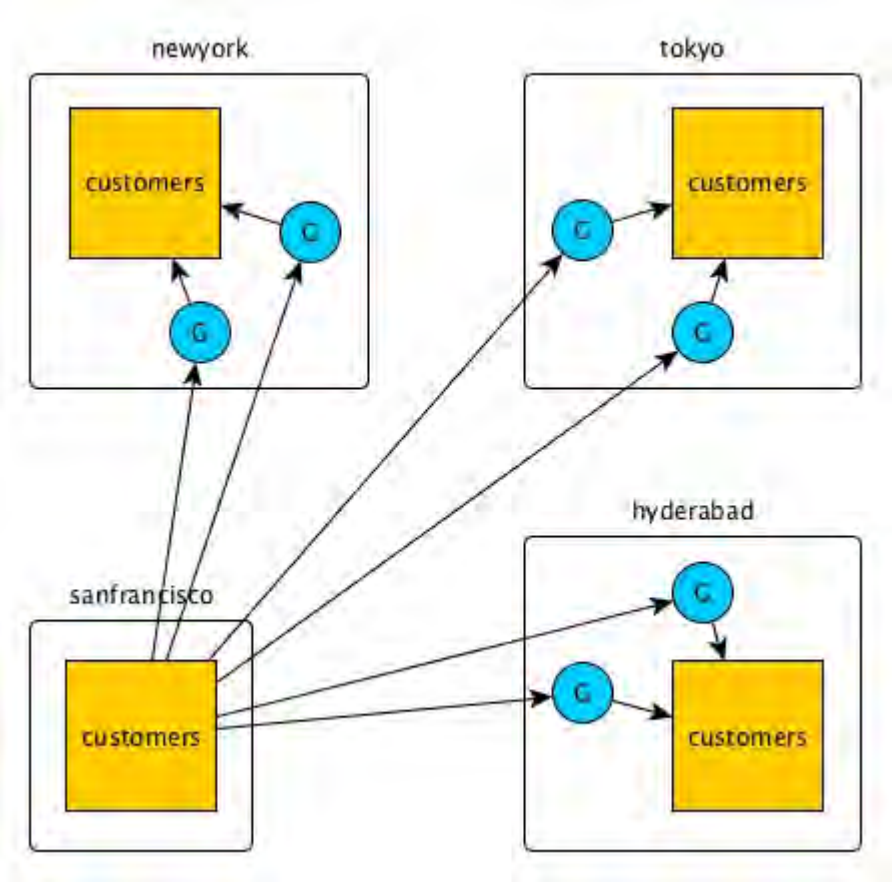
Multiple source tables can replicate to a single replica. In this diagram, operations on `customers` tables in three different clusters are replicated via gateways to the `customers` table in the `newyork` cluster.



One-to-many replication

A single source table can replicate to multiple replicas. In this diagram, operations on the `customers` table in the `sanfrancisco` cluster are replicated via gateways to replicas in three other clusters.

last-updated: Apr 11, 2022

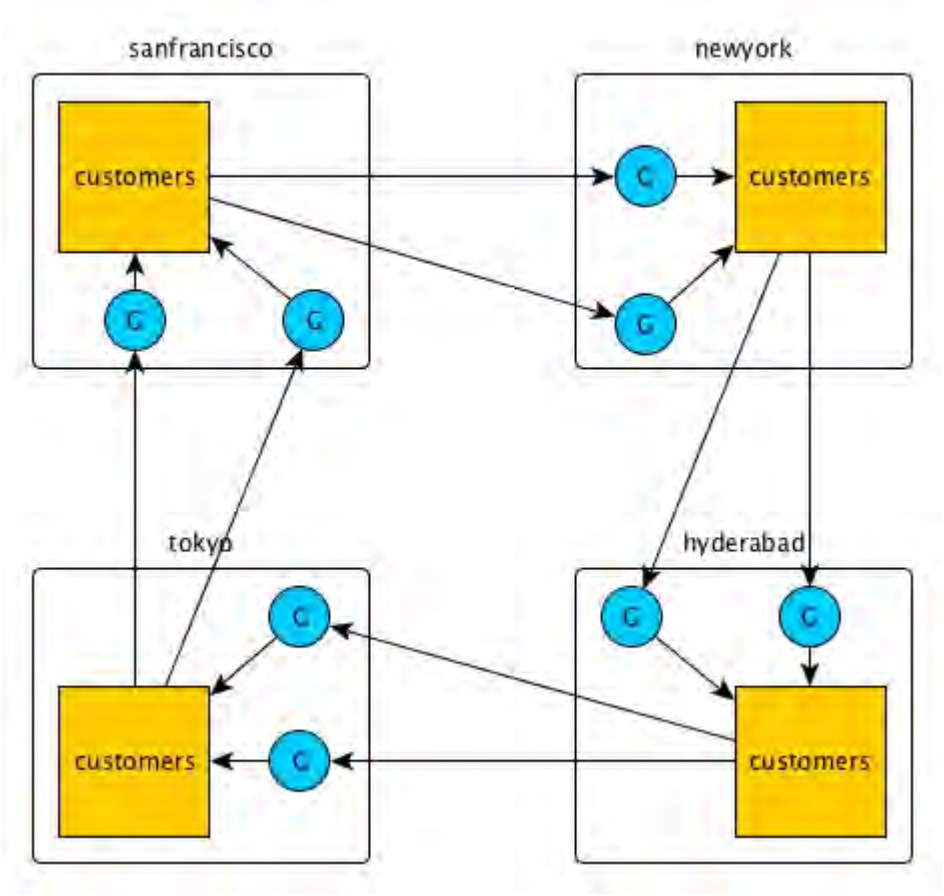


Replication loops

When three or more tables need to be kept in sync, you can set up primary-secondary replication between pairs of them to form a replication loop. Operations on a table are propagated to the other clusters in the loop, but there is no attempt to reapply the operations at the originating table. This is because the operations are tagged with a universally unique identifier (UUID) that identifies the table where the operations originated.

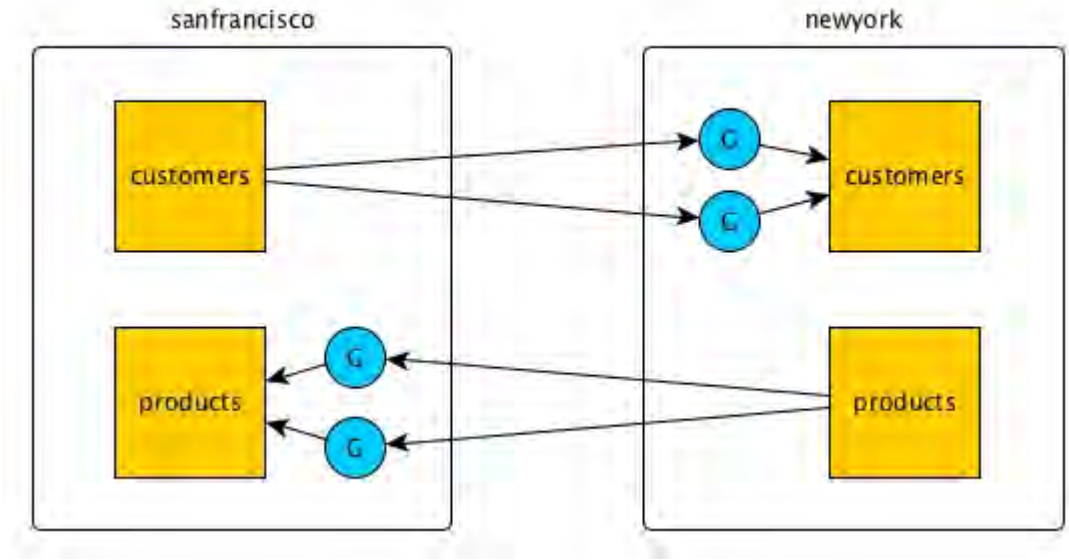
In this diagram, for example, operations on the `customers` table in the `hyderabad` cluster are replicated first to the `customers` table in the `tokyo` cluster. The operations are then replicated from the `tokyo` cluster to the `customers` table in the `sanfrancisco` cluster. Finally, the operations are replicated from the `sanfrancisco` cluster to the `customers` table in the `newyork` cluster. The `newyork` cluster does not replicate the operations to the `customers` table in the `hyderabad` cluster.

last-updated: Apr 11, 2022



Primary-Secondary replication in two directions

You can combine primary-secondary replication configurations to replicate data between clusters. Two clusters engaged in replication can each act as a source cluster and a destination cluster.

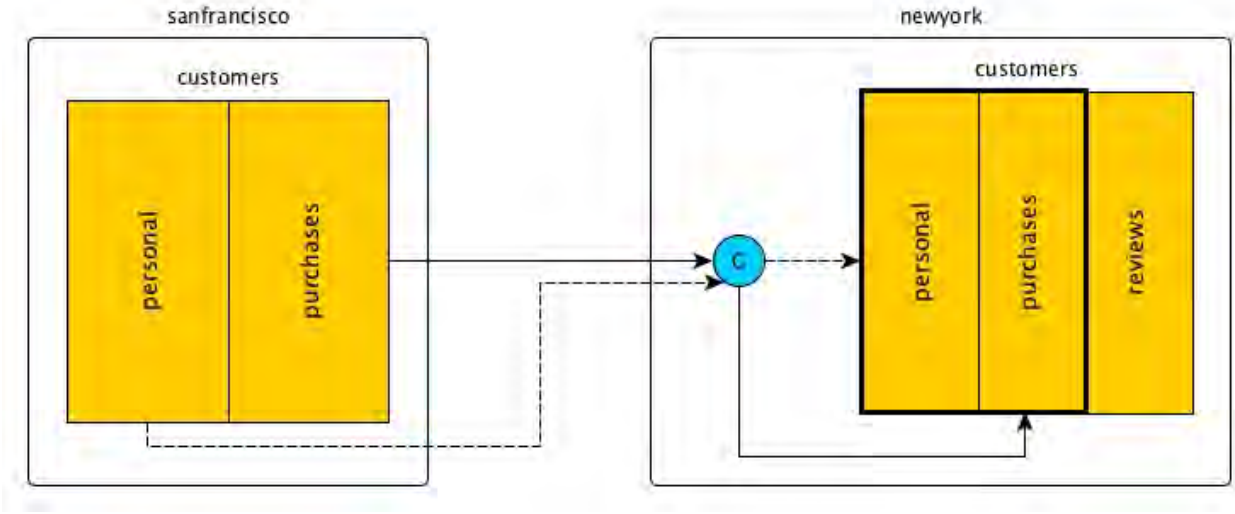


In this example, the data in the `customers` table in the cluster `sanfrancisco` is replicated to the `customers` table in the cluster `newyork`. At the same time, the data in the `products` table in the `newyork` cluster is replicated to the `products` table in the cluster `sanfrancisco`.

last-updated: Apr 11, 2022

In all primary-secondary configurations, changes made to replica tables are not replicated back to source tables. Therefore, if the replicated data is modified at the replica by client applications, the replica will become out of sync with the source table.

For example, you might replicate the two column families `personal` and `purchases` from the `customer` table in the `sanfrancisco` cluster to the `customers` table in the `newyork` cluster, as in this diagram. (For simplicity, the blue circle labeled G represents two or more gateways, rather than one as in the other diagrams in this topic.)



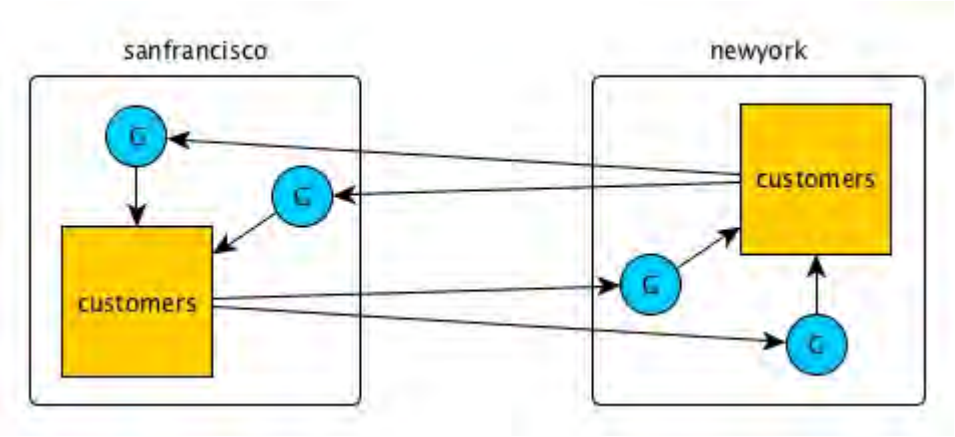
In primary-secondary replication, no updates to a replica are replicated back to the source. Any updates that applications might make to those two column families in the `customers` table in the `newyork` cluster will not be replicated to the `customers` table in the `sanfrancisco` cluster.

However, you do not have to protect a replica from all updates that are not due to replication. For example, the `customers` table in the `newyork` cluster might have an additional column family that is not populated with replicated data: `reviews`.

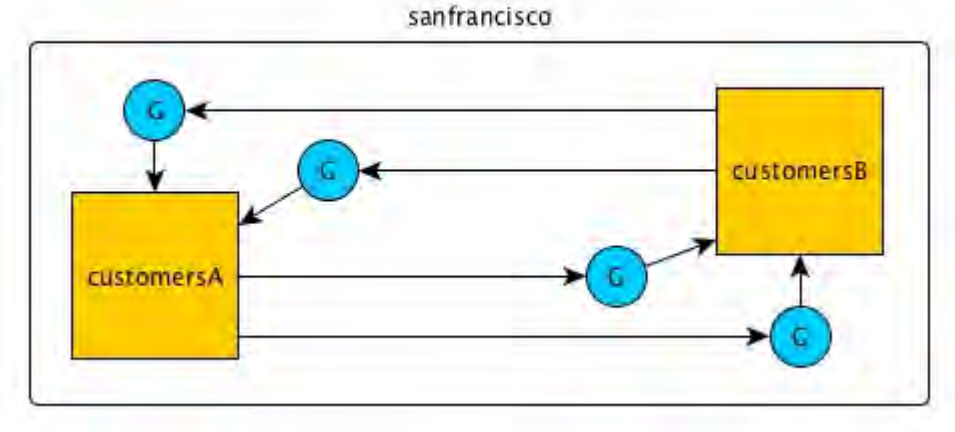
Multi-Master Replication
In this replication topology, there are two primary-secondary relationships, with each table playing both the primary and secondary roles. Client applications update both tables and each table replicates updates to the other.

All updates from a source table arrive at a replica after having been authenticated at a gateway. Therefore, access control expressions on the replica that control permissions for updates to column families and columns are irrelevant; gateways have the implicit authority to update replicas.

In this diagram, the `customers` table on the cluster `sanfrancisco` replicates updates to the `customers` table in the cluster `newyork`. The latter table in turn replicates updates to the former table. MapR Database tags each table operation with the universally unique ID (UUID) that it has assigned the table at which the operation originated. Therefore, operations are replicated only once and are not replicated back to the originating table.



In this diagram, both tables are in a single cluster. Operations on table `customersA` are replicated to table `customersB` and vice versa.

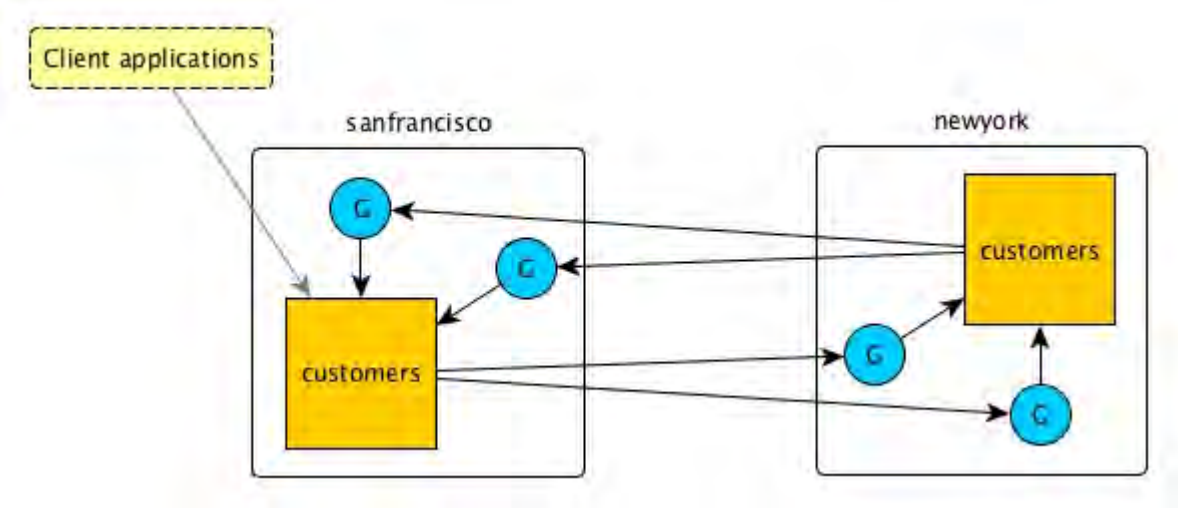


Offline Tables

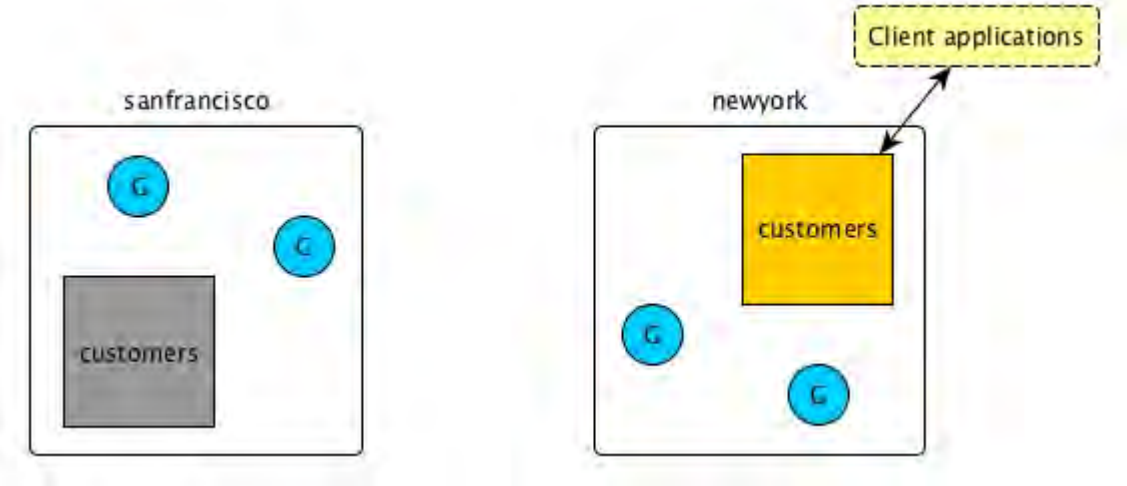
If one of the tables goes offline, you can direct client applications to the other table. When the offline table comes back online, replication between the two tables resumes automatically. When both tables are in synch again, you can redirect client applications back to the original table.

For example, assume that client applications are using the `customers` table that is in the cluster `sanfrancisco`.

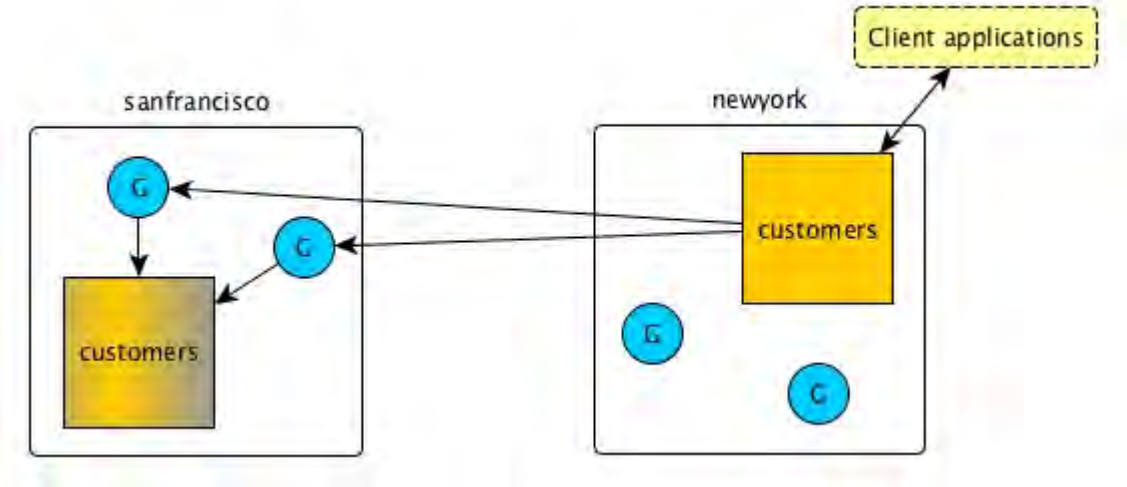
last-updated: Apr 11, 2022



The `customers` table in the `sanfrancisco` cluster becomes unavailable, so you redirect those client applications to the `customers` table in the `newyork` cluster.

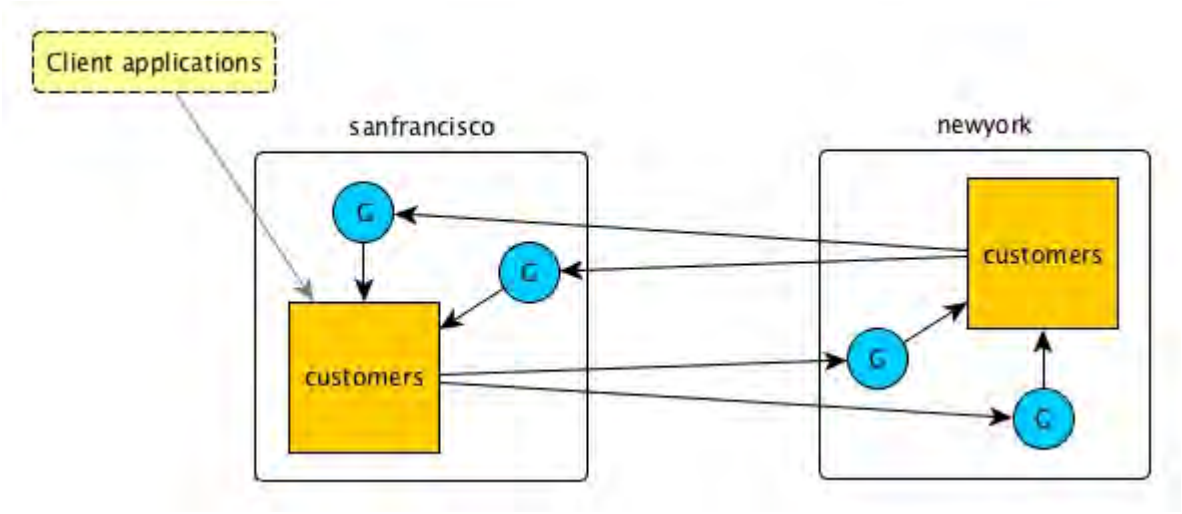


After the `customers` table in the `sanfrancisco` cluster comes back online, replication back to it starts immediately. As client applications are not yet using this table, there are no updates to replicate to the table in the `newyork` cluster.



last-updated: Apr 11, 2022

When the `customers` table in the `sanfrancisco` cluster is in sync with the other table, you can redirect your client applications back to it.



Conflict Resolution

The method that MapR Database uses to resolve conflicts depends on the type of table involved in a multi-master replication scenario.

Conflict resolution for binary tables	In the case of conflicting changes, MapR Database compares the cell timestamps of the two changes. In the rare event that the cell timestamps are identical, MapR Database compares the timestamps for when the changes arrived at their respective source tables. In the even rarer event that these latter timestamps are identical, MapR Database uses the C library function `memcmp` to compare the data that is being modified by the conflicting changes, favoring the greater value.
Conflict resolution for JSON tables	If two values conflict, MapR Database compares their types. The value of the type with the higher precedence is retained. Here is a list of the supported types in descending order of precedence: • Array • Document • Binary • Interval • Timestamp • Time • Date • Decimal • Double • Float • 64-bit integer

last-updated: Apr 11, 2022

- 32-bit integer
- 16-bit integer
- 8-bit integer
- UTF-8
- Boolean
- NULL

If both the conflicting values are of the same type, MapR Database compares the values themselves. All values are comparable except for values that are arrays or NULL.

Type	How Values Are Compared
Binary	The greater value is retained.
Interval	The later interval is retained.
Timestamp	The later timestamp is retained.
Time	The later time is retained.
Date	The later date is retained.
Decimal	The greater value is retained.
Double	The greater value is retained.
Float	The greater value is retained.
64-bit integer	The greater value is retained.
32-bit integer	The greater value is retained.
16-bit integer	The greater value is retained.
8-bit integer	The greater value is retained.
UTF-8	The greater lexicographic value is retained.
Boolean	TRUE is retained.

Time-to-Live for Deletes

Normally, delete operations are purged after the affected table cells are updated. Whereas the result of an update is saved in a table until another change overwrites or deletes it, the result of a delete is not saved. In multi-master replication, this difference can lead to tables being unsynchronized.

Example Scenario to Illustrate Time-to-Live for Deletes

1. On `/mapr/sanfrancisco/customers`, put row A at 10:00:00 AM.
2. On `/mapr/newyork/customers`, delete row A at 10:00:01 AM.

On `/mapr/sanfrancisco/customers`, the order of operations is:

- Put row A with a timestamp of 10:00:00 AM
- Delete row A with a timestamp of 10:00:01 AM (This operation is replicated from `/mapr/newyork/customers`.)

On `/mapr/newyork/customers`, the order of operations is:

- Delete row A with a timestamp of 10:00:01 AM
- Put row A with a timestamp of 10:00:00 AM (This operation is replicated from `/mapr/sanfrancisco/customers`.)

Now, though the put happened on `/mapr/sanfrancisco/customers` at 10:00:00 AM, the put reaches `/mapr/newyork/customers` several seconds after that. Assume that the actual time the put arrives at `/mapr/newyork/customers` is 10:00:03 AM.

To ensure that both tables stay synchronized, `/mapr/newyork/customers` should preserve the delete until after the put is replicated. Then, the delete can be applied after the put. Therefore, the time-to-live for the delete should be at least long enough for the put to arrive at `/mapr/newyork/customers`. In this case, the time-to-live should be at least 3 seconds.

In general, the time-to-live for deletes should be greater than the amount of time that it takes replicated operations to reach replicas. By default, the value is 24 hours. Configure the value with the `-deletettl` parameter in the `maprcli table edit` command.

For example, suppose (to extend the scenario above) that you pause replication during weekdays and resume it on weekends. The put takes place on Monday morning `/mapr/sanfrancisco/customers` at 10:00:00 AM and the delete takes place at `/mapr/newyork/customers` at 10:00:01 AM. Replication does not resume until 12:00:00 AM Saturday morning. Given the volume of operations to be replicated and the potential for network problems, it is possible that these operations will not be replicated until Sunday. In this scenario, a value of 7 days for `-deletettl` (7 multiplied by 24 hours) should provide sufficient margin.

Gateways for Replicating MapR Database Tables

In MapR Database table replication, MapR Database replicates updates to tables (binary and JSON) on source MapR clusters to replicas of those tables on destination MapR clusters. Gateways are services that receive these updates and apply them to the replicas. These gateways also propagate updates from JSON tables to their secondary indexes.

To set up gateways for replicating MapR Database tables:

- On the nodes of destination clusters, install the gateways.

  You must install gateways in the destination clusters.

  - If the destination cluster is remote from the source cluster, then the gateways must be in the remote cluster.
  - If the destination cluster is the source cluster, meaning that a source table and its replica are located in a single cluster, then the gateways must be in the local cluster.
  - If you have secondary indexes on your MapR Database JSON tables, then the gateways must be in the local cluster.

last-updated: Apr 11, 2022

- On the source clusters, configure the gateways by listing the destination cluster and the gateways that are running on them.

For information on configuring, and managing gateways, see:

- Configuring Gateways for Table and Stream Replication on page 1152
- Managing Gateways on page 1154

How Replication Works

During replication, MapR Database sends source table updates to the gateways on the destination clusters where the replicas of those source tables are located. Gateways batch the updates and then apply them to replicas.

All updates from a source table arrive at a replica after having been authenticated at a gateway. Therefore, *ACE* on the replica that control permissions for updates to column families and columns are irrelevant; gateways have the implicit authority to update replicas.

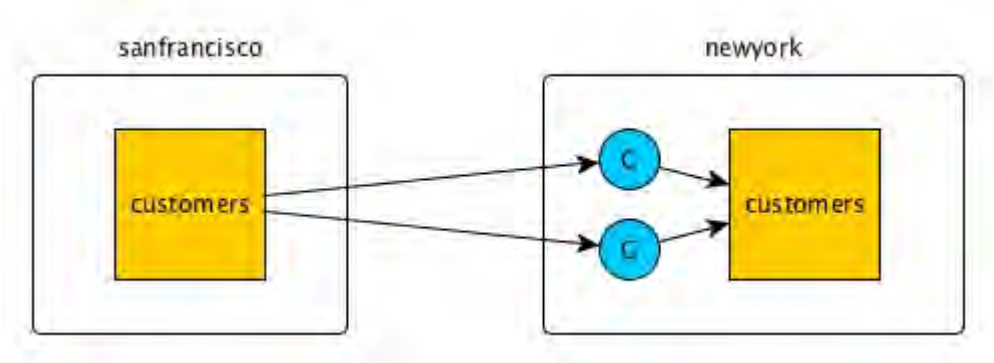
MapR Database distributes updates to a destination cluster's gateways in round-robin fashion. If a gateway is down or unreachable, MapR Database chooses another gateway or retries the operation on the same gateway.

Note: If a table is replicated to another table using the replication gateway, and you run a truncate operation on the source table, this operation is not replicated.

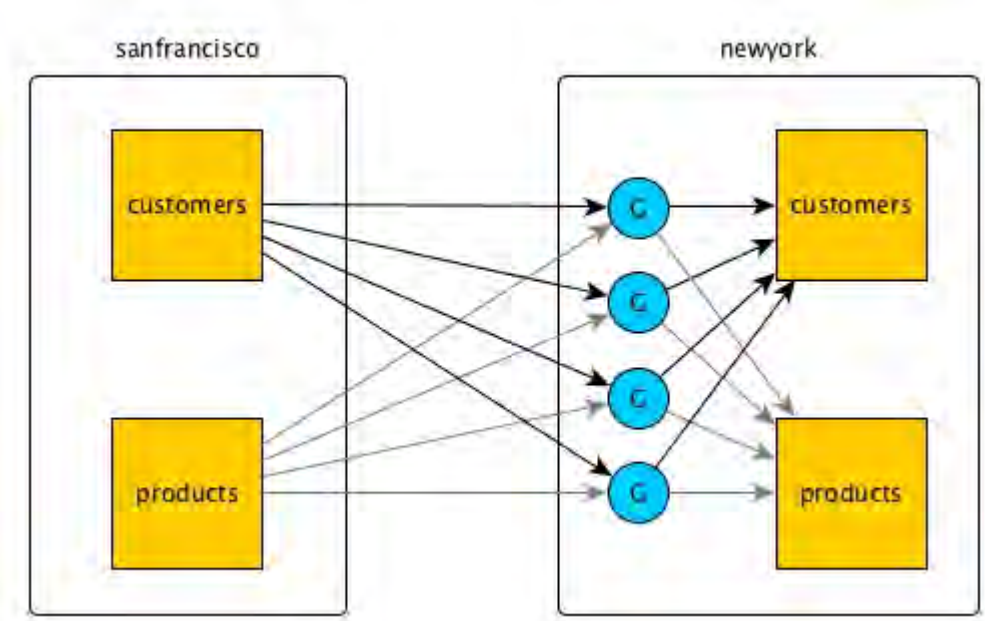
Gateways on nodes in remote destination MapR clusters

In this type of topology, gateways receive updates that are made to source tables, authenticate with the destination cluster on behalf of the source cluster, and apply the updates to the corresponding replicas.

This schematic diagram of basic inter-cluster primary-secondary replication shows updates to the customers table in the cluster `sanfrancisco` being sent to gateways. The gateways then apply the updates to the replica that is in the cluster `newyork`.



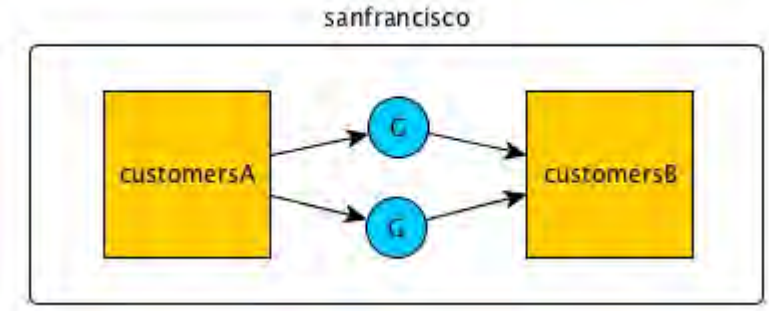
The gateways on a destination cluster are not assigned to particular replicas. They apply updates to all replicas on the destination cluster. For example, in this diagram, updates to two source tables in the cluster `sanfrancisco` are being replicated to two replicas in the cluster `newyork`. There are four gateways. Each gateway receives updates to both source tables, and each gateway applies those updates to both replicas.



Gateways on nodes within a MapR cluster serving as source and destination

In this type of topology, gateways again receive updates that are made to source tables and apply the updates to the replicas. However, all of this activity takes place within a single MapR cluster.

This schematic diagram of basic inter-cluster primary-secondary replication shows updates to the `customersA` table in the cluster `sanfrancisco` being sent to gateways. The gateways then apply the updates to the table `customersB`.



Related concepts

Administering MapR Gateways on page 1150
A MapR gateway mediates one-way communication between a source MapR cluster and a destination cluster. You can replicate MapR Database tables (binary and JSON) and MapR Event Store For Apache Kafka streams. MapR gateways also apply updates from JSON tables to their secondary indexes and propagate Change Data Capture (CDC) logs.

Configuring Gateways for Table and Stream Replication on page 1152
Configuring gateways involves installing the `mapr-gateway` package on nodes on a MapR destination cluster and then configuring the MapR source cluster to communicate with the destination cluster. The MapR source cluster is configured by specifying the destination cluster's CLDB node and gateway nodes.

gateway.conf on page 2197

Related tasks

Specifying the Location of Gateways on page 782

last-updated: Apr 11, 2022

Describes how to set the location of the MapR gateways using either the Control System or the CLI.

Related reference

cluster gateway delete on page 1564
Deletes the list of MapR gateways from a source MapR cluster.

cluster gateway get on page 1565
Lists the MapR gateways that a source MapR cluster is using.

cluster gateway list on page 1567
Lists all of the gateways that a source MapR cluster is using.

cluster gateway local on page 1569
Lists the gateways that are configured on the MapR cluster where this command is run.

cluster gateway resolve on page 1571
Lists the gateways configured on a MapR cluster that are up and running at the time that the command is issued.

cluster gateway set on page 1573
Specifies the locations of the MapR gateways that a source MapR cluster can use for table replication to a destination MapR cluster, or for indexing table data in an Elasticsearch cluster.

Related information

Managing Gateways on page 1154
Describes the commands for listing gateways, checking status of gateways, managing gateways if they fail, and troubleshooting gateways.

Replica Autosetup for MapR Database Tables

The option to automatically set up table replication, also known as replica autosetup, performs the steps to set up and start the replication of MapR Database binary table and MapR Database JSON tables. The replica autosetup option is available through the Control System and maprcli commands.

In general, replica autosetup performs the following steps to set up replication:

1. Creates a new table with metadata from the source table in the destination cluster. The primary table attributes are copied initially if an auto setup is used, because auto setup creates the destination table and any subsequent changes to the primary table's metadata are not propagated to the destination table. In the manual table replication setup, no metadata is propagated, even during the setup.
2. Declares the new table to be a replica of the source table and sets a paused replication state to ensure that replication does not begin immediately after the next step.
3. Declares the source table as an upstream source for the replica.
4. Loads a copy of the source data into the replica.
5. For multi-master replication, replica autosetup declares the source table to be a replica of the new table and then declares the new table to be an upstream source for the source table.
6. Clears the paused replication state to start the replication stream.

By default, replica autosetup uses the directcopy option. However, based on how you run replica autosetup, you also have the choice not to use directcopy.

Replica Autosetup with Directcopy (default)

The directcopy option uses gateways to perform all setup operations including the initial population of data into the replica table. Directcopy is the default option when you setup table replication using the Control System or with the `maprcli table replica autosetup` command.

When a client submits a request to automatically setup table replication to the cluster, the source cluster acknowledges the request and begins to track the replica autosetup request from start to finish.

If a failure occurs when replica autosetup operations are in progress, the source cluster resumes operations from the point of failure.

Note: To check the replication status of a table, run the `maprcli table replica list` command. To stop the automatic setup of table replication, run `maprcli table replica remove`, or delete the source or replica table.

Replica autosetup with directcopy provides the following benefits:

- **Replica autosetup operations do not block the client from submitting additional requests.** When setting up table replication, the process to copy source data to the replica can be time consuming. The client does not need to wait for the replica autosetup request to complete before submitting another request.
- **Source cluster retries replica autosetup operations in case of failure.** The source cluster keeps track of the replica autosetup progress. This allows the source cluster to resume autosetup operations in the event of an intermittent failure. If you choose to not use directcopy, user intervention is required if a failure occurs.
- **Throttling of copy table operations is done by default.** Throttling prevents the initial copy of data from the source to the replica table from consuming all cluster resources.

Replica Autosetup without Directcopy (not default)

Without the directcopy option, replica autosetup submits a majority of the replication setup requests through the client and then runs a copy table utility to populate the initial table data. To use replica autosetup without the directcopy option, run `maprcli table replica autosetup` command with the `-directcopy` parameter set to `false`.

Without the directcopy option, once a client submits a replica autosetup request to the cluster, it must wait until the source cluster sends a notification that the autosetup request is complete before it can submit another request to the cluster. In this case, replica autosetup uses the client connection to submit autosetup operation requests such as create replica, add replica, and add upstream source. To populate the initial table data, the client runs `mapr copytable` for JSON tables and the `CopyTable` utility for binary tables.

If a failure occurs when replica autosetup operations are in progress, the client hangs and any replica tables that were created during the failed autosetup operations must be manually deleted before trying to setup replication again.

Table Replication States

The replication state indicates when table replication is in progress and displays the status of operations related to replica autosetup.

The `maprcli table replica list` command displays the following replication states.

State	Description
REPLICA_STATE_WAIT_TILL_BULKLOAD	Replica autosetup with directcopy has not started because bulkload is in progress on the source table.
REPLICA_STATE_CREATE_SCHEDULE	Replica autosetup with directcopy had scheduled the creation of the replica table.
REPLICA_STATE_COPY_SCHEDULE	Replica autosetup with directcopy has not started the initial copying of source data to the replica because it is waiting for other in-progress copy operations to complete.
REPLICA_STATE_COPY_IN_RECOVER	Replica autosetup with directcopy is resuming the copy of source data to the replica after a connection failure.

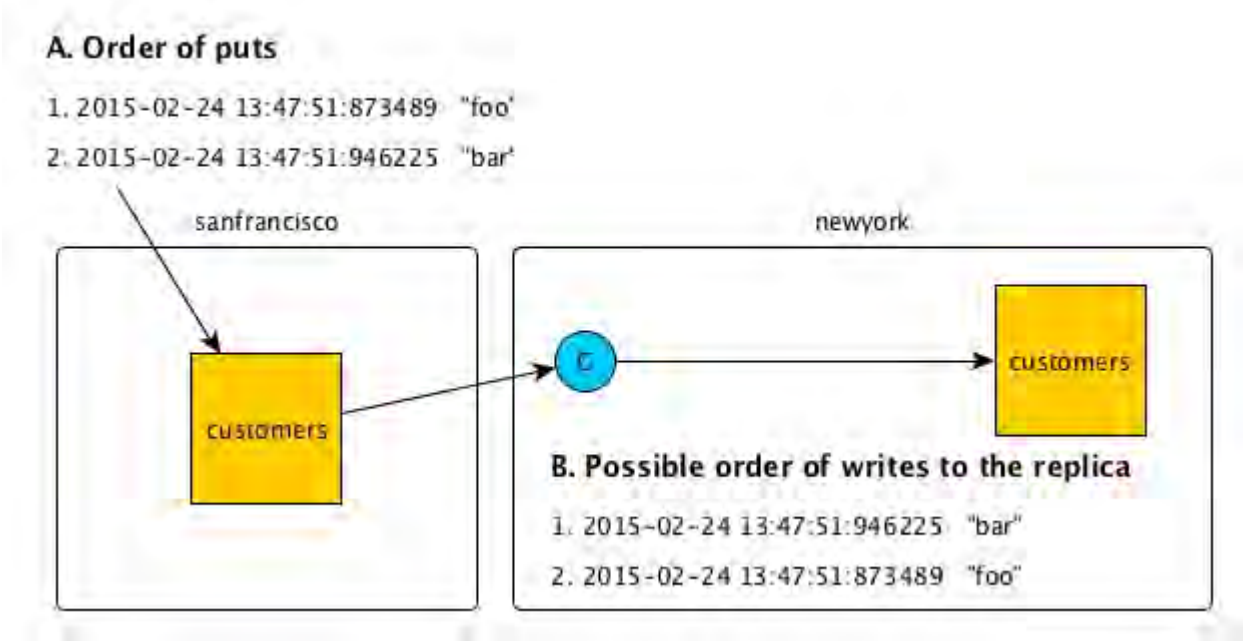
last-updated: Apr 11, 2022

State	Description
REPLICA_STATE_COPY_IN_PROGRESS	Replica autosetup with directcopy is copying the source data to the replica.
REPLICA_STATE_DELETING_CURSORS	Replica autosetup with directcopy is deleting progress cursors since the initial copy of source data to the replica is complete.
REPLICA_STATE_REPLICATING	Replication is in progress.
REPLICA_STATE_UNEXPECTED	Replica is in an unexpected state. See the `error` field in the output from `maprcli table replica list` for more information.

Order of Writes at Replicas

It is possible for replicated operations to arrive at and be written to a replica in an order different from the order they were written to a source table.

In this diagram, the values “foo” and then “bar” are written to the source table. However, due to network issues, the values are written to the replica in the reverse order: “bar”, “foo”



Client applications on the destination cluster should not depend on updates being written to the replica in the same order in which they were written to the source table.

Security and Replication

Describes how to replicate data between secure clusters.

Security is configured at all locations in the replication stream.

On clusters

You can replicate between clusters that are secure. See Configuring Secure Clusters for Cross-Cluster Mirroring and Replication on page 1486 for more information about replication between secure clusters.

At source tables

The `-replperm` parameter lets you specify an *ACE* to declare who has permission to replicate data from a table. This parameter is available in the `maprcli table create` and `maprcli table edit` commands.

Across a network

You can send data encrypted or unencrypted when replicating between secure clusters by using the `-networkencryption` parameter when adding a replica to a source table.

At gateways

Gateways ensure that replicas receive updates only from source tables that are designated as upstream sources.

Moreover, gateways handle authentication with secure destination clusters.

At replicas

Due to several upstream security checks, no parameters are needed for setting *ACE* to declare who has permission to update a replica through a replication stream. However, before replication begins, replicas can be loaded with a snapshot of the data in corresponding source tables. Permission to perform such a load is controlled by the *ACE* that you set in the `-bulkloadperm` parameter for a replica. You can set the *ACE* with either the `maprcli table create` or the `maprcli table edit` command.

All other *ACE* defined for a replica still apply for local updates and reads.

Licensing

Describes the licensing requirements for MapR.

Table replication requires a license for MapR Enterprise Database Edition (M7) on source and destination clusters.

Gateways for Indexing MapR Database Data in Elasticsearch

As of MapR 6.0, MapR Database Elastic Search integration capability is deprecated and no longer available in the MapR Database product.

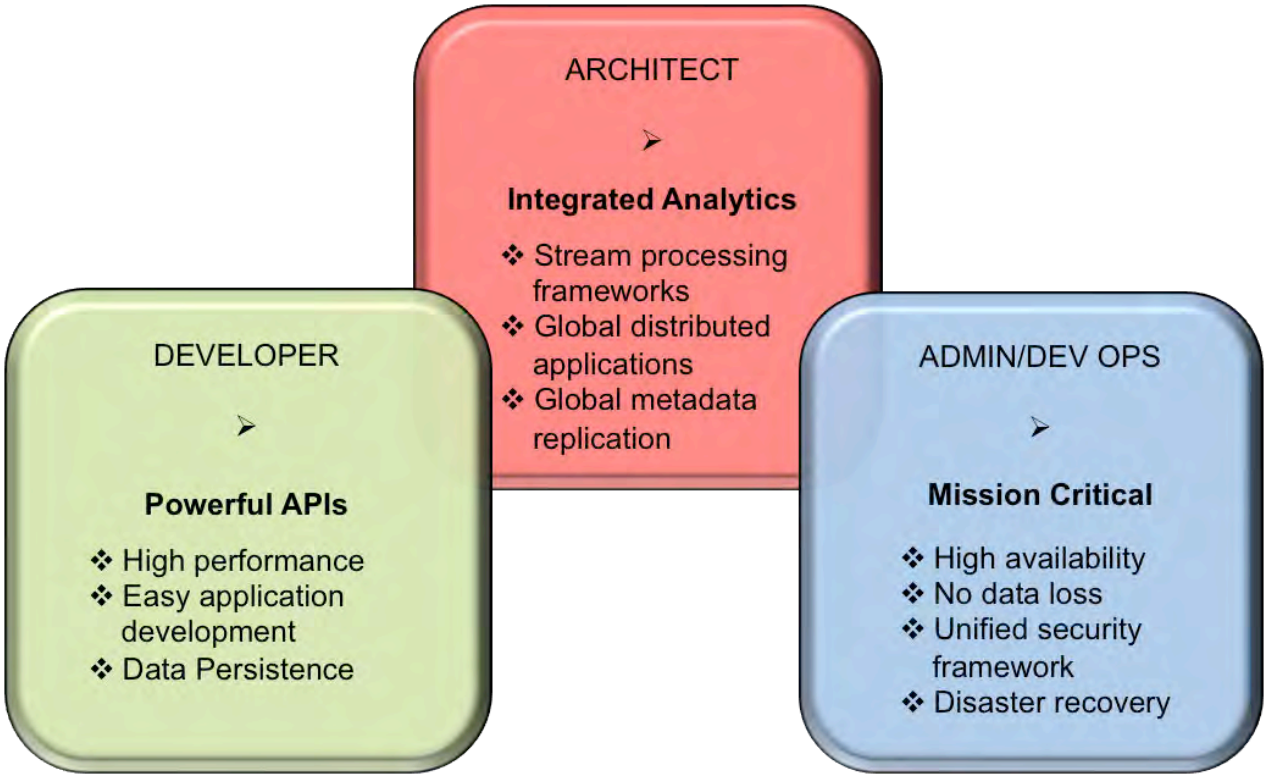
Attention: MapR Database Change Data Capture (CDC) framework can be used to integrate with latest versions of Elasticsearch. See Change Data Capture on page 597 for more information.

MapR Event Store For Apache Kafka

MapR Event Store For Apache Kafka brings integrated publish and subscribe messaging to the MapR Converged Data Platform.

Why MapR Event Store For Apache Kafka?

MapR Event Store For Apache Kafka is built into the MapR platform. It requires no additional process to manage, leverages the same architecture as the rest of the platform, and requires minimal additional management.



1. The MapR Event Store For Apache Kafka and Apps section information and examples for developing Producer and Consumer applications.
2. The MapR Event Store For Apache Kafka concepts section covers information associated with streams, topics, and messages.
3. The Administering Streams section provides information about creating and managing streams, topics, and stream replication.

How Do I Get Started?

Based on your role, review the MapR Event Store For Apache Kafka documentation. The following table identifies useful resources based on your role.

Developer	Architect	Administrator/Dev Ops
➢ MapR-ES and Applications	➢ MapR-ES Architecture	➢ Installing MapR
➢ MapR-ES Java Applications	➢ Stream Design	➢ Administering MapR-ES
➢ MapR-ES C Applications	➢ Life of a Message	➢ maprcli and REST API Syntax
➢ MapR-ES Python Applications	➢ Stream Replication	➢ Utilities for MapR-ES Streams

1. MapR Event Store For Apache Kafka and Apps
2. MapR Event Store For Apache Kafka Java Applications
3. MapR Event Store For Apache Kafka C Applications
4. MapR Event Store For Apache Kafka Python Applications
5. MapR Event Store For Apache Kafka architecture and concepts.
6. Determining stream design.
7. Describes the flow of a message from a producer to a consumer.
8. Describes the factors associated with stream replication.
9. Installing MapR
10. Administering MapR Event Store For Apache Kafka streams.
11. Using the maprcli for managing streams.
12. Utilities for MapR Event Store For Apache Kafka Streams.

Additional Resources

See the following MapR sites for more MapR Event Store For Apache Kafka information:

- MapR Event Store For Apache Kafka Product Page
- Blog: Kafka vs MapR Streams: Why MapR?
- Blog: Getting Started with MapR Streams
- Blog: Key Requirements for Streaming Platforms: A Micro-Services Advantage
- Blog: Streaming Data: How to Move from State to Flow
- Blog: Anomaly Detection in Telecommunications Using Complex Streaming Data

last-updated: Apr 11, 2022

- Blog: How to Persist Kafka Data as JSON in NoSQL Storage Using MapR Streams and MapR Database

Architecture

Streams contain topics that have logical collections of messages.

In MapR Event Store For Apache Kafka, topics are grouped into *streams*. Administrators can apply security, retention, and replication policies on streams. Combined with MapR File System and MapR Database in the MapR Data Platform, using these streams enables organizations to create a centralized, secure data lake that unifies files, database tables, and message topics.

Messages (topic data) are published to *topics* by Producer applications and are read by Consumer applications. All messages published to MapR Event Store For Apache Kafka are persisted, allowing future consumers to “catch-up” on processing and analytics applications to process historical data. Additionally, messages are specifically written to *topic partitions*.

Note: Topic partitions are stored in containers within volumes. Containers are written to storage pools, which are made up of disks on the nodes in the cluster. See Containers and the CLDB on page 453 for more information about containers.

Why Use MapR Event Store For Apache Kafka?

MapR Event Store For Apache Kafka is ideal for a variety of use cases, including the following:

Application event pipelines	Many types of applications generate event or log data that must be centrally stored and analyzed to gain insights about user activity or application performance. MapR Event Store For Apache Kafka simplifies these pipelines by transporting events to a central location, from which they can undergo event-by-event transformation and analysis.
Database change capture	Most modern databases enable users to generate an event each time an entry is added or modified. These events can be published to MapR Event Store For Apache Kafka to keep systems like search indexes and caches synchronized, as well as to feed security or notification applications.
Internet of Things	The explosion in the number of smart devices and sensors has created many situations in which billions of data points are created by millions of geographically dispersed sensors. MapR Event Store For Apache Kafka provides a reliable, global transport for these messages, enabling you to perform analytics both at the source and at a central location.

Replication

In addition to reliably delivering messages to applications within a single data center, MapR Event Store For Apache Kafka can continuously replicate data between multiple clusters, delivering messages globally. Like other MapR services, MapR Event Store For Apache Kafka has a distributed, scale-out design, allowing it to scale to billions of messages per second, millions of topics, and millions of producer and consumer applications.

Server and Client Libraries

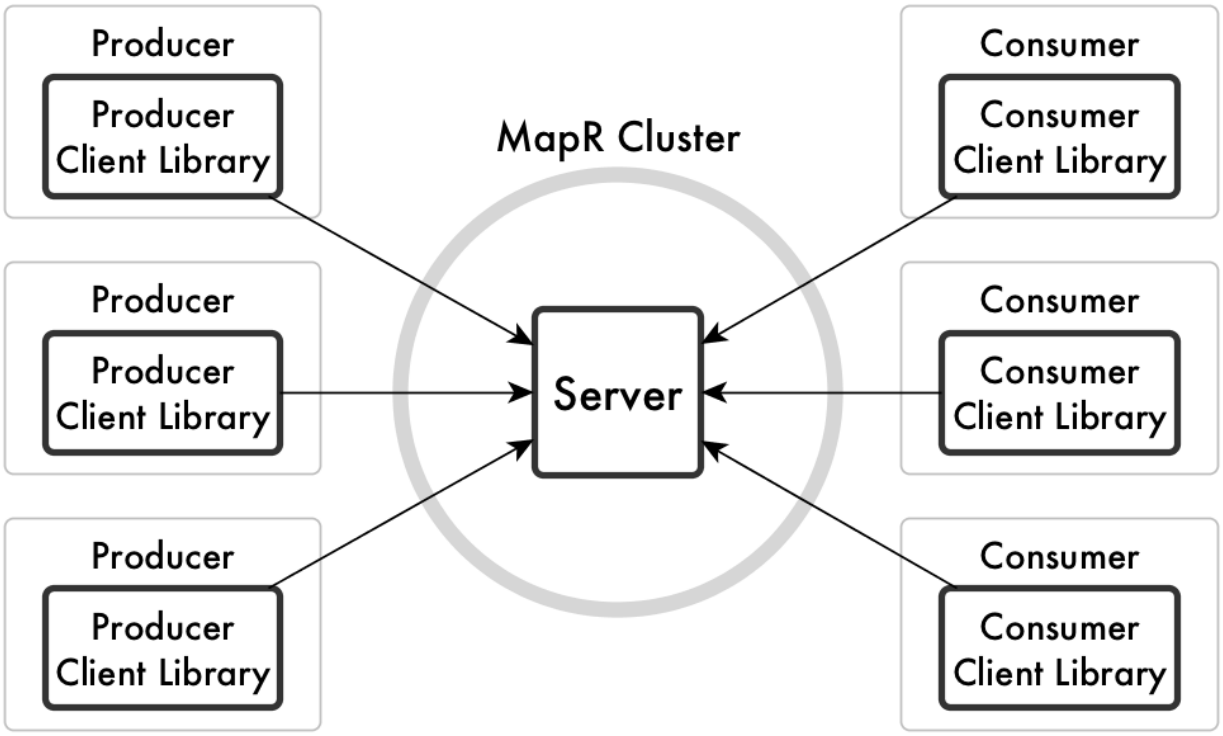


Figure 5: The relationship of the MapR Event Store For Apache Kafka server to producers, consumers, and client libraries

Server	The server manages streams, topics, and partitions and handles requests from the producer client library and the consumer client library.
Producer client library	This client side library which is part of the producer process receives the messages that are sent by producers, buffers the messages, and sends them to the server, which then publishes the messages and sends the client acknowledgements.
Consumer client library	This client side library which is part of the consumer process receives requests from consumers to poll subscriptions for unread messages, reads messages from topic partitions, and sends messages to consumers.

Stream Design

Streams are created in volumes and contain topics, which in turn, contain messages. Security, replication, retention, and compression policies are applied at the stream-level.

When designing the architecture, take in account the following factors:

- Security Permissions (using access-control expressions)

  Security permissions are set at the stream-level and, subsequently, topics inherit the stream permissions. See Stream Security on page 664 for more information.

- Data Replication

  MapR Event Store For Apache Kafka streams can be replicated to other streams in the same or different MapR clusters. For example, you can create a backup copy of a stream so that producers and consumers fail over to the backup if the original stream goes offline. See Stream Replication on page 656

- Data Retention

  The time-to-live of a message is the elapse time (in seconds) between the publication of a message in a topic in this stream and the expiration of that message. See Time-to-Live for Messages on page 637 for more information.

- Data Compression

  Topic messages can be sent to the server compressed. When compression is implemented, the messages are stored compressed, replicated to other containers compressed, and (if stream replication is configured) replicated to replica streams compressed. See Managing Compression on page 988 for more information.

Pollution Monitor Example

Suppose that you plan to create the stream `pollution_monitors` to collect various measurements about pollution levels in European cities. However, during a planning session, the representative from Amsterdam says that her country wants to analyze the data for its cities and would like your company to replicate the data to its own MapR cluster, where its own consumers can read the replicated messages.

In this scenario, you might do the following:

- Create a stream dedicated to the Netherland's pollution data or even for every country you are monitoring. For example, create streams named `pollution_monitors_netherlands`, `pollution_monitors_sweden`, `pollution_monitors_france`, and so on.
- Within each stream, create topics for each city in that county. For example, create a topic named `amsterdam` that contains data from Amsterdam's pollution sensors.
- Since, in this scenario, the Amsterdam representative also requested stream replication to their own MapR cluster, you would set up stream replication from your MapR cluster to Amsterdam's MapR cluster. See Managing Stream Replication on page 1130 for information about setting up and managing replication.

Alternatively, consider that the Netherlands did not request replication to their own MapR cluster. However, you want to restrict access to the pollution data where consumers could read only pollution data for their respective country.

In this scenario, you might do the following:

- Create streams for each country.
- Create topics for each city in that country.
- Set each stream's `consumeperm` permission for consumers associated with that country. See Stream Security on page 664 for more information about security permissions used with MapR Event Store For Apache Kafka streams. For general information about access-control expressions, see ACE Syntax on page 1448.

Stream Topics

Topics are created in streams and contain logical collections of messages. These collections of messages are published to partitions in the topic.

Using MapR Event Store For Apache Kafka with MapR File System and MapR Database in the MapR, enables organizations to create a centralized, secure data lake that unifies files, database tables, and message topics.

Note: Topics inherit security permissions, time-to-live data retention, and data compression policies at the stream-level.

You can design topic usage in a variety of ways. For example, you might have an application that monitors the logs for mission-critical software. Your monitoring application could send informational messages to a topic named `info`, warning messages to a topic named `warnings`, and error messages to a topic named `errors`. Different downstream applications might monitor each topic.

You can manage topics for different scenarios:

- Set security policies that apply all of the topics in that stream. Security policies are set at the stream-level. See Stream Security on page 664 for more information.
- Set a default number of partitions for each new topic that is created in the stream. The default number partitions is set at the stream-level, however, individual topics can override the default. See Topic Partitions on page 634 for more information.
- Set a time-to-live for messages in every topic in the stream. Every message in every topic in a stream expires after a duration of time, unless you set the time-to-live to 0, meaning messages never expire. Time-to-live is set at the stream-level. See Time-to-Live for Messages on page 637 for more information.

Restrictions

- After a topic is created in a stream, it is not possible to move that topic to a different stream.

  For example, suppose you create the topic `structural_integrity_sensors_us_western_region`, one of a number of topics that collect data from sensors that keep watch over various measurements for bridges, buildings, and other structures. However, you have mistakenly created the topic in the stream `ventilation_systems` instead of the stream `structural_integrity_sensors`.

  Note: There is no command that will move a topic from its current stream to a different stream.

  To rectify this mistake:

  - You must delete the topic and recreate it in the other stream.
  - Any producers that published messages to the topic and any consumers that read messages from the topic must be modified to point to the new location of the topic. This is because producers and consumers refer to topics with a combination of stream name and topic name.
- Only the following characters are allowed for stream topic names:
  - Alphanumeric characters
  - Period, underscore, and dash
- When producing or consuming stream topic messages, you must specify the stream's path and name along with the topic name in the following manner:

  ```
  /<stream name>:<topic name>
  ```

  Note: If a topic is specified but the stream's path and name is not, depending on the application's programming language, you might get an error or nothing. If nothing happens and you are using Java, the assumption is that you are publishing to Apache Kafka.

last-updated: Apr 11, 2022

Topic Partitions

Partitions, which exist within topics, are parallel, ordered, immutable sequences of messages that are continually appended to.

Topics can contain multiple partitions, which make topics scalable by spreading the load for a topic across multiple servers.

Downstream applications that read messages can read from multiple partitions within a topic for faster performance than would be possible if they read from a single partition per topic. Downstream applications can also scale by having separate instances read from separate partitions.

When creating or editing a stream, a default number of partitions can be specified for that stream's topics. Topics inherit the stream's partition default. However, topics can also override the stream's partition default by setting the number of partitions to be used.

Performance

The default number of partitions for MapR streams and topics can impact performance. Depending on the volume of messages being published to a topic, the default number of partitions might be increased for efficient consumption.

When there is a high volume of messages being published to a topic:

- Multiple consumers, in consumer groups, reading from multiple partitions are handled more efficiently.
- Individual consumers each reading from a single partition are handled less efficiently.

Reference

The following lists topics that have more detailed information.

- See the maprcli stream create on page 1758 for information about creating streams with the `-defaultpartitions` parameter.
- See the maprcli stream edit on page 1765 for information about editing streams with the `-defaultpartitions` parameter.
- See the maprcli stream topic create on page 1781 for information about creating topics with the `-partitions` parameter.
- See the maprcli stream topic edit on page 1784 for information about modifying topics with the `-partitions` parameter.
- See the maprcli stream topic info on page 1785 for information about topic data including the `-partitions` parameter.
- See the MapR Event Store For Apache Kafka Java API Library on page 2756 for the methods used to create and edit streams and to create and edit topics.

Topic Creation

Topics are created in streams and contain logical collections of messages. They can be created either automatically through your producer application or manually through the Control System or the `maprcli` commands.

Automatic Creation

If the topic does not already exist, a topic is created automatically when a producer first publishes a message to it. This is the default behavior.

For example, you created the stream **anonymous_usage** that you intend to use to collect data about the usage of a software application that is soon to be released. However, you did not create any stream topics because the topics were not known at the time. After the software is released to the public, at some point, a

producer application starts publishing messages to a topic that is created based on the range within which the producer's IP address falls. At another point in time, another producer application starts publishing messages to a topic based on a different range of IP addresses. Eventually, the stream contains a number of topics for different IP address ranges.

Note: Automatic creation of topics can be turned off by setting the `autocreate` parameter to **false** either when creating the stream or by editing the stream. See the `maprcli` stream create on page 1758 or stream edit on page 1765 command for more information. If you turn off automatic creation, you must manually create the stream topic, otherwise, the publishing of a message fails.

Manual Creation
To create topics manually, use either the Control System or the `maprcli` commands. See Administering Streams on page 1119 for information about managing MapR Event Store For Apache Kafka streams, topics and replication. See the stream topic create on page 1781 command for specific information about creating stream topics with the `maprcli` command.

For example, you created a stream called **systemMetrics** that you intend to use to collect operational metrics from systems in your enterprise. You did create several topics based on system, location, company department, project, or some other criterion. In this case, you could create topics in advance because they were pre-planned.

Note: When you manually create a topic, you can have the option of customizing the number of topic partitions used, otherwise, the default number of partitions is inherited from the stream.

Topic Messages
Messages are key/value pairs, where keys are optional. The values contain the data payload, which can be text, images, video files, or any other types of data.

Messages are published into *topic partitions* by Producer applications and are read by Consumer applications. All messages published to MapR Event Store For Apache Kafka are persisted, allowing future consumers to “catch-up” on processing and analytics applications to process historical data. See Producers on page 643 and Consumers on page 647 for more information.

Offsets

Messages are assigned offsets when published to partitions. Offsets are monotonically increasing and are local to partitions. The order of messages is preserved within individual partitions, but not across partitions.

Note: As of MapR 6.0, the message offset in a partition starts from zero (0). If you are upgrading and did not enable the MapR Database/MapR Event Store For Apache Kafka feature **mfs.feature.db.streams.v6.support**, the message offset in a partition starts from one (1).

Logical Schema of Messages
Each message has the same logical schema: _id, topic, partition, offset, timestamp, producer, key, and value.

As the logical schema of each message is the same, analytics applications can run queries on these fields. See MapR Event Store For Apache Kafka Java API Library on page 2756 for information about querying messages and mapr streamanalyzer on page 5348 for a sample application used to query and count messages in topics.

```
{
          "_id":<STRING>,
          "topic":<STRING>,
          "partition":<SHORT>,
          "offset":<LONG>,
          "timestamp":<LONG>,
          "producer":<VARCHAR>,
          "key":<BINARY>,
```

```
            "value":<VARBINARY>
            }
```

Field	Description
`_id`	A STRING value that represents the ID of the topic in which the message is located.
`topic`	A STRING value that represents the name of the topic in which the message is located.
`partition`	A SHORT value that represents the index of the partition in the topic.
`offset`	A LONG value that represents the position of the message within a partition.
`timestamp`	A LONG value that represents the date and time of the message. As of MapR 6.0.1, MapR Event Store For Apache Kafka supports an event-time timestamp. The timestamp type can be either `createtime` (default) or `logappendtime`. A `createtime` value (default) is the time defined by the user or application (when creating the message). If user or application does not define this value (or passes null), the client uses the current system timestamp. A `logappendtime` value is the time when the message (log) was appended to the server. **Tip:** Because each message is automatically produced into a topic-partition with an event-time timestamp as part of the message record, this allows the Consumer to seek based on the timestamp.
`producer`	A VARCHAR value that represents the value of the `client.id` configuration parameter for the producer that published the message. MapR Event Store For Apache Kafka does not require a value for this configuration parameter, so the value for this field could be empty.
`key`	A BINARY value that represents the key of the message. MapR Event Store For Apache Kafka does not require each message to have a key, so this value could be empty. The configuration parameter `key.serializer` for the producer that published the message specifies the means by which the key was serialized. Your application can deserialize the key by using the appropriate deserialization class in the `org.apache.kafka.common.serialization` package or a class that implements the `Deserializer` interface.
`value`	A VARBINARY value that represents the value of the message. The configuration parameter `value.serializer` for the producer that published the message specifies the means by which the value was serialized. Your application can deserialize the value by using the appropriate deserialization class in the `org.apache.kafka.common.serialization` package or a class that implements the `Deserializer` interface.

Resources

For more information about creating and editing streams or topics:

- maprcli
  - See maprcli stream create on page 1758 for information about creating streams.
  - See maprcli stream edit on page 1765 for information about editing streams.
  - See maprcli stream info on page 1768 for information about streams.
  - See maprcli stream topic create on page 1781 for information about creating topics.
  - See maprcli stream topic edit on page 1784 for information about modifying topics.

  - See maprcli stream topic info on page 1785 for information about topic data.
- MapR Event Store For Apache Kafka Java API
  - See the MapR Event Store For Apache Kafka Java API Library on page 2756 for the methods used to create and edit streams and to create and edit topics.

Time-to-Live for Messages
The time-to-live (TTL) for messages means that messages persist in the partitions of a stream topic for a specific time period. During that time, messages can be read or re-read by consumers. Once the TTL for a message expires, the message is marked for deletion.

Setting TTL for Message

Set the TTL for topic messages when you create or edit a stream. Since the TTL setting is specified at the stream-level, all messages in all topics associated with the stream will have the same TTL. The default TTL is 604,800 seconds (7 days).

Deleting Expired Messages

Expired messages are deleted by an automatic process that runs at periodic intervals of TTL/10 and no later than 24 hours. For example:

- If the TTL is set to 24 hours, expired messages are deleted once every 24 hours.
- If the TTL is set to 7 days (168 hours, which is the default), expired messages are deleted once every 24 hours because 24 is greater than 168/10.
- If the TTL is set to 20 days (480 hours), expired messages are deleted once every 48 hours because 48 is greater than 24.

Attention: The automatic process deletes messages from *active* streams that have an expired TTL. The automatic process does not purge deleted messages with an expired TTL from *idle* streams until producer or consumer operations are performed on the streams.

Monitor disk space utilization and manually delete messages from streams, as needed, to reclaim disk space.

To manually delete expired messages, run the `maprcli` command stream purge on page 1769.

For More Information

- See maprcli stream create on page 1758 for information about creating streams.
- See maprcli stream edit on page 1765 for information about editing streams.
- See maprcli stream purge on page 1769 for information about purging expired topic messages.
- See MapR Event Store For Apache Kafka Java API Library on page 2756 for the methods used to create and edit streams

Life of a Message
To show how the MapR Event Store For Apache Kafka concepts fit together, here is an example of the flow of one message from a producer to a consumer.

The Setup

Suppose that you are using MapR Event Store For Apache Kafka as part of a system to monitor traffic in San Francisco. Your producers are sensors in streets, freeways, bridges, overpasses, and other

infrastructure, as well as sensors reporting the weather in many different locations. Your consumers are various analytical and reporting tools.

In a volume in a MapR cluster, you create the stream `/somepath/traffic_monitoring`. In that stream, you create the topics `traffic`, `infrastructure`, and `weather_conditions`.

Of all of the sensors (producers) that your system uses to monitor traffic, let us choose a sensor that is under the pavement of Market Street and follow a message that it generates. We will follow a message that is generated by this sensor and published in the `traffic` topic.

Suppose that, when you created this topic, you created several partitions within it to help spread the load among the different nodes in your MapR cluster and to help improve the performance of your consumers. For simplicity, we will assume that the `traffic` topic has only one partition.

A Message Enters the System

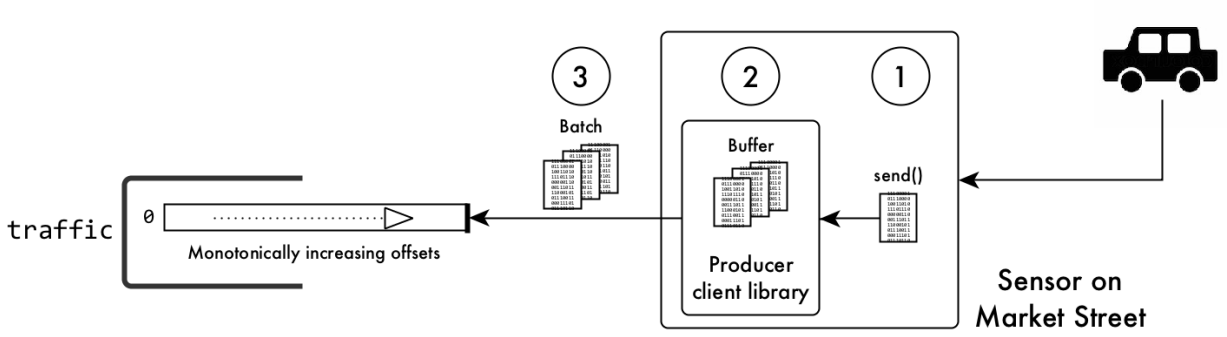


Figure 6: A car runs over a sensor, triggering the sending of a message

1. A car, one of hundreds on Market Street in morning rush-hour traffic, runs over the sensor. This action triggers the sensor to send a message to a MapR Event Store For Apache Kafka producer client library.

   Note: This message might list geospatial coordinates, time, date, direction, weight, distance between front and rear wheels, and more. MapR Event Store For Apache Kafka does not help you decide which data to collect.

2. The client buffers the message.

3. When the client has a large number of messages buffered (because other cars have subsequently triggered the sensor) or after an interval of time has expired, the client batches and sends the messages in the buffer. The message that we are following is published in the partition along with the rest of the messages in the batch. When the message is published, the MapR Event Store For Apache Kafka server assigns it the offset 001030 (which is only an example offset; real offsets are more sophisticated). These messages being the most recent to be published, are written to the head of the partition.

   For a moment, suppose that this example used more than one partition. In that case, the sensor could influence how the MapR Event Store For Apache Kafka server determines which messages go to which partition. In the example that we are following, the sensor could include a key with each message. The MapR Event Store For Apache Kafka server would hash the key to determine the partition to place the messages received from the sensor. More information about how partitions are selected if there are more than one in a topic is explained later in this documentation.

4. Each partition and all of its messages are replicated. The server owning the primary partition for the `traffic` topic assigns the offset 001030 to the message that we are following, and replicates the message to replica containers (replication rules are controlled at the volume level) within the MapR cluster.

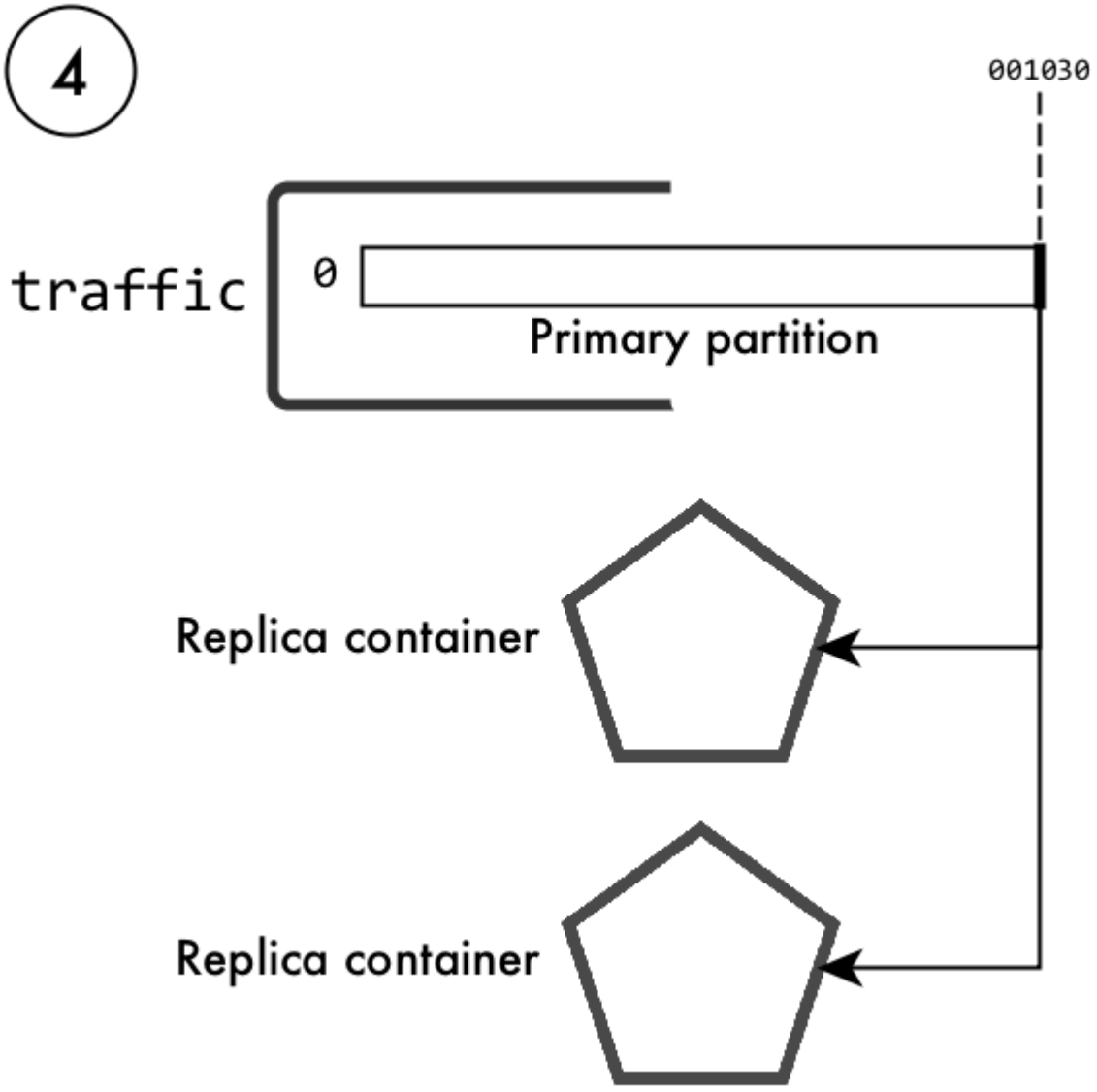


Figure 7: Replication of the partition in the topic `traffic`

5. The server acknowledges receiving the batch of messages and sends the offsets that it assigned to them.

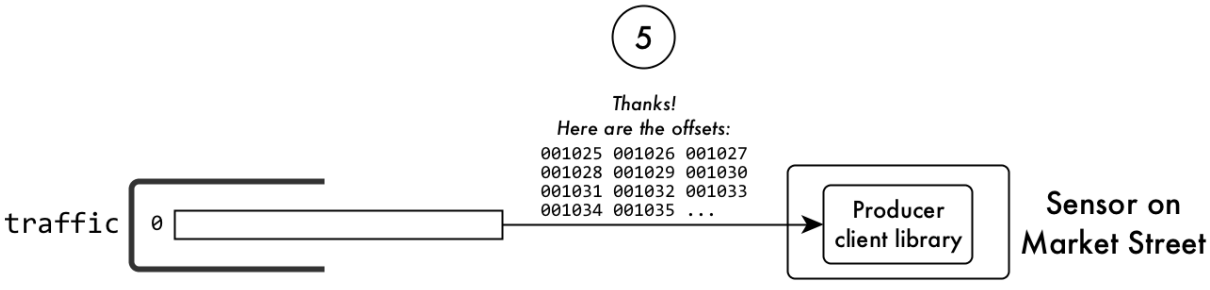


Figure 8: The server acknowledges receiving the messages

last-updated: Apr 11, 2022

The Message is Read from the System

An analytics application (consumer) that correlates traffic volume with weather conditions is subscribed to the `traffic` topic. Many more consumers could subscribe to it, too.

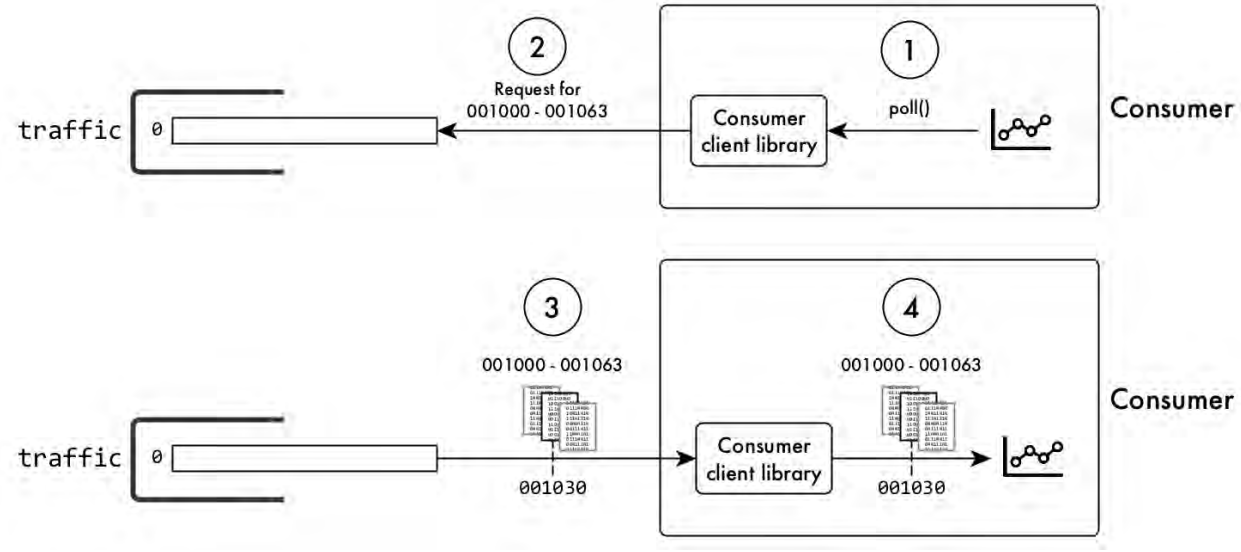


Figure 9: How messages are read

1. The application issues a request to the consumer client library to poll the topic for messages that the application has not yet read.
2. The client requests messages that are more recent than the consumer has yet read.
3. The primary partition returns multiple messages to the client. The originals of the messages remain on the partition and are available to other consumers.
4. The client passes the messages to the application, which extracts the data from them and processes it.
5. If more unread messages remain in the partition, the process repeats from step 2.

The Original Message is Deleted

Back in the cluster in San Francisco, messages are being continuously published to the partition in the `traffic` topic. Message 001030 is much further in the partition. More recent messages have filled the partition ahead of it.

When you created the stream, you set the time-to-live for messages to be six months. Message 001030 and messages around it have now been in the partition for that period, and are now expired. An automatic process eventually reclaims the disk space that message 001030 and the other expired messages are using.

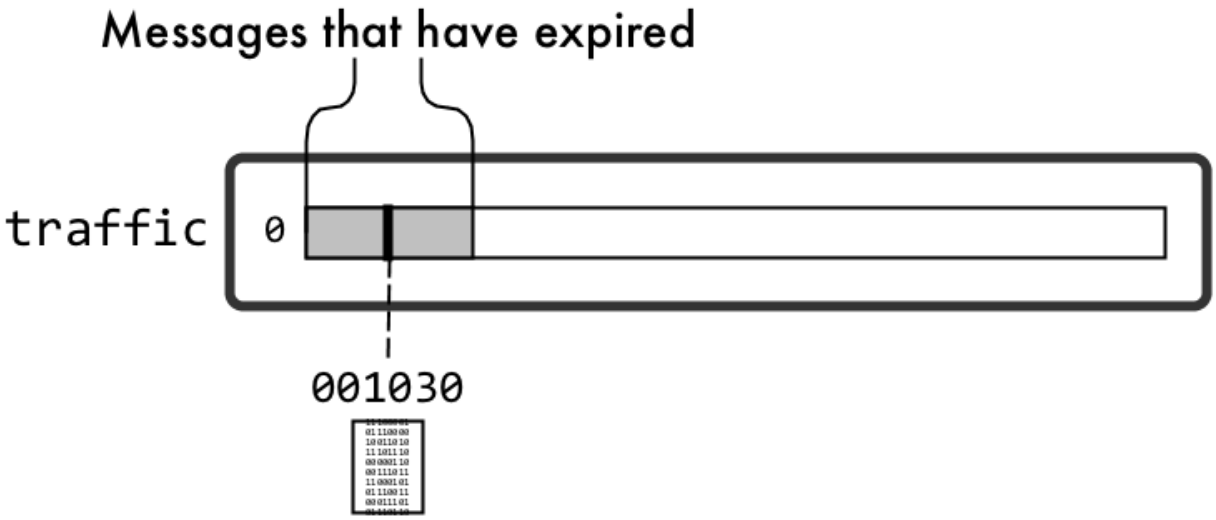


Figure 10: Messages to be deleted automatically

Log Compaction

Log compaction purges previous, older messages that were published to a topic-partition and retains the latest version of the record.

Log compaction reduces the size of a topic-partition by deleting older messages and retaining the last known value for each message key in a topic-partition. The `mincompactionlag` parameter provides a lower bound on how long each message remains prior to compaction and the `deleteretention` parameter provides a lower bound on how long a tombstone (a message with a null value) is retained. See the maprcli stream create on page 1758 and stream edit on page 1765 commands and Enabling Log Compaction on page 2765 for more information about these retention parameters.

Note: Log refers to the topic-partition pair. So when you are performing log compaction on the stream, you are compacting the stream and all the topic-partitions.

Log compaction is used for the following purposes:

- Application recovery time - Since log compaction retains the last known value, it is a full snapshot of the latest records. It is useful for restoring state after a crash or system failure.
- Storage space - This becomes noticeable when there is a high volume of messages.

Compaction Process

Log compaction is implemented by running a compaction process in the background that identifies duplicates, determines whether older messages exist, and purges older messages from the topic-partition.

The following diagram shows an initial message (published to a topic-partition) that is identified by the key-value pair, K1V1. When a subsequent message (K1V2) is published to the topic-partition, based on it's key-value pair, it is identified as a duplicate. The compactor then deletes the older message (K1V1) from the topic-partition.

Log compaction never re-orders messages, just deletes them. Any consumer reading from the start of the log sees at least the final state of all records in the order they were written. In addition, the offset for a message never changes.

Compaction and Gateways

The log compaction process uses a gateway that has an internal index and a compactor. The internal index tracks message key-value pairs. this allows duplicate messages in the topic-partition to be identified. Based on the identification of duplicate messages, the compactor runs the compaction process which purges the older message from the topic-partition. This process results in stream data being compacted.

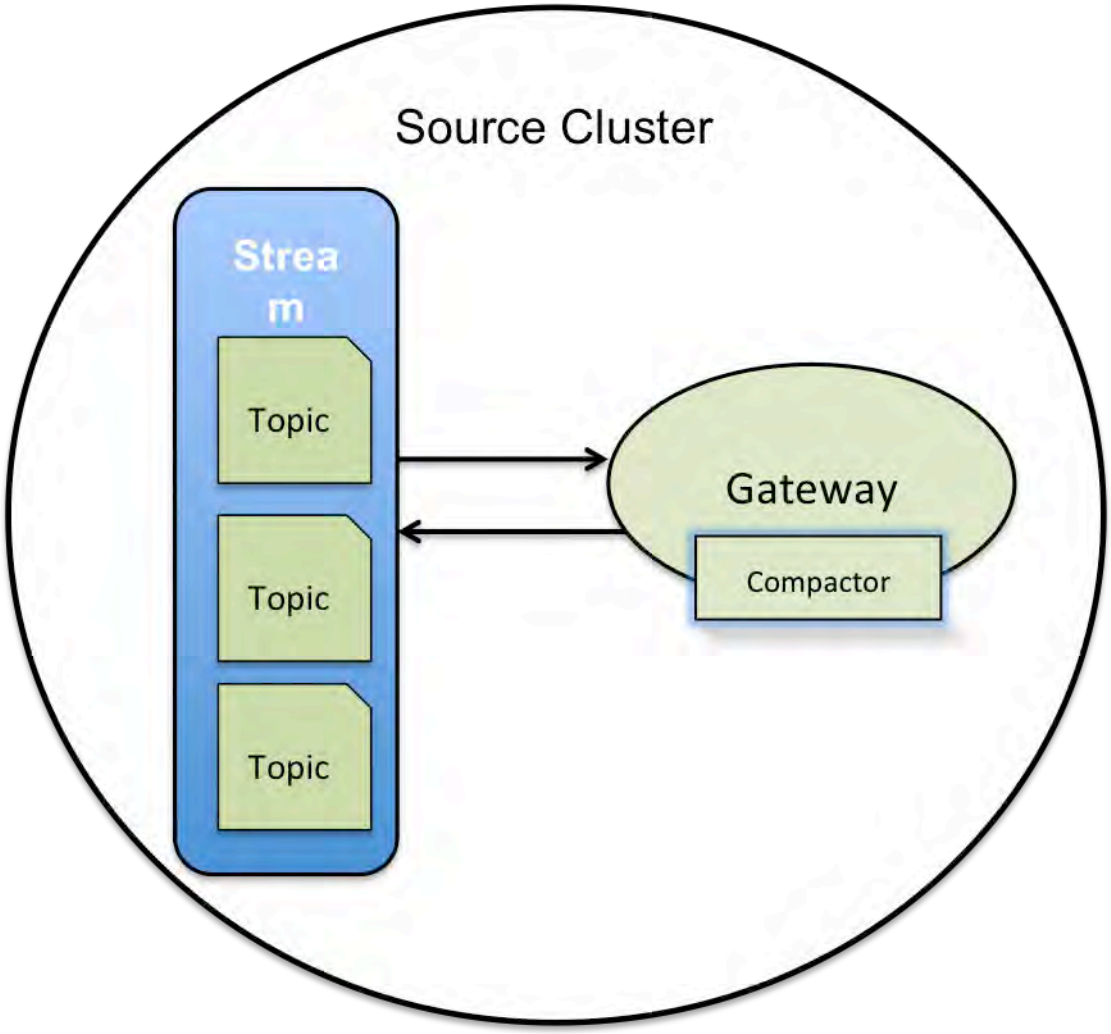


Figure 11: Log compaction with one gateway

The number of gateways impacts the compaction process, in that, increasing the number of gateways on the cluster improves the load distribution of the log compaction activity.

Important: Log compaction requires a gateway to be installed on the same cluster as the MapR stream. See Preparing Clusters for Log Compaction on page 1141 for more information about implementing gateways for this purpose. For example, if you are manually installing or upgrading, you must install a MapR gateway locally.

last-updated: Apr 11, 2022

Stream Replication

When a stream on a source cluster has both log compaction and replication enabled, the replica cluster does not automatically have log compaction enabled. You must explicitly enable log compaction on the replica cluster.

If a replica cluster has been upgraded and the stream data for a source cluster is compacted (that is, one or more messages have been deleted), then the source cluster replicates the compacted data to the replica cluster.

If a replica cluster has ***not*** been upgraded, the source cluster:

- Fails the replication.
- Automatically retries replication with an exponential backoff.
- Resumes replication when the replica cluster has been upgraded.

Note: The error message associated with the failed replication is displayed via the `maprcli stream replica status` command. This error requests that you upgrade the replica cluster.

Performance

Log compaction has a performance impact on other MapR Database and MapR Event Store For Apache Kafka applications running on the system. If log compaction is enabled on a very active stream (with more than 100K messages per second), all MapR Database and MapR Event Store For Apache Kafka applications running on the same cluster could see a drop in their performance (close to 2x).

Note: It is possible that the NODE_ALARM_TINY_BUCKET_FLUSH alarm may occur during high ingestion rates on source clusters with high topic-partition count. Under these circumstances, consider increasing the memory for MapR File System.

For More Information

See the following topics for more information:

- maprcli stream create on page 1758 and stream edit on page 1765
- Preparing Clusters for Log Compaction on page 1141
- MapR Event Store For Apache Kafka Java Applications on page 2754, MapR Event Store For Apache Kafka Java API Library on page 2756, and Enabling Log Compaction on page 2765

Producers

Producers are data-generating applications, such as sensors in automobiles or activity loggers in servers. Producers create messages with the collected data and publish the messages to MapR Event Store For Apache Kafka topics, specifically, to MapR Event Store For Apache Kafka topic-partitions.

Permissions

Before a producer can publish to topics, the user ID running the producer needs these permissions:

- The `writeAce` permission on the volume where the streams are located. For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.
- The `produceperm` permission on the streams where the topics are located. Users with the `adminperm` permission on those streams can grant the `produceperm` permission.

Producing Messages

Producers create messages about the collected data and send the collected data to a MapR Event Store For Apache Kafka producer client library. In addition to the actual message, the producer specifies the topic that the message is intended for and an optional partition ID. The producer client buffers incoming messages and sends them (in batches) to the MapR Event Store For Apache Kafka server.

Note: In case of server failure, the producer client automatically continues to retry sending messages.

Attention: As of MapR 6.1, the MapR Event Store For Apache Kafka API enforces a maximum of 4096 partitions for a topic. That is, when you create an application with the API, the maximum number of partitions is 4096. If you previously created an application with MapR Event Store For Apache Kafka 6.0.1 API (or older) and you have upgraded, the original number of partitions can be used. For example, if you were using more than 4096 partitions in MapR 6.0.1 or earlier, you will be able to continue with the same number of partitions after upgrading.

Event-time Timestamp

As of MapR 6.0.1, MapR Event Store For Apache Kafka supports an event-time timestamp. The timestamp type can be either `createtime` (default) or `logappendtime`. See the `maprcli` stream create on page 1758 and stream edit on page 1765 for more information about these parameters.

Tip: Since each message is automatically published into a topic-partition with an event-time timestamp as part of the message record, this allows the Consumer application to seek records based on the timestamp.

Idempotent (exactly once) Producers

An "exactly-once" message delivery semantic produces messages without duplication. Each message is delivered once and only once. Exactly-once is insured by uniquely identifying a group of messages that are atomically persisted. Exactly-once message delivery is set with the producer idempotence option. See Modes of Publishing on page 645 for more information.

The following failure scenarios are addressed with idempotence:

- The stream processor might take input from multiple source topics and the ordering across these source topics is not deterministic across multiple runs. So if you re-run your stream processor that takes input from multiple source topics, it might produce different results.
- The stream processor might produce output to multiple destination topics. If the producer cannot do an atomic write across multiple topics, then the producer output can be incorrect if writes to some (but not all) partitions fail.
- The stream processor might aggregate or join data across multiple inputs. If one of the instances of the stream processor fails, then you need to be able to rollback the state materialized by that instance of the stream processor. On restarting the instance, you also need to be able to resume processing and recreate its state.
- The stream processor might look up enriching information in an external database or by calling out to a service that is updated out of band. By depending on an external service, the stream processor can be fundamentally non-deterministic. For example, if the external service changes its internal state between two runs of the stream processor, it can lead to incorrect results downstream.

For More Information

For more information about creating and editing streams or topics:

- maprcli

- See maprcli stream create on page 1758 for information about creating streams.
- See maprcli stream edit on page 1765 for information about editing streams.
- See maprcli stream info on page 1768 for information about streams.
- See maprcli stream topic create on page 1781 for information about creating topics.
- See maprcli stream topic edit on page 1784 for information about modifying topics.
- See maprcli stream topic info on page 1785 for information about topic data.

- MapR Event Store For Apache Kafka Java API
  - See the MapR Event Store For Apache Kafka Java API Library on page 2756 for the methods used to create and edit streams and to create and edit topics.

How Messages are Published
To publish a message, a producer sends a record to the producer client library, which batches the records before sending them to the server.

The producer client library sends the records to the server when any of the following conditions are met:

- The producer client library has batched enough messages to make an efficient remote procedure call (RPC) to the server.
- A message has been queued for the amount of time that is specified for the `streams.buffer.max.time.ms` configuration parameter.

  For the Java client, the default interval for flushes is 3000 milliseconds. For clients based on librdkafka (for example, C, Python, and C#), the default interval for flushes is 0 (zero) milliseconds.
- The producer client library has batched messages beyond the value of the `buffer.memory` configuration parameter.
- The application explicitly flushes messages.

> **Tip:** The default number of threads used for flushing messages is 64. In most cases, this number provides excellent performance. However, you can adjust this number by setting a value for the `fs.mapr.threads` parameter in the `core-site.xml` file on your client node.

Modes of Publishing
Describes different modes of publishing.

When publishing a message, a producer sends a record to the producer client library. The producer client library batches messages into multiple publish requests which are sent to the MapR Event Store For Apache Kafka server.

At Least Once

The default message delivery semantics is "at-least-one". At-least-once means that the message delivery guarantees that a message is published at least once to the MapR Event Store For Apache Kafka server. Messages are never lost but may be re-delivered.

Exactly Once

An "exactly once" message delivery semantics produces messages without duplication. Each message is delivered once and only once. Exactly once is insured by uniquely identifying a group of messages that are atomically persisted. Exactly once message delivery is set with the producer `idempotence` option.

Note: Exactly-once message deliver semantics is enabled by setting the producer configurable option, `enable.idempotence` to **true**. By supporting an idempotent producer, retries no longer introduce duplicates. See Enabling an Idempotent Producer on page 2766 for more information.

The following unique identifiers are associated with each message:

- Producer ID - A unique identifier is generated internally for each client and group of messages that are atomically persisted.

  As a minimum, the ID is a unique ID for a given stream-topic-partition. Producer IDs expire if a producer ID is inactive for a period of time. The default Producer ID expiration is 7 days. At that point, a new Producer ID is requested once the Producer ID is expired. To change the expiration date, see the `pidexpirysecs` parameter in `maprcli` stream create on page 1758 and stream edit on page 1765 for more information.

- Sequence Number - A number that is monotonically incremented on every produced group of messages for the given Producer ID, assigned when received, and generated internally.

Note: If the producer `idempotence` option, is not set to **true**, then "at least once" message delivery semantics applies.

If the client resends a message after the producer ID has expired, then `UnknownProducerIdException` is thrown.

For example:

- If message1 from clientA is sent to a stream-topic-partition0 and 7 days go by, the Producer ID expires.
- Then, if clientA sends another message that has the same data to the same stream-topic-partition (stream-topic-partition0), then `UnknownProducerIdException` is thrown because the Producer ID has expired..

Note: With the alternative "at least once" message delivery, in some failure scenarios, a message can be produced more than once for a single send call. Common reasons for message duplication include network error or server failure. For example, if a network error occurs and the message has been processed and persisted by the server, if the client re-tries sending a message to a server node, then the result could be duplicate messages in the system.

Server Acknowledgements

By default, publishing requests for messages are sent without waiting for acknowledgement (ack) from the MapR Event Store For Apache Kafka server.

The acknowledgement behavior is determined by the producer configuration parameter `streams.parallel.flushers.per.partition`, which defaults to **true**.

With an "at-least-once" message delivery, in some failure scenarios, a message can be produced more than once for a single send call. A common reason for message duplication is when a network error occurs, a client may retry sending a message to a server node. If the network error occurs after the message is processed and persisted by the server, it can lead to duplicate messages in the system.

Publishing without Ack	When publishing without ack (default), it is possible for messages to be published to the partitions out of order due to the presence of multiple network interface controllers, network errors, or retries. For example, suppose a producer is sending messages that are specifically for Partition 1. The producer client library buffers the messages and sends a batch to Partition 1. Meanwhile, the producer keeps sending messages for Partition 1 and the client

	continues to buffer them. The next time the producer client library has enough messages for Partition 1, the client sends another batch, irresepctive of whether or notMapR Event Store For Apache Kafka server has acknowledged the previous batch.
Publishing with Ack	If you always want messages to arrive to partitions in the order in which they were sent, set the configuration parameter `streams.parallel.flushers.per.partition` to **false**. This causes the producer client library to wait for ack (acknowledgements) from the MapR Event Store For Apache Kafka server before sending subsequent publish requests.

How Partitions are Chosen for Messages

Since the number of partitions in a topic can change over time, producers regularly refresh the information that they have about the topics that they know. This refresh interval is controlled by the `metadata.max.age.ms` configuration parameter.

Partitions of a topic are identified by their index number. For example, if a topic has four partitions, their IDs are 0, 1, 2, and 3.

Partitions are chosen for a message in the following ways:

- If the producer specifies a partition ID or if the StreamsPartitioner interface specifies one, the MapR Event Store For Apache Kafka server publishes the message to the partition specified.
- If the producer does not specify a partition ID but provides a key, the MapR Event Store For Apache Kafka server hashes the key and sends the message to the partition that corresponds to the hash.
- If neither a partition ID nor a key is specified, the MapR Event Store For Apache Kafka server randomly chooses an initial partition and sends messages in a sticky round robin fashion. .

  For example, suppose that for topic `traffic_sensors`, the server chooses Partition 1. The server then accumulates enough messages for an RPC of optimal size and sends the batch of messages to Partition 1. The server then does the same with Partition 2, and so on, eventually returning to Partition 1.

Consumers

Consumers are applications that you create such as analytics applications, reporting tools, or enterprise dashboards.

Consumers use the MapR Event Store For Apache Kafka APIs to request messages from the topics in which they are interested. If the server fails, consumer clients automatically retry requests continuously. A consumer client library sends unread messages, from which consumers extract data.

Consumers can run as separate processes on a single machine and as processes on different machines.

Before a consumer can read messages from topics, the user ID running the consumer needs these permissions:

- The `readAce` permission on the volume where the streams are located. For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.
- The `consumeperm` permission on the streams where the topics are located. Users with the `adminperm` permission on those streams can grant the `consumeperm` permission.

Subscriptions

Consumers subscribe to topics. When a consumer subscribes to a topic or partition, it means that the consumer wants to receive messages from that topic or partition. For example, an analytics application might subscribe to the topics `rfids_productA`, `rfids_productB`, and more to track movement of products from factories to distribution centers. A reporting tool might subscribe to the topics `meters_NW`, `meters_SW`, and more to get a report of electricity usage in different geographic regions that a power company services.

A subscription is the list of the topics to which a consumer is subscribed.

Consumer Subscriptions

Consumers subscribe to topics. When a consumer subscribes to a topic or partition, it means that the consumer wants to receive messages from that topic or partition. A subscription is the list of the topics, specific partitions, or both to which a consumer is subscribed.

For example, an analytics application might subscribe to the topics `rfids_productA`, `rfids_productB`, and more to track movement of products from factories to distribution centers. A reporting tool might subscribe to the topics `meters_NW`, `meters_SW`, and more to get a report of electricity usage in different geographic regions that a power company services.

Consumers can subscribe to:

Topics

When a consumer subscribes to a topic, it reads messages from all of the partitions that are in the topic. The exception is when a consumer is part of a consumer group. Consumer groups and this exception are explained in Consumer Groups.

Consumers can subscribe to topics in two ways:

By name

Consumers specify the names of the topics to which they subscribe.

By regular expression

Consumers specify a regular expression and subscribe to all topics with names that match the regular expression.

The ability to use regular expressions is helpful when the `-autocreate` parameter for a stream is set to `true` and producers are allowed to create topics automatically at runtime.

To unsubscribe from topics to which you are subscribed with regular expressions, you must use the same regular expressions.

For example, suppose that you use this regular expression to subscribe to `topic0` and `topic1`:

```
topic[0-1]
```


last-updated: Apr 11, 2022

Next, you add `topic2`, `topic3`, and `topic4` to the subscription, as follows:

```
topic[0-4]
```

Trying subsequently to unsubscribe from, say, `topic0` has no effect. The consumer remains subscribed to it because `topic0` was subscribed to as part of a regular expression.

Trying to unsubscribe from `topic[0-1]` also has no effect because the regular expression `topic[0-4]` was used after `topic[0-1]`, and the latter is a superset of the former.

To unsubscribe from `topic0`, you have to follow these steps:

1. Unsubscribe from `topic[0-4]`. This step unsubscribes you from `topic2`, `topic3`, and `topic4`. You must follow this step because a) this regular expression was used last, and b) because it is a superset of `topic[0-1]`. The order in which regular expressions are used in subscriptions matters. If you were to unsubscribe from `topic[0-1]` first, you would still be subscribed to `topic[0-4]`.
2. Unsubscribe from `topic[0-1]`. This step unsubscribes you from `topic0` and `topic1`.

Partitions

Consumers can subscribe to individual partitions within topics. This is helpful when you want a consumer to read the messages published to a specific partition. For example, a producer might

publish messages for high-priority data to a specific partition for processing by a dedicated consumer.

When a consumer subscribes to individual partitions within a topic, the consumer does not receive messages from any of the other partitions in the topic.

Subscriptions to individual partitions can cause problems in consumer groups, as explained in the section Consumer Groups.

Consuming Messages
Describes the process by which consumers consume messages.

Consumers request the MapR Event Store For Apache Kafka consumer client library to check whether any new messages have been published in the topics or partitions to which they are subscribed, or the partitions that they are assigned. Consumers can do this at any time.

If a minimum number of bytes worth of messages is waiting across a consumer's subscription, MapR Event Store For Apache Kafka sends those messages to the consumer, up to a maximum number of bytes. You can configure this minimum and maximum in the configuration parameters for each consumer.

The MapR Event Store For Apache Kafka consumer client library sends the consumer messages that have been published by producers but not yet flushed to disk. If a consumer is able to consume data at the rate at which a producer publishes messages, the consumer client library continuously sends messages to consumers from its memory, increasing the speed of throughput from producer to consumer.

Time-based Consumption

As of MapR 6.0.1, MapR Event Store For Apache Kafka supports the consumption of messages based on the message's timestamp. When a consumer wants to search for messages based on a timestamp, the consumer provides the topic-partition and the timestamp, and then, MapR Event Store For Apache Kafka locates the message and returns the offset for that message. The returned message offset corresponds to the *earliest* message in a topic-partition whose timestamp is *equal to* or *greater than* (>=) the consumer-provided timestamp.

For example, with the following topic-partition, if your consumer-provided timestamp is 1522195205, then **offset 1** would be returned because it is the *earliest* message with a timestamp that is greater than or equal to the consumer-provided timestamp. In this case, greater than (>).

```
topic:partition0
        offset 0: 152219500
        offset 1: 152219510
        offset 2: 152219505
        offset 3: 152219515
```

> **Tip:** The consumer-provided timestamp and the returned message offset is in seconds since a Epoch Unix timestamp is used. In this example, the consumer-provided timestamp is March 26th, 2018 @ 12:00:05am and the message offsets are timestamped March 28th @ 12:00:00, 12:00:010, 12:00:05, and 12:00:15 in that order.

Resources

For information about MapR Event Store For Apache Kafka streams or topics, see:

- maprcli stream info on page 1768 for information about stream data.
- maprcli stream topic info on page 1785 for information about topic data.

Consumer Groups
Group consumers together by setting the same value for the `group.id` configuration parameter when you start each consumer.

For example, if you create three consumers and give each of them the group ID `clickstream_consumers`, together these consumers form the consumer group `clickstream_consumers`. MapR Event Store For Apache Kafka does not generate IDs for consumer groups. You can create IDs that make sense for your purposes. You specify the group ID by using the `group.id` configuration parameter when you create a consumer. IDs are strings that can be up to 2457 bytes long.

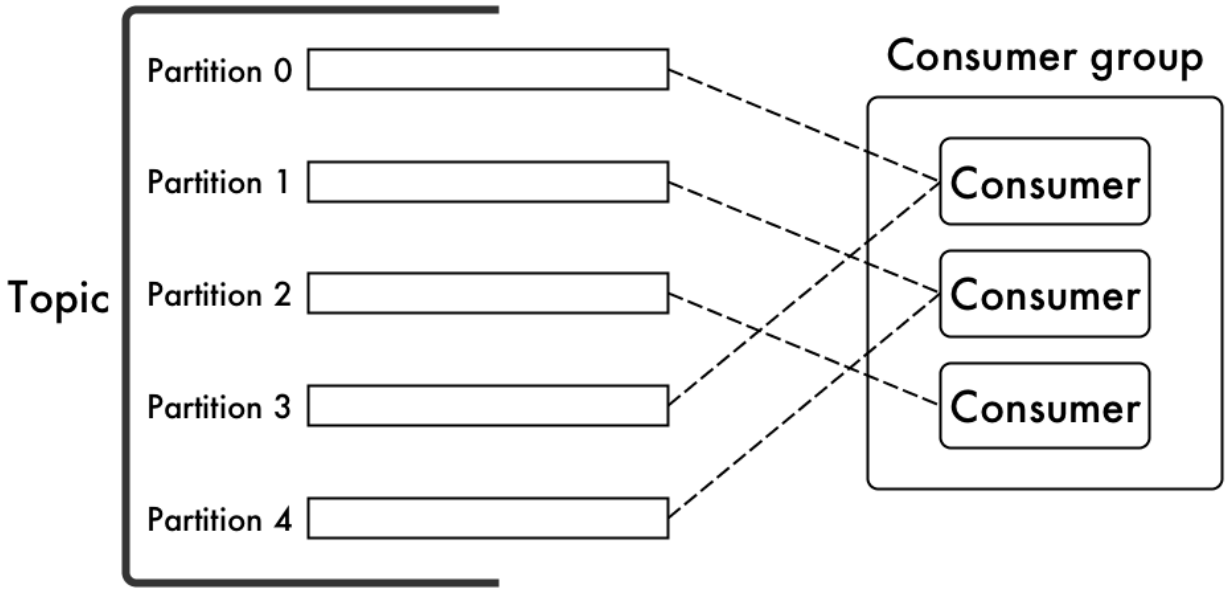
You can even create a consumer group that consists of only one consumer. In such a case, the unique ID that identifies the group would be shared with no other consumers.

The following are the benefits to creating consumer groups:

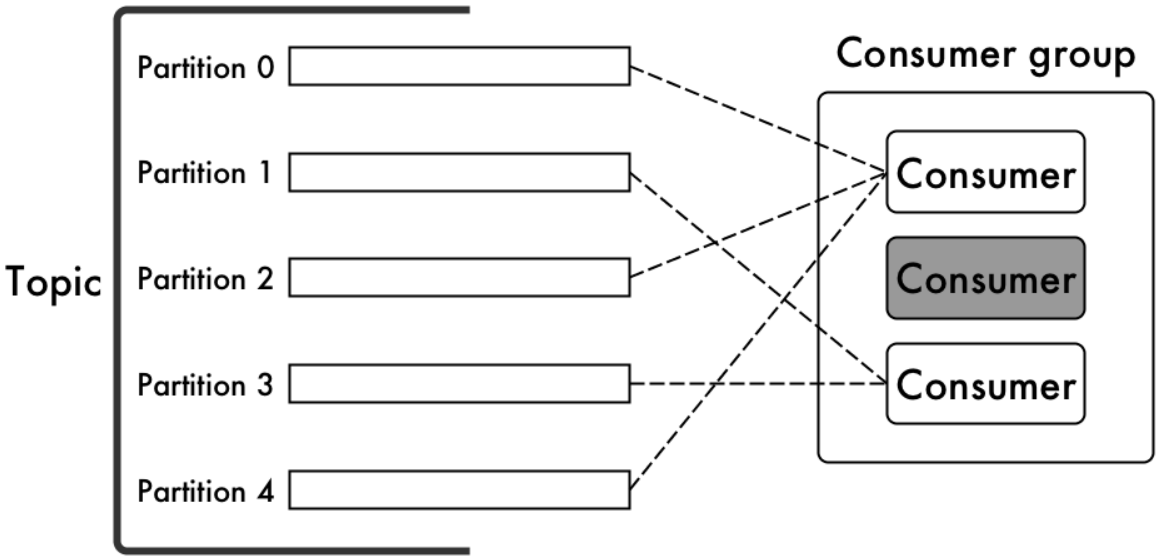
Parallelism when Consuming Messages
For parallelism when reading messages from topics, you can create consumer groups. These groups consist of consumers that are associated with an ID that you set for each of the participating consumers with the `group.id` configuration parameter. The partitions in each topic to which all of the consumers are subscribed, are assigned dynamically to the consumers in round-robin fashion.

These groups consist of consumers that are associated with an ID that you set for each of the participating consumers with the `group.id` configuration parameter. The partitions in each topic to which all of the consumers are subscribed, are assigned dynamically to the consumers in round-robin fashion.

For example, suppose that there are three consumers in a group and each consumer is subscribed to the same topic. There are five partitions in the topic. MapR Event Store For Apache Kafka assigns each partition to a consumer, with two consumers both being assigned two partitions.



If one of the consumers goes offline, the partitions are reassigned dynamically among the remaining consumers in the group.



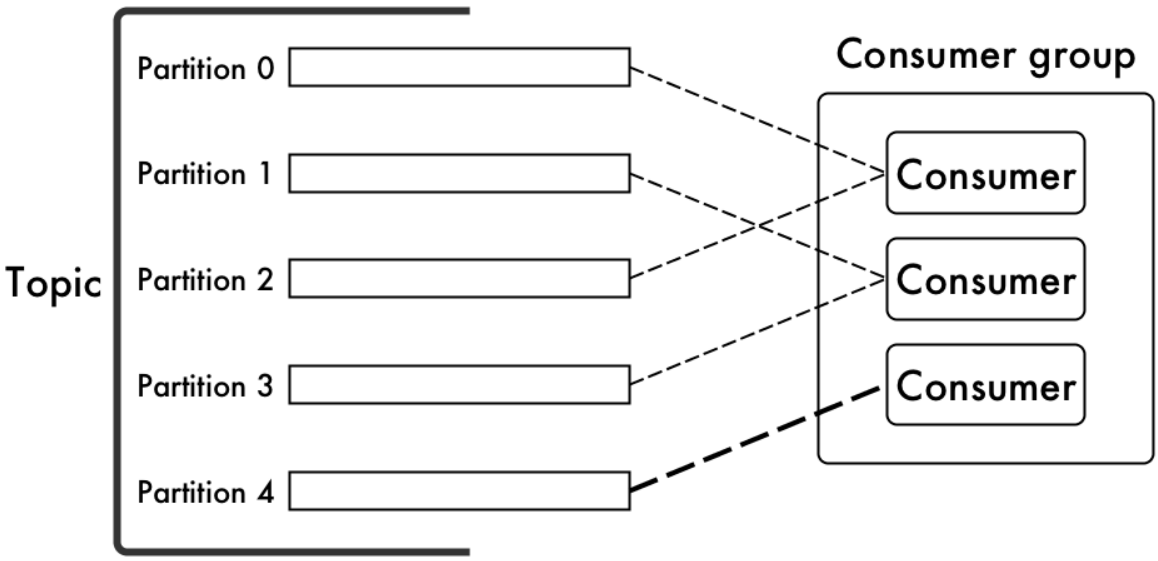
If the offline consumer comes back online or a different consumer is added to the group, again the partitions are redistributed among the consumers in the group.

This parallelism and dynamic reassignment is possible only when none of the consumers in a consumer group subscribe to individual partitions.

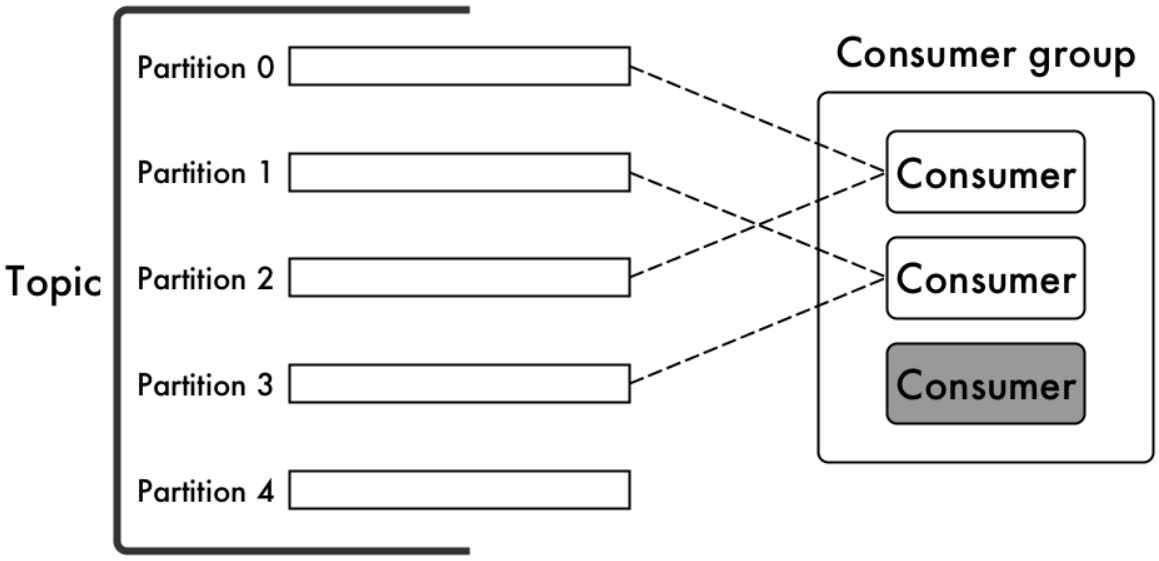
For example, suppose that from three consumers in a consumer group:

- Two subscribe to the same topic.
- One subscribes to a single partition within that topic.

If the topic has five partitions, MapR Event Store For Apache Kafka assigns four of them via round robin to two of the consumers. Only the remaining partition is read from the third consumer.



If that third consumer fails, MapR Event Store For Apache Kafka does not reassign its partition to either of the other consumers.

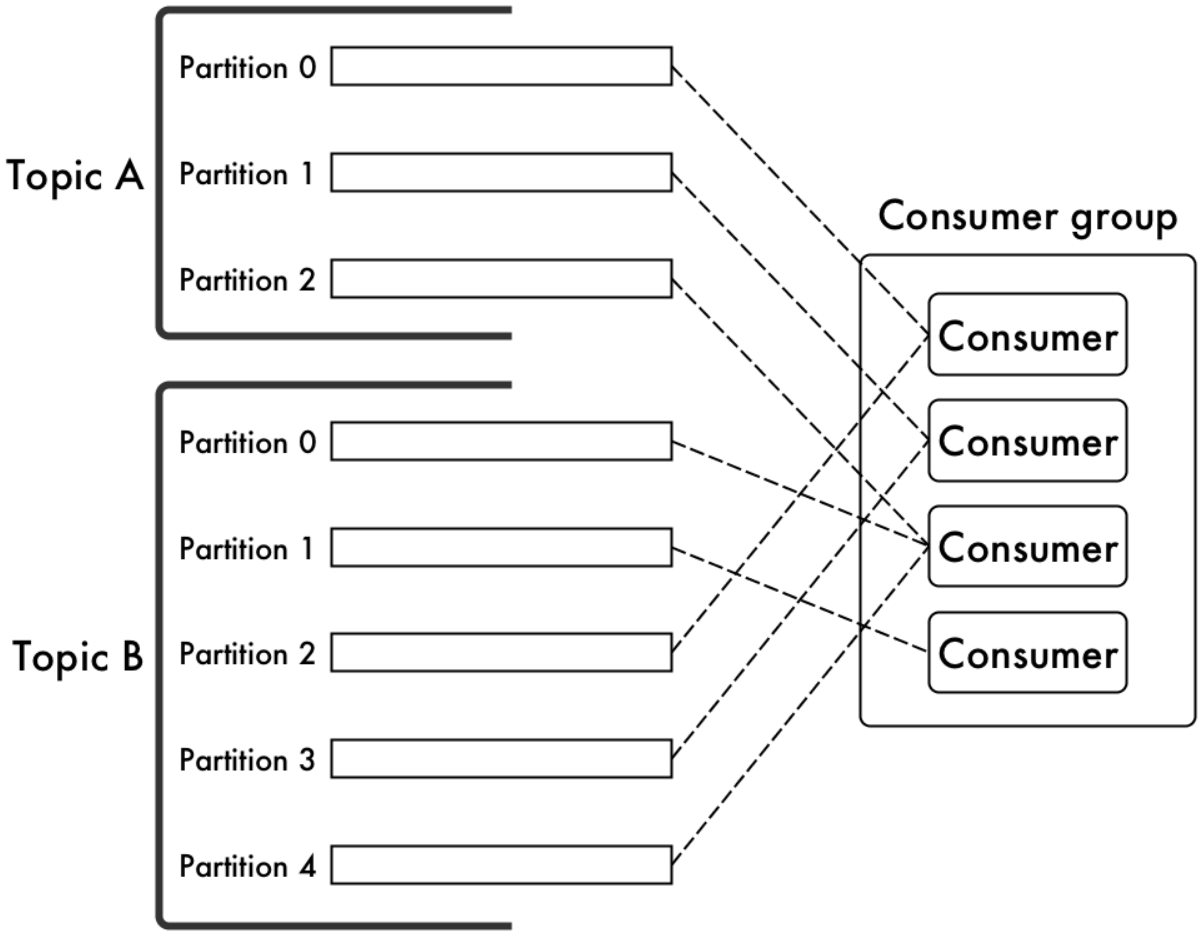


Now that you understand how partitions are assigned when the number of partitions is equal to or greater than the number of consumers in a consumer group, you might be wondering what happens if the number of partitions in a topic is less than the number of consumers in a consumer group. The answer is simply that one or more consumers in the consumer group will not be assigned any partitions from the topic.

That does not necessarily mean those consumers will be idle. There could be other topics to which the consumer group is subscribed, and those consumers could be assigned partitions from those other topics.

For example, in this diagram there is a consumer group with four consumers. Topic A has only three partitions, and those are assigned to the first three consumers shown in the group. However, the fourth consumer is not idle. The consumer group also subscribes to Topic B, which has more partitions than there are consumers. Each of the consumers in the group is assigned at least one partition from Topic B.

last-updated: Apr 11, 2022



Moreover, if a consumer that is assigned a partition from Topic A happens to fail, its partition will be reassigned to the fourth consumer.

Saving Cursor Position
The MapR Event Store For Apache Kafka server uses cursors to keep track of the messages that consumers in consumer groups have read.

There is one cursor per partition per consumer group. There are two kinds of cursors: read cursors and committed cursors.

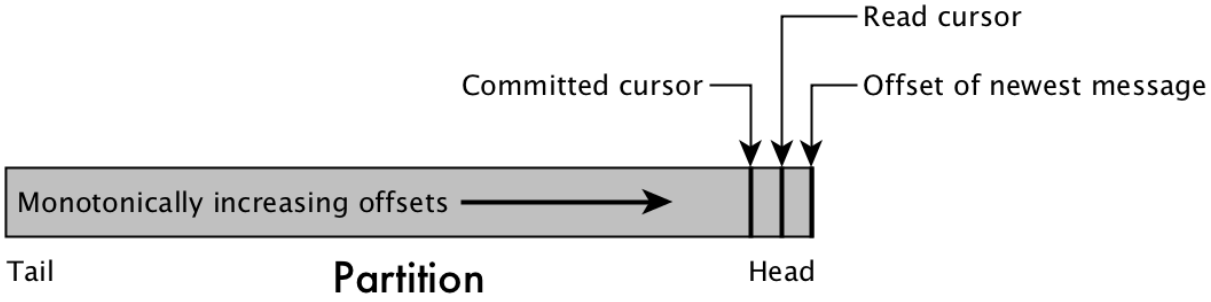


Figure 12: A topic partition and the cursors of a consumer group

A consumer's read cursor is the offset of the most recent message that MapR Event Store For Apache Kafka has sent to a consumer from a partition.

Consumers that are part of a consumer group can save the current position of their read cursor. Consumers can do this either automatically or manually. The saved cursor is called a committed cursor because it indicates that the consumer has processed all messages in a partition up to and including the one with this offset.

There are two benefits to committing cursors:

Failover on consumer failure	One benefit is that if a consumer fails and MapR Event Store For Apache Kafka reassigns the consumer's partitions to other consumers in a group, those consumers can start reading from the next offset after the committed cursor in each of those partitions.
Failover on cluster failure	When you backup a stream by replicating it to another cluster, committed cursors are also replicated. If the main cluster fails, consumers that are redirected to the standby copy of a stream can start reading from the next offset after committed cursors.
Read cursors	A consumer's read cursor is the offset of the most recent message that MapR Event Store For Apache Kafka has sent to a consumer from a partition.
Committed cursors	Consumers that are part of a consumer group can save the current position of their read cursor. Consumers can do this either automatically or manually. The saved cursor is called a committed cursor because it indicates that the consumer has processed all messages in a partition up to and including the one with this offset.

How often a consumer should commit depends on how much read duplication you are willing to tolerate. The more often a consumer commits, the less read duplication with which the consumer must contend.

The length of time since the failed consumer last committed determines (together with the rate at which messages are published to its partitions) how many messages are read a second time. For example, suppose that the auto-commit interval is five seconds. A consumer saves its commit cursor and then fails after three seconds. During those three seconds, the consumer's read cursor has continued to move through the messages. When its partitions are reassigned to other consumers in the group, those consumers will read three seconds of messages that the failed consumer already read.

There are two ways of committing cursors:

Automatic commits	The MapR Event Store For Apache Kafka server commits the cursors for a consumer that is in a consumer group based on the value of the `enable.auto.commit` configuration parameter. Set this parameter to `true` to enable auto-commit. The default value is `true`. The `auto.commit.interval.ms` configuration parameter determines the frequency of the commits in milliseconds. The default is value is 1000.
Manual commits	The Java API provides a method of committing cursors manually.

Consumer Failure and Recovery

When a consumer that is not associated with a consumer-group ID recovers from failure and comes back online, it can either start reading its partitions from the earliest offsets or from the latest offset. This choice is determined by the `auto.offset.reset` configuration parameter.

last-updated: Apr 11, 2022

If the consumer reads from the earliest offset in a partition, which is the offset of the message that has been in the partition longest without being deleted because of the expiration of the time-to-live interval for the stream, it might re-read a large number of messages before reading messages that were published after it failed.

If the consumer reads from the latest offset in a partition, which is the offset of the most current message at the time the consumer requests new messages from MapR Event Store For Apache Kafka, the consumer starts off up-to-date, but skips over the messages between its time of failure and the current time.

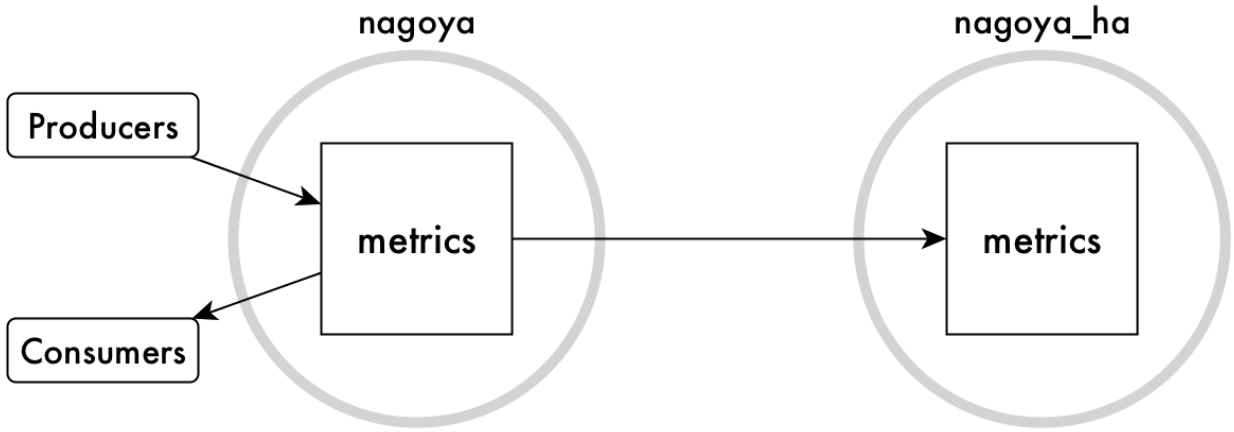
Stream Replication

You can replicate streams to other MapR clusters worldwide, or to other streams within a MapR cluster.

There are many scenarios in which replicating MapR Event Store For Apache Kafka streams can be useful.

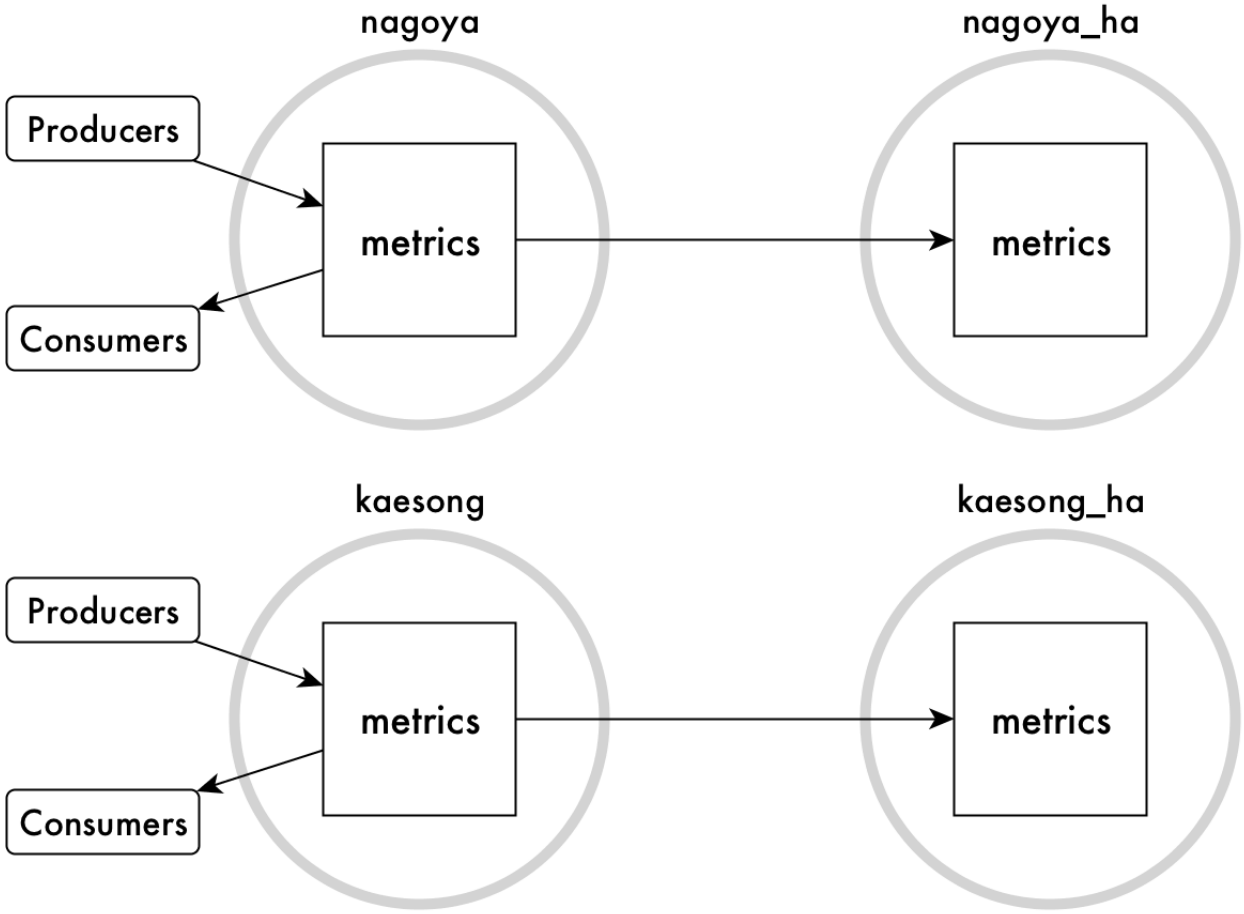
Basic Primary-Secondary Replication

For example, suppose that your company has a factory in Nagoya, and sensors in the equipment track different metrics. The sensors are producers publishing messages to a stream named `metrics`. The applications that use the collected metrics would read the messages from the stream, playing the role of consumers. With replication, the factory could create a stream in the `nagoya` cluster and maintain a backup of the stream in the `nagoya_ha` cluster.



This type of replication is called *basic primary-secondary replication* because replication is in one direction only. The metrics stream in the `nagoya_ha` cluster is considered to be a *replica*. The original metrics stream is considered to be the *upstream source* for the replica. This type of replication is simple to set up with the command `maprcli stream replica autosetup`.

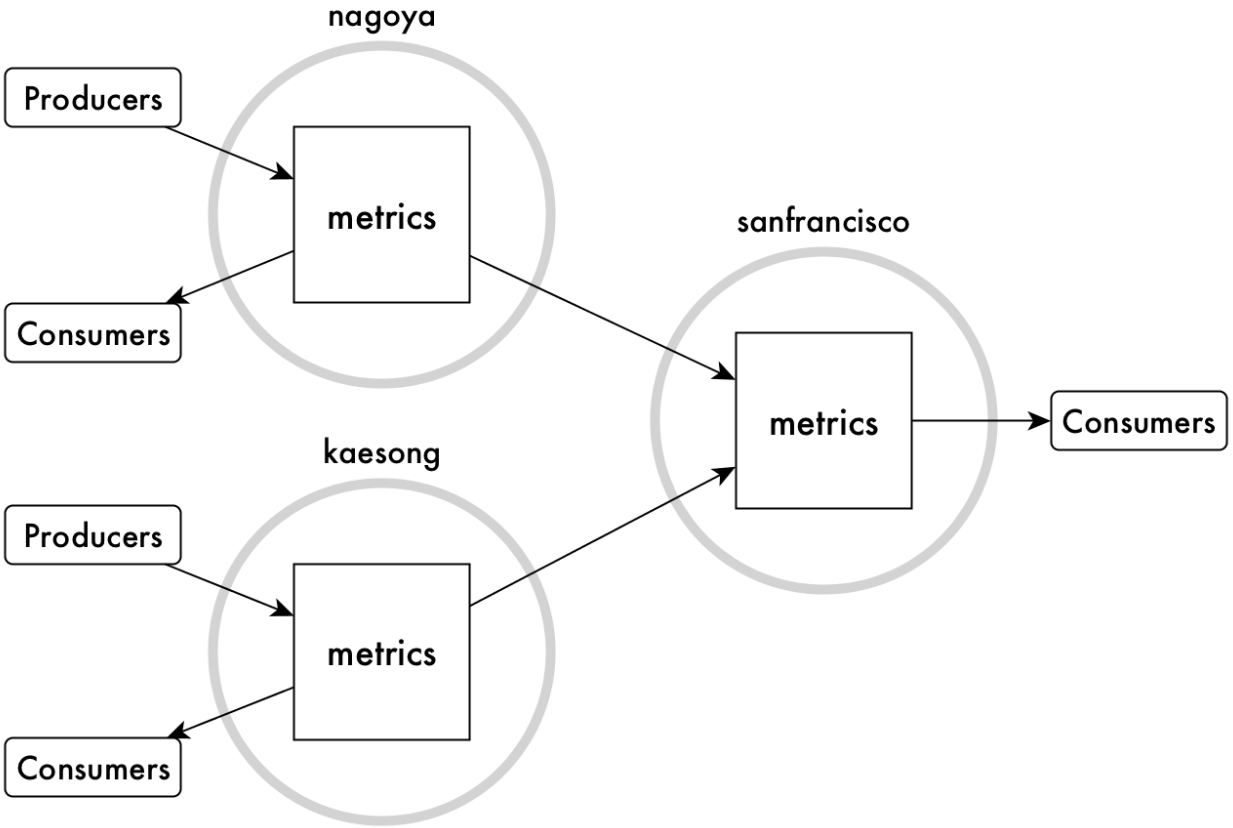
Suppose further that your company also has a factory in Kaesong that collects metrics from its equipment, analyzes the data, and replicates its own `metrics` streams to a backup.



Many-to-One Replication

Your company's headquarters are in San Francisco and you want data analysts there to analyze all data company-wide. You can replicate the two `metrics` streams that are in the your factories to the `metrics` stream in the `sanfrancisco` cluster. In this scenario, the replica is the `metrics` stream in the `sanfrancisco` cluster. This replica has two upstream sources: the `metrics` streams that are replicated from the two factories.

last-updated: Apr 11, 2022

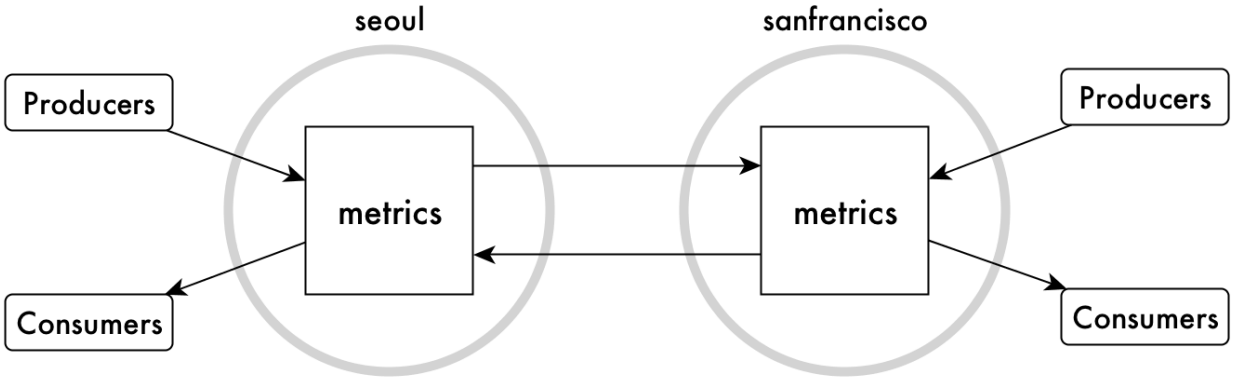


This type of replication, called *many-to-one replication*, requires that the topics in each stream have unique names, so that message offsets do not conflict. For example, suppose both factories have an assembly line named Line 2 and the topic in each factory's stream for collecting metrics from this line is named `line_2`. At some point, the Nagoya factory and the Kaesong factory both replicate messages that use the same offsets. Since offsets are replicated together with messages, messages can be overwritten in this case.

To avoid this type of problem, the sensors for Line 2 in the Nagoya factory might publish to a topic named `line_2_nagoya`, the sensors for Line 2 in the Kaesong factory might publish to a topic named `line_2_kaesong`, and so on. The consolidated stream in San Francisco would contain the topics `line_2_nagoya` and `line_2_kaesong`.

Multi-Master Replication

Another kind of of replication that can be useful is *multi-master replication*. You can use it when you need two streams, both to send updates to and receive updates from the other stream. Each stream is a replica and an upstream source. MapR Event Store For Apache Kafka keeps both streams synchronized with each other. This type of replication is also simple to set up with the command `maprcli stream replica autosetup`.



As with many-to-one replication, the names of the topics in each stream must be unique across both streams, so that offsets for messages do not conflict.

Updates are applied to replica streams by MapR gateways. See Gateways and Stream Replication on page 662 for more information.

Modes of Stream Replication
You can replicate streams in one of two replication modes. You specify the mode per source-replica pair.

Asynchronous replication

In this replication mode, MapR Event Store For Apache Kafka confirms to producers that messages are published after the messages are placed in partitions. Messages are replicated in the background. Therefore, the latency of message publishing is not affected by the time required for the network round trip between the source cluster and the destination cluster.

This type of replication is well-suited for clusters that are geographically separated in wide-area networks.

Asynchronous replication is the default replication mode.

Synchronous replication

In this replication mode, MapR Event Store For Apache Kafka confirms to producers that messages have been placed in partitions only after the messages are sent to a gateway in the destination cluster.

Due to the confirmations that MapR Event Store For Apache Kafka receives on source clusters, synchronous replication is especially well-suited for creating a backup of your data for disaster recovery.

When the latency of a replication stream is high, MapR Event Store For Apache Kafka switches to asynchronous replication temporarily so that producers are not blocked indefinitely. After the latency is sufficiently reduced, MapR Event Store For Apache Kafka switches back to synchronous replication.

The same switching from synchronous to asynchronous replication occurs if all gateways fail. MapR Event Store For Apache Kafka does not resume synchronous replication until a new gateway is established or at least one of the failed gateways is restarted.

Replica Autosetup for Streams
The option to automatically set up stream replication, also known as replica autosetup, performs the steps to set up and start the replication of streams. The replica autosetup option is available through the Control System and the CLI.

In general, replica autosetup performs the following steps to set up replication:

1. Creates a stream in the destination cluster.
2. Declares the new stream to be a replica of the source stream and ensures that replication does not begin immediately after the next step.

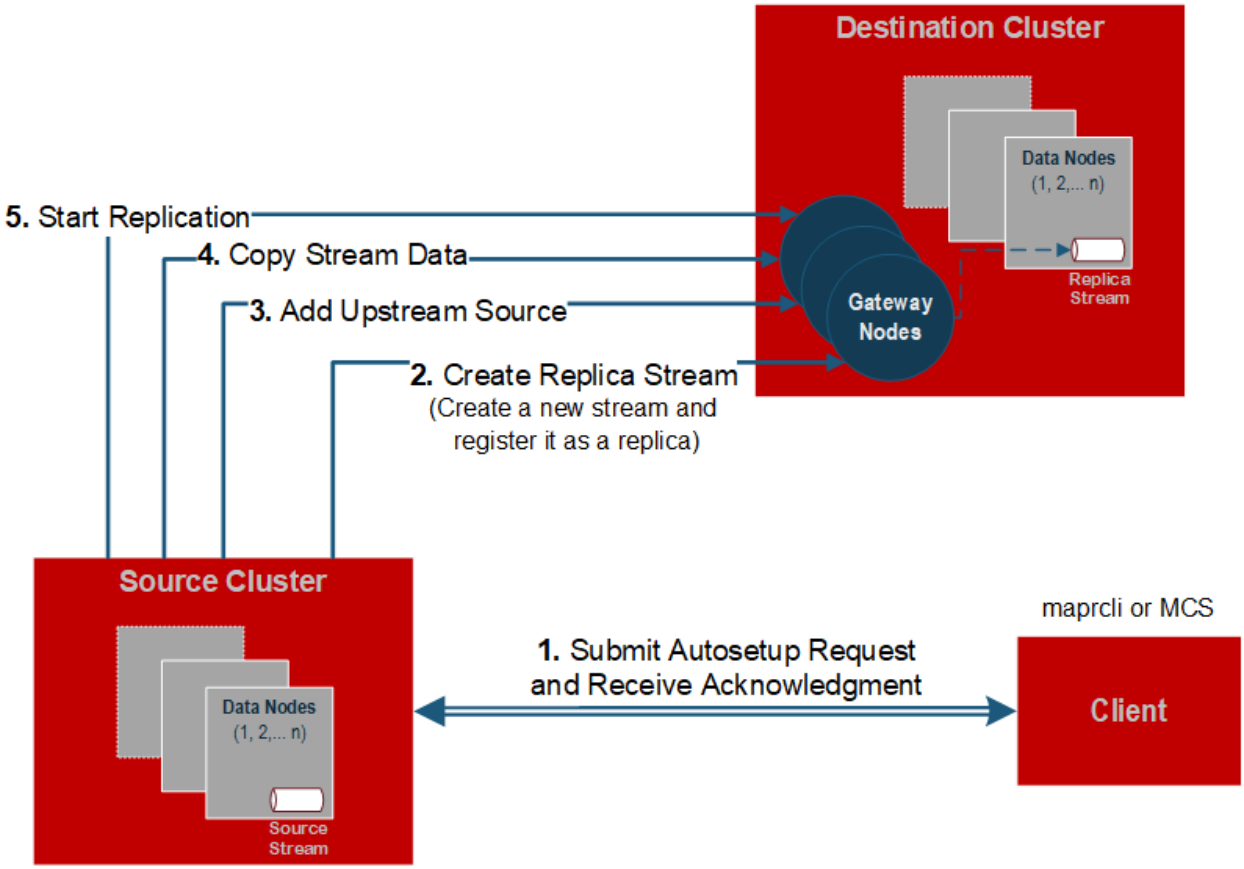
3. Declares the source stream as the original of the replica stream.
4. Loads a copy of the source stream into the replica.
5. For multi-master replication, replica autosetup declares the source stream to be a replica of the new stream and then declares the new stream to be an upstream source for the source stream.
6. Clears the paused replication state to start replication.

By default, replica autosetup uses the directcopy option. However, based on how you run replica autosetup, you also have the option not to use directcopy.

Replica Autosetup with Directcopy (default)

The directcopy option uses gateways to perform all setup operations including the initial population of data into the replica stream. Directcopy is the default option when you setup stream replication using the Control System or with the `maprcli stream replica autosetup` command.

When a client submits a request to automatically setup stream replication to the cluster, the source cluster acknowledges the request and begins to track the replica autosetup request from start to finish.



If a failure occurs when replica autosetup operations are in progress, the source cluster resumes operations from the point of failure.

Note: To check the replication status of a stream, run the `stream replica list` on page 1775 command. To stop the automatic setup of stream replication, run `stream replica remove` on page 1778, or delete the source or replica stream.

Replica autosetup with directcopy provides the following benefits:

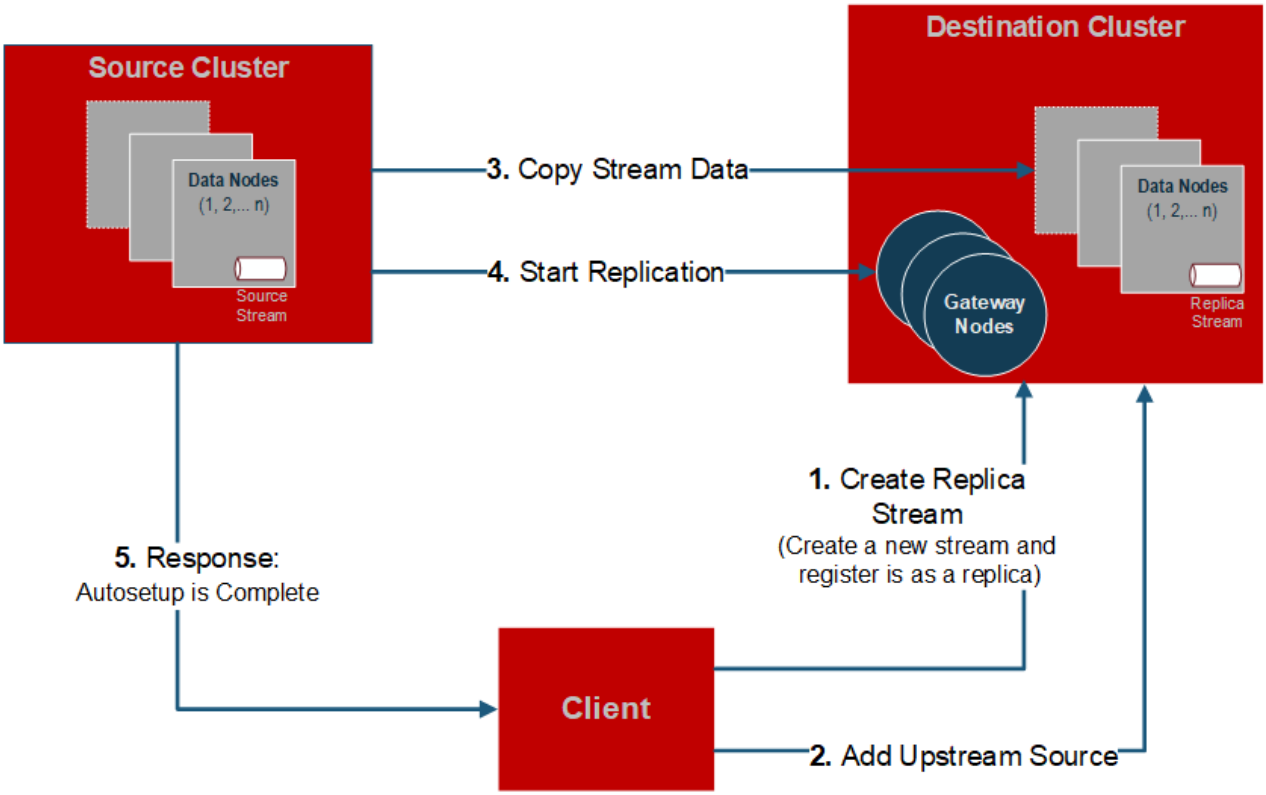
last-updated: Apr 11, 2022

- **Replica autosetup operations do not block the client from submitting additional requests.** When setting up stream replication, the process to copy source data to the replica can be time consuming. The client does not need to wait for the replica autosetup request to complete before submitting another request.
- **Source cluster retries replica autosetup operations in case of failure.** The source cluster keeps track of the replica autosetup progress. This allows the source cluster to resume autosetup operations in the event of an intermittent failure. If you choose to not use directcopy, user intervention is required if a failure occurs.
- **Throttling of copy table operations is done by default.** Throttling prevents the initial copy of data from the source to the replica stream from consuming all cluster resources.

Replica Autosetup without Directcopy (not default)

Without the directcopy option, replica autosetup submits a majority of the replication setup requests through the client and then runs the `mapr copystream` utility to populate the initial table data. To use replica autosetup without the directcopy option, run `maprcli stream replica autosetup` command with the `-directcopy` parameter set to `false`.

Without the directcopy option, once a client submits a replica autosetup request to the cluster, it must wait until the source cluster sends a notification that the autosetup request is complete, before it can submit another request to the cluster. In this case, replica autosetup uses the client connection to submit autosetup operation requests such as create replica, add replica, and add upstream source. Then, to populate the initial table data, it runs `mapr copystream`.



If a failure occurs when replica autosetup operations are in progress, the client hangs and any replica streams that were created during the failed autosetup operations must be manually deleted before you can try to setup replication again.

last-updated: Apr 11, 2022

States of Stream Replication

The replication state indicates when stream replication is in progress and it also displays the status of operations related to replica autosetup with directcopy. The `maprcli stream replica list` command displays the following replication states.

State	Description
REPLICA_STATE_WAIT_TILL_BULKLOAD	Replica autosetup with directcopy has not started because bulkload is in progress on the source table.
REPLICA_STATE_CREATE_SCHEDULE	Replica autosetup with directcopy had scheduled the creation of the replica table.
REPLICA_STATE_COPY_SCHEDULE	Replica autosetup with directcopy has not started the initial copying of source data to the replica because it is waiting for other in-progress copy operations to complete.
REPLICA_STATE_COPY_IN_RECOVER	Replica autosetup with directcopy is resuming the copy of source data to the replica after a connection failure.
REPLICA_STATE_COPY_IN_PROGRESS	Replica autosetup with directcopy is copying the source data to the replica.
REPLICA_STATE_DELETING_CURSORS	Replica autosetup with directcopy is deleting progress cursors since the initial copy of source data to the replica is complete.
REPLICA_STATE_REPLICATING	Replication is in progress.

Security for Stream Replication

Describes where security can be implemented for stream replication.

On clusters	You can replicate between streams that are in secure clusters.
At source streams	The `-adminperm` parameter in the commands `maprcli stream create` and `maprcli stream edit` lets you specify an *ACE* to declare who has administrative permissions on a stream, including permission to replicate the stream. The *ACE*s themselves are copied the first time during auto setup but are not replicated continuously. You need to manage the replicated and source *ACE*s separately after the initial setup.
Across a network	You can send data encrypted or unencrypted when replicating streams by using the `-networkencryption` parameter when you create or edit a replica.
At gateways	Gateways ensure that replicas receive data only from approved source streams.

Gateways and Stream Replication

When replicating streams, MapR Event Store For Apache Kafka replicates messages that are published to a source stream. Gateways are services that receive messages from source streams and publish them in replica streams.

You configure gateways on nodes that are in destination clusters. On source clusters, you list the destination clusters and the gateways that are running on them.

For information on configuring and managing gateways, see:

- Configuring Gateways for Table and Stream Replication on page 1152
- Managing Gateways on page 1154

last-updated: Apr 11, 2022

During replication, MapR Event Store For Apache Kafka sends messages from source streams to the gateways on the destination clusters, where the replicas of those source streams are located. Gateways batch the messages and then apply them to replicas. All messages from a source stream arrive at a replica after having been authenticated at a gateway. Therefore, access control expressions on the replica that control permission to publish messages are irrelevant; gateways have the implicit authority to publish messages to replicas.

MapR Event Store For Apache Kafka distributes messages to a destination cluster's gateways in round-robin fashion. If a gateway is down or unreachable, MapR Event Store For Apache Kafka chooses another gateway. If all of the gateways are down, MapR Event Store For Apache Kafka retries the operation periodically until a gateway comes online.

You must configure gateways in destination clusters. If the destination cluster is remote from the cluster in which a source stream is located, then the gateways must be in the remote cluster. If the destination cluster is the source cluster, meaning that a source stream and its replica are located in a single cluster, then the gateways must be in the local cluster.

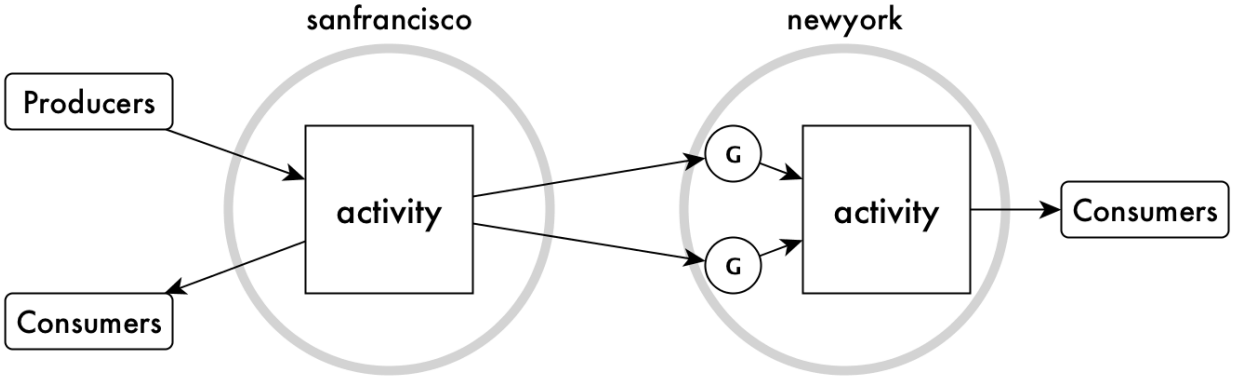
In a Primary-Secondary setup, you cannot have two primary instances with the same topic name replicating to the same secondary instance. It creates a conflict for that topic name. This is similar to Multi-Master replication where you must have separate topic names for Master1 (Cluster1) and Master2 (Cluster2).

For more information about replicating streams, see Stream Replication on page 656.

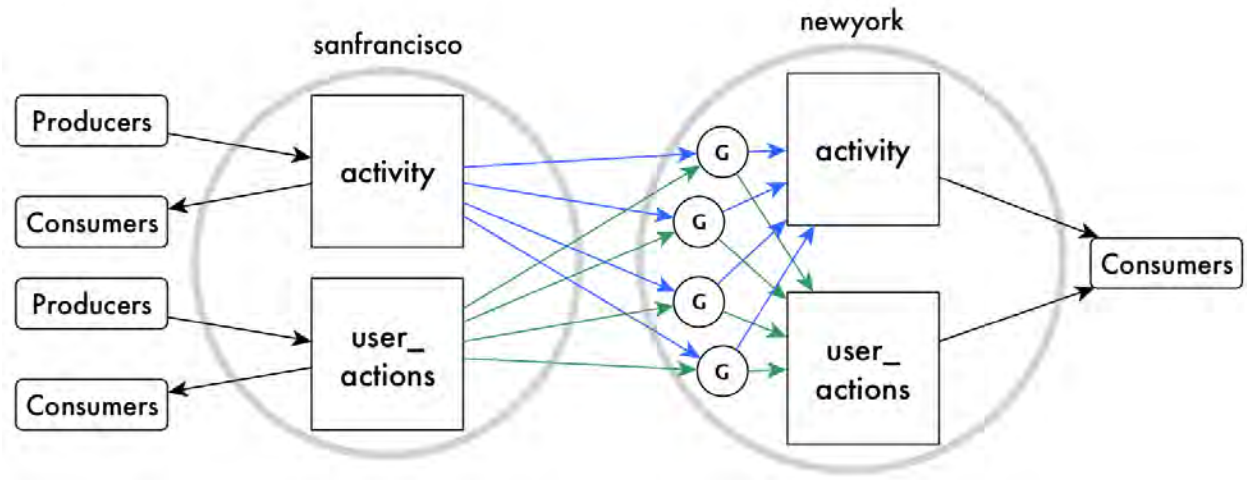
Gateways on nodes in remote destination MapR clusters

In this type of topology, gateways receive messages that are published to source streams, authenticate with the destination cluster on behalf of the source cluster, and publish the messages to the corresponding streams.

This diagram of basic intercluster primary-secondary replication shows messages from the `activity` stream in the cluster `sanfrancisco` being sent to gateways. The gateways then publish the messages to the replica stream that is in the cluster `newyork`.



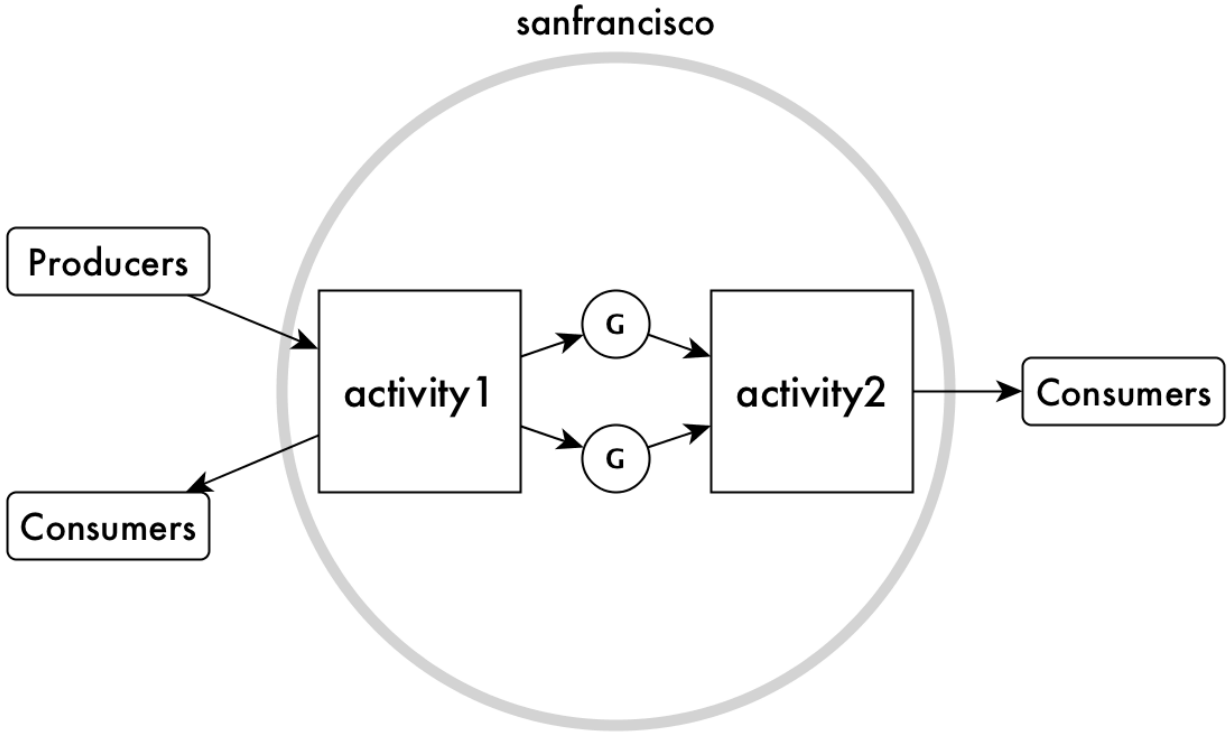
The gateways on a destination cluster are not assigned to particular replicas. They publish messages to all replicas on the destination cluster. For example, in the following diagram, messages from two source streams in the cluster `sanfrancisco` are replicated to two replicas in the cluster `newyork`. There are four gateways. Each gateway receives messages from both source streams, and each gateway applies those messages to the corresponding replicas.



Gateways on nodes within a MapR cluster serving as source and destination

In this type of topology, gateways also receive messages that are published to source streams and publish the streams to the replicas. However, all of this activity takes place within a single MapR cluster.

The following schematic diagram of basic intracluster primary-secondary replication shows messages from the `activity1` stream in the cluster `sanfrancisco` being sent to gateways. The gateways then publish the messages to the stream `activity2`.



Stream Security

The adminperm, copyperm, comsumeperm, produceperm, and topicperm security permissions protect topics in a stream from unauthorized access. In addition, MapR supports user impersonation.

ACE Permissions

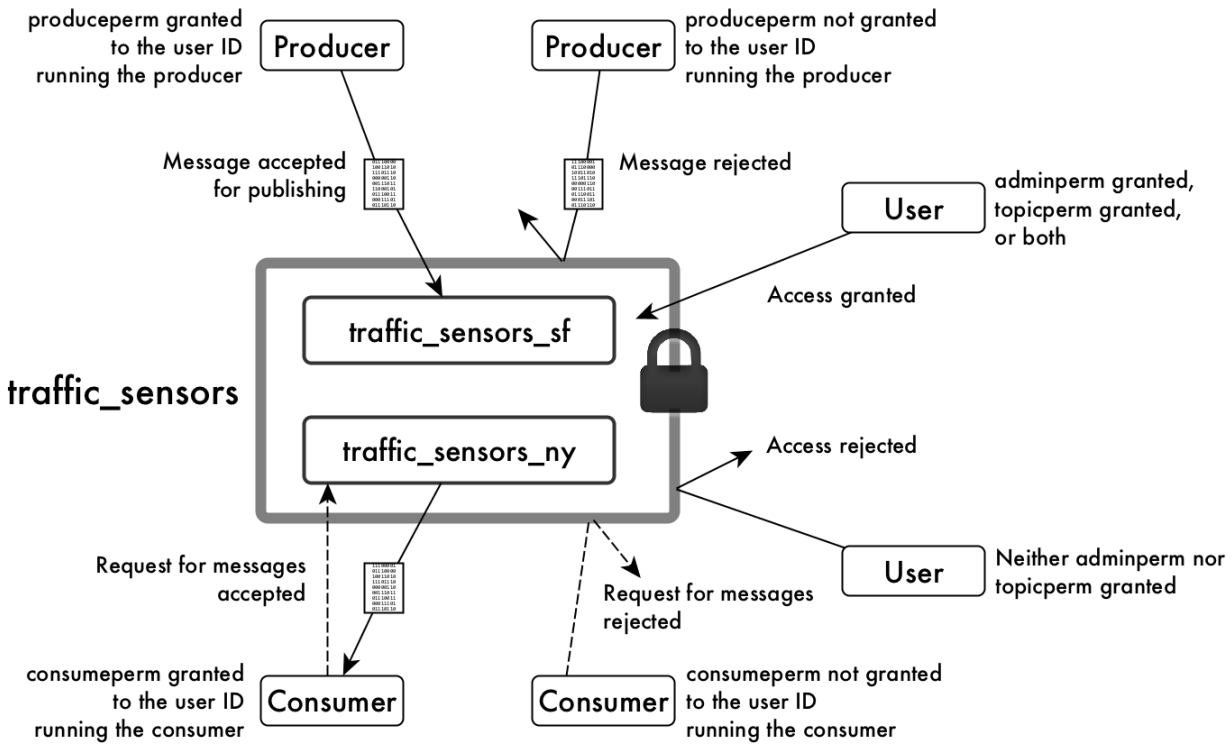
The following *ACE*s are used to protect topics in a stream from unauthorized access. *ACE*s are set when you create or edit a stream.

`adminperm`	Determines which users can modify *ACE*s for a stream, set up replication of a stream, and modify other attributes of a stream. **Important:** By default, only the stream owner can modify this setting; however, a patch is available that changes this behavior. After applying the patch, both the stream owner and the MAPR user can modify this setting. The patch works with MapR Core-6.1.0. To install patches, see Applying a Patch.
`copyperm`	Determines the users who can run the `mapr copystream` and `mapr diffstreams` utilities on the stream. Users with this permission can publish messages to topics in a stream, read messages in topics from a stream, and create or remove topics in a stream. This permission is a combination of the `consumeperm`, `produceperm`, and `topicperm` permissions.
`consumeperm`	Determines the users who can read messages in topics from a stream.
`produceperm`	Determines the users who can publish messages to topics in a stream.
`topicperm`	Determines the users who can create topics in a stream or remove them.

The following example shows the `adminperm`, `consumeperm`, `produceperm`, and `topicperm` permissions on a stream named `traffic_sensors`, which includes the topics `traffic_sensors_sf` and `traffic_sensors_ny`.

Figure 13: How permissions grant or deny access to a stream

last-updated: Apr 11, 2022



For general information about *ACE*s, see ACE Syntax on page 1448.

User Impersonation

MapR Event Store For Apache Kafka supports user impersonation through the Java API. See MapR Event Store For Apache Kafka Java API Library on page 2756 for more information. MapR Event Store For Apache Kafka does not support user impersonation through the C API or Python API.

Kafka REST supports outbound user impersonation. See User Impersonation on page 3871 for more information.

MapR Data Fabric for Kubernetes

This section describes the MapR Data Fabric for Kubernetes, which include the Container Storage Interface (CSI) driver for multiple container-orchestration systems, and the FlexVolume driver for Kubernetes.

The following table describes these features:

MapR CSI Storage Plugin	The CSI Storage Plugin is a volume driver that uses the industry-standard container-storage interface to expose the MapR Platform to workloads on container-orchestration systems.
MapR Data Fabric for Kubernetes FlexVolume Driver	The FlexVolume Driver is a set of Docker containers that provide persistent storage for Kubernetes objects through the MapR File System.

MapR Container Storage Interface (CSI) Storage Plugin Overview

This page describes how the MapR Container Storage Interface (CSI) Storage Plugin can be used to expose the MapR Data Platform to the containerized workload on Kubernetes.

To install or use the MapR Container Storage Interface (CSI) Storage Plugin, see:

last-updated: Apr 11, 2022

- Installing, Uninstalling, and Upgrading the MapR Container Storage Interface (CSI) Storage Plugin on page 228
- Using the MapR Container Storage Interface (CSI) Storage Plugin on page 3104

About the MapR Container Storage Interface (CSI) Storage Plugin

The MapR Container Storage Interface (CSI) Storage Plugin is an industry-standard interface that Container Orchestration systems can use to expose MapR Data Platform to their containerized workloads. Traditionally, storage vendors had either to write and support multiple volume drivers for different Container Orchestration systems or choose not to support Container Orchestration systems. Using CSI, you can use the same volume driver with different Container Orchestration systems. Also, CSI enables the volume plug-ins to be containerized to make it agnostic to the host underneath, which might run other software such as MapR, allowing both Kubernetes and MapR to co-exist on the same node with CSI support.

The MapR CSI driver:

- Allows other software to run on the same node.
- Does not require volume plug-ins to be built into the Kubernetes binaries.
- Does not require direct access to the machine to deploy the volume plug-in.

The CSI Driver for MapR consists of `.yaml` configuration files for installation into Kubernetes. Once installed, a Kubernetes Container Storage Interface (CSI) driver for the file system and a Kubernetes Dynamic Volume Provisioner are available for both static and dynamic provisioning of MapR storage.

The CSI driver uses sidecar containers, which are containers included with the driver for handling Kubernetes events and for communicating with CSI drivers for storage provisioning. Specifically:

- The `csi-provisioner` provisions and creates volumes for the MapR Data Platform.
- The `csi-driver-registrar` registers the driver to the kubelet.
- The `csi-attacher` attaches volumes to the node and mounts the volumes.
- The `livenessprobe` probes the driver for health and readiness.
- The `csi-snapshotter` and `snapshot-controller` provision and create snapshots on the MapR Data Platform.

When you install the CSI driver, it creates a DaemonSet Pod for the CSI node service and StatefulSet Pod for CSI controller service.

Additional Resources

For more information about application containers and Kubernetes, see the following MapR references:

- Blog: Overview of Application Containers
- Blog: Using Kubernetes to Manage Containers and Cluster Resources
- Blog: Containers: Best Practices for Running in Production
- Blog: How to Mount a PersistentVolume for Static Provisioning Using MapR CSI in GKE

Static and Dynamic Volume Provisioning Using MapR Container Storage Interface (CSI) Storage Plugin

Explains static and dynamic volume provisioning using the CSI plugin.

Kubernetes makes a distinction between static and dynamic provisioning of storage.

last-updated: Apr 11, 2022

Static Provisioning

In static provisioning, a MapR administrator first creates MapR volumes (mount points) and then ensures that they are mounted, and a Kubernetes administrator exposes those MapR mount points in Kubernetes through Kubernetes PersistentVolumes. In a typical static-provisioning scenario, a Pod author requests that a Kubernetes admin create a PersistentVolume that references an existing MapR mount point with a dataset that the Pod author is interested in. This PersistentVolume references the CSI driver. The CSI Driver mounts and unmounts MapR mount points for the requesting Pod. In addition, CSI supports the creation of a PersistentVolume directly by creating a PersistentVolumeClaim. The Pod author requests that a Kubernetes admin to create a PersistentVolume that points to the CSI driver and references an existing MapR mount point.

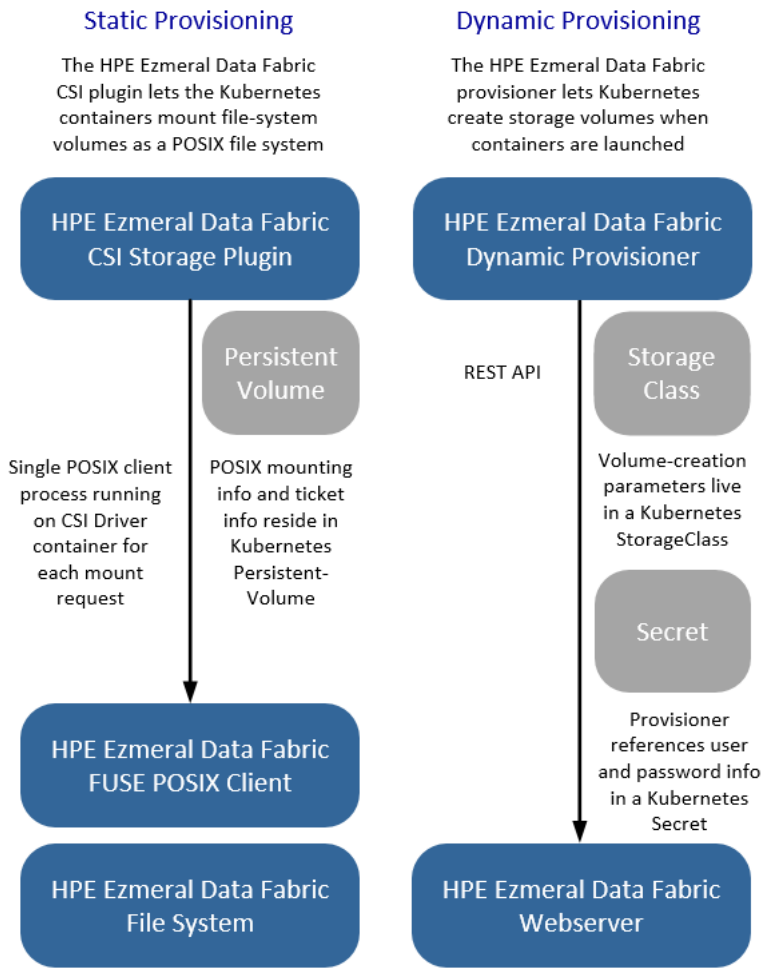
Dynamic Provisioning

In dynamic provisioning, a Kubernetes administrator creates a set of StorageClasses pointing to the CSI provisioner for MapR Data Fabric. Each StorageClass has a predefined set of storage characteristics. Examples of these characteristics include the CLDB hosts, REST server hosts, provisioner secret name and namespace, MapR volume name prefix, MapR volume mount path, and volume advisory quota size. The Pod creator searches the predefined Storage Classes for the one that best matches the creator's requirements. When the Pod references this StorageClass through a PersistentVolumeClaim, the StorageClass calls the CSI Provisioner for MapR Data Fabric to allocate storage for the requesting Pod dynamically and creates the volume.

To leverage the MapR file system with a Kubernetes cluster, you can create a PersistentVolume in Kubernetes.

The following diagram shows the two ways in which the PersistentVolume can be provisioned for the POSIX client. In the case of the Loopback NFS plugin, the Loopback NFS server performs the functions of the POSIX client shown in the diagram:

last-updated: Apr 11, 2022



Static Provisioning Implementation

To accomplish static provisioning, the CSI Driver for MapR Data Fabric for Kubernetes is deployed to all nodes in the Kubernetes cluster via a Kubernetes DaemonSet. The CSI Driver uses the Basic, which is the default, or the optional Platinum POSIX client to mount the MapR filesystem. The information that the POSIX client uses to connect to MapR is contained in a Kubernetes Volume or PersistentVolume. A MapR ticket inside a Secret, referenced by the Kubernetes Volume or PersistentVolume specification, is used by the POSIX client to pass secure data to the filesystem.

Dynamic Provisioning Implementation

To accomplish dynamic provisioning, the CSI provisioner is deployed as a StatefulSet in the Kubernetes cluster.

A Kubernetes Administrator must configure at least one storage class with MapR parameters (for example, mirroring, snapshots, quotas, and other parameters) for use during creation of the MapR volume. The storage class passes MapR administrative credentials to the provisioner through a Kubernetes Secret. Security for the provisioner is handled through role-based access control (RBAC) in Kubernetes.

Related tasks

Example: Statically Provisioning a Volume Using the MapR Container Storage Interface (CSI) Storage Plugin on page 3111

Example: Mounting a PersistentVolume for Static Provisioning on page 3113

Example: Mounting a PersistentVolume for Dynamic Provisioning Using MapR Container Storage Interface (CSI) Storage Plugin on page 3120

Raw Block Volumes
This page describes support for raw block volumes by the MapR Container Storage Interface (CSI) Storage Plugin.

The HPE implementation of CSI supports raw block volumes. Inside a container, this feature enables a persistent volume to appear as a block device instead of as a mounted file system. This feature can be useful for applications that do not work with NFS or FUSE or perform better on a standard Linux file system, such as EXT4 or XFS.

The following (or later) releases of the storage plugin support raw block volumes:

- MapR Container Storage Interface (CSI) Storage Plugin Release 1.2.x (FUSE POSIX) on page 6550
- MapR Container Storage Interface (CSI) Storage Plugin Release 1.0 (Loopback NFS) on page 6552

For more information, see Raw Block Volume Support in the Kubernetes documentation.

Comparing the FUSE POSIX and Loopback NFS Plugins
This page compares the two types of MapR Container Storage Interface (CSI) Storage Plugins and describes when to use them.

Features Common to Both Plugins

The FUSE POSIX and Loopback NFS plugins both support the following features:

- Create a volume
- Delete a volume
- Expand a volume
- Clone a volume
- Create a snapshot
- Delete a snapshot
- Restore a snapshot

Current versions of both plugins include the release 6.2 binaries and can be used with releases 6.1 or 6.2. In addition, both plugins can exist on the same Kubernetes cluster at the same time. Both plugins can also be used without a license; however, the FUSE POSIX plugin provides a `license` parameter that allows you to control the number of resources used, as described in Example: Mounting a PersistentVolume for Static Provisioning on page 3113.

How the Loopback NFS Plugin is Different

The Loopback NFS plugin leverages the MapR loopbacknfs POSIX Client. For more information about this client, see MapR loopbacknfs POSIX Client on page 1230.

The plugins differ in how they handle I/O. The Loopback NFS plugin uses asynchronous I/O, allowing more I/O operations. The Loopback NFS plugin also uses less memory and provides better performance for small-file writes and raw block storage.

How the FUSE POSIX Plugin is Different

The FUSE POSIX plugin leverages the MapR FUSE-Based POSIX Client. For more information about this client, see MapR FUSE-Based POSIX Client on page 1238.

The FUSE POSIX plugin uses synchronous I/O. FUSE POSIX runs with at most five clients (platinum license) and incurs resource overhead because of the high number of client threads, but works better for use cases that require high throughput.

MapR Data Fabric for Kubernetes FlexVolume Driver Overview

Describes how the FlexVolume driver for MapR Data Fabric for Kubernetes integrates with Kubernetes to provide persistent data for containers.

Important: The MapR Data Fabric FlexVolume Driver for Kubernetes is officially deprecated and becomes an unsupported product on October 31, 2022. Users of the FlexVolume Driver are encouraged to migrate to one of the available CSI drivers. See CSI Version Compatibility on page 5596.

To review the FlexVolume Driver end-of-life announcement, see support advisory 4822. For a comparison of the CSI and FlexVolume technologies, see FlexVolume.

To install or use the MapR Data Fabric for Kubernetes, see:

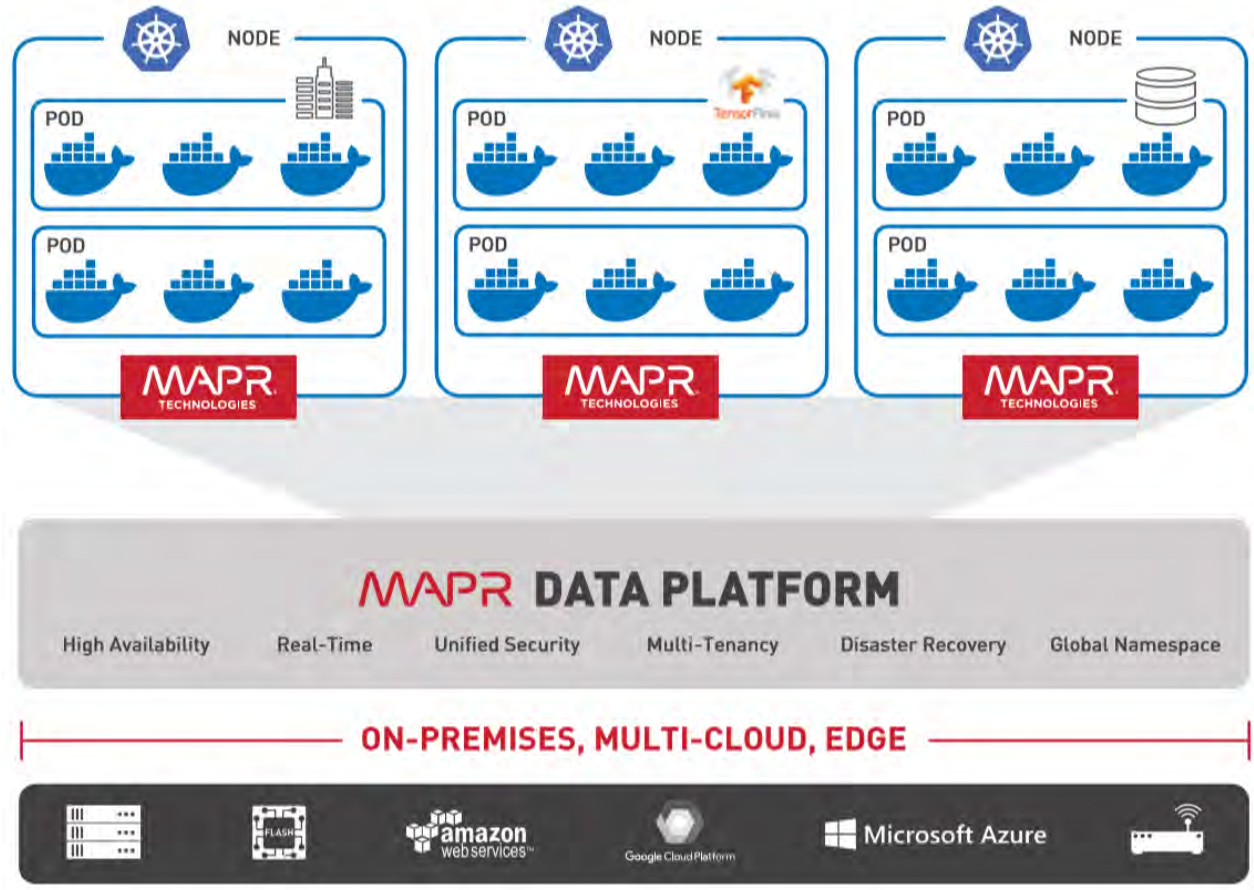
- Installing the MapR Data Fabric for Kubernetes FlexVolume Driver on page 241
- Using the MapR Data Fabric for Kubernetes FlexVolume Driver on page 3151

About the MapR Data Fabric for Kubernetes

Most Pods in a Kubernetes environment should be portable, short-lived, and stateless. Traditionally, when a Pod is stopped or moved, the state of its containers could be lost. The MapR Data Fabric for Kubernetes:

- Provides long-lived, persistent storage for Pods and their containers.
- Allows containers running in Kubernetes to use the MapR filesystem for all of their storage needs.
- Allows secure storage of all container states in MapR XD Distributed File and Object Store.

last-updated: Apr 11, 2022



The MapR Data Fabric for Kubernetes consists of a set of Docker containers and their respective `.yaml` configuration files for installation into Kubernetes. Once installed, both a Kubernetes FlexVolume Driver for MaprFS and a Kubernetes Dynamic Volume Provisioner are available for both static and dynamic provisioning of MapR storage.

Containers

Containers are stand-alone, executable images of applications. They freeze all code needed to run an application, including an OS. Unlike VMs, containers run directly on an operating system without the need for a HyperVisor. Both Linux- and Windows-based applications can be packaged as containers. Containers represent an easy way to deploy applications in development and test environments. Using containers, developers can quickly create a development platform to test their code.

Containers are ephemeral by nature and light-weight. They enable setting up compute clusters quickly. They also allow a cluster to be dismantled quickly. To accomplish this task, containers are designed to be ephemeral. That is, they are designed to be somewhat stateless. However, truly stateless containers would eliminate many classes of applications. It is therefore important to provide containers with persistent data independent of the container lifecycle. A natural solution is to have persistent storage (data) presented to the containers, just as persistent storage is presented today for VMs and in bare-metal environments.

Container Management

Simple container solutions are somewhat limited when orchestrating multiple containers to solve complex business challenges. Managing containers for production is challenging. With many workloads transitioning to fully production-grade containers, cluster admins need something beyond a container engine like Docker. Several container-orchestration engines are now available to manage containers in production. Kubernetes is the most prominent example of these container-orchestration solutions.

last-updated: Apr 11, 2022

Kubernetes Volume Drivers

Kubernetes introduced the concept of FlexVolume drivers. FlexVolume drivers are intended to allow storage vendors to provide storage to containers managed by Kubernetes. The MapR Data Fabric for Kubernetes leverages Kubernetes FlexVolume drivers. There are additional Kubernetes components and concepts you should also be aware of:

- Kubernetes Volumes: A Kubernetes volume is a Kubernetes-managed resource concept. Kubernetes Volumes are associated with Kubernetes Pods. Kubernetes Volumes are different from MapR volumes. The lifecycle of a Kubernetes volume is tied to the lifecycle of a Kubernetes Pod, and the Kubernetes Volume is destroyed when the Pod is deleted.
- Kubernetes Persistent Volumes: As the name indicates, a Kubernetes Persistent Volume (PV) lifecycle is separate from the Pod that uses it. Persistent Volumes are referenced by Persistent Volume Claims (PVC), which are in turn referenced by Pods. Multiple Pods can claim a single PVC, but only a single PVC can bind with a PV.
- Storage Classes: A Storage Class is a way for administrators to advertise the different classes of storage they offer. For example, the admin can provide parameters in the storage class that define the frequency of snapshots or the number of mirrors associated with the storage. Storage Classes are used to dynamically provision a new storage volume for use by containers.
- MapR Volumes: The MapR Glossary defines a MapR volume as a tree of files and directories grouped for the purpose of applying a policy or set of policies to all of them at once. To avoid confusion, this document uses the terms *Kubernetes volume* and *MapR volume* to distinguish between the different types of volumes.

Kubernetes and MapR Volumes

In general, Kubernetes is not aware of MapR volumes. When static provisioning a MapR path, Kubernetes simply uses a MapR POSIX client to obtain a specific mount point within the MapR file system. When dynamically provisioning a new MapR volume for a container to use, the dynamic provisioner issues REST calls to the MapR REST server to create actual MapR volumes.

Additional Resources

See the following MapR sites for more information about application containers and Kubernetes:

- MapR Data Fabric for Kubernetes Product Page
- Tutorial: How to Install and Deploy Applications at Scale on Kubernetes
- Blog: Overview of Application Containers
- Blog: Using Kubernetes to Manage Containers and Cluster Resources
- Blog: Containers: Best Practices for Running in Production

Static and Dynamic Provisioning Using FlexVolume Driver

Describes static and dynamic storage provisioning using the FlexVolume driver on a Kubernetes cluster.

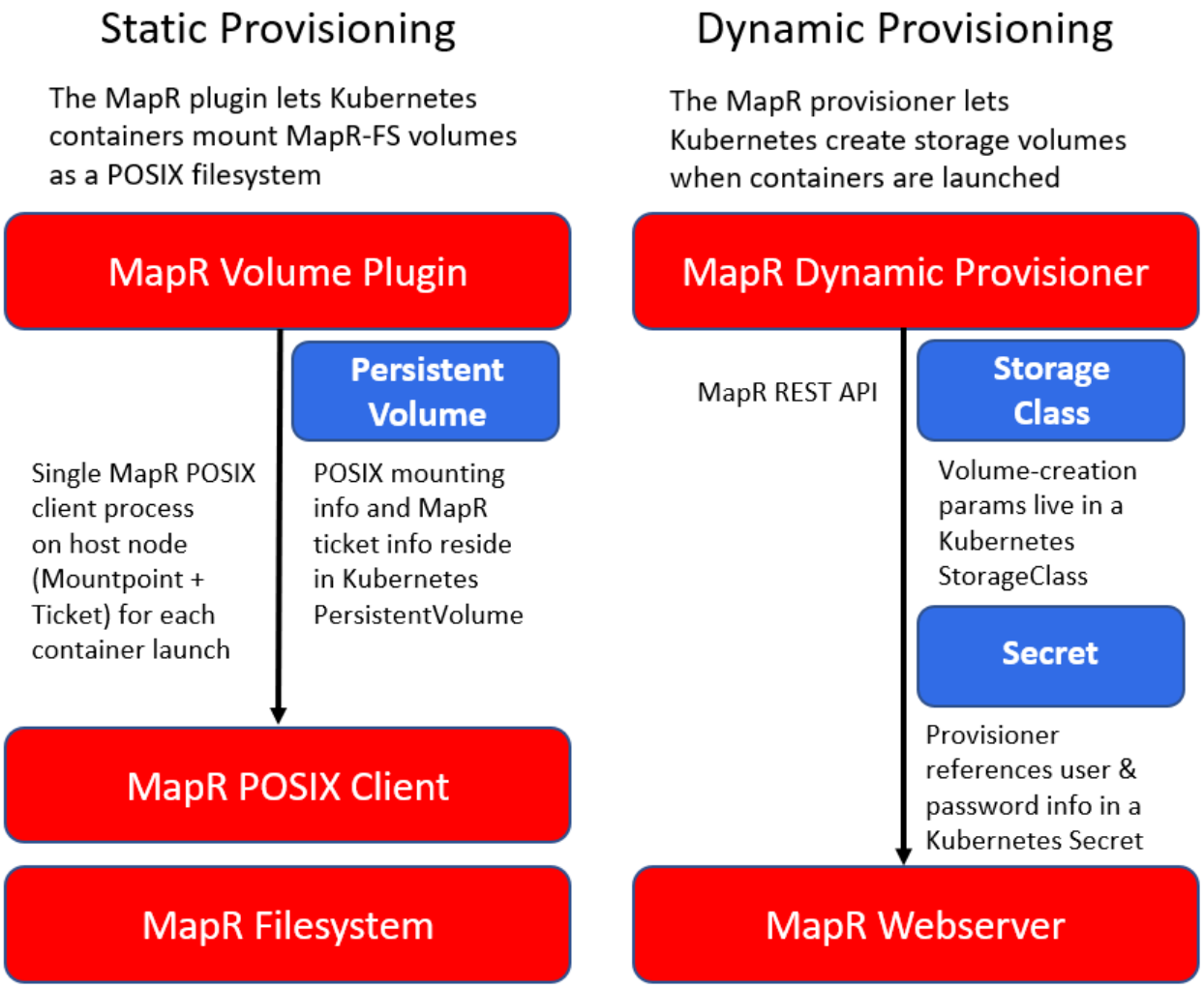
Kubernetes makes a distinction between static and dynamic provisioning of storage.

In static provisioning, a MapR administrator first creates MapR volumes (mount points) and then ensures that they are mounted. A Kubernetes administrator exposes these MapR mount points in Kubernetes through Kubernetes PersistentVolumes. In a typical static-provisioning scenario, a Pod author requests that a Kubernetes administrator create a PersistentVolume that references an existing MapR mount point with a dataset that the Pod author is interested in. This PersistentVolume references the MapR FlexVolume plug-in. The FlexVolume plug-in mounts and unmounts MapR mount points for the requesting Pod.

last-updated: Apr 11, 2022

In dynamic provisioning, a Kubernetes administrator creates a set of StorageClasses for Pods to invoke. Each StorageClass has a predefined set of storage characteristics. Examples of these characteristics include the MapR volume advisory quota size and snapshot rules. The Pod creator searches the predefined Storage Classes for the one that best matches the creator's requirements. When the Pod references this StorageClass through a PersistentVolumeClaim, the StorageClass calls the MapR Dynamic Provisioner to allocate storage for the requesting Pod dynamically.

To leverage MapR File System with a Kubernetes cluster, you can create a PersistentVolume in Kubernetes. This diagram shows the two ways in which the PersistentVolume can be provisioned:



Static Provisioning Implementation

To accomplish static provisioning, the MapR KDF FlexVolume plug-in is deployed to all nodes in the Kubernetes cluster via a Kubernetes DaemonSet. The volume plug-in uses the Basic or Platinum POSIX client to mount the MapR filesystem The information that the POSIX client uses to connect to MapR is contained in a Kubernetes Volume or PersistentVolume. A MapR ticket inside a Secret, referenced by the Kubernetes Volume or PersistentVolume specification, is used to pass secure data to the file system.

Dynamic Provisioning Implementation

To accomplish dynamic provisioning, the MapR KDF provisioner is deployed as a Kubernetes Deployment to a single node in the Kubernetes cluster. The provisioner requests the creation of MapR volumes when a container is launched. You can scale your provisioner deployment to multiple nodes for high availability. If a provisioner Pod is deleted, a new provisioner is started on another worker node in the cluster.

last-updated: Apr 11, 2022

A Kubernetes Administrator must configure at least one storage class with MapR parameters (for example, mirroring, snapshots, quotas, and other parameters) for use during creation of the MapR volume. The storage class passes MapR administrative credentials to the provisioner through a Kubernetes Secret. Security for the provisioner is handled through role-based access control (RBAC) in Kubernetes.

POSIX Integration and Licensing
Explains how the basic and platinum POSIX clients are supported on a Kubernetes cluster,

The MapR POSIX client provides fast-data access between the container and the MapR filesystem. For FlexVolume plug-in, the POSIX client is installed onto all Kubernetes worker nodes when you install the volume plug-in through its `.yaml` configuration file. For CSI Driver, the POSIX client is installed onto the CSI Driver container only.

For static provisioning, the volume plug-in uses the POSIX client to mount the MapR filesystem. The provisioner does not use the POSIX client to provision volumes, but a provisioned volume is mounted through POSIX when the plug-in is called after PV creation.

Support for Basic and Platinum Licenses

By default, the product includes the Basic POSIX client package, but you can enable the Platinum license, if needed. See Enabling the Platinum Posix Client for FlexVolume Driver and CSI Driver. Only the POSIX client is supported. NFSv3 and NFSv4 are currently not supported.

While the Platinum POSIX client offers up to five times better performance than the Basic POSIX client, resource utilization is significantly higher for the Platinum client. For a comparison of the Basic and Platinum packages, see Preparing for Installation.

Mounting Multiple MapR Paths

It is inefficient in both host resources and licenses to mount multiple MapR paths in the same Pod. In FlexVolume Driver, multiple mount points will consume additional resources on the Kubernetes host node. A more resource-efficient strategy is to use subpaths. See Using subpaths in the Kubernetes documentation.

Cluster Management

Provides a synopsis of the various cluster components and their management.

MapR provides high availability management and data processing services for automatic continuity throughout the cluster. You can use the Control System, command-line interface, or REST API to start, stop, and monitor services at the node or cluster level. MapReduce services such as the ResourceManager, management services such as the ZooKeeper, and data access services such as NFS provide continuous service during any system failure.

MapR Monitoring (part of the Spyglass initiative) provides the ability to collect, store, and view metrics and logs for nodes, services, and jobs/applications.

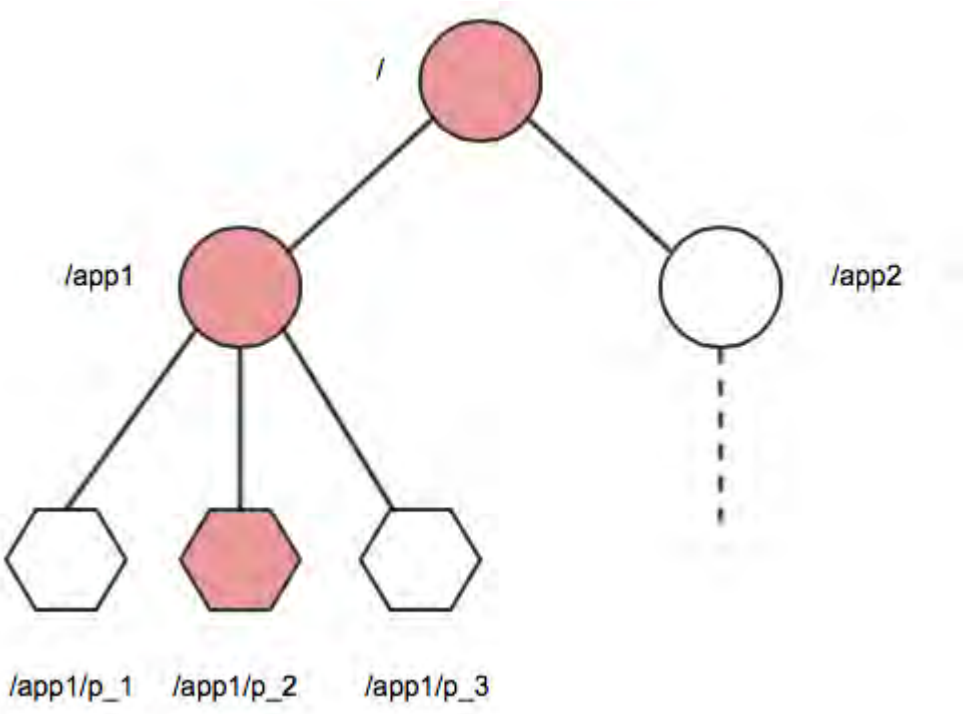
This section describes the following components and services, and also describes their roles in managing a MapR cluster:

ZooKeeper

Provides an overview of the ZooKeeper service.

ZooKeeper is a coordination service for distributed applications. It provides a shared hierarchical namespace that is organized like a standard filesystem. The namespace consists of data registers called znodes, for ZooKeeper data nodes, which are similar to files and directories. A name in the namespace is a sequence of path elements where each element is separated by a / character, such as the path `/app1/p_2` shown here:

last-updated: Apr 11, 2022



Namespace

The znode hierarchy is kept in-memory within each ZooKeeper server in order to minimize latency and to provide high throughput of workloads.

The ZooKeeper Ensemble

The ZooKeeper service is replicated across a set of hosts called an ensemble. One of the hosts is designated as the leader, while the other hosts are followers. ZooKeeper uses a leader election process to determine which ZooKeeper server acts as the leader, or master. If the ZooKeeper leader fails, a new leader is automatically chosen to take its place.

Establishing a ZooKeeper Quorum

As long as a majority (a quorum) of the ZooKeeper servers are available, the Zookeeper service is available. For example, if the ZooKeeper service is configured to run on five nodes, three of them form a quorum. If two nodes fail (or one is taken off-line for maintenance and another one fails), a quorum can still be maintained by the remaining three nodes. An ensemble of five ZooKeeper nodes can tolerate two failures. An ensemble of three ZooKeeper nodes can tolerate only one failure. As a quorum requires a majority, an ensemble of four ZooKeeper nodes can only tolerate one failure, and therefore offers no advantages over an ensemble of three ZooKeeper nodes. In most cases, you should run three or five ZooKeeper nodes on a cluster. Larger quorum sizes result in slower write operations.

Ensuring Node State Consistency

Each ZooKeeper server maintains a record of all znode write requests in a transaction log on the disk. The ZooKeeper leader issues timestamps to order the write requests, which when executed, updates elements in the shared data store. Each ZooKeeper server must sync transactions to disk and wait for a majority of ZooKeeper servers (a quorum) to acknowledge an update. Once an update is held by a quorum of nodes, a successful response can be returned to clients. By ordering the write requests with timestamps and waiting for a quorum to be established to validate updates, ZooKeeper avoids race conditions and ensures that the node state is consistent.

last-updated: Apr 11, 2022

Warden

Describes the Warden daemon that monitors and restarts services if they terminate.

Warden is a light Java application that runs on all the nodes in a cluster and coordinates cluster services. Warden's job on each node is to start, stop, or restart the appropriate services, and allocate the correct amount of memory to them. Warden makes extensive use of the znode abstraction discussed in the ZooKeeper section of this document to monitor the state of cluster services.

Each service running in a cluster has a corresponding znode in the ZooKeeper namespace, named in the pattern `/services/<servicename>/<hostname>`. Warden's Watcher interface monitors znodes for changes and acts when a znode is created or deleted, or when child znodes of a monitored znode are created or deleted.

Warden configuration is contained in the `warden.conf` file, which lists service triplets in the form `<servicename>:<number of nodes>:<dependencies>`. The `number of nodes` element of this triplet controls the number of concurrent instances of the service that can run on the cluster. Some services are restricted to one running instance per cluster, while others, such as the File Server, can run on every node. The Warden monitors changes to its configuration file in real time.

When a configuration triplet lists another service as a dependency, the Warden only starts that service after the dependency service is running.

Note: When Warden is started/restarted, the irqbalancer is enabled on nodes running MapR File System because it balances IRQ SMP affinities, which provide better performance.

Memory Management with the Warden

System administrators can configure how the cluster's memory is allocated to running the operating system, MapR File System, and Hadoop services. The configuration files `/opt/mapr/conf/warden.conf` and `/opt/mapr/conf/conf.d/warden.<servicename>.conf` include parameters that define how much of the memory on a node is allocated to the operating system, MapR File System, and Hadoop services.

You can edit the following memory parameters to reserve memory:

- The `service.<servicename>.heapsize.percent` parameter controls the percentage of system memory allocated to the named service.
- The `service.<servicename>.heapsize.max` parameter defines the maximum heapsize used when invoking the service.
- The `service.<servicename>.heapsize.min` parameter defines the minimum heapsize used when invoking the service.

For example, the `service.command.os.heapsize.percent`, `service.command.os.heapsize.max`, and `service.command.os.heapsize.min` parameters in the `warden.conf` file control the amount of memory that Warden allocates to the host operating system before allocating memory to other services.

The actual heap size used when invoking a service is a combination of the three parameters according to the formula:

```
max(heapsize.min, min(heapsize.max, total-memory * heapsize.percent / 100))
```

For more information, see Memory Allocation for Nodes.

last-updated: Apr 11, 2022

The Warden and Failover

The Warden on each node watches appropriate znodes to determine whether to start or stop services during failover. The following paragraphs provide failover examples for the CLDB and ResourceManager. Note that not all failover involves the Warden; NFS failover is accomplished using VIPs.

CLDB Failover	The ZooKeeper contains a znode corresponding to the active primary CLDB. This znode is monitored by the secondary CLDBs. When the primary CLDB znode is deleted, the secondary CLDBs recognize that the primary CLDB is no longer running. The secondary CLDBs contact ZooKeeper in an attempt to become the new primary CLDB. The first CLDB to get a lock on the znode in ZooKeeper becomes the new primary instance.
ResourceManager Failover	Starting in version 4.0.2, if the node running the ResourceManager fails and the Warden on the ResourceManager node is unable to restart it, Warden starts a new instance of the ResourceManager on another node. The Warden on every ResourceManager node watches the ResourceManager's znode for changes. When the active ResourceManager's znode is deleted, the Wardens on other ResourceManager nodes attempt to launch the ResourceManager. The Warden on each ResourceManager node works with the ZooKeeper to ensure that only one ResourceManager is running in the cluster. In order for failover to occur in this manner, at least two nodes in the cluster should include the ResourceManager role and your cluster must be use the zero configuration failover implementation.

The Warden and Pluggable Services

Services can be plugged into the Warden's monitoring infrastructure by setting up an individual configuration file for each supported service in the `/opt/mapr/conf/conf.d` directory, named in the pattern `warden.<servicename>.conf`. The `<servicename>:<number of nodes>:<dependencies>` triplets for a pluggable service are stored in the individual `warden.<servicename>.conf` files, not in the main `warden.conf` file.

The following services/packages have configuration files pre-configured at installation:

- Hue
- HTTP-FS
- The Hive metastore
- HiveServer2
- Oozie
- Spark-Master
- mapr-apiserver
- mapr-collectd
- mapr-drill
- mapr-elasticsearch

last-updated: Apr 11, 2022

- mapr-fluentd
- mapr-grafana
- mapr-hbase
- mapr-hbasethrift
- mapr-historyserver
- mapr-hive
- mapr-hivemetastore
- mapr-hiveserver2
- mapr-hivewebchat
- mapr-httpfs
- mapr-hue
- mapr-impala
- mapr-impalacatalog
- mapr-impalaserver
- mapr-impalastore
- mapr-kafka
- mapr-kibana
- mapr-ksql
- mapr-livy
- mapr-nodemanager
- mapr-objectstore
- mapr-oozie
- mapr-opentsdb
- mapr-resourcemanager
- mapr-schema
- mapr-sentry
- mapr-spark
- mapr-sqoop2
- mapr-storm
- mapr-tez
- mapr-timelineserver

last-updated: Apr 11, 2022

- mapr-webserver

A package can contain multiple services. For example, `mapr-spark` contains all of Spark services including Spark Thrift Server and Spark Master.

After you install a package and run the configure.sh on page 2053 utility, the associated Warden files are present in `/opt/mapr/conf/conf.d`.

The Warden daemon monitors the znodes for a configured component's service and restarts the service as specified by the configuration triplet. The configuration file also specifies resource limits for the service, ports used by the service (if any), and a location for log files.

In the triplet `<servicename>:<number of nodes>:<dependencies>`, the `<number of nodes>` can be set to `all`. The value `all` specifies that the service is to be started on every node on which the service is installed.

For example, consider the entry `services=kvstore:all;cldb:all:kvstore;hoststats:all:kvstore`. This entry specifies the following:

1. Start `kvstore` on all the nodes on which it is installed.
2. Start `cldb` on all the nodes on which it is installed, but wait until `kvstore` is up on all nodes. In other words, `cldb` depends on `kvstore` to be up.
3. Start `hoststats` on all nodes on which it is installed but wait until `kvstore` is up on all nodes. In other words, `hoststats` depends on `kvstore` to be up.

As another example, consider the entry: `resourcemanager:1:cldb`. Here, only one instance of `resourcemanager` is started, after `cldb` is up.

If this instance of `resourcemanager` goes down, Warden notices that the number of running instances is below the specified count, and automatically handles the failover. If multiple instances of `resourcemanager` get started, Warden terminates all the extra instances.

Dependencies are usually handled internally. Some non-core components do have dependencies among themselves, such as for example:

```
services=nodemanager:all:resourcemanager
hbmaster:all:cldb
hbregionserver:all:hbmaster
```

Here:

1. `nodemanager` depends on `resourcemanager`
2. `hbmaster` depends on `cldb`
3. `hbregionserver` depends on `hbmaster`

CLDB

Describes the Container Location Database (CLDB).

CLDB

The Container Location Database (CLDB) service tracks the following information about every container in the MapR file system:

- The node where the container is located.
- The container's size.

- The volume to which the container belongs.
- The policies, quotas, and usage for that volume.

For more information on containers, see File System on page 452.

The CLDB also tracks fileservers in the cluster and node activity. Running the CLDB service on multiple nodes distributes lookup operations across those nodes for load balancing, and also provides high availability.

When a cluster runs the CLDB service on multiple nodes, one node acts as the primary CLDB and the others act as secondary instances. The primary node has read and write access to the filesystem, while secondary nodes only have read access. The kvstore (key-value store) container has the container ID 1, and holds cluster-related information. The ZooKeeper tracks container information for the kvstore container. The CLDB assigns a container ID to each new container it creates. The CLDB service tracks the location of containers in the cluster by the container ID.

When a client application opens a file, the application queries the CLDB for the container ID of the root volume's name container. The CLDB returns the container ID and the IP addresses of the nodes in the cluster where the replicas of that container are stored. The client application looks up the volume associated with the file in the root volume's name container, then queries the CLDB for the container ID and IP addresses of the nodes in the cluster with the name container for the target volume. The target volume's name container has the file ID and inode for the target file. The client application uses this information to open the file for a read or write operation.

Each fileserver heartbeats to the CLDB periodically, at a frequency ranging anywhere from 1-3 seconds depending on the cluster size, to report its status and container information. The CLDB may raise alarms based on the status communicated by the FileServer.

Central Configuration

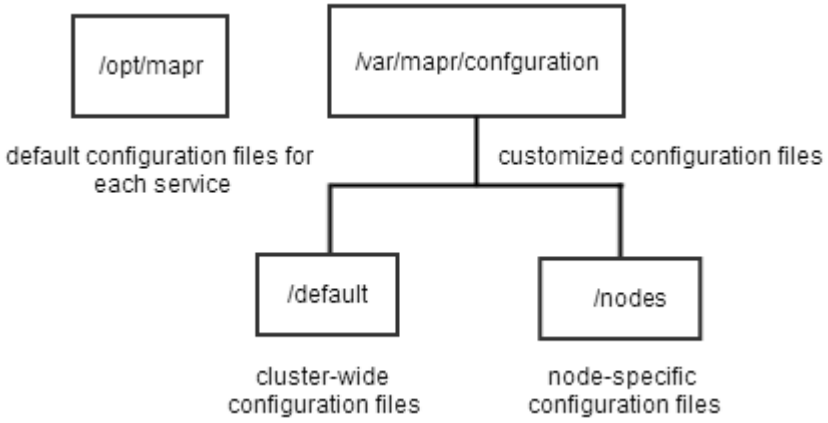
Describes the location where you can store customized configuration files, and the `pullcentralconfig` utility that is used to detect such files.

Each service on a node has one or more configuration files associated with it. The default version of each configuration file is stored locally under `/opt/mapr/`.

Customized versions of the configuration files are placed in the `mapr.configuration` volume, which is mounted at:

```
/var/mapr/configuration
```

The following diagram illustrates where each configuration file is stored:



MapR uses the `pullcentralconfig` script to detect customized configuration files in:

```
/var/mapr/configuration
```

This script is launched every five minutes by default. When the script finds a customized file, it overwrites the local files in `/opt/mapr`. First, the script looks for node-specific custom configuration files under:

```
/var/mapr/configuration/nodes/<hostname>
```

If the script does not find any configuration files at that location, the script searches for cluster-wide configuration files under:

```
/var/mapr/configuration/default
```

The `/default` directory stores cluster-wide configuration files that apply to all nodes in the cluster by default.

MapR Control System

Provides a brief description of the MapR Control System.

The MapR Control System provides a graphical control panel for cluster administration with all the functionality of the command-line or REST APIs. The Control System provides job monitoring metrics and helps you troubleshoot issues, such as which jobs required the most memory in a given week, or which events caused job and task failures.

The Control System provides various views, which you can use to configure and monitor your cluster:

Overview	The Control System **Overview** page provides a summary of information about the cluster including a cluster heat map that displays the health of each node organized by service, an alarms summary, cluster utilization that shows the CPU, memory, and disk space usage, the number of available, unavailable, and under replicated volumes, and MapReduce applications.
Services	The Control System **Services** page provides a summary of the services running across the cluster.
Nodes	The Control System **Nodes** page provides a summary of information about the nodes on the cluster including a heat map that displays the health of each node, resource utilization that shows the CPU and memory usage, all active alarms, and a list of all the nodes on the cluster with links that provide shortcuts to more detailed information about the node.
Data	The Control System **Data** drop-down menu contains links to pages that provide summary of information about volumes, tables, and streams.
Admin	The Control System **Admin** drop-down menu contains links to pages for user and cluster management tasks such as setting up permissions, quotas, and email settings for users, enabling cluster-level and data auditing, configuring balancer settings, and adding licenses.

Note: During installation using the MapR Installer, you can configure metrics and logging using settings on the Monitoring page of the MapR Installer user interface. The metrics collection infrastructure must be installed because the Control System relies on these metrics to provide graphs and charts. If the metrics collection infrastructure is not installed, you cannot visualize the metrics in the panes on the Control System.

Related concepts

Setting Up the Control System on page 423
Describes how to configure and access the Control System.

Security

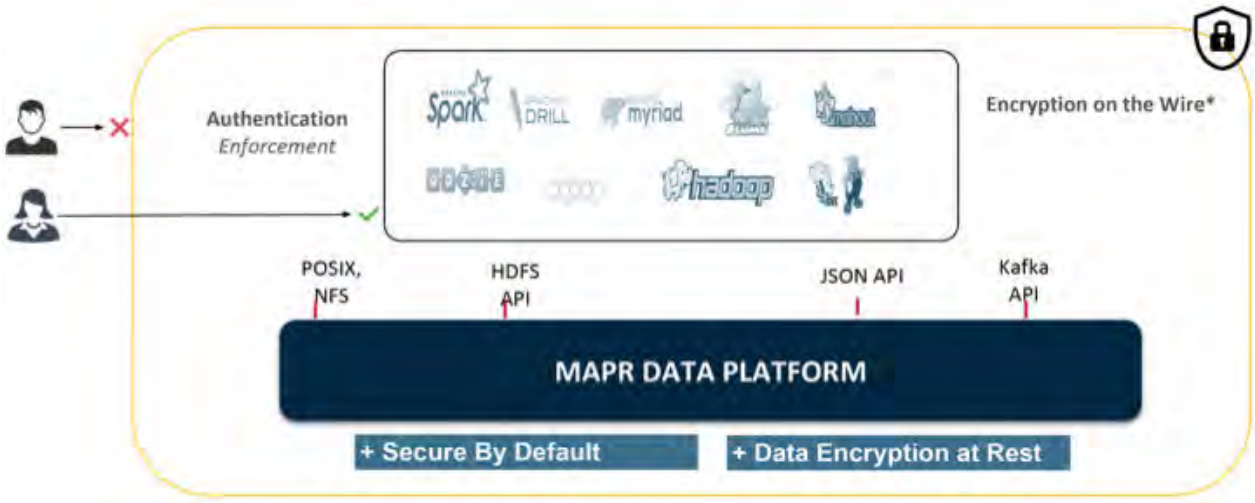
Provides an overview of the MapR security features.

Securing enterprise data is critical. To make securing data in clusters easy, the MapR Data Platform has a data protection scheme built directly into the platform that is enabled by default, simplifying the process of protecting critical data. You can take advantage of the default security settings, or you can implement data security manually. Either way, it is important to identify which data to secure.

Since data must be shared between nodes on the cluster, data transmissions between nodes, and from the cluster to the client are vulnerable to interception. Networked computers are also vulnerable to attacks where an intruder successfully pretends to be another authorized user and then acts improperly as that user. Additionally, networked machines share the security vulnerabilities of a single node. The MapR Data Platform supports the ability to apply protection directly as data enters and exits the platform. You do not need to apply an external management server or particular security plug-in.

Secure by Default	MapR, which includes the MapR Data Platform and EEP components, is secure out-of-the-box on all new installations, ensuring all network connections require authentication and all data in motion is protected with wire-level encryption. MapR provides the ability to apply security protection directly for data as it comes into and out of the platform without requiring an external security manager server or a particular security plug-in for each ecosystem component. The security semantics are applied automatically on data being retrieved or stored by any ecosystem component, application, or users.
Platform-Based Security	The MapR Data Platform applies security semantics automatically as data is being stored and retrieved from the platform. It supports all four pillars of security (authentication, authorization, auditing, and encryption), using platform-level capabilities that do not require external security tools or plugins.
Encryption	On the MapR Data Platform, data is protected by encrypting all data being transmitted over the wire and encrypting all data that is stored in the platform.

last-updated: Apr 11, 2022



The following sections describe the MapR security capabilities and security architecture.

Security Capabilities

A secure MapR environment is predicated on authentication, authorization, auditing, and encryption capabilities. You can use policy-based security to classify and manage these capabilities.

Authentication	Restricting access to a specified set of users. Robust authentication prevents third parties from representing themselves as legitimate users. MapR supports a wide range of authentication mechanisms depending on the network transport that includes MapR tickets, kerberos, Pluggable Access Module (PAM), Basic Authentication, MapRSASL, and SPNEGO. See Configuring Authentication on page 1424 for more information.
Authorization	Restricting an authenticated user's capabilities on the system. MapR provides sophisticated authorization controls to ensure that users can perform only the activities for which they have permissions, such as data access, job submission, cluster administration, and more. These permissions can be granted by an administrator through the browser-based Control System management and monitoring interface, or by using the command-line utilities. See Managing Access Controls on page 1445 for more information.
Auditing	Logging audit records of operations. MapR allows you to log audit records of cluster-administration operations and operations on directories, files, and tables. See Managing Auditing on page 757 for more information.
Encryption	Restricting an external party's ability to read data. Encryption is used to avoid exposure to breaches, such as packet sniffing and theft of storage devices.

last-updated: Apr 11, 2022

	In a secure MapR cluster, data transmission between nodes, and between a MapR cluster and ecosystem application is encrypted, preventing an attacker with access to that communication from gaining information about the contents of the transmission. Optionally, you can enable encryption for data at rest to prevent unauthorized users from accessing sensitive data, and protect against data theft through sector-level disk access. Data is protected by encrypting all data being transmitted over the wire and optionally encrypting all that is stored on the MapR platform. The MapR data encryption scheme is built directly into the platform and is enabled by default. See Managing Encryption for MapR Core on page 1410 for more information.

Security Architecture

MapR provides the following authentication and authorization functionality:

Filesystem permissions	For files and directories on the MapR cluster, you can leverage standard Unix-style permissions to grant access to authorized users. Since MapR File System is a POSIX-like filesystem, you can set user permissions as you would on any other Linux system. See Setting MapR File System Permissions on page 981 for more information.
Cluster, volume, and job queue Access Control Lists (ACLs)	You can specify the actions that a given user can perform on each of these cluster elements. You can use access control lists (ACLs) to grant permissions for performing administrative tasks at both the cluster and the volume level. See Managing Access Control Lists on page 1445 for more information.
Access Control Expressions for filesystem and natively stored MapR Database tables	ACEs control which files, directories, volumes, streams, and tables users or groups can access. ACEs are a powerful and flexible mechanism to grant permissions on structured and unstructured data. See Managing Access Control Expressions on page 1448 for more information.
Impersonation for centralized control of access to resources	Impersonation, also known as identity assertion, is one user accessing data and submitting jobs on behalf of another user. See Managing Impersonation on page 1476 for more information.

What to do Next

The MapR secure-by-default data platform provides security through a single option in the MapR Installer on page 5395 or by running the `configure.sh on page 2053` script with the `-secure` option after a manual installation. You can enable security on your cluster using the procedure described in the following topics:

- Using the Enable MapR Secure Cluster Option on page 5427 if you are installing with the MapR Installer on page 5395.
- Enabling Security on page 1405 if you are Installing without the MapR Installer on page 141.

After enabling security, optionally, you can perform the following tasks:

- Understand the security exceptions and take corrective action, where applicable.

last-updated: Apr 11, 2022

- Configure authorization on the resources.
- Configure auditing on administration and resources.
- Configure encryption for data at rest.
- If you have Hive installed, enable storage-based authorization for the Hive Metastore server.

Authentication in MapR

Describes types of authentication available with the MapR Data Platform and how to manage user authentication with the `maprlogin` utility.

Authentication ensures that who you really are and who you claim to be, match when identifying the end user to the system. MapR authentication supports standard Basic Authentication and SPNEGO authentication for web-based interfaces, and supports MapR tickets for many of the core system component non-web-based interfaces. A ticket is an object that contains specific information about a user, an expiration time, and a key. Tickets uniquely identify a user and are encrypted to protect their contents. You can use tickets to establish sessions between a user and the cluster.

Types of Authentication in MapR

MapR supports two methods of authenticating a user and generating a ticket: a username-password pair and Kerberos. Both of these methods are mediated by the maprlogin on page 2130 utility. When you authenticate with a username-password pair, the system verifies your credentials using Pluggable Authentication Modules (PAM). You can configure the cluster to use any registry that has a PAM module.

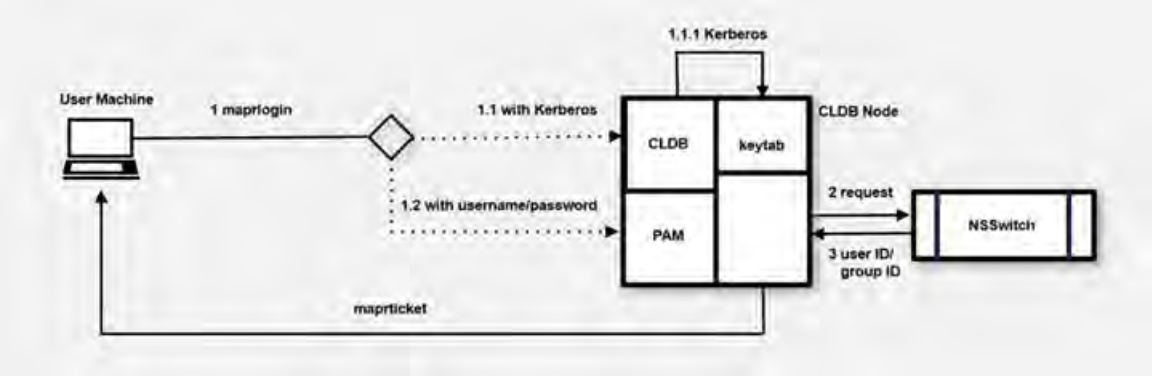
MapR tickets contain the following information:

- UID (generated from the UNIX user ID)
- GIDs (group IDs for each group the user belongs to)
- Ticket creation time
- Ticket expiration time (by default, 14 days)
- Renewal expiration time (by default, 30 days from the date of ticket creation)

A MapR ticket determines the user's identity. The system uses the ticket as the basis for authorization. A MapR cluster with security features enabled does not rely on the client-side operating system identity.

The maprlogin Utility for Generating Tickets

The `maprlogin on page 2130` utility supports user authentication with either username and password, or Kerberos to generate a unique session token called a *ticket*. The following diagram outlines the process flow:



MapR tickets are either implicitly or explicitly generated. On clusters that use Kerberos for authentication, a user that runs a MapR command without first using the `maprlogin` utility implicitly obtains a MapR ticket. During usage, the client runtime process first checks for a valid user ticket, and uses that ticket if it exists. If a ticket does not exist, the runtime process checks if Kerberos is enabled for the cluster and then checks for an existing valid Kerberos identity. When a valid Kerberos identity is found, the client implicitly generates a ticket for that Kerberos identity.

When you explicitly generate a ticket, you can authenticate either with your username and password, or with Kerberos:

1. The user on the client machine invokes the `maprlogin` utility, which connects to a CLDB node in the cluster using HTTPS. The host name for the CLDB node is specified in the `mapr-clusters.conf` file.
   - For username-password authentication, the node authenticates using PAM modules with the Java Authentication and Authorization Service (JAAS).

     The JAAS configuration is specified in the `mapr.login.conf` file. The system can use any registry that has a PAM module available.
   - For Kerberos authentication, the CLDB node verifies the Kerberos principal with the `keytab` file.
2. After authenticating, the CLDB node uses the standard UNIX APIs `getpwnam_r` and `getgrouplist`, which are controlled by the `/etc/nsswitch.conf` file, to determine the user IDs and group IDs.
3. The CLDB node generates a ticket and returns it to the client machine, completing the login communication between the client and the CLDB.
4. After login, the MapR server validates that the ticket is properly encrypted, to verify that the ticket was issued by the cluster's CLDB.
5. The server also verifies that the ticket has not expired or been blacklisted.
6. The server checks the ticket for a privileged identity such as the `mapr` user.

   Privileged identities have impersonation functionality enabled.
7. The ticket's user and group information are used for authorization to the cluster, unless impersonation is in effect.

Authorization in MapR

Describes the basics of authorization including Access Control Lists and Access Control Expressions.

Authorization restricts what an authenticated user can do with data. MapR enables you to create flexible authorization systems that grant a user capabilities to perform desired tasks, but prevents the user from performing tasks outside of that scope. Use a combination of Access Control Lists and Access Control Expressions to set up a flexible authorization system.

Access Control Lists	MapR supports Access Control Lists (ACLs) in several areas, including for regulating user privileges to the job queue and cluster. MapR also uses ACLs to control administrative access to volumes (administrative access is distinct from data access). An Access Control List (ACL) is a list of users or groups. Each user or group in the list is paired with a defined set of permissions that limit the actions that the user or group can perform on the object secured by the ACL. In MapR, the objects secured by ACLs are the job queue, volumes, and the cluster itself.

last-updated: Apr 11, 2022

	A job queue ACL controls who can submit jobs to a queue, kill jobs, or modify their priority. A volume-level ACL controls which users and groups have administrative access to that volume, and what actions they may perform, such as mirroring the volume, altering the volume properties, dumping or backing up the volume, or deleting the volume.
Access Control Expressions	MapR also provides a more powerful authorization known as Access Control Expressions. *ACE*s allow you to control access using powerful boolean logic expressions. You can use *ACE*s to control data access to MapR tables, files, directories, volumes, and streams. The MapR File System also supports standard POSIX filesystem permission levels. An *ACE* is a combination of user, group, and role definitions. A *role* is a custom defined name that is determined and implemented by your custom authorization code. It can be a property of a user or group that defines a set of behaviors that the user or group performs regularly. You can use ACEs to secure files, directories, volumes, tables, and streams that use native storage.

See the Configuring Data-Fabric Security on page 1402 section for information about the procedures for setting up and modifying ACLs and ACEs for the cluster, the volumes on the cluster, the job queue, the MapR filesystem, and the natively stored MapR tables and streams.

Encryption in MapR

Describes encryption types available on the MapR Data Platform.

MapR encryption restricts an external party's ability to read or modify data.

MapR supports encryption of data on wire and data at rest for preventing unauthorized access to sensitive data. These encryption methods are in addition to authentication and authorization protections. Encryption can be used to avoid exposure to breaches such as packet sniffing and theft of storage devices.

Data transmission between nodes on a secure MapR cluster is encrypted, preventing an attacker with access to that communication from gaining information about the contents of the transmission. Encryption of data-at-rest prevents unauthorized users from accessing sensitive data and protects against data theft through sector-level disk access.

On-Wire Encryption

Data transmission between nodes on a secure MapR cluster over any network connection supported by MapR is encrypted. When you run the `configure.sh` on page 2053 utility with the `-secure` option, you are enabling the cluster for security, authentication, and wire-level encryption for the platform and all ecosystem components. In secure mode, MapR automatically encrypts all data traffic. Enabling encryption ensures that data to and from the locations you specify is encrypted as it travels over the network.

MapR uses the following technologies to protect network traffic:

- The Secure Sockets Layer/Transport Layer Security (SSL/TLS) protocol secures several channels of HTTP traffic supporting TLS 1.0, 1.1(default), and 1.2.
- In compliance with the NIST standard, the 256-bit Advanced Encryption Standard in Galois/Counter Mode (AES256/GCM) secures several communication channels between cluster components.

The information in Security Protocols Used by MapR on page 689 includes details on the specific technologies used by particular elements of a cluster.

last-updated: Apr 11, 2022

Nodes with CPUs that support AES encryption at the hardware level provide superior performance on encryption tasks. You can determine if the CPU of a node supports the AES instruction set, by running the following command:

```
$ cat /proc/cpuinfo | grep flags | grep aes
```

Data-at-Rest Encryption

Data on disk (or data-at-rest) on a secure MapR cluster can be encrypted, enabling you to protect the data if a disk is compromised. Encryption of data-at-rest not only prevents unauthorized users from accessing sensitive data, but it also protects against data theft via sector-level disk access. When you run the `configure.sh` on page 2053 utility with the `-dare` option, you are enabling data at rest encryption feature at the cluster level. If encryption of data at rest is enabled, new volumes are encrypted by default with the option to create a volume without encryption. For example, if you have a volume that contains data that is not at all sensitive, you might not want to encrypt it. For encrypted volumes, MapR automatically encrypts data at rest and manages the keys used to encrypt data seamlessly; you do not need special utilities to encrypt or decrypt the data. MapR uses AES256/XTS to protect data on the disk.

SSL Certificates
Describes how certificates are used to perform authentication and encryption for websites that use the HTTPS protocol.

The TLS (Transport Layer Security formally SSL Secure Sockets Layer) certificate performs authentication and encryption for websites that use the HTTPS protocol. A certificate contains information about an entity and contains a public key. The public key is related to a private key which is NOT part of the certificate, but it is used by one entity when it communicates with another entity.

MapR stores the private key and certificate in a key store file called `ssl_keystore`. A certificate is also digitally signed so that it cannot be altered. The signer is known as the signing certificate.

In order for a HTTPS connection to be established, the following criteria must be met:

- The *server* must have a key file that contains a certificate and a private key
- The *client* must provide a trust file that contains a signer who signed the certificate used by the server
- The server certificate must be valid and not expired
- The client must determine that the SubjectDN in the certificate is acceptable

The process of enabling MapR security generates the common `ssl_keystore` and `ssl_truststore` files on the first CLDB server that are used by all clients and servers.

- The `ssl_keystore` contains a single self-signed certificate with a wildcard SubjectDN (for example, if the hostname of the CLDB is `a.b.com` the SubjectDN would be `CN=*.b.com`).
- The `ssl_truststore` contains the signer for the certificate in the `ssl_keystore`.

By default, MapR uses a unique self-signed certificate. You can use your own signing certificate by following the steps for Importing a Certificate Authority Signed (CA Signed) SSL Certificate Into a MapR Cluster.

The REST API calls in a MapR cluster communicate over the HTTPS protocol on port 8443. These calls are secured with SSL certificates that identify a node to the cluster.

Security Protocols Used by MapR
Lists the various security protocols that MapR uses for encryption and authentication.

last-updated: Apr 11, 2022

Protocol	Encryption	Authentication
MapR RPC	AES/GCM	maprticket
Hadoop RPC and MAPRSASL	AES/GCM	maprticket
Hadoop RPC and Kerberos	Kerberos	Kerberos ticket
Generic HTTP Handler	HTTPS using SSL/TLS	maprticket, username and password, or Kerberos SPNEGO

For detailed information about component-level support for authentication, impersonation, and wire-level encryption, see MapR Security Support Matrix on page 5602.

HTTPS Excluded Ciphers
Lists the weak ciphers that are excluded from the MapR HTTPS implementation.

By default, the following weak TLS/SSL ciphers are excluded from the MapR HTTPS implementation:

- `SSL_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA`
- `SSL_RSA_EXPORT_WITH_DES40_CBC_SHA`
- `SSL_RSA_EXPORT_WITH_RC4_40_MD5`

You can modify this list of excluded ciphers by editing the `hadoop.ssl.exclude.cipher.suites` property in the `core-site.xml` file. Restart the web servers that use the HTTPS protocol after changing the list of excluded ciphers. The following web servers use HTTPS:

- Control System
- NodeManager
- ResourceManager
- HistoryServer
- CLDB
- HBase

Impersonation in MapR

Describes impersonation in MapR, which allows centralized control of access to resources in the MapR File System, MapR Database, and MapR Event Store For Apache Kafka.

Also known as identity assertion, impersonation is one user (authorized to impersonate another) or the `mapr` super user accessing data and submitting jobs on behalf of another user. Implementing impersonation provides authoritative, end-to-end security for your MapR installation, independent of remote authentication and security mechanisms that control user access to application features.

To implement impersonation in MapR, there are both MapR core and ecosystem component requirements that must be met as well as requirements at the application development level. These requirements are described in Access Control and Impersonation in MapR.

When all other requirements are met, enabling impersonation for the mapr superuser or for any other user is a simple task.

Auditing in MapR

MapR allows you to log audit records of cluster-administration operations, and operations on directories, files, streams and tables.

last-updated: Apr 11, 2022

The auditing capabilities in MapR are critical for regulatory compliance as well as for understanding user behavior. Regulations often require the ability to prove which user accessed which data. Logging user behavior helps to identify suspicious activities on sensitive data.

What Information is Collected?

If you enable auditing, MapR records information about data access, operations on data objects, and execution of `maprcli` commands, including the following:

- All administrator activities that use `maprcli` commands, REST API calls, and actions performed on a cluster through the Control System
- Authentication to the Control System
- Operations on directories and files
- Operations on MapR Database objects
- Operations on MapR Event Store For Apache Kafka

How is Auditing Typically Used?

By analyzing audit records, security analysts can answer questions such as these:

- Who accessed customer records outside of business hours?
- What actions did users take in the days before leaving the company?
- What operations were performed without following change control?
- Are users accessing sensitive files from protected or secured IP addresses?
- Why do my reports sourced from the same underlying data look different?

Data scientists can analyze audit records to answers these questions:

- Which data is used most frequently, is therefore of high value, and should be shared more broadly?
- Which data is least commonly used, is therefore of low value, and could be purged?
- Which data should be used more, is therefore underused, and needs better advertising?
- Which administrative actions are most commonly performed and are therefore candidates for automation?

How does Auditing Work?

For a comprehensive explanation on how auditing works, see How Does Auditing Work? on page 760.

What are the Levels of Auditing?

Levels of Auditing on page 693 explains the two levels of auditing.

What are the Prerequisites to Enable Auditing?

Ensure that you perform the prerequisites mentioned in Managing Auditing on page 757 before enabling auditing.

How to Enable or Disable Auditing of Data Access Operations?

To enable or disable auditing of data access operations, see Enabling and Disabling Auditing of Data Access Operations on page 758.

What is Audited for Data Access Operations?

Auditing Data Access Operations on page 698 describes the data access operations that are audited.

How to Enable or Disable Auditing of Cluster Administration Operations?

To enable or disable auditing of cluster administration operations, see Enabling and Disabling Auditing of Cluster Administration on page 758.

What is Audited for Cluster Administration Operations?

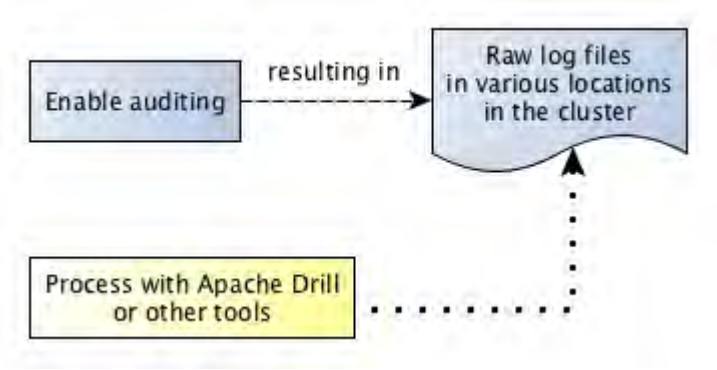
Auditing Cluster Operations on page 696 describes the operations that are audited on a cluster.

How to Selectively Audit MapR Objects?

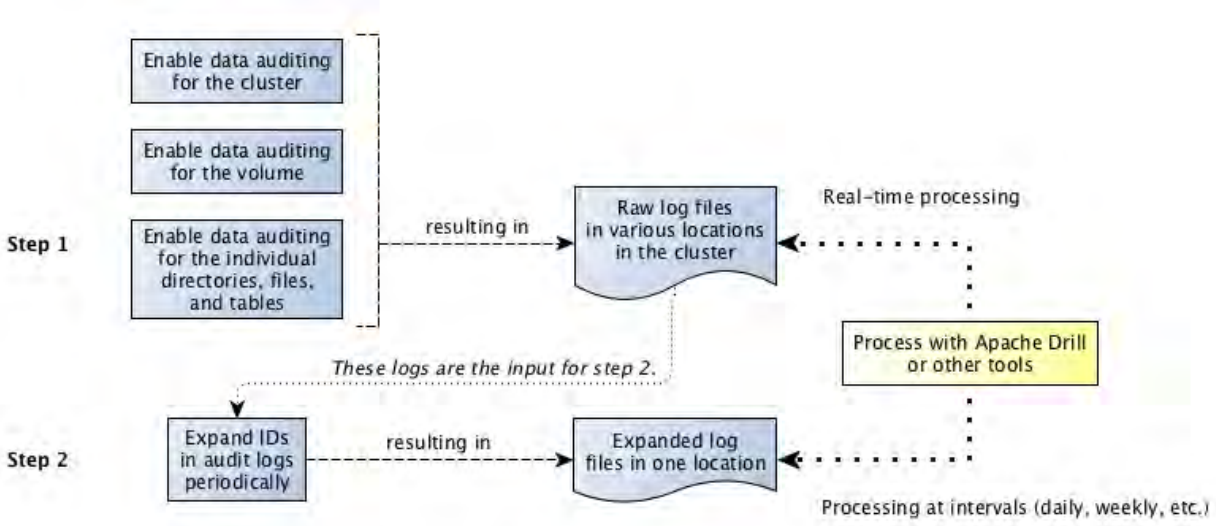
To selectively audit MapR Objects, see Selective Auditing of MapR File System, MapR Database Table, and MapR Event Store For Apache Kafka Operations Using the CLI on page 761.

How to use Audit Logs?

After you enable auditing, audit records immediately start to be recorded in audit logs. You can use Apache Drill or other tools to process these logs. The following diagram shows the workflow for processing audit logs of cluster-administration operations:



The next diagram shows the workflow for processing audit logs of filesystem and table operations.



The step "Expand IDs in log files periodically" refers to the use of the `expandaudit` utility. Raw audit logs contain file identifiers, volume identifiers, and user identifiers. The `expandaudit` utility looks up the names that are associated with those identifiers and puts them in new copies of the audit logs. In addition, the MapR audit streaming feature uses an API to convert file and volume IDs. The information on audit log files can be used to interpret auditing messages.

How to Stream Audit Logs?

To stream audit logs, see Streaming Audit Logs on page 701.

How to Enable or Disable Audit Streaming

To enable or disable audit streaming, see Enabling and Disabling Audit Streaming Using the CLI on page 764.

Levels of Auditing

Describes the two levels of auditing and the requirements to enable each level.

There are two levels of auditing:

- Auditing for cluster level operations
- Auditing of filesystem, table, and stream operations

In contrast to auditing cluster-level operations, auditing of filesystem, table, and stream operations needs to be enabled at multiple levels. For auditing file, table, and stream operations, you must first enable auditing at the cluster level and then enable auditing at the volume level. If you want:

- Granular or selective auditing of content in the volume, you must also enable auditing on each individual directory, file, table, and/or stream in the volume, recursively from the root directory, using the `hadoop` command. If auditing is enabled at the root directory, all new files inherit the property.
- To audit all content (files, tables, and/or streams) in the volume, you can set the `forceaudit` parameter at the volume level, irrespective of what is set (or whether or not auditing is enabled) at the individual file, table, and/or stream level.

The following table summarizes the requirements:

For this type of auditing...	You must enable...	Using...
Cluster-level operations	Auditing at the cluster level	`audit cluster` on page 1553 command
Granular or selective auditing of content (files, tables, and streams) in the volume	1. Auditing at the cluster level 2. Auditing at the volume level 3. Auditing on each individual file, table, and/or stream in the volume, recursively from the root directory	1. `audit cluster` on page 1553 command 2. `audit data` on page 1553, `volume create` on page 1931, or `volume modify` on page 2005 command 3. `hadoop mfs` on page 5373 command
Auditing all content (files, tables, and streams) in the volume (whether or not auditing is selectively enabled or disabled on the individual file, table, or stream)	1. Auditing at the cluster level 2. Auditing at the volume level	1. `audit cluster` on page 1553 command 2. `audit data` on page 1553, `volume create` on page 1931, or `volume modify` on page 2005 command

In the following diagram, the illustration on the left shows data auditing enabled at three levels: the cluster level, through the `maprcli audit data` command; the volume level, through any of the three volume commands shown in the diagram; and the level of the individual directory, file, table, or stream, recursively from the root directory, using the `hadoop` command. This allows you to include and/or exclude specific directories, files, tables, and streams for auditing. If auditing is not enabled at any one of these levels, operations on an object are not logged.

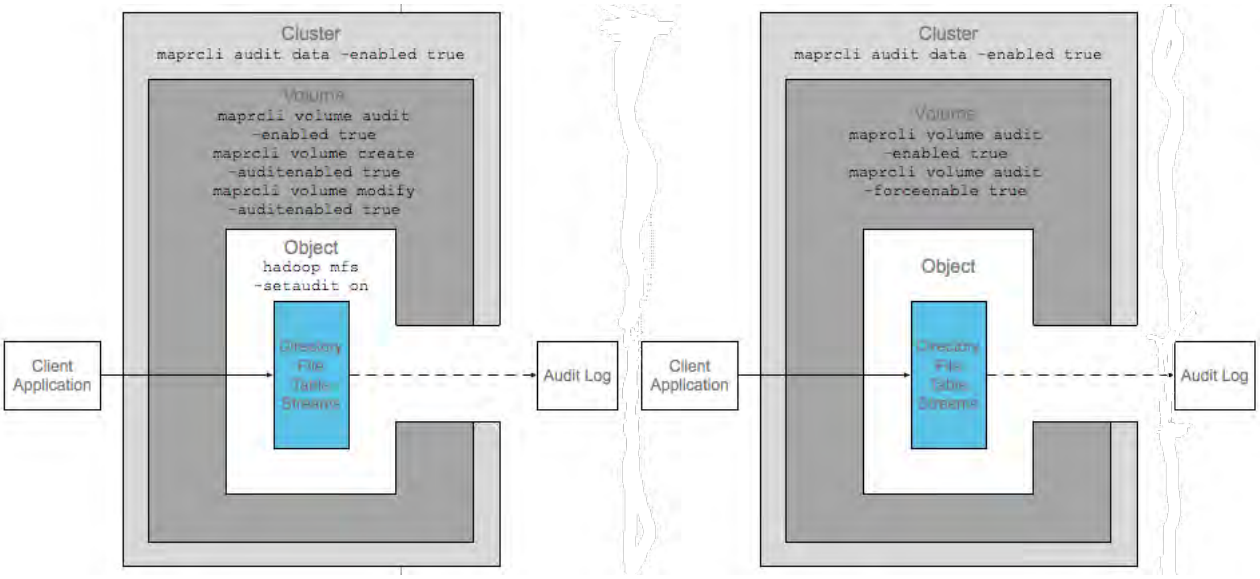
Alternatively, after enabling auditing at the cluster level, you can enforce auditing for all directories, files, tables, and streams at the volume level itself, irrespective of audit setting at the individual file, table, and/or stream level, using:

- `auditenabled` and `forceauditenable` parameters with the `volume create` on page 1931 or `volume modify` on page 2005 command.
- `enabled` and `forceenable` parameters with the `volume audit` on page 1922 command.

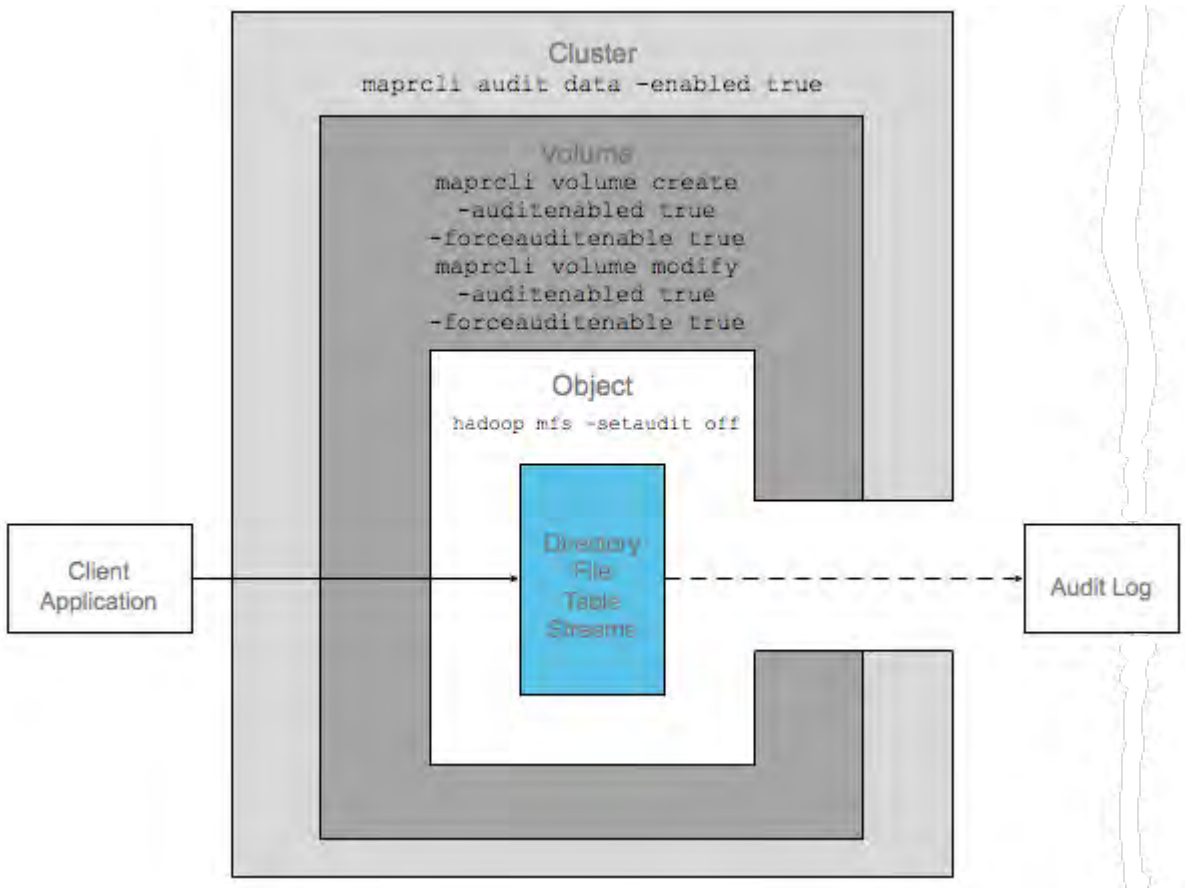
The illustration on the right shows auditing enabled at two levels: the cluster level, through the `audit data` on page 1553 command and the volume level through `volume audit` on page 1922 command (`enabled` and `forceenable` parameters).

Note: You can enable auditing at the volume level using the `volume create` on page 1931 and `volume modify` on page 2005 commands also.

As all levels are enabled, operations that, for example, a client application makes on a directory, file, table, or stream are recorded in an audit log.

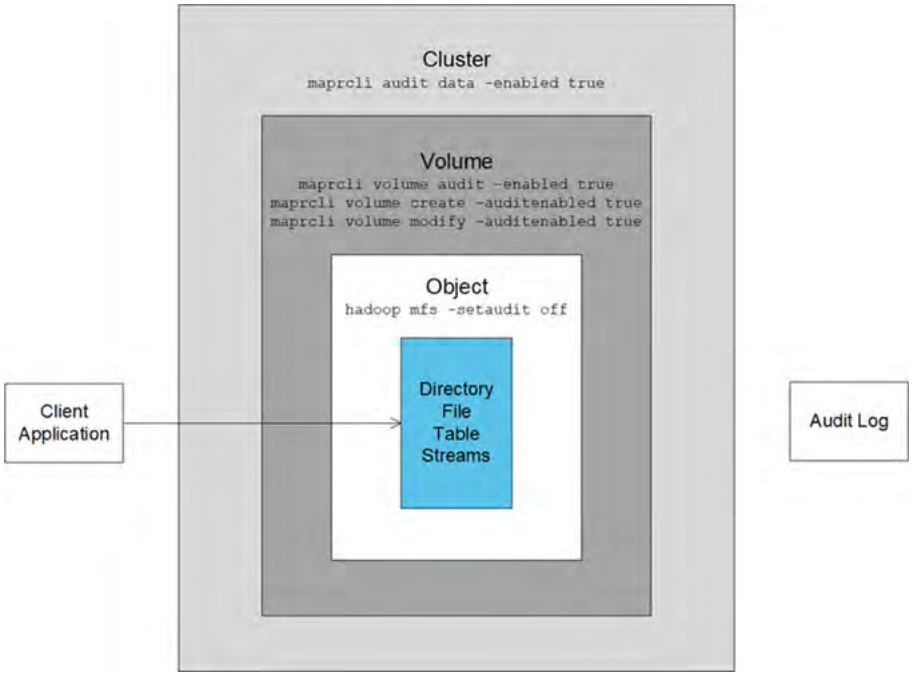


To state another example, in the following diagram, auditing is enabled at the cluster level using the `audit data` on page 1553 command and at the volume level through the `auditenabled` and `forceauditenable` parameters set using any one of the volume commands. Also note that although auditing is explicitly disabled at the directory, file, table, and/or stream level, operations on all directories, files, tables, and streams in the volume are audited because `forceauditenable` is set to `true` at the volume level.

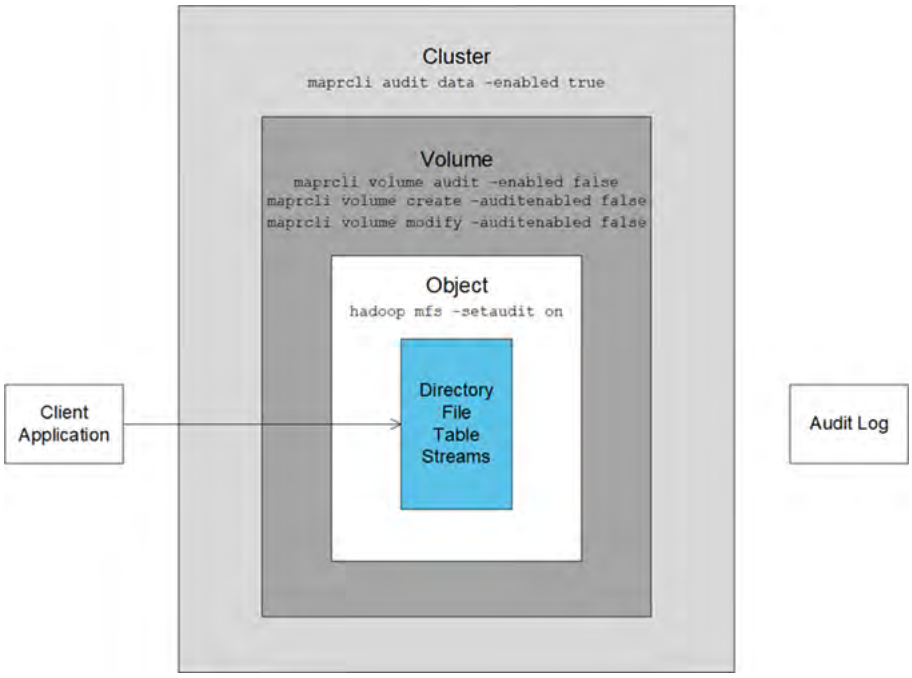


For granular or selective auditing, the following diagram shows auditing enabled at the cluster level and the volume level (with just the `auditenabled` parameter), but not on the directory, file, table, or stream on

which an operation is performed. Although the two higher levels are enabled for auditing, the operation is not logged in an audit log because the objects on which auditing has to be performed is not enabled for auditing.



For granular or selective auditing, as a final example, in the next diagram auditing is enabled on the individual directory, file, table, or stream, and at the cluster level. However, auditing is not enabled at the volume level. Therefore, the operation that the client application performs on the object is not recorded in an audit log.



Auditing Cluster Operations
Explains the operations that are audited for a cluster.

The following types of operations are audited when you run the `maprcli audit cluster` command on a cluster:

last-updated: Apr 11, 2022

- All `maprcli` commands, REST calls, and actions in the Control System that have effects at the cluster level, including those that enable auditing, are audited.
- All authentications to the Control System and authentications to MapR clusters via `maprlogin` are audited.
- All volume level tiering operations are audited.

Audit records for these operations are recorded in the following audit logs:

Audit logs for operations related to cluster management and authentications to clusters via maprlogin

Every CLDB operation is logged in the local filesystem of the CLDB node that responded to the operation. The log file is `/opt/mapr/logs/cldbaudit.log.json`.

Audit logs for maprcli commands, REST API calls, and actions in the Control System

Executions of `maprcli` commands, REST API calls, and actions in the Control System are logged in the local filesystem on the nodes where they are executed. Log files are located at `/opt/mapr/mapr-cli-audit-log/audit.log.json`. To see what information is recorded in typical log entries, see Example Log Entries for Audited maprcli Command Executions, REST API Calls, and Actions in the Control System.

The following `maprcli` commands, as well as their equivalent REST API calls and actions in the Control System, are also logged in audit logs on the servers where they are processed.

Command Family	Commands
acl	`acl edit, acl set, acl show`
audit	`audit cluster, audit data, audit info`
blacklist	`blacklist listusers, blacklist user`
cluster	`cluster mapreduce get, cluster mapreduce set`
config	`config load, config save`
entity	`entity info, entity list, entity modify`
license	`license add, license addcrl, license apps, license list, license listcrl, license remove, license showid`
nagios	`nagios generate`
rlimit	`rlimit get, rlimit set`
schedule	`schedule create, schedule list, schedule modify, schedule remove`
virtualip	`virtualip add, virtualip edit, virtualip list, virtualip move, virtualip remove`

last-updated: Apr 11, 2022

volume	`volume compact, volume container move, volume container switchmaster, volume create, volume fixmountpath, volume info, volume list, volume mirror push, volume mirror start, volume mirror stop, volume modify, volume mount, volume move, volume offload, volume recall, volume remove, volume rename, volume showmounts, volume snapshot list, volume snapshot preserve, volume snapshot remove, volume tierstats, volume tierjobabort, volume tierjobstatus, volume unmount`  **Note:** These commands are not audited: `volume dump create, volume dump restore, volume link create, volume link remove, volume snapshot create`

Audit logs for authentications to the Control System

Every attempt at authentication to the Control System, whether successful or unsuccessful, is logged to the local filesystem in `/opt/mapr/logs/authaudit.log.json` on the webserver node where an attempt was made.

Audit logs for volume level tiering operations

All volume level tiering operations, whether successful or unsuccessful, are logged in the `/opt/mapr/logs/cldbaudit.log.json` file.

Auditing Data Access Operations

Describes MapR File System, MapR Database, and MapR Event Store For Apache Kafka operations that are audited by default, and operations that can be selectively enabled or disabled for auditing.

This type of auditing is for operations that are managed by the MapR File System, MapR Database, and MapR Event Store For Apache Kafka. These operations take place within volumes and have effects at the level of the MapR filesystem.

Auditing of Operations on Directories and Files

The following table shows whether (`Y`) or not (`N`) the following operations on files and directories are audited. In the table, the operations with `Y` in the **Selective Auditing Support** column can be included and/or excluded from auditing. Operations with `N` in the **Selective Auditing Support** column are audited by default and cannot be excluded from auditing. Use the name specified in the **Operation Name to use for Selective Auditing** column when you run the maprcli command to enable or disable auditing for that operation.

Operation	Name in Audit Logs	Operation Name to use for Selective Auditing	Directories	Files	Selective Auditing Support
Change group owner	CHGRP	chgrp	Y	Y	Y
Change owner	CHOWN	chown	Y	Y	Y
Change permissions	CHPERM	chperm	Y	Y	Y
Create	CREATE	create	N/A	Y	Y
Create symbolic link	CREATESYM	createsym	Y	Y	Y
Delete file	DELETE	delete	N/A	Y	Y
Disable auditing	DISABLEAUDIT	N/A	Y	Y	N

last-updated: Apr 11, 2022

Operation	Name in Audit Logs	Operation Name to use for Selective Auditing	Directories	Files	Selective Auditing Support
Enable auditing	ENABLEAUDIT	N/A	Y	Y	N
Offload file to tiered storage	FILE_OFFLOAD	fileoffload or filetieroffloadevent	N/A	Y	Y
Recall file from tiered storage	FILE_RECALL	filerecall or filetierrecallevent	N/A	Y	Y
Scan offset ranges owned by given FID. Used in tiered operations to get owned offsets during offload and recall operations.	FILE_SCAN	filescan	N/A	Y	Y
Abort ongoing offload or recall of file	FILE_TIER_JOBABORT	filetierjobabort	N/A	Y	Y
Retrieve status for an existing file level tier job (offload/recall)	FILE_TIER_JOBSTATUS	filetierjobstatus	N/A	Y	Y
Audit event generated on file server while purging data during offload operation	FILE_TIER_OFFLOAD_EVENT	filetieroffloadevent	N/A	N	Y
Audit event generated on file server while recalling data during recall operation	FILE_TIER_RECALL_EVENT	filetierrecallevent	N/A	N	Y
Get attributes	GETATTR	geattr	N	N	Y
Get extended attributes	GETXATTR	getxattr	Y	Y	Y
Get the mode bits for files/directories accessed over NFS	GETPERM	getperm	Y	Y	Y
List extended attributes	LISTXATTR	listxattr	Y	Y	Y
Lookup	LOOKUP	lookup	Y	Y	Y
Create directory	MKDIR	mkdir	Y	N/A	Y
Read a file	READ	read	N/A	Y	Y
Read a directory	READDIR	readdir	Y	N/A	Y
Remove extended attributes	REMOVEXATTR	removexattr	Y	Y	Y
Rename	RENAME	rename	Y	Y	Y
Delete a directory	RMDIR	rmdir	Y	N/A	Y
Set attributes	SETATTR	setattr	Y	Y	Y
Set extended attributes	SETXATTR	setxattr	Y	Y	Y
Truncate a file	TRUNCATE	truncate	N/A	Y	Y
Write to a file	WRITE	write	N/A	Y	Y

last-updated: Apr 11, 2022

Auditing of Operations on MapR Database Binary Tables and JSON Tables

The following operations on both types of MapR Database tables are audited by default. Operations with `Y` in the **Selective Auditing Support** column can be included or excluded from auditing. Operations with `N` in the **Selective Auditing Support** column are audited by default and cannot be excluded from auditing. Use the name specified in the **Operation Name to use for Selective Auditing** column when you run the maprcli command to enable or disable auditing for that operation.

Operation	Name in Audit Logs	Operation Name to use for Selective Auditing	Selective Auditing Support
Create a column family	DB_CFCREATE	tablecfcreate	Y
Modify a column family	DB_CFMODIFY	tablecfmodify	Y
Delete a column family	DB_CFREMOVE	tablecfdelete	Y
Scan a column	DB_CFSCAN	tablecfscan	Y
Get data	DB_GET	tableget	Y
Perform incremental bulk load	DB_IMPORTBUCKET	N/A	N
Perform full bulk load	DB_IMPORTSEGMENT	N/A	N
Put data	DB_PUT	tableput	Y
Compact a table region	DB_REGIONCOMPACT	N/A	N
Look up a region on the current node	DB_REGIONLOOKUP	N/A	N
Merge two consecutive regions	DB_REGIONMERGE	N/A	N
Split a region into two	DB_REGIONSPLIT	N/A	N
Configure a replica for a table	DB_REPLICAADD	N/A	N
Edit the replica for a table	DB_REPLICAEDIT	N/A	N
List the replicas for a table	DB_REPLICALIST	N/A	N
Remove a replica for a table	DB_REPLICAREMOVE	N/A	N
Scan a table	DB_SCAN	tablescan	Y
Create a table	DB_TABLECREATE	tablecreate	Y
View information about a table	DB_TABLEINFO	tableinfo	Y
Modify a table	DB_TABLEMODIFY	tablemodify	Y
Add an upstream source to a replica	DB_UPSTREAMADD	N/A	N
List all upstream sources for a replica	DB_UPSTREAMLIST	N/A	N
Remove an upstream source for a replica	DB_UPSTREAMREMOVE	N/A	N

Auditing of Operations on MapR Event Store For Apache Kafka

The following operations on MapR Event Store For Apache Kafka are audited by default. Operations with `Y` in the **Selective Auditing Support** column can be included or excluded from auditing. Operations with `N` in the **Selective Auditing Support** column are audited by default and cannot be excluded from auditing. Use the name specified in the **Operation Name to use for Selective Auditing** column when you run the maprcli command to enable or disable auditing for that operation.

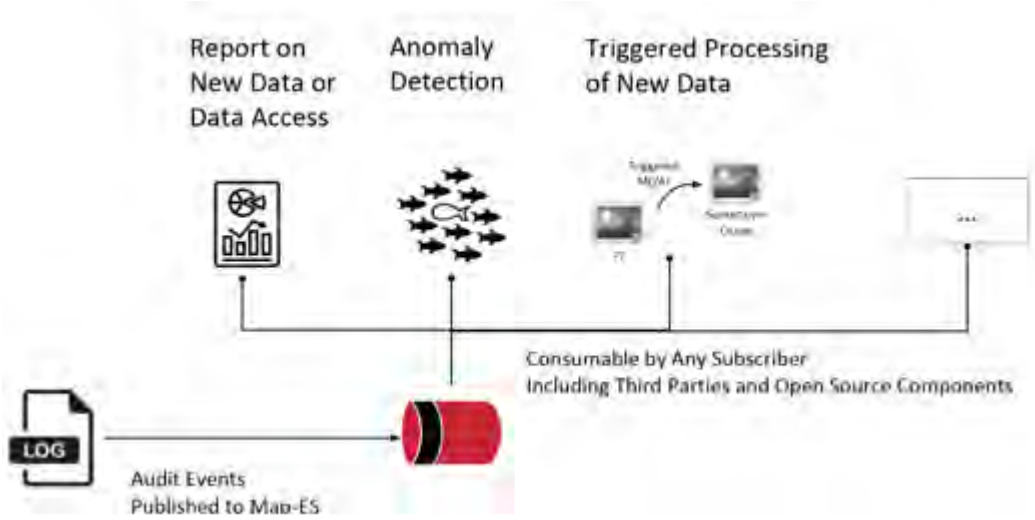
last-updated: Apr 11, 2022

Operation	Name in Audit Logs	Operation Name to use for Selective Auditing	Selective Auditing Support
Modify attributes or permissions of a stream	DB_CFMODIFY	tablecfmodify	Y
Produce messages to topics of a stream	DB_PUT	tableput	Y
Add a replica	DB_REPLICAADD	N/A	N
Edit a replica	DB_REPLICAEDIT	N/A	N
List the replicas for a stream	DB_REPLICALIST	N/A	N
Remove a replica	DB_REPLICAREMOVE	N/A	N
Consume messages from topics of a stream	DB_SCAN	tablescan	Y
Add an upstream source to a replica	DB_UPSTREAMADD	N/A	N
List all upstream sources for a replica	DB_UPSTREAMLIST	N/A	N
Remove an upstream source from a replica	DB_UPSTREAMREMOVE	N/A	N

Streaming Audit Logs
Describes the audit streaming feature and how to consume the audit stream messages.

Audit-streaming (available from v6.0.1) eliminates the need to process the logs nightly using the `expandaudit on page 2096` utility and provides a way to process the audit data in real-time. The audit data is sent as a audit stream as the audit data is generated, opening the possibility for real-time processing of the audit data. You can use it to monitor data access such as:

- Who accessed certain files, tables, and/or streams at certain times
- What type of action is/was performed on the files, tables, and/or streams
- How many failed attempts were made on the files, tables, and/or streams in a certain period
- When did a particular property or configuration change and who changed it



Audit Stream Creation, Location, and Topic

Audit streaming is not enabled by default; you can enable audit streaming using the CLI. If the feature is enabled, MapR File System, MapR Database, and MapR Event Store For Apache Kafka operation-related audit logs and CLDB and auth audit logs are available as MapR Event Store For Apache Kafka topics. The audit-streaming consumer can view all audited operations on a node in the cluster in near real-time by subscribing to one or more topics associated with a node.

The audit stream is created when the hoststats process starts. If the hoststats process is restarted, the audit stream starts publishing to topics from where it left off processing audit logs; some audit log entries might be republished.

The audit log stream topic is available at the following location:

```
/var/mapr/auditstream/
```

Topics named `<clusterName>_<logType>_<nodeName>` are published to the stream (`/var/mapr/auditstream/auditlogstream:<clustername>_<logType>_<nodename>`). Here:

- `<clustername>` is the name of the cluster.
- `<logType>` is the type of the log. Valid types are `cldb`, `auth`, `fs`, and `db` (for both MapR Database and MapR Event Store For Apache Kafka logs).
- `<nodeName>` is the hostname of the node on which the operation was logged.

The message is in JSON format and is identical to the audit log content, as in the following example:

```
{"timestamp":
{"$date":"2017-04-27T10:53:37.239Z"},"operation":"CREATE","uid":0,"ipAddress
":"10.20.30.140","nfsServer":"10.20.30.140","parentFid":"2066.32.131358","ch
ildFid":"2066.33.262630","childName":"abc.txt","volumeId":106738640,"status"
:0}
```

Duration of Audit Stream Topics

Messages in the topics are stored by default for 7 days.

Consuming Audit Stream Messages

Only the `mapr` user can consume the stream. Refer to Sample Cached Consumer Application for Audit Stream on page 2741 and Sample Uncached Consumer Application for Audit Stream on page 2747 for information on consuming the messages using the sample consumers.

Security for Ecosystem Components

Whether you install MapR software by using the MapR Installer or by using manual steps, the platform and its ecosystem components are installed with security ON by default.

MapR Installer: Security with a Single Click

A single option in the MapR Installer controls security for the platform and ecosystem components. The **Enable MapR Secure Cluster** option is checked by default for new installations.

Before starting a new installation, if you want to disable security for the platform and ecosystem components, you can deselect the **Enable MapR Secure Cluster** option. Later, after the cluster is installed, if you want to add or remove security, you can select or deselect the option during an **Incremental Install** operation. For more information, see Enable MapR Secure Cluster.

last-updated: Apr 11, 2022

Note: Note that some exceptions to secure by default can require manual intervention. Also, before enabling security using the Incremental Install function, be sure to review the known issue (IN-1084) related to custom certificates. See MapR Installer Known Issues.

Manual Installation: Security with configure.sh

When you install a MapR cluster by using the manual steps, you configure security on all nodes by using the `configure.sh` script with the `-secure -genkeys` options, as described in Enabling Security on page 154.

Manual installation also creates a cluster that is *secure by default*. For individual ecosystem components, additional security measures are supported, depending on the component. See the notes in the following table.

Security and Ecosystem Components

The MapR platform and the majority of ecosystem components are installed to be secure by default (with some exceptions). The following table lists the EEP 6.0.0 ecosystem components that are secure by default when installed using the MapR Installer or manual installation steps.

Component	Supports Secure by Default	Notes
AsynchHBase	N/A	Security is not applicable. This component acts as a library.
Data Access Gateway 2.0	Yes	For more information, see Understanding the MapR Data Access Gateway on page 750.
Drill	Yes	For more information about Drill security, see Securing Drill on page 3275.
Flume	No	Flume is installed as a library but works like a service after the agents are started. To configure security for Flume, see Configuring Flume on page 3368. Security Exceptions on page 737 notes a security exception for Avro clients.
HBase	Yes	For more information, see HBase Configuration Properties on page 3390.
HBase REST / Thrift Gateway	Yes	For more information, see HBase REST Gateway and HBase Thrift Gateway Secured By Default to Use SSL on page 3404.
Hive	Yes	For more information, see Hive Security on page 3427.
Httpfs	Yes	For more information, see Configuring HttpFS on page 3610.
Hue	Yes	For more information, see Configure Hue with Security on page 3636.
Impala	No	This component can be configured to run on a secure MapRdata-fabric cluster. Security must be configured manually. See Impala Security on page 3811.

last-updated: Apr 11, 2022

Component	Supports Secure by Default	Notes
Kafka-Connect	Yes	For more information, see Worker Configuration on page 3908.
Kafka-REST	Yes	For more information, see User Impersonation on page 3871 and SSL Security Configuration on page 3870.
KSQL	No	For more information, see KSQL Security on page 3845.
Kafka Streams	No	For more information, see Kafka Streams Security on page 3862.
Livy	Yes	For more information, see Configure Livy on page 3839.
MapR Installer	Yes	For more information, see Using the Enable MapR Secure Cluster Option and Using the Enable MapR DARE Option.
MapR Object Store with S3-Compatible API	Yes	For more information, see S3 Gateway on page 3959.
Myriad	N/A	This component can be configured to run on a secure MapR cluster.
Oozie	Yes	For more information, see Configuring Oozie on a Secure Cluster on page 3990.
Pig	N/A	Security is not applicable. This component acts as a library.
Sentry	No	This component can be configured to run on a secure MapR cluster. Security must be configured manually.
Spark	Yes	For more information, see Spark configure.sh on page 4038.
Sqoop 1	N/A	Security is not applicable. This component acts as a library.
Sqoop2	Yes	For more information, see Configuring Sqoop2 on page 3680.
Timeline Server	Yes	For more information, see Configuring the Timeline Server to Use the Hive-on-Tez User Interface on page 3507.
MapR Monitoring Components		
collectd	Yes	Communicates over MapR streams. See Spyglass on Streams on page 1332.
ElasticSearch	Yes	For additional steps that you can take to enhance security, see Security Exceptions on page 737.
FluentD	Yes	For additional steps that you can take to enhance security, see Security Exceptions on page 737.

Component	Supports Secure by Default	Notes
Grafana	Yes	For additional steps that you can take to enhance security, see Security Exceptions on page 737.
Kibana	Yes	For additional steps that you can take to enhance security, see Security Exceptions on page 737.
OpenTSDB	Yes	Communicates over MapR streams. See Spyglass on Streams on page 1332.

Security Settings for Ecosystem Components

Lists the security settings for all MapR ecosystem components.

The security settings for the various MapR ecosystem components are as follows:

Security Settings for Hadoop/Yarn

File or command: `core-default.xml`	*Description*: Authentication used for the HTTP web-consoles *Default Secure Setting*: `hadoop.http.authentication.type:org.apache.hadoop.security.authentication.server.MultiMechsAuthenticationHandler` *Alternate Value or Change Command*: simple | kerberos | #AUTHENTICATION_HANDLER_CLASSNAME# *Notes*: None
File or command: `core-default.xml`	*Description*: Custom principal of the service *Default Secure Setting*: `hadoop.security.custom.auth.principal.class:com.mapr.security.MapRPrincipal` *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml` & `core-site.xml`	*Description*: LDAP Configuration *Default Secure Setting*: `hadoop.security.group.mapping.ldap.search.filter.user:(&(objectClass=user)(sAMAccountName={0}))` *Alternate Value or Change Command*: None *Notes*: An additional filter to use when searching for LDAP users. The default filter is usually appropriate for Active Directory installations. If connecting to an LDAP server with a non-AD schema, replace the default filter with `(&(objectClass=inetOrgPerson)(uid={0})`. {0} is a special string used to denote where the username fits into the filter. If the LDAP server supports posixGroups, Hadoop can enable the feature by setting the value of this property to *posixAccount* and the value of the `hadoop.security.group.mapping.ldap.search.filter.group` property to *posixGroup*.
File or command: `core-default.xml` & `core-site.xml`	*Description*: Client authentication types

	Default Secure Setting: `hadoop.security.authentication: CUSTOM` *Alternate Value or Change Command*: None *Notes*: None.
File or command: `core-default.xml & core-site.xml`	*Description*: Java class that handles HTTP auth secret *Default Secure Setting*: `hadoop.http.authentication.signature.secret:com.mapr.security.maprauth.MaprSignatureSecretFactory` *Alternate Value or Change Command*: None *Notes*: None.
File or command: `core-default.xml & core-site.xml`	*Description*: Group authentication cache duration *Default Secure Setting*: `hadoop.security.groups.cache.secs:300` *Alternate Value or Change Command*: None *Notes*: None.
File or command: `core-default.xml & core-site.xml`	*Description*: Name of the SignerSecretProvider class to use *Default Secure Setting*: `hadoop.http.authentication.signer.secret.provider:org.apache.hadoop.security.authentication.util.MapRSignerSecretProvider` *Alternate Value or Change Command*: None *Notes*: None.
File or command: `core-default.xml & core-site.xml`	*Description*: Service that manages the MapR ticket *Default Secure Setting*: `yarn.external.token.manager:com.mapr.hadoop.yarn.security.MapRTicketManager` *Alternate Value or Change Command*: None *Notes*: None.
File or command: `core-default.xml & core-site.xml`	*Description*: OS security random device file path *Default Secure Setting*: `hadoop.security.random.device.file.path:/dev/urandom` *Alternate Value or Change Command*: None *Notes*: None.
File or command: `core-default.xml & core-site.xml`	*Description*: Key to set if the registry is secure *Default Secure Setting*: hadoop.registry.secure:false *Alternate Value or Change Command*: true *Notes*: Turning it on, changes the permissions policy from `open access` to restrictions on kerberos with the option of a user adding one or more auth key pairs down their own tree.
File or command: `core-default.xml & core-site.xml`	*Description*: Authentication class name *Default Secure Setting*: `hadoop.log.level.authenticator.class:com`

	`.mapr.security.maprauth.MaprAuthenticator` *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml & core-site.xml`	*Description*:Indicates if administrator ACLs are required to access instrumentation servlets (JMX, METRICS, CONF, STACKS) *Default Secure Setting*: hadoop.security.instrumentation.requires.admin:false *Alternate Value or Change Command*: true *Notes*: None
File or command: `core-default.xml & core-site.xml`	*Description*:The keystores factory to use for retrieving certificates *Default Secure Setting*: hadoop.ssl.keystores.factory.class:org.apache.hadoop.security.ssl.FileBasedKeyStoresFactory *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml & core-site.xml`	*Description*: Comma-separated list of crypto codec implementations for AES/CTR/NoPadding *Default Secure Setting*: hadoop.security.crypto.codec.classes.aes.ctr.nopadding:org.apache.hadoop.crypto.OpensslAesCtrCryptoCodec,org.apache.hadoop.crypto.JceAesCtrCryptoCodec *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml & core-site.xml`	*Description*: The attribute of the group object that identifies the users that are members of the group. *Default Secure Setting*: hadoop.security.group.mapping.ldap.search.attr.member:member *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml & core-site.xml`	*Description*: Logs a warning message, if looking up a single user to group takes longer than the specified number of milliseconds *Default Secure Setting*: hadoop.security.groups.cache.warn.after.ms:5000 *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml & core-site.xml`	*Description*: The attribute applied to the LDAP Search Control properties to set a maximum time limit when searching and waiting for a result *Default Secure Setting*: `hadoop.security.group.mapping.ldap.directory.search.timeout:10000` *Alternate Value or Change Command*: The unit is in milliseconds. Set to `0` if an infinite wait period is desired. Default is `10` seconds.

	Notes: None
File or command: `core-site.xml`	*Description*: MapR service account ("mapr") impersonation *Default Secure Setting*: • `hadoop.proxyuser.mapr.hosts:*` • `hadoop.proxyuser.mapr.groups:*` *Alternate Value or Change Command*: None *Notes*: Set by default in version 6.1 secure install.
File or command: `yarn-site.xml`	*Description*: Defines the authentication used for the timeline server HTTP endpoint. *Default Secure Setting*: `yarn.timeline-service.http-authentication.type:com.mapr.security.maprauth.MaprDelegationTokenAuthenticationHandler` *Alternate Value or Change Command*: Supported values are: `simple / kerberos / #AUTHENTICATION_HANDLER_CLASSNAME # Defaults to simple.` *Notes*: None.
File or command: `yarn-default.xml`	*Description*: The allowed pattern for UNIX user names enforced by the Linux-container-executor when used in Nonsecure mode (use case for this is using cgroups). *Default Secure Setting*: `yarn.nodemanager.linux-container-executor.nonsecure-mode.user-pattern:^[_.A-Za-z0-9][-@_.A-Za-z0-9]{0,255}?[$]?$` *Alternate Value or Change Command*: None *Notes*: The default value is taken from `/usr/sbin/adduser`.
File or command: `core-default.xml` & `core-site.xml`	*Description*: Indicates whether or not to use SSL when connecting to the LDAP server. *Default Secure Setting*: `hadoop.security.group.mapping.ldap.ssl:false` *Alternate Value or Change Command*: true *Notes*: None
File or command: `core-default.xml` & `core-site.xml`	*Description*: An additional filter to use when searching for LDAP groups *Default Secure Setting*: `hadoop.security.group.mapping.ldap.search.filter.group:(objectClass=group)` *Alternate Value or Change Command*: None *Notes*: Change this filter when resolving groups against a non-Active Directory installation. See the description of `hadoop.security.group.mapping.ldap.search.filter.user` to enable `posixGroups` support.

last-updated: Apr 11, 2022

File or command: `core-default.xml & core-site.xml`	*Description*: This setting is the configuration controlling the validity of the entries in the cache containing the `userId` to `userName` and `groupId` to `groupName` mappings that are used by `NativeIO getFstat()`. *Default Secure Setting*: `hadoop.security.uid.cache.secs:14400` *Alternate Value or Change Command*: None *Notes*:None
File or command: `yarn-default.xml`	*Description*: Determines which of the two modes LCE should use on a nonsecure cluster. *Default Secure Setting*: `yarn.nodemanager.linux-container-executor.nonsecure-mode.limit-users:true` *Alternate Value or Change Command*: false *Notes*:Set this value to `true`, to launch all containers as the user specified in `yarn.nodemanager.linux-container-executor.nonsecure-mode.local-user`. Set this value to `false` to run containers as the user who submitted the application.
File or command: `yarn-default.xml`	*Description*: Disable insecure protocols *Default Secure Setting*: `hadoop.ssl.exclude.insecure.protocols:SSLv3,TLSv1` *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml & core-site.xml`	*Description*: Class for user to group mapping (get groups for a given user) for ACL. *Default Secure Setting*: `hadoop.security.group.mapping:org.apache.hadoop.security.JniBasedUnixGroupsMappingWithFallback` *Alternate Value or Change Command*: None *Notes*: The default implementation `org.apache.hadoop.security.JniBasedUnixGroupsMappingWithFallback` determines if the Java Native Interface (JNI) is available. If JNI is available, the implementation uses the API within Hadoop to resolve a list of groups for a user. If JNI is not available, then the shell implementation `ShellBasedUnixGroupsMapping`, is used. This implementation shells out to the Linux/Unix environment with the `bash -c groups` command to resolve a list of groups for a user.
File or command: `core-default.xml & core-site.xml`	*Description*: Class for the 'custom type of authentication' method *Default Secure Setting*: `hadoop.security.custom.rpc.auth.method.class:org.apache.hadoop.security.rpcauth.MaprAuthMethod` *Alternate Value or Change Command*: None *Notes*: None

last-updated: Apr 11, 2022

File or command: `core-default.xml & core-site.xm`	*Description*: The attribute of the group object that identifies the group name *Default Secure Setting*: `hadoop.security.group.mapping.ldap.search.attr.group.name:cn` *Alternate Value or Change Command*: None *Notes*: The default setting is usually appropriate for all LDAP systems.
File or command: `core-default.xml & core-site.xm`	*Description*: The Java secure random algorithm. *Default Secure Setting*: `hadoop.security.java.secure.random.algorithm:SHA1PRNG` *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml & core-site.xm`	*Description*: Indicates whether service-level authorization is enabled *Default Secure Setting*: `hadoop.security.authorization:true` *Alternate Value or Change Command*: false *Notes*: None
File or command: `core-default.xml & core-site.xm`	*Description*: Expiration time for entries in the the negative user-to-group mapping caching, in seconds *Default Secure Setting*: `hadoop.security.groups.negative-cache.secs:30` *Alternate Value or Change Command*: None *Notes*: This setting is useful when invalid users retry frequently. Set a low value for this expiration, since a transient error in group lookup could temporarily lock out a legitimate user. Set this parameter to zero or a negative value, to disable negative user-to-group caching.
File or command: `yarn-default.xml`	*Description*: Linux-container-executor setting *Default Secure Setting*: `yarn.nodemanager.linux-container-executor.nonsecure-mode.local-user:nobody` *Alternate Value or Change Command*: None *Notes*: The UNIX user that containers run as when Linux-container-executor is used in Nonsecure mode (a use case for this is using cgroups) if the `yarn.nodemanager.linux-container-executor.nonsecure-mode.limit-users` is set to `true`.
File or command: `core-default.xml & core-site.xml`	*Description*: Cipher suite for crypto codec. *Default Secure Setting*: `hadoop.security.crypto.cipher.suite:AES/CTR/NoPadding` *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml & core-site.xml`	*Description*: Denotes the buffer size used by `CryptoInputStream` and `CryptoOutputStream`.

last-updated: Apr 11, 2022

	Default Secure Setting: `hadoop.security.crypto.buffer.size:8192` *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml & core-site.xml`	*Description*: Path to the JAAS configuration file *Default Secure Setting*: `hadoop.security.java.security.login.config.jar.path:/mapr.login.conf` *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml & core-site.xml`	*Description*: Indicates if anonymous requests are allowed when using `simple` authentication. *Default Secure Setting*: `hadoop.http.authentication.simple.anonymous.allowed:true` *Alternate Value or Change Command*: false *Notes*: None
File or command: `yarn-default.xml`	*Description*: Indicates if anonymous requests are allowed by the timeline server when using `simple` authentication. *Default Secure Setting*: `yarn.timeline-service.http-authentication.simple.anonymous.allowed:true` *Alternate Value or Change Command*: false *Notes*: None
File or command: `core-default.xml & core-site.xml`	*Description*: Indicates how long (in seconds) an authentication token is valid before it has to be renewed. *Default Secure Setting*: `hadoop.http.authentication.token.validity:36000` *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml & core-site.xml`	*Description*: IPC client fallback. *Default Secure Setting*: `ipc.client.fallback-to-simple-auth-allowed:false` *Alternate Value or Change Command*: true *Notes*: When a client is configured to attempt a secure connection, but attempts to connect to an insecure server, that server may instruct the client to switch to SASL SIMPLE (unsecure) authentication. This setting controls whether or not the client accepts this instruction from the server. When false (the default), the client does not allow the fallback to SIMPLE authentication, but aborts the connection.
File or command: `yarn-default.xml`	*Description*: Initial duration of the MapR ticket *Default Secure Setting*: `yarn.mapr.ticket.expiration:604800000` *Alternate Value or Change Command*: None

	Notes: None
File or command: `core-default.xml` & `core-site.xml`	*Description*: Protocols supported by SSL. *Default Secure Setting*: `hadoop.ssl.enabled.protocols:TLSv1` *Alternate Value or Change Command*: true *Notes*: When a client is configured to attempt a secure connection, but attempts to connect to an insecure server, that server may instruct the client to switch to SASL SIMPLE (unsecure) authentication. This setting controls whether or not the client accepts this instruction from the server. When false (the default), the client does not allow the fallback to SIMPLE authentication, but aborts the connection.
File or command: `core-default.xml` & `core-site.xml`	*Description*: List of excluded ciphers *Default Secure Setting*: `hadoop.ssl.exclude.cipher.suites:SSL_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA,SSL_RSA_EXPORT_WITH_DES40_CBC_SHA, SSL_RSA_EXPORT_WITH_RC4_40_MD5,TLS_DHE_RSA_WITH_AES_128_CBC_SHA,TLS_DHE_RSA_WITH_AES_256_CBC_SHA, TLS_DHE_RSA_WITH_AES_256_CBC_SHA256,TLS_DHE_DSS_WITH_AES_256_CBC_SHA256,TLS_DHE_DSS_WITH_AES_256_CBC_SHA, TLS_DHE_RSA_WITH_AES_128_CBC_SHA256,TLS_DHE_DSS_WITH_AES_128_CBC_SHA256, TLS_DHE_DSS_WITH_AES_128_CBC_SHA` *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml` & `core-site.xml`	*Description*: Indicates whether client certificates are required *Default Secure Setting*: `hadoop.ssl.require.client.cert:false` *Alternate Value or Change Command*: true *Notes*: None
File or command: `core-default.xml` & `core-site.xml`	*Description*: The hostname verifier to provide for `HttpsURLConnections` *Default Secure Setting*: `hadoop.ssl.hostname.verifier:DEFAULT` *Alternate Value or Change Command*: Valid values are: `DEFAULT`, `STRICT`, `STRICT_I6`, `DEFAULT_AND_LOCALHOST`, and `ALLOW_ALL` *Notes*: None
File or command: `core-default.xml` & `core-site.xml`	*Description*: Resource file from which SSL client keystore information is extracted *Default Secure Setting*: `hadoop.ssl.client.conf:ssl-client.xml` *Alternate Value or Change Command*: None *Notes*: This file is looked up in the classpath, and is usually present in the Hadoop `conf/` directory.
File or command: `mapred-default.xml`	*Description*: Buffer size for reading spills from file when using SSL.

	Default Secure Setting: `mapreduce.shuffle.ssl.file.buffer.size:65536` *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml` & `core-site.xml`	*Description*: The keystores factory to use for retrieving certificates. *Default Secure Setting*: `hadoop.ssl.keystores.factory.class:org.apache.hadoop.security.ssl.FileBasedKeyStoresFactory` *Alternate Value or Change Command*: None *Notes*: None
File or command: `core-default.xml` & `core-site.xml`	*Description*: Comma-separated list of crypto codec implementations for AES/CTR/NoPadding. *Default Secure Setting*: `hadoop.security.crypto.codec.classes.aes.ctr.nopadding:` `org.apache.hadoop.crypto.OpensslAesCtrCryptoCodec,org.apache.hadoop.crypto.JceAesCtrCryptoCodec` *Alternate Value or Change Command*: None *Notes*: The first implementation is used, if available. Other implementations are fallbacks.
File or command: `core-default.xml` & `core-site.xml`	*Description*: Resource file from which SSL server keystore information is extracted. *Default Secure Setting*: `hadoop.ssl.server.conf:ssl-server.xml` *Alternate Value or Change Command*: None *Notes*: This file is looked up in the classpath, and is usually present in the Hadoop `conf/` directory.
File or command: `core-default.xml` & `core-site.xml`	*Description*: Configures the HTTP endpoint for Yarn daemons. *Default Secure Setting*: `yarn.http.policy:HTTP_ONLY` *Alternate Value or Change Command*: The following values are supported: • HTTP_ONLY: Service is provided only on HTTP • HTTPS_ONLY: Service is provided only on HTTPS *Notes*: None.
File or command: `core-default.xml` & `core-site.xml`	*Description*: Indicates whether or not to use SSL when connecting to the LDAP server. *Default Secure Setting*: `hadoop.security.group.mapping.ldap.ssl:false` *Alternate Value or Change Command*: None *Notes*: None.

File or command: `core-default.xml & core-site.xml`	*Description*: Enables or disables SSL connections to S3. *Default Secure Setting*: `fs.s3a.connection.ssl.enabled:true` *Alternate Value or Change Command*: false *Notes*: None.
File or command: `mapred-default.xml`	*Description*: Indicates whether to use SSL for for the Shuffle HTTP endpoints. *Default Secure Setting*: `mapreduce.shuffle.ssl.enabled:false` *Alternate Value or Change Command*: true *Notes*: None.

Security Settings for Hive

File or command: `hive-site.xml`	*Description*: Hive client authenticator manager class name *Default Secure Setting*: `hive.security.authenticator.manager:org.apache.hadoop.hive.ql.security.HadoopDefaultAuthenticator` *Alternate Value or Change Command*: None *Notes*: None.
File or command: `hive-site.xml`	*Description*: Enables or disables Hive client authorization *Default Secure Setting*: `hive.security.authorization.enabled:true` *Alternate Value or Change Command*: false *Notes*: None.
File or command: `hive-site.xml`	*Description*: The Hive client authorization manager class name *Default Secure Setting*: `hive.security.authorization.manager:org.apache.hadoop.hive.ql.security.authorization.plugin.fallback.FallbackHiveAuthorizerFactory` *Alternate Value or Change Command*: None *Notes*: None.
File or command: `hive-site.xml`	*Description*: List of comma separated Java regexes *Default Secure Setting*: `hive.security.authorization.sqlstd.confwhitelist:hive\.exec\.pre\.hooks` *Alternate Value or Change Command*: None *Notes*: You can modify configurations parameters that match these regexes when you enable SQL standard authorization.
File or command: `hive-site.xml`	*Description*: Authorization DDL task factory implementation *Default Secure Setting*: `hive.security.authorization.task.factory`

	`:org.apache.hadoop.hive.ql.parse.authorization.HiveAuthorizationTaskFactoryImpl` *Alternate Value or Change Command*: None *Notes*: None
File or command: `hive-site.xml`	*Description*: Comma-separated list of non-SQL Hive commands that users are authorized to execute *Default Secure Setting*: `hive.security.command.whitelist:set,reset,dfs,add,list,delete,reload,compile` *Alternate Value or Change Command*: None *Notes*: None
File or command: `hive-site.xml`	*Description*: Authenticator manager class name to be used in the metastore for authentication. *Default Secure Setting*: `hive.security.metastore.authenticator.manager:org.apache.hadoop.hive.ql.security.HadoopDefaultMetastoreAuthenticator` *Alternate Value or Change Command*: None *Notes*: None
File or command: `hive-site.xml`	*Description*: When set to true, the metastore authorizer authorizes read actions on the database and table *Default Secure Setting*: `hive.security.metastore.authorization.auth.reads:true` *Alternate Value or Change Command*: false *Notes*: None
File or command: `hive-site.xml`	*Description*: Names of authorization manager classes (comma-separated) to be used in the metastore for authorization. *Default Secure Setting*: `hive.security.metastore.authorization.manager:org.apache.hadoop.hive.ql.security.authorization.StorageBasedAuthorizationProvider` *Alternate Value or Change Command*: None *Notes*: The user defined authorization class should implement interface `org.apache.hadoop.hive.ql.security.authorization.HiveMetastoreAuthorizationProvider`. All authorization manager classes have to successfully authorize the metastore API call for the command execution to be allowed.
File or command: `hive-site.xml`	*Description*: If true, the HiveServer2 WebUI is secured with PAM *Default Secure Setting*: `hive.server2.webui.use.pam=true` *Alternate Value or Change Command*: false *Notes*: None

File or command: `hive-site.xml`

Description: Class for PAM authentication

Default Secure Setting:
`hive.server2.webui.pam.authenticator:org.apache.hive.http.security.PamAuthenticator`

Alternate Value or Change Command: None

Notes: None

File or command: `hive-site.xml`

Description: Determines whether the metastore performs authorization checks against the underlying storage for operations such as drop-partition

Default Secure Setting:
`hive.metastore.authorization.storage.check.externaltable.drop:true`

Alternate Value or Change Command: false

Notes: Disallow the drop-partition if the user in question does not have permissions to delete the corresponding directory on the storage

File or command: `hive-site.xml`

Description: Determines whether the metastore performs authorization checks against the underlying storage for operations such as drop-partition

Default Secure Setting:
`hive.metastore.authorization.storage.checks:false`

Alternate Value or Change Command: true

Notes: Disallow the drop-partition if the user in question does not have permissions to delete the corresponding directory on the storage

File or command: `hive-site.xml`

Description: Client authentication types.

Default Secure Setting:
`hive.server2.authentication:PAM`

Alternate Value or Change Command:

- NONE: no authentication check – plain SASL transport
- LDAP: LDAP/AD based authentication
- KERBEROS: Kerberos/GSSAPI authentication
- CUSTOM: Custom authentication provider (use with property `hive.server2.custom.authentication.class`)
- PAM: Pluggable authentication module (added in Hive 0.13.0 with HIVE-6466)
- NOSASL: Raw transport (added in Hive 0.13.0)

Notes: None

File or command: `hive-site.xml`

Description: Use this property in LDAP search queries for finding LDAP group names to which a user belongs

Default Secure Setting:
`hive.server2.authentication.ldap.groupClassKey:groupOfNames`

Alternate Value or Change Command: None

Notes: Use this property to construct a LDAP group search query, and to indicate the `objectClass` of a group. Every LDAP group has a certain `objectClass`. For example: `group`, `groupOfNames`, and `groupOfUniqueNames`.

File or command: `hive-site.xml`

Description: LDAP attribute name on the group object that contains the list of distinguished names for the user, group, and contact objects that are members of the group.

Default Secure Setting: `hive.server2.authentication.ldap.groupMembershipKey:member`

Alternate Value or Change Command: None

Notes: For example: `member`, `uniqueMember`, or `memberUid`. Use this property in LDAP search queries when finding LDAP group names to which a particular user belongs. The value of the LDAP attribute as indicated by this property, should be a full DN for the user or the short username or userid.

For example, a group entry for `fooGroup` containing `member : uid=fooUser,ou=Users,dc=domain,dc=com` helps determine that `fooUser` belongs to LDAP group `fooGroup`.

See Group Membership for a detailed example. You can use this property to find the users, if a custom-configured LDAP query returns a group instead of a user (as of Hive 2.1.1). For details, see Support for Groups in Custom LDAP Query.

File or command: `hive-site.xml`

Description: This property indicates the prefix to use when building the `bindDN` for LDAP connection (when using only `baseDN`).

Default Secure Setting: `hive.server2.authentication.ldap.guidKey:uid`

Alternate Value or Change Command: None

Notes: `bindDN` is `<guidKey>=<user/group>,<baseDN>`. If the configuration uses `userDNPattern` and/or `groupDNPattern`, the `guidKey` is not required. The `guidKey` is required when only the `baseDN` is being used.

File or command: `hive-site.xml`

Description: When `true`, HiveServer2 in HTTP transport mode uses a cookie-based authentication mechanism.

Default Secure Setting: `hive.server2.thrift.http.cookie.auth.enabled:true`

Alternate Value or Change Command: None

Notes: None

File or command: `hive-site.xml`

Description: Sasl QOP value; set it to one of the following values to enable higher levels of protection for HiveServer2 communication with clients.

Default Secure Setting: `hive.server2.thrift.sasl.qop:auth-conf`

	Alternate Value or Change Command: One of: • auth – authentication only (default) • auth-int – authentication plus • integrity protection auth-conf – authentication plus integrity and confidentiality protection *Notes*: Note that setting `hadoop.rpc.protection` to a higher level than HiveServer2 does not make sense in most situations. HiveServer2 ignores `hadoop.rpc.protection` in favor of `hive.server2.thrift.sasl.qop`. This setting is applicable only if HiveServer2 is configured to use Kerberos authentication.
File or command: `hive-site.xml`	*Description*: Applies test settings for HS2 (for example for standard base authorization verification in `FallbackHiveAuthorizer` or in `SQLAuthorizationUtils`). *Default Secure Setting*: `hive.test.authz.sstd.hs2.mode:false` *Alternate Value or Change Command*: None *Notes*: None
File or command: `hive-site.xml`	*Description*: Setting this property to `true` enables HiveServer2 to execute Hive operations as the user making the calls. *Default Secure Setting*: `hive.server2.enable.doAs=true` *Alternate Value or Change Command*: false *Notes*: None
File or command: `hive-site.xml`	*Description*: Indicates whether metastore should use SSL *Default Secure Setting*: `hive.metastore.use.SSL:false` *Alternate Value or Change Command*: false *Notes*: None
File or command: `hive-site.xml`	*Description*: SSL certificate keystore location. *Default Secure Setting*: `hive.server2.keystore.path:/opt/mapr/conf/ssl_keystore` *Alternate Value or Change Command*: None *Notes*: None
File or command: `hive-site.xml`	*Description*: Set this to `true` to use SSL encryption in HiveServer2. *Default Secure Setting*: `hive.server2.use.SSL:true` *Alternate Value or Change Command*: false *Notes*: None
File or command: `hive-site.xml`	*Description*: SSL certificate keystore location for HiveServer2 WebUI.

	Default Secure Setting: `hive.server2.webui.keystore.path:/opt/mapr/conf/ssl_keystore` *Alternate Value or Change Command*: None *Notes*: None
File or command: `hive-site.xml`	*Description*: Set this to `true` to use SSL encryption for HiveServer2 WebUI. *Default Secure Setting*: `hive.server2.webui.use.ssl:true` *Alternate Value or Change Command*: true *Notes*: None
File or command: `hive-site.xml`	*Description*: SSL protocols that need to be disabled *Default Secure Setting*: `hive.ssl.protocol.blacklist:SSLv2,SSLv3` *Alternate Value or Change Command*: None *Notes*: None

Security Settings for Oozie

File or command: `oozie-default.xml` & `oozie-site.xml`	*Description*: Authentication used for Oozie HTTP endpoint *Default Secure Setting*: `oozie.authentication.type=simple` *Alternate Value or Change Command*: The supported values are: • simple • keberos • #AUTHENTICATION_HANDLER_CLASSNAME# *Notes*: None
File or command: `oozie-default.xml` & `oozie-site.xml`	*Description*: Denotes how long (in seconds) an authentication token is valid before it has to be renewed *Default Secure Setting*: `oozie.authentication.token.validity=36000` *Alternate Value or Change Command*: None *Notes*: None
File or command: `oozie-default.xml` & `oozie-site.xml`	*Description*: If not set, a random secret is generated at startup time *Default Secure Setting*: `oozie.authentication.signature.secret=` *Alternate Value or Change Command*: The signature secret for signing the authentication tokens *Notes*: None
File or command: `oozie-default.xml` & `oozie-site.xml`	*Description*: The domain to use for the HTTP cookie that stores the authentication token *Default Secure Setting*: `oozie.authentication.cookie.domain=`

	Alternate Value or Change Command: Set the domain appropriately to enable authentication to work correctly across all Hadoop node web-consoles. *Notes*: None
File or command: `oozie-default.xml & oozie-site.xml`	*Description*: Indicates whether anonymous requests are allowed *Default Secure Setting*: `oozie.authentication.simple.anonymous.allowed=true` *Alternate Value or Change Command*: false *Notes*: This setting is applicable only when using `simple` authentication
File or command: `oozie-site.xml`	*Description*: Controls whether SSL encryption is enabled *Default Secure Setting*: `oozie.https.enabled:true` *Alternate Value or Change Command*: false *Notes*: None
File or command: `oozie-site.xml`	*Description*: Path to a TrustStore file *Default Secure Setting*: `oozie.https.truststore.file:/opt/mapr/conf/ssl_truststore` *Alternate Value or Change Command*: None *Notes*: None
File or command: `oozie-site.xml`	*Description*: Path to a KeyStore file *Default Secure Setting*: `oozie.https.keystore.file:/opt/mapr/conf/ssl_keystore` *Alternate Value or Change Command*: None *Notes*: None
File or command: `oozie-site.xml`	*Description*: Password to the KeyStore *Default Secure Setting*: `oozie.https.keystore.pass:<password>` *Alternate Value or Change Command*: None *Notes*: None
File or command: `oozie-client-env.sh`	*Description*: Configuration for Oozie clients to use SSL *Default Secure Setting*: `export OOZIE_CLIENT_OPTS="${OOZIE_CLIENT_OPTS} -Djavax.net.ssl.trustStore=/opt/mapr/conf/ssl_truststore"` *Alternate Value or Change Command*: None *Notes*: None
File or command: `oozie-site.xml`	*Description*: User impersonation for Oozie *Default Secure Setting*: • `oozie.service.ProxyUserService.proxyuser.mapr.hosts:*`

last-updated: Apr 11, 2022

- `oozie.service.ProxyUserService.proxyuser.mapr.groups:*`

Alternate Value or Change Command: None

Notes: None

Security Settings for HTTPFS

File or command: `httpfs-site.xml`

Description: PAM authentication for `HttpFS`

Default Secure Setting:

- `httpfs.hadoop.authentication.type:multiauth`
- `httpfs.authentication.type:multiauth`

Alternate Value or Change Command: None

Notes: None

File or command: `httpfs-site.xml`

Description: User impersonation for `HttpFS`

Default Secure Setting:

- `httpfs.proxyuser.mapr.hosts:*`
- `httpfs.proxyuser.mapr.groups:*`

Alternate Value or Change Command: None

Notes: None

Security Settings for Hue

File or command: `hue.ini`

Description: Configure HTTPS for Hue UI

Default Secure Setting:

```
[desktop]
  ssl_certificate=${ssl_certificate}
  ssl_private_key=${ssl_private_key}
  ssl_password=${ssl_password}
```

Alternate Value or Change Command: true

Notes: Value is picked from the following files:

```
cat /opt/mapr/hue/hue-4.2.0/desktop/
conf/.isSecure true

cat /opt/mapr/hue/hue-4.2.0/bin/env.d/
20secure

#!/bin/sh HUE_SECURE_FILE="$
{HUE_HOME}/desktop/conf/.isSecure"
if [ -e "$HUE_SECURE_FILE" ]
      && [ $(cat "$HUE_SECURE_FILE")
= "true" ] ; then export
      mechanism=$
{mechanism:-"MAPR-SECURITY"} export
      security_enabled=$
{security_enabled:-"true"} export
      ssl_cacerts=${ssl_cacerts:-"$
{MAPR_HOME}/conf/ssl_truststore.pem"}
```

```
export
      ssl_validate=$
{ssl_validate:-"true"} export
      ssl_certificate=$
{ssl_certificate:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
      ssl_private_key=$
{ssl_private_key:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
      ssl_password=$
{ssl_password:-"mapr123"}
fi
```

File or command: `hue.ini`

Description: Path to PEM truststore, and option to enable/disable certificate verification for SSL-encrypted connections to other services (RM, HS, NM, Spark HS, Oozie, Livy, Sqoop2, HBase, Hive, Impala)

Default Secure Setting:

```
[desktop]
  ssl_cacerts=${ssl_cacerts}
  ssl_validate=${ssl_validate}
```

Alternate Value or Change Command: true

Notes: Values are picked in the same way, as values for the previous parameter. Also, the installer overrides this property with value `false` by creating the following file:

```
cat /opt/mapr/hue/hue-4.2.0/bin/env.d/
30installer
# Do not edit this file. It
was generated automatically by MapR
Installer.
# Disable cetificate verification,
as Installer allows to use node IPs
instead of proper
# hostnames:
export ssl_cacerts=""
export ssl_validate="false"
```

File or command: `hue.ini`

Description: Configure Hue to use MapR-SASL for YARN (RM, NM, HS, Spark HS)

Default Secure Setting:

```
[hadoop]
  [[yarn_clusters]]
    [[[default]]]
      # ...
      # Change this if your YARN
cluster is secured
      # security_enabled=$
{security_enabled}
      # Security mechanism
of authentication none/GSSAPI/
MAPR-SECURITY
      # mechanism=${mechanism}
      # In secure mode(HTTPS), if SSL
```

```
certificates from Resource Manager's
      # Rest Server have to be
verified against certificate authority
      # ssl_cert_ca_verify=false
```

Alternate Value or Change Command: true

Notes: Value is picked from the following files:

```
cat /opt/mapr/hue/hue-4.2.0/desktop/
conf/.isSecure true

cat /opt/mapr/hue/hue-4.2.0/bin/env.d/
20secure

#!/bin/sh
HUE_SECURE_FILE="${HUE_HOME}/desktop/
conf/.isSecure"
if [ -e "$HUE_SECURE_FILE" ] && [ $
(cat "$HUE_SECURE_FILE") = "true" ] ;
then export
   mechanism=$
{mechanism:-"MAPR-SECURITY"} export
   security_enabled=$
{security_enabled:-"true"} export
   ssl_cacerts=${ssl_cacerts:-"$
{MAPR_HOME}/conf/ssl_truststore.pem"}
export
   ssl_validate=$
{ssl_validate:-"true"} export
   ssl_certificate=$
{ssl_certificate:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
   ssl_private_key=$
{ssl_private_key:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
   ssl_password=$
{ssl_password:-"mapr123"}
fi
```

File or command: `hue.ini`

Description: Configure Hue to use MapR-SASL for `HttpFS`

Default Secure Setting:

```
[hadoop]
  [[hdfs_clusters]]
    [[[default]]]
      ...
      # Change this if your HDFS
cluster is secured
      security_enabled=$
{security_enabled}
      # Security mechanism
of authentication none/GSSAPI/
MAPR-SECURITY
      mechanism=${mechanism}
      # Enable mutual SSL
authentication
      # mutual_ssl_auth=False
      # Certificate for SSL connection
      # ssl_cert=keys/cert.pem
```

```
      # Private key for SSL connection
      # ssl_key=keys/
hue_private_keystore.pem
      # In secure mode (HTTPS), if
SSL certificates from YARN Rest APIs
      # have to be verified against
certificate authority
      ## ssl_cert_ca_verify=True
```

Alternate Value or Change Command: true

Notes: Value is picked from the following files:

```
cat /opt/mapr/hue/hue-4.2.0/desktop/
conf/.isSecure true

cat /opt/mapr/hue/hue-4.2.0/bin/env.d/
20secure
#!/bin/sh
HUE_SECURE_FILE="${HUE_HOME}/desktop/
conf/.isSecure"
if [ -e "$HUE_SECURE_FILE" ]
     && [ $(cat "$HUE_SECURE_FILE")
= "true" ] ; then export
     mechanism=$
{mechanism:-"MAPR-SECURITY"} export
     security_enabled=$
{security_enabled:-"true"} export
     ssl_cacerts=${ssl_cacerts:-"$
{MAPR_HOME}/conf/ssl_truststore.pem"}
export
     ssl_validate=$
{ssl_validate:-"true"} export
     ssl_certificate=$
{ssl_certificate:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
     ssl_private_key=$
{ssl_private_key:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
     ssl_password=$
{ssl_password:-"mapr123"}
fi
```

File or command: `hue.ini`

Description: Configure Hue to use MapR-SASL for Oozie

Default Secure Setting:

```
[liboozie] ...
# Requires FQDN in oozie_url if
enabled
security_enabled=${security_enabled}
# Security mechanism
of authentication: none/GSSAPI/
MAPR-SECURITY
mechanism=${mechanism}
```

Alternate Value or Change Command: true

Notes: Value is picked from the following files:

```
cat /opt/mapr/hue/hue-4.2.0/desktop/
conf/.isSecure true
cat /opt/mapr/hue/hue-4.2.0/bin/env.d/
20secure
#!/bin/sh
HUE_SECURE_FILE="${HUE_HOME}/desktop/
conf/.isSecure"
if [ -e "$HUE_SECURE_FILE" ]
     && [ $(cat "$HUE_SECURE_FILE")
= "true" ] ; then export
     mechanism=$
{mechanism:-"MAPR-SECURITY"} export
     security_enabled=$
{security_enabled:-"true"} export
     ssl_cacerts=${ssl_cacerts:-"$
{MAPR_HOME}/conf/ssl_truststore.pem"}
export
     ssl_validate=$
{ssl_validate:-"true"} export
     ssl_certificate=$
{ssl_certificate:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
     ssl_private_key=$
{ssl_private_key:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
     ssl_password=$
{ssl_password:-"mapr123"}
fi
```

File or command: `hue.ini`

Description: Configure Hue to use MapR-SASL for Livy

Default Secure Setting:

```
[spark] ...
  # Whether Livy requires client to
perform Kerberos authentication.
  security_enabled=$
{security_enabled}
  # Security mechanism
of authentication: none/GSSAPI/
MAPR-SECURITY
  mechanism=${mechanism}
```

Alternate Value or Change Command: true

Notes: Value is picked from the following files:

```
cat /opt/mapr/hue/hue-4.2.0/desktop/
conf/.isSecure true
cat /opt/mapr/hue/hue-4.2.0/bin/env.d/
20secure
#!/bin/sh
HUE_SECURE_FILE="${HUE_HOME}/desktop/
conf/.isSecure"
if [ -e "$HUE_SECURE_FILE" ]
     && [ $(cat "$HUE_SECURE_FILE")
= "true" ] ; then export
     mechanism=$
{mechanism:-"MAPR-SECURITY"} export
```

```
     security_enabled=$
{security_enabled:-"true"} export
     ssl_cacerts=${ssl_cacerts:-"$
{MAPR_HOME}/conf/ssl_truststore.pem"}
export
     ssl_validate=$
{ssl_validate:-"true"} export
     ssl_certificate=$
{ssl_certificate:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
     ssl_private_key=$
{ssl_private_key:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
     ssl_password=$
{ssl_password:-"mapr123"}
 fi
```

File or command: `hue.ini`

Description: Configure Hue to use MapR-SASL for Hive

Default Secure Setting:

```
[beeswax] ...
  # Security mechanism
of authentication none/GSSAPI/
MAPR-SECURITY
    mechanism=${mechanism}

  # For secure cluster:

  # Use SASL framework to establish
connection to host.
    use_sasl=true
```

Alternate Value or Change Command: true

Notes: Value is picked from the following files:

```
cat /opt/mapr/hue/hue-4.6.0/desktop/
conf/.isSecure true

cat /opt/mapr/hue/hue-4.6.0/desktop/
conf/env.d/20secure
HUE_SECURE_FILE="${HUE_HOME}/desktop/
conf/.isSecure"
if [ -e "$HUE_SECURE_FILE" ] && [ $
(cat "$HUE_SECURE_FILE") = "true" ] ;
thee
  export mechanism=$
{mechanism:-"MAPR-SECURITY"}
  export security_enabled=$
{security_enabled:-"true"}
  export ssl_cacerts=${ssl_cacerts:-"$
{MAPR_HOME}/conf/ssl_truststore.pem"}
  export ssl_validate=$
{ssl_validate:-"true"}
  export ssl_certificate=$
{ssl_certificate:-"${MAPR_HOME}/conf/
ssl_keystore.pem"}
  export ssl_private_key=$
{ssl_private_key:-"${MAPR_HOME}/conf/
```

```
ssl_keystore.pem"}
fi
```

```
cat /opt/mapr/hue/hue-4.2.0/desktop/
conf/.isSecure true
cat /opt/mapr/hue/hue-4.2.0/bin/env.d/
20secure
#!/bin/sh
HUE_SECURE_FILE="${HUE_HOME}/desktop/
conf/.isSecure"
if [ -e "$HUE_SECURE_FILE" ]
     && [ $(cat "$HUE_SECURE_FILE")
= "true" ] ; then export
     mechanism=$
{mechanism:-"MAPR-SECURITY"} export
     security_enabled=$
{security_enabled:-"true"} export
     ssl_cacerts=${ssl_cacerts:-"$
{MAPR_HOME}/conf/ssl_truststore.pem"}
export
     ssl_validate=$
{ssl_validate:-"true"} export
     ssl_certificate=$
{ssl_certificate:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
     ssl_private_key=$
{ssl_private_key:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
     ssl_password=$
{ssl_password:-"mapr123"}
fi
```

File or command: `hue.ini`

Description: Configure Hue to use MapR-SASL for HBase Thrift (MapR-DB)

Default Secure Setting:

```
[hbase] ...
  # Security mechanism
of authentication none/GSSAPI/
MAPR-SECURITY
  mechanism=${mechanism}
```

Alternate Value or Change Command: true

Notes: Value is picked from the following files:

```
cat /opt/mapr/hue/hue-4.2.0/desktop/
conf/.isSecure true
cat /opt/mapr/hue/hue-4.2.0/bin/env.d/
20secure
#!/bin/sh
HUE_SECURE_FILE="${HUE_HOME}/desktop/
conf/.isSecure"
if [ -e "$HUE_SECURE_FILE" ]
     && [ $(cat "$HUE_SECURE_FILE")
= "true" ] ; then export
     mechanism=$
{mechanism:-"MAPR-SECURITY"} export
     security_enabled=$
{security_enabled:-"true"} export
```

```
     ssl_cacerts=${ssl_cacerts:-"$
{MAPR_HOME}/conf/ssl_truststore.pem"}
export
     ssl_validate=$
{ssl_validate:-"true"} export
     ssl_certificate=$
{ssl_certificate:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
     ssl_private_key=$
{ssl_private_key:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
     ssl_password=$
{ssl_password:-"mapr123"}
fi
```

File or command: `hue.ini`

Description: Configure Hue to use MapR-SASL for Drill

Default Secure Setting:

```
[librdbms]
  [[databases]]
    # ...
    [[[drill]]]
      # ...
      # Security mechanism
of authentication none/GSSAPI/
MAPR-SECURITY.
      mechanism=${mechanism}
```

Alternate Value or Change Command: true

Notes: Value is picked from the following files:

```
cat /opt/mapr/hue/hue-4.2.0/desktop/
conf/.isSecure true
cat /opt/mapr/hue/hue-4.2.0/bin/env.d/
20secure
#!/bin/sh
HUE_SECURE_FILE="${HUE_HOME}/desktop/
conf/.isSecure"
if [ -e "$HUE_SECURE_FILE" ]
     && [ $(cat "$HUE_SECURE_FILE")
= "true" ] ; then export
     mechanism=$
{mechanism:-"MAPR-SECURITY"} export
     security_enabled=$
{security_enabled:-"true"} export
     ssl_cacerts=${ssl_cacerts:-"$
{MAPR_HOME}/conf/ssl_truststore.pem"}
export
     ssl_validate=$
{ssl_validate:-"true"} export
     ssl_certificate=$
{ssl_certificate:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
     ssl_private_key=$
{ssl_private_key:-"${MAPR_HOME}/conf/
ssl_keystore.pem"} export
     ssl_password=$
{ssl_password:-"mapr123"}
fi
```

File or command: `hue.ini`

Description: PAM/LDAP authentication between Hue and Hive

Default Secure Setting:

```
[desktop]
  # ...
  # Default LDAP/PAM/.. username and
password of the Hue user used for
authentication with other services.
  # Inactive if password is empty.
  # e.g. LDAP pass-through
authentication for HiveServer2 or
Impala.
  # Apps can override them
individually.
  auth_username=${MAPR_USER}
  auth_password=<user_password>
 ...

[beeswax]
  # ...
  # Security mechanism
of authentication none/GSSAPI/
MAPR-SECURITY
  mechanism=none
```

Alternate Value or Change Command: true

Notes: None

File or command: `hue.ini`

Description: PAM/LDAP authentication between Hue and Drill

Default Secure Setting:

```
[librdbms]
  [[databases]]
    # ...
    [[[drill]]]
      # ...
      # Security mechanism
of authentication none/GSSAPI/
MAPR-SECURITY.
      mechanism=none
      # Username to authenticate with
when connecting to the database.
      # Used with plain
authentication (mechanism set to
"none").
      user=<username>
      # Password matching the
username to authenticate with when
      # connecting to the database.
      # Used with plain
authentication (mechanism set to
"none").
      password=<password>
      # Execute this script to
produce the database password.
      # This will be used when
password is required and `password`
```

```
is not set.
      # password_script=
```

Alternate Value or Change Command: true

Notes: None

File or command: `hue.ini`

Description: User impersonation between Hue and YARN services (RM, NM, HS) + Spark HS

Default Secure Setting: Enabled by default

Alternate Value or Change Command: false

Notes: Hue always send requests to RM, NM, HS and SparkHS with the `doAs=<impersonation_target>` parameter

File or command: `hue.ini`

Description: User impersonation between Hue and HttpFS

Default Secure Setting: Enabled by default

Alternate Value or Change Command: false

Notes: Hue always send requests to HttpFS with the `doAs=<impersonation_target>` parameter

File or command: `hue.ini`

Description: User impersonation between Hue and Oozie

Default Secure Setting: Enabled by default

Alternate Value or Change Command: false

Notes: Hue always send requests to Oozie with the `doAs=<impersonation_target>` parameter

File or command: `hue.ini`

Description: User impersonation between Hue and Livy

Default Secure Setting: Enabled by default

Alternate Value or Change Command: false

Notes: Hue always send requests to Livy with the the `proxyUser=<impersonation_target>` option

File or command: `hue.ini`

Description: User impersonation between Hue and Hive

Default Secure Setting: `true` (enabled)

Alternate Value or Change Command: false

Notes: Hue automatically detects impersonation settings of Hive from `hive-site.xml`

File or command: `hue.ini`

Description: User impersonation between Hue and HBase Thrift (MapR-DB)

Default Secure Setting: `false` (disabled)

Alternate Value or Change Command: false

Notes: Hue automatically detects impersonation settings of Hive from `hbase-site.xml`

File or command: `hue.ini`

Description: User impersonation between Hue and Drill

Default Secure Setting:

```
[librdbms]
  [[databases]]
    # ...
    [[[drill]]]
      # ...
      # Available options:
      # "impersonation" to enable or
disable outbound impersonation.
      # "principal" of Drill service.
Used when Kerberos authentication is
enabled.
      options='{"impersonation":true}
```

Alternate Value or Change Command: true

Notes: None

File or command: `hue.ini`

Description: Authenticating Hue users with LDAP credentials

Default Secure Setting: TDB

Alternate Value or Change Command: None

Notes: None

File or command: `hue.ini`

Description: Determines which authentication method to use: `search and bind`, or `direct bind`

Default Secure Setting: `search_bind_authentication`

Alternate Value or Change Command: None

Notes: When set to `true`, Hue performs an LDAP search using `bind_dn` and `bind_password` as provided in `hue.ini`. The search can be further limited by the search filter `user_filter`. When set to `false`, Hue performs a direct bind to LDAP using the credentials provided from one of these sources:

- The UPN, formed by concatenating <shortname> (the user name provided on the Hue login page) and nt_domain (if nt_domain is specified)
- The ldap_username_pattern (if nt_domain is not specified)

File or command: `hue.ini`

Description: The NT domain to connect. This parameter is only used with Active Directory.

Default Secure Setting: `nt_domain`

Alternate Value or Change Command: None

Notes: Used with the `direct bind` method of authentication. If `nt_domain` is specified, then `ldap_username_pattern` is ignored.

File or command: `hue.ini`

Description: Used to connect to directory services other than Active Directory.

Default Secure Setting: `ldap_username_pattern`

Alternate Value or Change Command: None

	Notes: Used with the `direct bind` method of authentication. Usually takes the form `cn=<username>,dc=example,dc=com`
File or command: `hue.ini`	*Description*: The backend to use for authenticating users. *Default Secure Setting*: `backend` *Alternate Value or Change Command*: None *Notes*: Set it to `desktop.auth.backend.LdapBackend` for Hue authentication.

Security Settings for Drill

File or command: `drill-override.conf`	*Description*: Determines if encryption on the server is enabled for negotiating privacy with the Drill client. *Default Secure Setting*: `drill.exec.security.user.encryption.sasl.enabled=false` *Alternate Value or Change Command*: true *Notes*: None.
File or command: `drill-override.conf`	*Description*: Determines if the server is enabled for negotiating privacy with another Drillbit. *Default Secure Setting*: `drill.exec.security.bit.encryption.ssl.enabled=true` *Alternate Value or Change Command*: false *Notes*: None.
File or command: `drill-override.conf`	*Description*: TLS/SSL versions allowed *Default Secure Setting*: `drill.exec.impersonation.ssl.protocol: TLSv1.2` *Alternate Value or Change Command*: Other versions are possible *Notes*: None.
File or command: `drill-override.conf`	*Description*: Format of the keystore file *Default Secure Setting*: `javax.net.ssl.keyStoreType: JKS` *Alternate Value or Change Command*: jks, jceks, pkcs12 *Notes*: None.
File or command: `drill-override.conf`	*Description*: Location of the Java keystore file *Default Secure Setting*: `drill.exec.ssl.keyStorePath` *Alternate Value or Change Command*: ssl.server.keystore.location: /opt/mapr/conf/ssl_keystore *Notes*: Using it from MapR Hadoop properties, leveraging it from `drill-distrib.conf` property `drill.exec.ssl.useHadoopConfig: true`

File or command: `drill-override.conf`	*Description*: Password to access the private key from the keystore file. *Default Secure Setting*: `drill.exec.ssl.keyStorePassword` *Alternate Value or Change Command*: ssl.server.keystore.password *Notes*: Using it from MapR Hadoop properties, leveraging it from `drill-distrib.conf` property `drill.exec.ssl.useHadoopConfig: true`
File or command: `drill-override.conf`	*Description*: Format of the truststore file *Default Secure Setting*: `drill.exec.ssl.trustStoreType: JKS` *Alternate Value or Change Command*: jks, jceks, pkcs12 *Notes*: None
File or command: `drill-override.conf`	*Description*: Location of the Java keystore file containing the collection of CA certificates trusted by the Drill client. *Default Secure Setting*: `drill.exec.ssl.trustStorePath` *Alternate Value or Change Command*: ssl.server.truststore.location: /opt/mapr/conf/ssl_truststore *Notes*: None
File or command: `drill-override.conf`	*Description*: Password to access the private key from the keystore file specified as the truststore *Default Secure Setting*: `drill.exec.ssl.trustStorePassword` *Alternate Value or Change Command*: ssl.server.truststore.password *Notes*: None
File or command: `drill-distrib.conf`	*Description*: Changes the underlying implementation to the chosen value *Default Secure Setting*: `drill.exec.ssl.provider: JDK` *Alternate Value or Change Command*: OpenSSL/JDK *Notes*: None
File or command: `drill-distrib.conf`	*Description*: Use MapR SSL trust and key store *Default Secure Setting*: `drill.exec.ssl.useHadoopConfig` *Alternate Value or Change Command*: true *Notes*: None
File or command: `drill-distrib.conf`	*Description*: Drill Web UI HTTPS protocol for encryption *Default Secure Setting*: `drill.exec: { http.ssl_enabled: true, ssl.useHadoopConfig: true }` *Alternate Value or Change Command*: Default from Drill 1.13

	Notes: None
File or command: `drill-distrib.conf`	*Description*: Zookeeper znode ACL for Drill cluster info and query info *Default Secure Setting*: `zk.apply_secure_acl: true` *Alternate Value or Change Command*: false *Notes*: Set by default on MapR Secure cluster with installer in `drill-distrib.conf`. `drill.exec.zk.apply_secure_acl: true`
File or command: `drill-distrib.conf`	*Description*: Drill user impersonation, needed for MapR-DB to work properly with CF access *Default Secure Setting*: `drill.exec.impersonation.enabled: true` *Alternate Value or Change Command*: false *Notes*: Set by default on MapR Secure cluster with installer in `drill-distrib.conf`. `drill.exec.impersonation.enabled: true`, also see impersonation inbound policies for information on setting which users can impersonate others.
File or command: `drill-override.conf`	*Description*: Drill user impersonation, maximum number of hops - when one user creates a view on data and shares with other, how many hops are allowed *Default Secure Setting*: `drill.exec.impersonation.max_chained_user_hops: 3` *Alternate Value or Change Command*: Other numeric values *Notes*: Set by default on MapR Secure cluster with installer in `drill-distrib.conf`.
File or command: `drill-override.conf`	*Description*: Authentication mechanisms *Default Secure Setting*: `drill.exec.security.auth.mechanisms: ["MAPRSASL", "PLAIN"]` *Alternate Value or Change Command*: KERBEROS *Notes*: Set by default on MapR Secure cluster with installer in `drill-distrib.conf`.
File or command: `drill-override.conf`	*Description*: End user encryption mechanism *Default Secure Setting*: `drill.exec.security.user.encryption.sasl.enabled: true` *Alternate Value or Change Command*: Can set `drill.exec.security.user.encryption.ssl.enabled: true` *Notes*: Set by default on MapR Secure cluster with installer in `drill-distrib.conf`. To use SSL, set `drill.exec.security.user.encryption.ssl.enabled: true`.

	To use PLAIN (user/pass) authentication, SASL encryption cannot be set to `true`. You have to set SSL encryption to use PLAIN authentication. You can also use MapR tickets (SASL) with SSL encryption, but only with SSL encryption for both.

Security Settings for Spark

File or command: `spark-defaults.conf`	*Description*: SSL option for file download client (used to download jars and files from HTTPS-enabled servers). *Default Secure Setting*: `spark.ssl.fs.enabled true` *Alternate Value or Change Command*: https://spark.apache.org/docs/2.3.1/security.html *Notes*: None
File or command: `spark-defaults.conf`	*Description*: The password to the private key in the key store. *Default Secure Setting*: `spark.ssl.keyPassword <ssl-keystore-password>` *Alternate Value or Change Command*: None *Notes*: None
File or command: `spark-defaults.conf`	*Description*: Path to the key store file. The path can be absolute or relative to the directory in which the process is started. *Default Secure Setting*: · `spark.ssl.keyStore /opt/mapr/conf/ssl_keystore` *Alternate Value or Change Command*: None *Notes*: None
File or command: `spark-defaults.conf`	*Description*: Password to the key store. *Default Secure Setting*: · `spark.ssl.keyStorePassword <ssl-keystore-password>` *Alternate Value or Change Command*: None *Notes*: None
File or command: `spark-defaults.conf`	*Description*: Path to the trust store file. The path can be absolute or relative to the directory in which the process is started. *Default Secure Setting*: · `spark.ssl.trustStore /opt/mapr/conf/ssl_truststore` *Alternate Value or Change Command*: None *Notes*: None
File or command: `spark-defaults.conf`	*Description*: Password for the trust store. *Default Secure Setting*: · `spark.ssl.trustStorePassword <ssl-truststore-password>` *Alternate Value or Change Command*: None *Notes*: None

File or command: `spark-defaults.conf`	*Description*: The TLS protocol to use. The protocol must be supported by the JVM. *Default Secure Setting*: `· spark.ssl.protocol TLSv1.2` *Alternate Value or Change Command*: None *Notes*: None
File or command: `spark-defaults.conf`	*Description*: Configure encryption for the Spark HTTP file and broadcast servers *Default Secure Setting*: `spark.ssl.enabledAlgorithms TLS_RSA_WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_256_CBC_SHA` *Alternate Value or Change Command*: None *Notes*: None

Security Settings for Livy

File or command: `livy.conf`	*Description*: MapR-SASL authentication *Default Secure Setting*: `livy.server.auth.type = multiauth` *Alternate Value or Change Command*: true *Notes*: None

File or command: `livy.conf`	*Description*: User impersonation with Livy
	Default Secure Setting:
	`livy.impersonation.enabled = true`
	`livy.superusers = <MAPR_USER>`
	Alternate Value or Change Command: true
	Notes: None

File or command: `livy.conf`	*Description*: HTTPS
	Default Secure Setting:
	`livy.keystore`
	`livy.keystore.password`
	`livy.key-password`
	Alternate Value or Change Command: true
	Notes: Values automatically filled on runtime using `com.mapr.web.security.WebSecurityManager`

Security Settings for Tez

File or command: `/opt/mapr/tez/tez-0.8/tomcat/apache-tomcat-9.0.1/conf/server.xml`	*Description*: SSL Config for Tez *Default Secure Setting*: `<Connector port="9444" protocol="org.apache.coyote.http11.Http11NioProtocol" maxThreads="150" SSLEnabled="true" scheme="https" secure="true" keyAlias="edl-dev-r01-tezui" keystoreFile="/opt/mapr/tez/tez-0.8/tomcat/apache-tomcat-9.0.1/conf/bdx1xxx0125.xxxxx.com.jks" keystorePass="xxxxxxxxxx"`

last-updated: Apr 11, 2022

	`keystoreType="JKS" clientAuth="false" sslProtocol="TLS" /> <!-- Define an AJP 1.3 Connector on port 8009 --> <Connector port="8009" protocol="AJP/1.3" redirectPort="9444" />` *Alternate Value or Change Command*: None *Notes*: Tez UI redirectPort value changed to 9444 (default value 8443 conflicts with the Control System)
File or command: `/opt/mapr/elasticsearch/elasticsearch-5.4.1/usr/share/elasticsearch/plugins/search-guard-5/sgconfig/sg_internal_users.yml`	*Description*: Kibana and ElasticSearch login account and password file *Default Secure Setting*: `admin:hash: <$2a$12$6ASxMQEBKYPyGUc10RyleOhz3c8RrvPGb7oqLC9xGGwPxJFwOLJtq>` *Alternate Value or Change Command*: https://docs.datafabric.hpe.com/home/AdministratorGuide/Changing_Password_for_ES_Kibana.html *Notes*: None
File or command: `/opt/mapr/conf/ssl_truststore*` and `/opt/mapr/conf/ssl_keystore*`	*Description*: SSL Keys *Default Secure Setting*: `Created at install, should rarely change, used by all web and REST HTTPS interfaces.` *Alternate Value or Change Command*: Add site specific certificates with keytool utiliity *Notes*: None

Security Settings for Grafana

File or command: /opt/mapr/grafana/grafana-version/etc/grafana/grafana.ini	*Description*: Certificate File *Default Secure Setting*: `/opt/mapr/grafana/grafana-4.6.1/etc/grafana/cert.pem` *Alternate Value or Change Command*: None *Notes*: None
File or command: /opt/mapr/grafana/grafana-version/etc/grafana/grafana.ini	*Description*: Certificate Key *Default Secure Setting*: `/opt/mapr/grafana/grafana-4.6.1/etc/grafana/key.pem` *Alternate Value or Change Command*: None *Notes*: None

Security Exceptions

"Secure by default" means network-safe authentication and encryption. This page describes areas in which secure-by-default capabilities are not yet implemented for the MapR platform or ecosystem components. Included where applicable, are links to more information to help you work around those issues.

Flume

Flume does not support any authentication mechanism for an Avro client. See Configuring Flume on page 3368.

Hive

MapR-SASL is not supported for Hive in HTTP mode.

last-updated: Apr 11, 2022

Hue

Certificate verification is disabled on Hue.

Impala

Impala is not secure by default, but encryption and authentication can be enabled. See Impala Security on page 3811.

KSQL

KSQL does not support encryption between a KSQL client and KSQL server.

NFSv3

NFSv3 is not secure by default, and there are no provisions for authentication or network encryption.

NFSv4

NFSv4 is not secure by default, but it can be secured using Kerberos to enable both encryption and authentication. See Configuring NFSv4 Server for Kerberos on page 1209.

OpenTSDB

There is no authentication or network encryption by default for read access over REST, and authentication and encryption cannot be enabled. However, note that no updates are allowed over REST; therefore, intruders cannot alter cluster metric data.

ZooKeeper

ZooKeeper supports server-to-server authentication by default, but ZooKeeper does not support encryption and cannot be configured to do so.

YARN

YARN is a resource-management and scheduling framework that distributes resource-management and job-management duties. YARN assigns the resource-management and job-management duties as follows:

- ResourceManager: manages cluster resources and tracks resource usage and node health.
- ApplicationMaster: a framework-specific process that negotiates resources for a single application (a single job or a directed acyclic graph of jobs), which runs in the first *container* allocated for the application.
- A YARN component called the HistoryServer archives job metrics and metadata. Status on completed applications is available via REST APIs.

The ResourceManager allocates resources among all the applications running the cluster. The ResourceManager includes a pluggable scheduler, which is responsible for allocating resources according to the resource requirements of the running applications. Current MapReduce schedulers, including the Capacity Scheduler and the Fair Scheduler, can be plugged into the YARN scheduler directly.

Label-based scheduling provides job placement control on a multi-tenant Hadoop cluster. Administrators can control exactly which nodes are chosen to run jobs submitted by different users and groups. An administrator assigns node labels in a text file, then composes queue labels or job labels based on the node labels. When users run jobs, they can place them on specified nodes on a per-job basis (using a job label) or on a per-queue level (using a queue label).

last-updated: Apr 11, 2022

The ResourceManager caches the mapping file, and checks every two minutes (the default monitoring period) for updates. If the file has been modified, the ResourceManager updates the labels for all active ApplicationMasters immediately.

Each application runs an ApplicationMaster to negotiate resources from the ResourceManager. The ApplicationMaster works with the NodeManagers to execute and monitor tasks. The duties of the ApplicationMaster are divided as follows:

- NodeManager: One instance runs on each node, to manage that node's resources.
- Container: An abstraction representing a unit of resources on a node.

The NodeManager provides containers to an application. The ResourceManager and the NodeManager provide the system for distributed management of applications and resources.

ResourceManager

Describes the role of the ResourceManager.

The ResourceManager is mainly concerned with arbitrating available resources in the cluster among competing applications, with the goal of maximum cluster utilization. The ResourceManager includes a pluggable scheduler called the YarnScheduler, which allows different policies for managing constraints such as capacity, fairness, and service level agreements.

The ResourceManager manages resources as follows:

- Each NodeManager takes instructions from the ResourceManager, reporting and handling containers on a single node
- Each ApplicationMaster requests resources from the ResourceManager, then works with containers provided by NodeManagers

The ResourceManager communicates with application clients via an interface called the ClientService. A client can submit or terminate an application and gain information about the scheduling queue or cluster statistics through the ClientService.

Administrative requests are served by a separate interface called the AdminService, through which operators can get updated information about cluster operation.

Behind the scenes, the ResourceTrackerService receives node heartbeats from the NodeManager to track new or decommissioned nodes. The NMLivelinessMonitor and NodesListManager keep an updated status of which nodes are healthy so that the scheduler and the ResourceTrackerService can allocate work appropriately.

A component called the ApplicationMasterService manages ApplicationMasters on all nodes, keeping the scheduler informed. A component called the AMLivelinessMonitor keeps a list of ApplicationMasters and their last heartbeat times, in order to let the ResourceManager know what applications are healthy on the cluster. Any ApplicationMaster that does not heartbeat within a certain interval is marked as dead and re-scheduled to run on a new container.

At the core of the ResourceManager is an interface called the ApplicationsManager, which maintains a list of applications that have been submitted, are running, or are completed. The ApplicationsManager accepts job submissions, negotiates the first container for an application (in which the ApplicationMaster will run) and restarts the ApplicationMaster if it fails.

The ResourceManager and NodeManagers communicate via heartbeats.

Configure the ResourceManager for high availability so that the failure of the ResourceManager service is a not single point of failure for the cluster. High availability of the ResourceManager is configured by default when you run `configure.sh` without specifying the `-RM` parameter.

last-updated: Apr 11, 2022

ApplicationMaster

Describes the role of the ApplicationMaster.

The ApplicationMaster is an instance of a framework-specific library that negotiates resources from the ResourceManager and works with the NodeManager to execute and monitor the granted resources (bundled as containers) for a given application. An application can be a process or set of processes, a service, or a description of work.

The ApplicationMaster is run in a container like any other application. The ApplicationsManager, part of the ResourceManager, negotiates for the container in which an application's ApplicationMaster runs when the application is scheduled by the YarnScheduler.

While an application is running, the ApplicationMaster manages the following:

- Application life cycle
- Dynamic adjustments to resource consumption
- Execution flow
- Faults
- Providing status and metrics

The ApplicationMaster is designed to support a specific framework, and can be written in any language since its communication with the NodeManagers and the ResourceManager is accomplished using extensible communication protocols. The ApplicationMaster can be customized to extend the framework or run any other code. For this reason, the ApplicationMaster is not considered trustworthy, and is not run as a trusted service.

An ApplicationMaster typically requests resources on multiple nodes to complete a job by sending the ResourceManager requests that include locality preferences and attributes of the containers. When the ResourceManager is able to allocate a resource to the ApplicationMaster, it generates a lease that the ApplicationMaster pulls on a subsequent heartbeat. A security token associated with the lease guarantees its authenticity when the ApplicationManager presents the lease to the NodeManager to gain access to the container.

The Application Master heartbeats to the ResourceManager to communicate its changing resource needs, and to let the ResourceManager know it is still alive. In response, the ResourceManager can return a lease on additional containers on other nodes, or cancel the lease on some containers. The ApplicationMaster can then adjust its execution strategy to fit the increase or decrease in available resources. When cluster resources become scarce, the ResourceManager can also request that the ApplicationMaster relinquish some resources. The ApplicationMaster can move work to other running containers in order to give up resources gracefully.

Containers

A YARN container is a result of a successful resource allocation, meaning that the ResourceManager has granted an application a lease to use a specific set of resources in certain amounts on a specific node. The ApplicationMaster presents the lease to the NodeManager on the node where the container has been allocated, thereby gaining access to the resources.

To launch the container, the ApplicationMaster must provide a container launch context (CLC) that includes the following information:

- Environment variables
- Dependencies (local resources such as data files or shared objects needed prior to launch)
- Security tokens

- The command necessary to create the process the application plans to launch

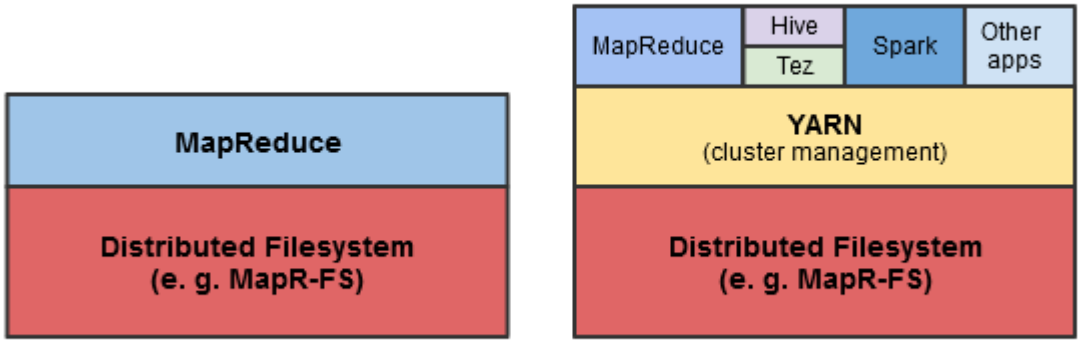
The CLC makes it possible for the ApplicationMaster to use containers to run a variety of different kinds of work, from simple shell scripts to applications to virtual machines.

MapReduce Version 2

Provides an overview of how MapReduce works.

YARN dynamically allocates resources for applications as they execute. The MapReduce version 1 (MRv1) has been rewritten to run as an application on top of YARN; this new version is called MapReduce version 2.0 (MRv2).

Figure 2. A comparison between MapReduce 1.0 and MapReduce 2.0



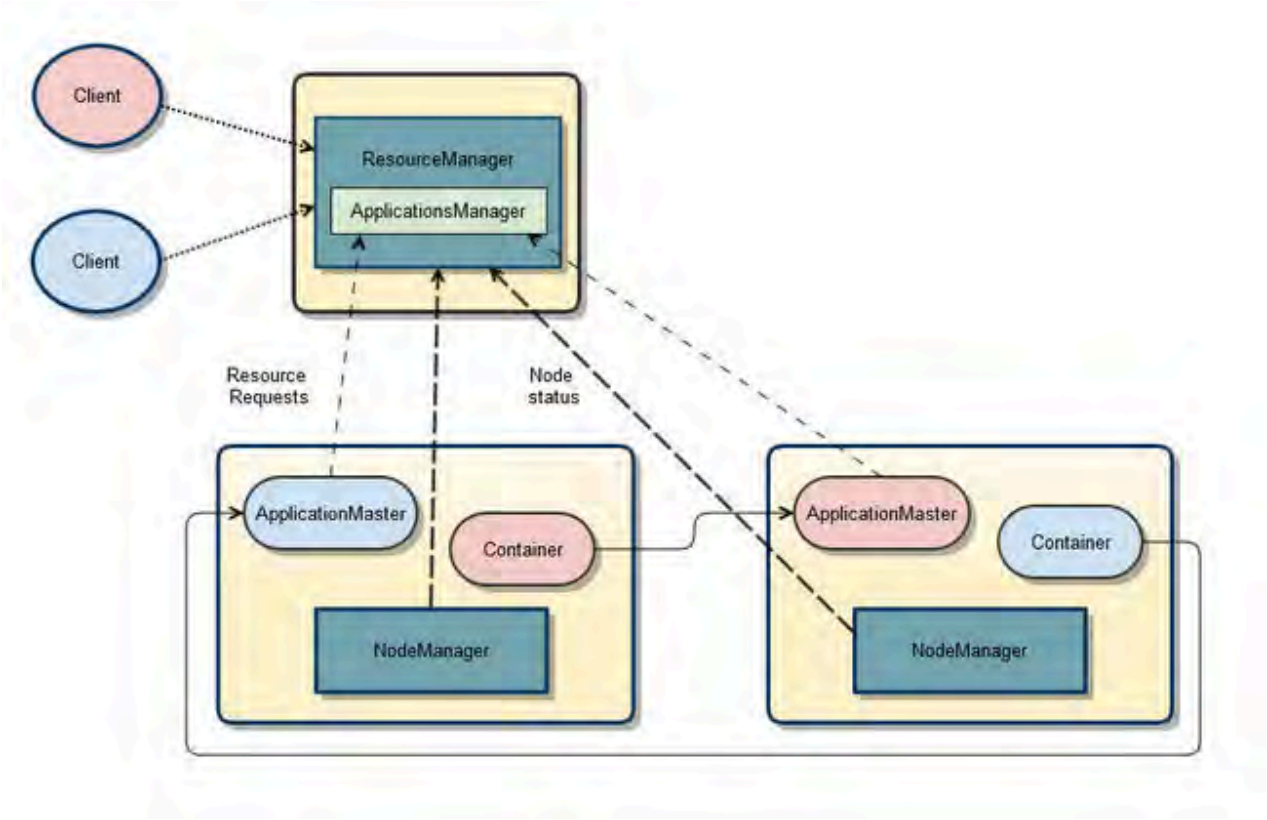
The main advancement in YARN architecture is the separation of resource management and job management, which were both handled by the same process (the JobTracker) in Hadoop 1.x. Cluster resources and job scheduling are managed by the ResourceManager, while resource negotiation and job monitoring are managed by an ApplicationMaster for each application running on the cluster. In MapReduce, each node advertises a relatively fixed number of map slots and reduce slots. This can lead to resource under-utilization, for example, when there is a heavy reduce load and map slots are available, because the map slots cannot accept reduce tasks (and vice versa).

YARN generalizes resource management for use by new engines and frameworks, allowing resources to be allocated and reallocated for different concurrent applications sharing a cluster. Existing MapReduce applications can run on YARN without any changes. At the same time, because MapReduce is now merely another application on YARN, MapReduce is free to evolve independently of the resource management infrastructure.

How Applications Work in YARN

Describes the data flow during application execution in YARN.

The following diagram and steps describe how data flows during application execution in YARN.



The following steps summarize execution of the application:

1. A client submits an application to the YARN ResourceManager, including the information required for the Container Life Cycle (CLC).
2. The ApplicationsManager (in the ResourceManager) negotiates a container and bootstraps the ApplicationMaster instance for the application.
3. The ApplicationMaster registers with the ResourceManager and requests containers.
4. The ApplicationMaster communicates with NodeManagers to launch the containers it has been granted, specifying the CLC for each container.
5. The ApplicationMaster manages application execution. During execution, the application provides progress and status information to the ApplicationMaster. The client can monitor the application's status by querying the ResourceManager or by communicating directly with the ApplicationMaster.
6. The ApplicationMaster reports completion of the application to the ResourceManager.
7. The ApplicationMaster deregisters with the ResourceManager, which then cleans up the ApplicationMaster container.

Direct Shuffle on YARN

Explains the shuffle phase of a MapReduce application.

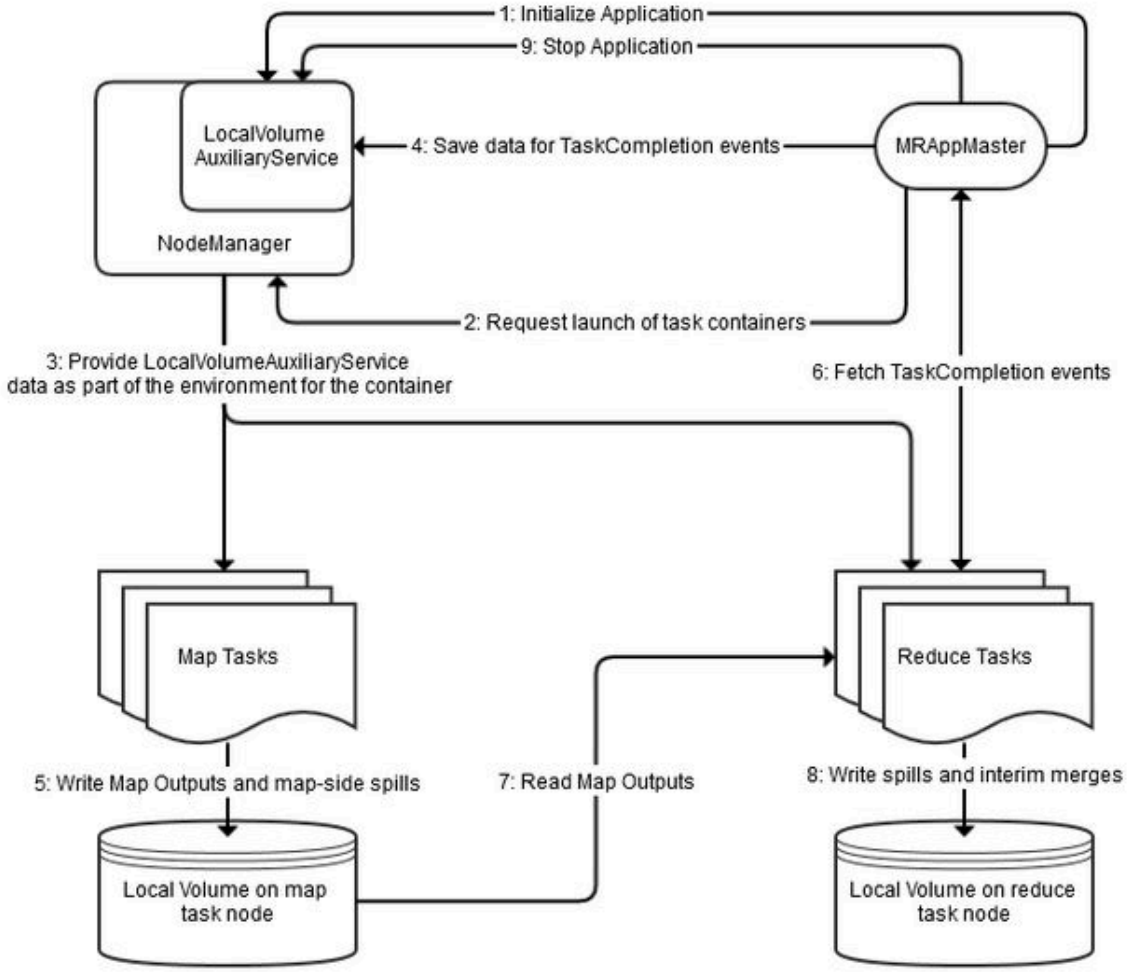
Overview of Direct Shuffle

During the shuffle phase of a MapReduce application, MapR writes to a MapR File System volume limited by its topology to the local node instead of writing intermediate data to local disks controlled by the operating system. This improves performance and reduces demand on local disk space while making the output available cluster-wide.

last-updated: Apr 11, 2022

Direct Shuffle is the default shuffle mechanism for MapR Data Platform. However, you can modify the `yarn-site.xml` and `mapred-site.xml` configuration files to enable Apache Shuffle for MapReduce applications. See Apache Shuffle on YARN.

The LocalVolumeAuxiliaryService runs in the NodeManager process. The LocalVolumeAuxiliaryService manages the local volume on each node and cleans up shuffle data after a MapReduce application has finished executing.



1. The MRAppMaster service initializes the application by calling `initializeApplication()` on the LocalVolumeAuxiliaryService.
2. The MRAppMaster service requests task containers from the ResourceManager. The ResourceManager sends the MRAppMaster information that MRAppMaster uses to request containers from the NodeManager.
3. The NodeManager on each node launches containers using information about the node's local volume from the LocalVolumeAuxiliaryService.
4. Data from map tasks is saved in MRAppMaster for later use in TaskCompletion events, which are requested by reduce tasks.
5. As map tasks complete, map outputs and map-side spills are written to the local volumes on the map task nodes, generating Task Completion events.

last-updated: Apr 11, 2022

6. ReduceTasks fetch Task Completion events from the Application Manager. The task Completion events include information on the location of map output data, enabling reduce tasks to copy data from MapOutput locations.
7. Reduce tasks read the map output information.
8. Spills and interim merges are written to local volumes on the reduce task nodes.
9. MRAppMaster calls `stopApplication()` on the LocalVolumeAuxiliaryService to clean up data on the local volume.

Configuration for Direct Shuffle

The deafult YARN parameters for Direct Shuffle are as follows:

```
<property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle,mapr_direct_shuffle</value>
    <description>shuffle service that needs to be set for Map Reduce to
run</description>
</property>
<property>
    <name>yarn.nodemanager.aux-services.mapr_direct_shuffle.class</name>
    <value>org.apache.hadoop.mapred.LocalVolumeAuxService</value>
</property>
```

The default mapred parameters for Direct Shuffle are as follows:

```
<property>
    <name>mapreduce.job.shuffle.provider.services</name>
    <value>mapr_direct_shuffle</value>
</property>
<property>
    <name>mapreduce.job.reduce.shuffle.consumer.plugin.class</name>
    <value>org.apache.hadoop.mapreduce.task.reduce.DirectShuffle</value>
</property>
<property>
    <name>mapreduce.job.map.output.collector.class</name>
    <value>org.apache.hadoop.mapred.MapRFsOutputBuffer</value>
</property>
<property>
    <name>mapred.ifile.outputstream</name>
    <value>org.apache.hadoop.mapred.MapRIFileOutputStream</value>
</property>
<property>
    <name>mapred.ifile.inputstream</name>
    <value>org.apache.hadoop.mapred.MapRIFileInputStream</value>
</property>
<property>
    <name>mapred.local.mapoutput</name>
    <value>false</value>
</property>
<property>
    <name>mapreduce.task.local.output.class</name>
    <value>org.apache.hadoop.mapred.MapRFsOutputFile</value>
</property>
```


last-updated: Apr 11, 2022

Apache Shuffle on YARN

You can disable Direct Shuffle and enable Apache Shuffle by modifying the configuration options in the `yarn-site.xml` and `mapred-site.xml` files. This page describes how to configure Apache Shuffle for MapReduce applications.

The shuffling phase in Hadoop is the process of transferring mappers intermediate output to the reducers. Direct shuffle increases the load on MapR File System disks. You can enable the Apache Shuffle to reduce the load on MapR File System disks.

Configuration for Apache Shuffle

Add the following property to `yarn-site.xml` file:

```
<property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
</property>
```

Add the following properties to `mapred-site.xml` file:

```
 <property>
    <name>mapreduce.job.shuffle.provider.services</name>
    <value>mapreduce_shuffle</value>
</property>
<property>
    <name>mapreduce.job.reduce.shuffle.consumer.plugin.class</name>
    <value>org.apache.hadoop.mapreduce.task.reduce.Shuffle</value>
</property>
<property>
    <name>mapreduce.job.map.output.collector.class</name>
    <value>org.apache.hadoop.mapred.MapTask$MapOutputBuffer</value>
</property>
<property>
    <name>mapred.ifile.outputstream</name>
    <value>org.apache.hadoop.mapred.IFileOutputStream</value>
</property>
<property>
    <name>mapred.ifile.inputstream</name>
    <value>org.apache.hadoop.mapred.IFileInputStream</value>
</property>
<property>
    <name>mapred.local.mapoutput</name>
    <value>true</value>
</property>
<property>
    <name>mapreduce.task.local.output.class</name>
    <value>org.apache.hadoop.mapred.YarnOutputFiles</value>
</property>
```

Logging Options on YARN

Describes the logging options that are available on YARN.

For YARN applications, there are various logging options to choose from based on the MapR version and the types of applications that you run. In 4.0.2 and later versions, you have the following logging options:

- For MapReduce version 2 (MRv2) applications, the default logging option is to log files on the local filesystem. However, central logging and YARN log aggregation are also available.

last-updated: Apr 11, 2022

- For non-MapReduce applications, the default logging option is to log files on the local filesystem. However,YARN log aggregation is also available.

Centralized Logging for MRv2	Centralized logging provides an application-centric view of all the log files generated by NodeManager nodes throughout the cluster. It enables users to gain a complete picture of application execution by having all the logs available in a single directory, without having to navigate from node to node. The MapReduce program generates three types of log output: • Standard output stream: captured in the `stdout` file • Standard error stream: captured in the `stderr` file • Log4j logs: captured in the `syslog` file Centralized logs are available cluster-wide as they are written to the following local volume on the MapR filesystem: / `var/mapr/local/<NodeManager node>/logs/yarn/userlogs` Since the log files are stored in a local volume directory that is associated with each NodeManager node, you run the `maprcli job linklogs` command to create symbolic links for all the logs in a single directory. You can then use tools such as `grep` and `awk` to analyze them from an NFS mount point. You can also view the entire set of logs for a particular application using the HistoryServer UI.
YARN Log Aggregation	The YARN log aggregation option aggregates logs from the local filesystem and moves log files for completed applications from the local filesystem to the MapR filesystem. This allows users to view the entire set of logs for a particular application using the HistoryServer UI or by running the `yarn logs` command.

Client Connections

The following sections describe how a client connects to local and remote MapR clusters.

How MapR clients Connect to the cluster

Explains how clients connect to a MapR cluster.

The MapR client connects to the cluster via CLDB nodes. When a connection attempt fails, the MapR client returns an error. When an existing connection is no longer available, the MapR client attempts to reconnect to a CLDB node.

For information about installing the MapR client, see MapR Client on page 388.

How the MapR Client Establishes connections to the Cluster

Client applications connect to a cluster via CLDB nodes. To identify the CLDB nodes, check the connection request or the `mapr-clusters.conf` file on the node that submits the connection request. When a client application attempts to connect to the cluster for the first time, the following scenarios can occur:

last-updated: Apr 11, 2022

- At least one of the CLDB nodes is online, in which case the connection is successful.
- None of the CLDB nodes is online, in which case the connection attempt fails.
- The CLDB nodes are listed incorrectly (for example, the IP addresses are incorrect), in which case the connection attempt fails.

When a connection attempt fails, the MapR client returns one of the following errors based on the application type:

Application Type	Error
C Application using HBase API	`ErrorCode = 1`.
C Applications using HDFS API	`NULL handle`
Java Application using HBase, OJAI, or MapR File System API	`java.io.IOException: Could not create FileClient`
Java Application using Kafka API for MapR Event Store For Apache Kafka	`org.apache.kafka.common.errors.NetworkException.`

How the MapR Client Re-establishes Failed Connections to the Cluster

If the CLDB goes down after a client application establishes its first connection to MapR, the client behavior depends on the setting for the `fs.mapr.hardmount` property in the `core-site.xml` file. The `core-site.xml` file is located in the client installation directory.

- If `fs.mapr.hardmount` is set to `true`, the MapR client is nonresponsive as it continuously attempts to reconnect to the CLDB. The MapR client responds when the CLDB comes back online. This is the default behavior.
- If `fs.mapr.hardmount` is set to `false`, the MapR client attempts to connect to each CLDB node that is listed in the `mapr-clusters.conf` file on the node that submitted the connection request. If all of the CLDB nodes are down, the MapR client returns the error EAGAIN/-EAGAIN to the client application. This error indicates that a connection could not be established because the CLDB nodes were not available or because the request timed out for specific reasons. For example, heavy traffic might have caused the network to be slow, or other nodes were unavailable.

Configuring Timeout for Inactive Connections

Configure `fs.mapr.binding.inactive.threshold` in `core-site.xml`. This parameter accepts a value in seconds, and refreshes existing bindings before performing the next I/O, if the time from the previous I/O crosses the given threshold. For example:

```
<property>
<name>fs.mapr.binding.inactive.threshold</name>
<value>600</value>
</property>
```

In this example, when the client tries to send data to the CLDB after a certain idle time, the system checks if the specified time (here 600 seconds, which is 10 minutes) is crossed after the previous request was sent. If so, the system tears down the existing TCP connection and creates a new TCP connection for the file client and CLDB to use for communication.

How Clients Connect to the Replica

Provides an overview of what Replicas are, and how MapR clients connect to them.

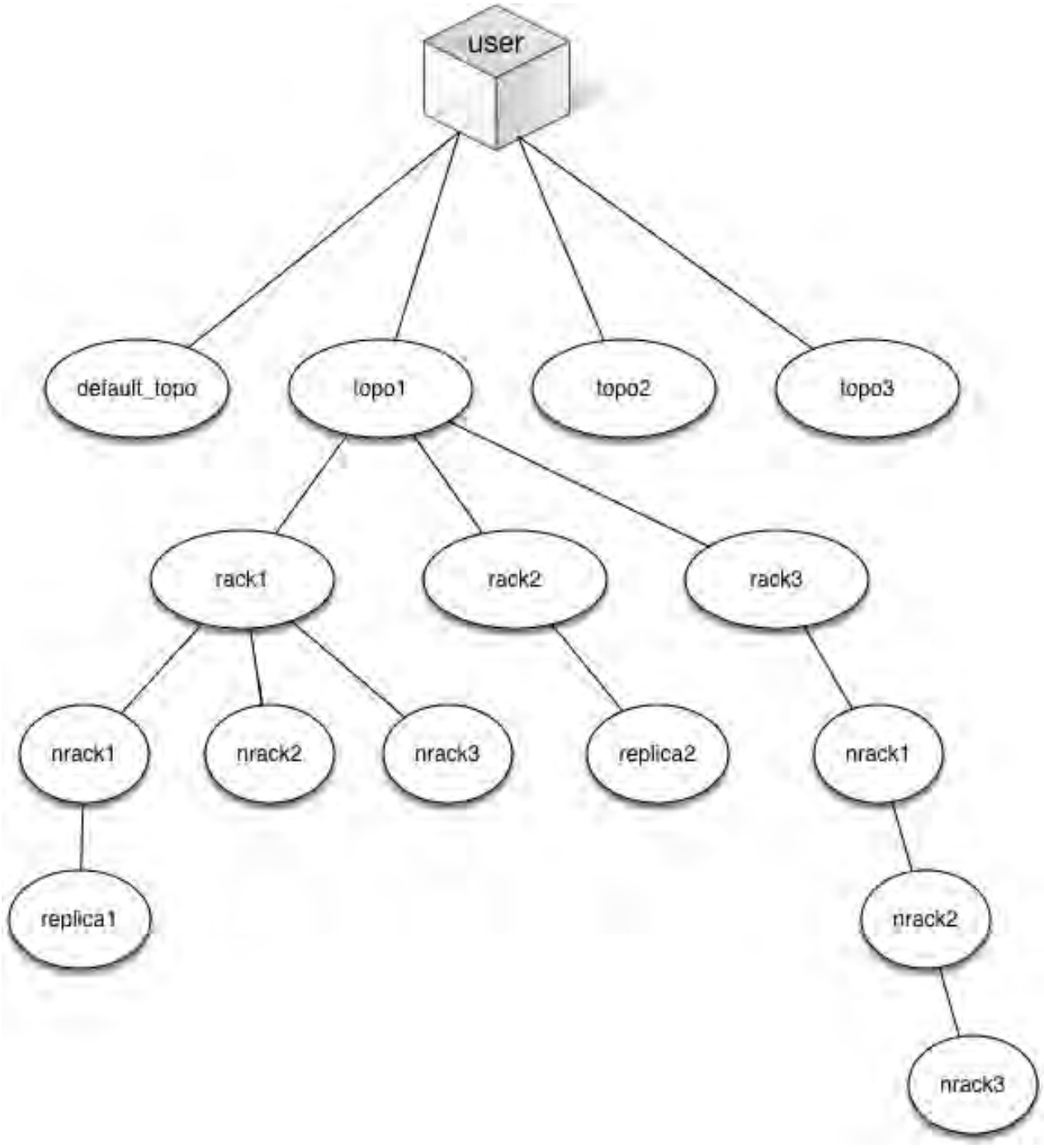
When a client connects to the MapR cluster for I/Os, the client is directed to the replica with the shortest distance. To calculate the distance, every client is given a specific path based on whether the client is connecting from within the cluster or from outside the cluster.

last-updated: Apr 11, 2022

For clients connecting from outside the cluster, because the topology is unknown, and MapR defines the path based on the topology configured in the CLDB configuration property (`cldb.default.node.topology`), the default rack (which is hard-coded as default-rack), and the client IP address. For example, suppose the client IP address is 10.10.10.1 and the value for the CLDB configuration property is `default_topo`. The client topology (or path) is: `/default_topo/default_rack/10.10.10.1`.

For clients on the cluster, the client node has a known topology and the path is built based on that topology, rack, and the client IP address. For example, suppose the client IP address is 10.10.10.1 and the client node topology is `/topo1/rack1`, the client path is: `/topo1/rack1/10.10.10.1`.

Assume the following node topology:



For the client connecting from outside or within the cluster, the replica that the client connects to is calculated based on the client's node topology (or path) and the distance between the nodes on the cluster. When trying to find the nearest replica, the system does a distance calculation based upon how far away the replica is from the MapR client looking for the replica and chooses the replica with the shortest topology or least number of hops from the client node. In the above example, the client connecting from:

- Outside the cluster with the path `/default_topo/default_rack/10.10.10.1` will connect to replica2
- Within the cluster with the path:
  - `/topo1/rack1/10.10.10.1` will connect to replica1
  - `/topo1/rack2/10.10.10.2` will connect to replica2
  - `/topo1/rack3/10.10.10.3` will connect to replica2

Locking Support in MapR

Provides a synopsis of how MapR supports locking for clients.

The MapRMapR File System does not support server-side locking. This means that locks are held by components outside of the filesystem layer and are therefore not shared or enforced globally. As a result, when locking is available, you will need to carefully understand exactly where this enforcement occurs and ensure that all programs using the same locks access them through the same enforcement point. Also, locks, when supported in MapR, are always whole file locks and advisory, not mandatory. The following table describes the locking support in MapR for the clients.

Client Type	Locking Support	Default Value	Notes
Hadoop	No	N/A	Hadoop does not support locking APIs.
FUSE-based POSIX	Yes	Lock	MapR supports advisory locking using kernel locking. This lock is locally enforced on that single Linux host and not shared with other hosts. The lock is only meaningful if all users accessing the file are using FUSE and are on the same host.
Loopback NFS POSIX	No	N/A	No support in MapR for locking. However, you can use Network Lock Manager to lock with the `nolock` option on the `mount` command. This lock is locally enforced on that single Linux host and not shared with other hosts. The lock is only meaningful if all lock users are using loopbacknfs and are on the same host.

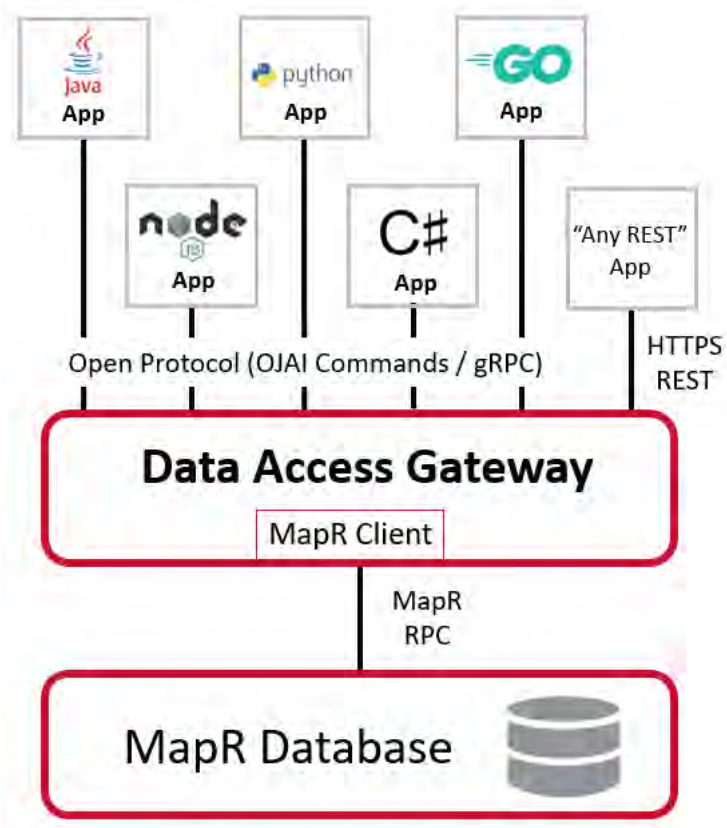
Client Type	Locking Support	Default Value	Notes
NFS v3	No	N/A	No support for locking in MapR. However, you can use Network Lock Manager to lock with the `nolock` option on the `mount` command. This lock is locally enforced on that single Linux host and not shared with other hosts. The lock is only meaningful if all lock users are using NFS v3 and are on the same host.
NFS v4	Yes	NoLock	MapR supports advisory locking for NFS v4 server. The lock is enforced locally on the NFS v4 server, which means the lock is only meaningful if all lock users are using the same NFS v4 server. See Advisory Locking in NFS v4 on page 1218 for more information.

Understanding the MapR Data Access Gateway

The MapR Data Access Gateway is a service that acts as a proxy and gateway for translating requests between lightweight client applications and the MapR cluster.

For the EEP 5.0 release, the service is used by the MapR Database JSON REST API. Starting with the EEP 6.0 release, the service can be used by the Node.js, Python, C#, and Go OJAI clients. Beginning with the EEP 6.3.0 release, the service can be used by the Java OJAI thin client.

The OJAI clients that connect to the Data Access Gateway use gRPC, an open-source RPC framework, to expose the MapR Database OJAI API to the client. RPC message headers for individual messages are encoded using Protocol Buffers. The message payload is encoded using OJAI JSON encoding. Depending on whether your MapR cluster has security enabled, TLS is either enabled or disabled, by default, for the gRPC service. When TLS is enabled, the SSL provider is OpenSSL.



The service runs on nodes in your MapR cluster. You can install the service manually or by using the MapR Installer on page 5395. Both installation methods support upgrades of existing MapR clusters. When installing the service, you can decide the number of nodes on which to install the service. The number of nodes you need depends on your scalability requirements.

Regardless of your scalability requirements, you should install the service on at least two nodes to provide high availability. To load balance requests and to achieve high availability and failover, you must use an external load balancer. For recommendations and best practices when using an external load balancer with gRPC, see gRPC Load Balancing.

The service runs as user `mapr`. However, the service issues all data access calls on behalf of the user requesting the data. For example, if user `john` is running the client application, the service reads data using the authorization of `john`, not `mapr`.

All traffic between the Data Access Gateway and other MapR services is encrypted. This is done regardless of whether the underlying MapR file system volume has encryption enabled.

Warden on page 677 manages the MapR Data Access Gateway. It handles stopping and starting of the service during node failovers and also controls the amount of memory assigned to the service.

Related concepts

Using the MapR Database JSON REST API on page 2696
Starting in the EEP 5.0 release, you can use a REST API to access MapR Database JSON tables. The REST API allows you to use HTTP calls to perform basic operations on MapR Database JSON tables.

Using the Node.js OJAI Client on page 2673
Starting with EEP 6.0, you can use the Node.js OJAI client to write MapR Database JSON applications. The client provides you with a lightweight library that supports the OJAI API. You can connect to MapR Database JSON from middleware components, and add, update, and query documents in a MapR Database JSON table.

Using the Python OJAI Client on page 2678

last-updated: Apr 11, 2022

Starting with EEP 6.0, you can use the Python OJAI client to write MapR Database JSON applications. The client provides you with a lightweight library that supports the OJAI API. You can connect to MapR Database JSON, and add, update, and query documents in a MapR Database JSON table.

Using the C# OJAI Client on page 2688
Starting with EEP 6.1.0, you can use the C# OJAI client to write MapR Database JSON applications. The client provides you with a lightweight library that supports the OJAI API. You can connect to MapR Database JSON, and add, update, and query documents in a MapR Database JSON table.

Using the Go OJAI Client on page 2692
Starting with EEP 6.0.0, you can use the Go OJAI client to write MapR Database JSON applications. The client provides you with a lightweight library that supports the OJAI API. You can connect to MapR Database JSON, and add, update, and query documents in a MapR Database JSON table.

Using the Java OJAI Thin Client on page 2670
Starting with EEP 6.3.0, you can use the Java OJAI Thin Client to write MapR Database JSON applications. The Java OJAI Thin Client provides a lightweight library that supports the OJAI API. You can connect to MapR Database JSON, and add, update, and query documents in a MapR Database JSON table.

Administering the MapR Data Access Gateway on page 1492
The MapR Data Access Gateway is a service that acts as a proxy and gateway for translating requests between lightweight client applications and the MapR cluster. This section describes considerations when upgrading the service, how to modify configuration settings, and how to administer and manage the service.

Related tasks

Installing MapR Data Access Gateway on page 203
This topic includes instructions for using package managers to download and install the MapR Data Access Gateway from the EEP repository.

6.1 Administration

This section describes how to manage the nodes and services that make up a cluster.

This section is for system administrators tasked with managing MapR clusters. Topics include how to manage data by using volumes, how to monitor the cluster for performance, how to manage users and groups, how to add and remove nodes from the cluster, and more. The focus of this section is managing the nodes and services that make up a cluster using the Control System and the CLI or REST API.

Administering Users and Clusters

Lists topics that help manage a MapR cluster.

This section describes processes involved in managing a MapR cluster. Topics include setting up users and groups, adding licenses to the cluster, enabling/disabling auditing of cluster administration, configuring disk and role balancers, and allocating quotas for users and groups and setting cluster reserve limit.

Managing Users and Groups

Provides a brief introduction to user management on a MapR cluster.

The following two users are important when installing and setting up MapR software:

- `root` is used to install MapR software on each node.

last-updated: Apr 11, 2022

- The “MapR user” is the user that MapR services run as (typically named **mapr** or **hadoop**) on each node. The MapR user has full privileges to administer the cluster. Administrative privilege with varying levels of control can be assigned to other users as well.

Before installing MapR, decide on the name, user ID (UID) and group ID (GID) for the MapR user. The MapR user must exist on each node, and the user name, UID and primary GID must match on all nodes.

- When adding a *user* to a cluster node, specify the `--uid` option with the `useradd` command to guarantee that the user has the same UID on all machines.
- When adding a *group* to a cluster node, specify the `--gid` option with the `groupadd` command to guarantee that the group has the same GID on all machines.

MapR uses the native operating system configuration of each node to authenticate users and groups for access to the cluster. If you are deploying a large cluster, you should consider configuring all nodes to use LDAP or another user management system. You can use the Control System to grant specific permissions to particular users and groups. For more information, see Setting User Permissions. Each user can be restricted to a specific amount of disk usage. For more information, see Setting Quotas for Users and Groups.

By default, MapR grants the user `root` full administrative permissions. If the nodes do not have an explicit `root` login, grant full permissions to another user after deployment. See Adding Cluster Permissions on page 755.

On the node where you plan to run the `mapr-apiserver` (the Control System), install Pluggable Authentication Modules (PAM). See PAM Configuration for more information.

You can perform the following procedures to manage users and groups in a MapR cluster using the Control System (click **Admin** > **User Settings**) and the CLI:

Setting Up Email Addresses
Describes how to set up email addresses using the Control System and the CLI.
Setting Up Email Addresses Using the Control System

To set up email addresses for cluster users, in the Control System under **Admin** > **User Settings** > **Email Address**:

1. Choose one of the following to configure the cluster to use an SMTP server to send email:
   - **Use Company Domain** to specify a domain to append after each user name to complete each user's email address
   - **Use LDAP** to obtain each user's email address from an LDAP server.
2. Specify, for:

Use Company Domain

Domain to append after each user name to complete each user's email address in the **user** @ field.

Use LDAP

LDAP Server	The LDAP server address.
LDAP Port	The LDAP server port number. You can select the **Use Secured Port** checkbox to use port 636.
Bind Domain	The bind domain for the users.

Bind Domain Password	The bind password for the users.
Base Domain	The base domain.
UID Attribute	The user ID.
Mail Attribute	The mail attribute.

3. Click **Save Changes**.

Setting Up Email Addresses Using the CLI or the REST API

The basic command to set up email address for a user is:

```
maprcli entity modify -name <entityname> -type 0 -email <email>
```

For complete reference information, see `entity modify` on page 1649.

Setting Up SMTP
Describes how to set up SMTP information using the Control System and the CLI.

You can specify SMTP server information for the cluster using the Control System and the CLI.

Setting Up SMTP Using the Control System

Use the following procedure to configure the cluster to use an SMTP server to send email:

1. Log in to the Control System and click **Admin** > **User Settings** > **SMTP**.
2. Set up the email account the MapR cluster must use to send alerts and other notifications.

Provider	Select **Gmail**, **Office 365**, or **SMTP** from the drop-down menu. If you select **Gmail** or **Office 365**, the SMTP server and port information will be pre-filled for you.
SMTP Server	The SMTP server to use for sending mail.
This server requires an encrypted connection (SSL)	Specifies an SSL connection to SMTP.
SMTP Port	The SMTP port to use for sending mail.
Sender's Full Name	The name that MapR should use when sending email. Example: `MapR Cluster`
Sender's Email Address	The email address that MapR should use when sending email.
Sender's Username	The user name that MapR should use when logging on to the SMTP server.
Sender's SMTP Password	The password that MapR should use when logging on to the SMTP server.

3. Click **Save Changes**.
   An email request is sent to the specified email address. You can check the `/opt/mapr/logs/cldb.log` file if there is a problem sending the email. If there is a problem, also check the fields to make sure that the SMTP information is correct. Click **Revert** if you wish to cancel the changes.

Setting Up SMTP Using the CLI

Use the `maprcli config save` command to set the SMTP server. For example:

```
maprcli config save -values '{"mapr.smtp.provider":"gmail",
               "mapr.smtp.server":"smtp.gmail.com",
               "mapr.smtp.sslrequired":"true",
               "mapr.smtp.port":"465",
               "mapr.smtp.sender.fullname":"Ab  Cd",
               "mapr.smtp.sender.email":"xxx@gmail.com",
               "mapr.smtp.sender.username":"xxx@gmail.com",
               "mapr.smtp.sender.password":"abc"}'
```

For complete reference, see `config save` on page 1586.

Managing Permissions

Provides an overview of managing user permissions at the cluster, volume and file system levels.

You can manage user permissions at the cluster, volume, and filesystem levels. Cluster and volume permissions use access control lists (ACLs) to specify which actions a user can perform on a cluster or volume. File system permissions and *ACE*s control user access to volumes, directories, and files, similar to Linux file permissions. Users get the permissions that are directly assigned to them as well as the permissions assigned to the groups they are in. You must have `fc` permissions to manage permissions.

Adding Cluster Permissions

Describes how to set cluster permissions for users and groups through the Control System and the CLI.

The following table lists the actions that a user can perform on a cluster with the corresponding UI columns and codes used in the cluster *Access Control List (ACL)*:

UI	ACL	Allowed Action
Login	login	Log in to the Control System, use the API and command-line interface, read access on cluster and volumes
Start/Stop Service	ss	Start and stop services
Create Volumes	cv	Create volumes
Create Security Policy	cp	Required to create security policies. Users with Administrator (a) access can assign this permission to other administrators.
Administrator	a	Administrative access (can edit and view *ACL*s, but cannot perform cluster operations)
Full Control	fc	Full control over the cluster. This enables all cluster-related administrative options with the exception of changing the cluster *ACL*s.

Setting Permissions Using the Control System

Complete the following steps to add cluster permissions in the Control System:

1. Log in to the Control System and click **Admin** > **User Settings** > **Permissions**.
2. Under **USER PERMISSIONS**, select the type and specify the name of the user or group in the **Name** field.

last-updated: Apr 11, 2022

3. Select the checkbox associated with the permissions you want to grant to the user or group.
4. Click **Add Another** to add permissions for another user or group.
   Each row lets you assign permissions to a single user or group.

   Note: A user gets the permissions directly granted to the user as well as permissions granted to any group to which the user belongs.
5. Click **Save Changes** to save the changes.

Setting Permissions Using the CLI or the REST API

To set permissions using the CLI, run the following command:

```
/opt/mapr/bin/maprcli acl set
    [ -cluster <cluster name> ]
    [ -group <group> ]
    [ -name <name> ]
    -type cluster|volume
    [ -user <user> ]
```

See `acl set` on page 1531 for complete reference information.

Granting a User Full Control from the Command-Line

The user who has full control over the cluster can manage all aspects of the cluster operation except assign permissions for other users.

Complete the following steps to give full administrative control to a user:

1. Log on to any cluster node as `root` (or use `sudo`).
2. Execute the following command, replacing `<user>` with the username of the account that gets administrative control: `sudo /opt/mapr/bin/maprcli acl edit -type cluster -user <user>:fc`

   For general information about users and groups in the cluster, see Managing Users and Groups.

Removing Cluster Permissions
Describes how to remove cluster permissions using the Control System or the CLI.
Removing Cluster Permissions Using the Control System

Complete the following steps to remove cluster permissions in the Control System:

1. Log in to the Control System and click **Admin** > **User Settings** > **Permissions**.
2. Remove the desired permissions:
   - To remove all permissions for a user or group, click ⛔ associated with the row.
   - To change the permissions for a user or group, deselect the checkbox associated with the permissions you wish to revoke from the user or group.
3. Click **Save Changes** to save the changes.

Removing Cluster Permissions Using the CLI or REST API

The `acl set` command specifies the entire *ACL* for a cluster or volume. Any previous permissions are overwritten by the new values, and any permissions omitted are removed. To remove cluster permissions, run the `acl set` command and omit the permissions to remove. See `acl set` on page 1531 for complete reference information.

last-updated: Apr 11, 2022

Blocking Users Using the CLI

Explains how to block users using the CLI.

You can block users using the CLI. When a user is blocked, all existing tickets for the user are canceled and any request sent by the user that has a ticket older than the blocked timestamp is rejected. For more information, see How Tickets Work on page 1426.

Blocking Users Using the CLI or REST API

The basic command to blacklist a user is:

```
maprcli blacklistuser -user <user name>
```

For complete reference information, see `blacklist user` on page 1555.

Managing the Cluster

Lists the settings for managing the MapR cluster.

You can add licenses, set up auditing of cluster administration, configure disk space balancer tool settings and Role Balancer settings, configure how MapReduce programs run, allocate quotas for users and groups and set the cluster reserve limit, and generate the DNS Gateway record for table replication using both the Control System (click **Admin** > **Cluster Settings**) and the CLI.

Managing Auditing

Provides instructions for using MapR auditing features.

You can enable auditing of cluster administration and data-access operations using the Control System and the CLI. Enabling auditing of the filesystem, table, and streams operations requires running a command on a cluster, a command on individual volumes in the cluster, and a command on individual directories, files, and MapR Database tables and streams within those volumes.

These steps are summarized in the following table:

	Steps to enable auditing			
	Enable auditing of cluster administration	Enable data auditing on the cluster	Enable auditing of individual volumes	Enable auditing of individual directories, files, and MapR Database tables
Auditing of cluster administration		Not applicable	Not applicable	Not applicable

last-updated: Apr 11, 2022

Auditing of directories, files, and MapR Database tables	Not applicable			

Prerequisites for enabling auditing

- If you upgraded your MapR cluster from version 4.1 or earlier, you must enable the auditing feature.
  - Run `maprcli config save -values {"mfs.feature.audit.support":"1"}`
  - To verify that the feature is enabled, run `maprcli config load -json | grep "mfs.feature.audit.support"`
- Only the `root` user or `mapr` user can enable or disable auditing.

Enabling and Disabling Auditing of Cluster Administration
Describes how to enable and disable cluster administration auditing using the Control System and the CLI.

You can enable or disable auditing of cluster-management operations on a MapR cluster using the Control System and the CLI. See Auditing Cluster Operations on page 696 for the complete list of cluster management commands that are audited.

Enabling and Disabling Auditing of Cluster Administration Using the Control System

1. Log in to the Control System and go to the **Auditing** tab in the **Admin** > **Cluster Settings** page.
2. Move the **Enabled** slider to **Yes** to enable and **No** to disable cluster auditing.
3. Click **Save Changes** for the changes to take effect.

Enabling and Disabling Auditing of Cluster Administration Using the CLI or REST API

To enable or disable auditing of cluster-management operations on a MapR cluster, run the `maprcli audit cluster` command.

```
maprcli audit cluster -enabled <true | false>
```

For complete reference information, see `audit cluster` on page 1553.

Enabling and Disabling Auditing of Data Access Operations
Describes how to enable or disable auditing of data-access operations using the Control System and the CLI.

See Auditing Data Access Operations on page 698 for the complete list of data-access operations that can be audited.

Enabling and Disabling Auditing of Data Access Operations Using the Control System

To enable or disable auditing of data-access operations on a cluster:

1. Log in to the Control System and go to the **Auditing** tab in the **Admin** > **Cluster Settings** page.
2. Set the following:

Enabled	Move the slider to **Yes** to enable or to**No** to disable data auditing.
Maximum Size	Set the size in GB, which when reached causes an alarm to be sent to the dashboard on the Control System. The alarm is to notify the cluster administrator that the audit log size is large enough to need administrator intervention. The audit log continues to grow until the administrator takes action or until the retention period ends.
Retain Logs for	Set the period of time in days to keep the data in the audit log. After this period elapses, the content of the file is deleted and new entries are added to the file until the retention period elapses.

3. Click **Save Changes** for the changes to take effect.

Note: This action does not cause auditing to start for operations within the volumes. It only sets a flag that indicates that you allow auditing of individual volumes to be enabled when volume is created or modified.

Enabling and Disabling Auditing of Data Access Operations Using the CLI or REST API

1. To enable or disable auditing of the filesystem, table, and streams operations on a cluster, run the `maprcli audit data` command.

   This command does not cause auditing to start for operations within those volumes. It only sets a flag that indicates you allow auditing of individual volumes to be enabled with the `maprcli volume audit` command. The audit logs for file operations, table operations, and stream operations are affected by the value that you set for the `-retention` parameter.

2. To enable or disable auditing for a particular volume, run the `maprcli volume audit` command. To verify that auditing is enabled for a volume, run the `maprcli volume info` command.
   You can grep with the search term `'audited\|coalesce'`.

   ```
   maprcli volume info -name <volume_name> -json | grep -i 'audited\|
   coalesce'
   ```

   The output of the command should be as follows, with a 1 for the `audited` key and the value for the `coalesceinterval` key: `"audited":1, "coalesceInterval":2`

3. To enable or disable auditing for a particular directory, file, MapR Database table, or streams that existed in a volume at the time that you ran the `maprcli volume audit` command, run the `hadoop mfs` command with the `-setaudit` parameter.

   ```
   hadoop mfs -setaudit <on|off> <directory|file|table>
   ```

   Note: Wildcards are not supported for the names of filesystem objects in this command.

   Enabling auditing on a directory does not enable auditing on the files that already exist in the directory, though new files and directories created in the directory will have auditing enabled. For example, if you run this command on the root directory of a volume, all new files, directories, and tables that are subsequently created in the volume are audited. The creation of those objects is also audited.

After enabling auditing, if you create a:

- Snapshot of a volume, the snapshot inherits the audit settings of the original volume.

last-updated: Apr 11, 2022

- Local mirror or remote mirror of a volume, you must run the `maprcli volume audit` command to enable auditing on the mirror volume. Auditing for particular directories, files, and MapR Database tables in a mirror volume is automatically enabled if auditing is enabled for them in the source volume.

How Does Auditing Work?
Explains how auditing works on MapR objects.

When you enable the auditing of a particular directory, file, table, or stream, you set the *audit bit* to "on" for that object. You can tell whether auditing is enabled for a directory, file, or table by checking the status of the object's audit bit.

For example, the volume as shown in the following tree diagram, consists of the root directory, the two directories `dir1` and `dir2`, and two files in directory `dir1`. Every directory, file, table, and stream in a volume has an “audit bit” associated with it. You can tell whether, say, `dir1` has its audit bit on and is therefore enabled for auditing by running the `hadoop mfs -ls` command. The output of the command might look like as follows:

```
drwxrwxrwx Z U U 3 root root 100 2015-05-20 21:09 192473738 /dir1
```

The second `U` indicates that auditing is not enabled on the directory.

However, an `A` in place of that `U` indicates that auditing is enabled on the directory:

```
drwxrwxrwx Z U A 3 root root 100 2015-05-20 23:41 192473738 /dir1
```

In the first diagram, as well as in the next two diagrams, `U` indicates that the audit bit is turned off for a filesystem object and `A` indicates that the audit bit is on for that object. After you run `maprcli volume audit` on the volume, none of the audit bits are on:

```
/              U
-/dir1          U
 -file1         U
 -file2         U
-/dir2          U
```

Suppose you enable auditing on the root directory by running this command:

```
hadoop mfs -setaudit on /
```

Then, you create the file `file3` in `dir2` and you create the directory `dir3` and the file `file4` in it. The tree diagram now looks as follows :

```
/           A
-/dir1          U
 -file1         U
 -file2         U
-/dir2          U
 -file3         U
-/dir3          A
 -file4         A
```

The audit bit is still `U` on `dir1`, and the files are in `dir1`, and `dir2`. The new file `file3` in `dir2` inherits the audit bit from `dir2`.

`dir3` inherits the audit bit from the root folder, so the audit bit for `dir3` is `A`. Moreover, `file4` inherits the audit bit from `dir3`, so its audit bit is `A`, as well.

last-updated: Apr 11, 2022

Next, you run the following command to enable auditing in `dir1`:

```
hadoop mfs -setaudit on /dir1
```

Then, you create the file `file5`. The new file inherits the audit bit from its parent folder, so it is enabled for auditing immediately after it is created. However, `file1` and `file2` still have the audit bit turned off.

```
/         A
-/dir1           A
 -file1          U
 -file2          U
 -file5          A
-/dir2           U
 -file3          U
-/dir3           A
 -file4          A
```

As file1 and file2 existed before you turned on the audit bit for their parent folder, you need to enable auditing for them as follows:

```
hadoop mfs -setaudit on /dir1/file1
```

```
hadoop mfs -setaudit on /dir1/file2
```

Selective Auditing of MapR File System, MapR Database Table, and MapR Event Store For Apache Kafka Operations Using the CLI
Explains how to selectively audit MapR objects.

Administrators can specify filesystem, table, or stream operations to include or exclude from auditing. The operations that can be included or excluded from auditing are listed here.

Including or excluding specific operations from auditing requires running the `maprcli` command. You can specify the list of operations to include or exclude from auditing during volume creation using the `maprcli volume create` command, and afterwards using the `maprcli volume modify` or `maprcli volume audit` command.

To:

- Include operations for auditing, use the plus sign (+) before the operation.
- Exclude operations from auditing, use the minus sign (-) before the operation.

  Note: If the first operation in the list is to be excluded from auditing, it must be preceded by two minus (--) signs. Subsequent operations to exclude from auditing must be preceded by only a single minus (-) sign, whether or not the first operation was included (using a plus (+) sign) or excluded (using two minus (--) signs).

Note: If neither (plus (+) or minus (-)) sign is specified before an operation, the given operation is included for auditing.

Including and/or Excluding Operations

Including or excluding specific operations from auditing requires running the `maprcli` command.

Include or Exclude Operations During Volume Creation

During volume creation, the specified list of operations must either be included for auditing or excluded from auditing. You cannot specify a mixed list of included and excluded operations.

By default, all other operations other than the specified operations are:

last-updated: Apr 11, 2022

- Included for auditing if the specified list is a list of excluded operations.
- Excluded from auditing if the specified list is a list of included operations.

Examples

The following example shows how to enable auditing and exclude specific operations (such as `lookup`, `read`, and `write`) from auditing:

```
maprcli volume create -name test-volume -path /test/
test-volume -auditenabled true -dataauditops --lookup,-read,-write
```

In the above example, operations other than the ones specified are included for auditing.

The following example shows how to include all operations except `lookup` for auditing:

```
maprcli volume create -name test-volume -path /test/
test-volume -dataauditops --lookup
```

The following example shows how to include only `chown` operation for auditing and exclude all other operations from auditing:

```
maprcli volume create -name test-volume -path /test/
test-volume -dataauditops +chown
```

Include and Exclude Operations After Volume Creation

After volume creation, you can include and exclude certain operations from auditing using the `volume modify` or `volume audit` command. When you modify a volume (by running the `volume modify` command) or when you enable volume auditing (by running the `volume audit` command), you can specify a mixed list of included and excluded operations. There are no changes to operations that are not specified with the command.

For the list of operations that can be included and/or excluded from auditing, see Auditing of Filesystem Operations and Table Operations.

Examples

The following example shows how to include `create` operation for auditing and exclude `lookup` operation from auditing:

```
maprcli volume modify -name test-volume -dataauditops +create,-lookup
```

The following example shows how to include all operations except `lookup` for auditing:

```
maprcli volume audit -name test-volume -dataauditops +all,-lookup
```

Grouping of Operations

You can group all filesystem and table operations using the keyword `all`. If the operations to:

- Include for auditing are specified using the keyword `all`, you cannot specify other individual operations to include as well.

  For example, the following is *not* allowed:

  ```
  maprcli volume modify -name v1 -dataauditops +all,+mkdir
  ```


last-updated: Apr 11, 2022

- Exclude from auditing are specified using the keyword `all`, you cannot specify other individual operations to exclude as well.

  For example, the following is *not* allowed:

  ```
  maprcli volume modify -name v1 -dataauditops --all,-mkdir
  ```

If operations are specified using the keyword `all`, ensure that the individual operations specified with the same command are used to:

- Include, if `all` is used to exclude from auditing.
- Exclude, if `all` is used to include for auditing.

For example, the following is a valid combination of operations to audit:

```
maprcli volume modify -name v1 -dataauditops +all,-mkdir,-lookup
```

Verifying Selective Auditing of Operations

After you set up the list of operations to include and/or exclude from auditing, you can retrieve and verify the list of included and/or excluded operations using the `maprcli volume info` command. When you run the `volume info` command, the output will show the list of operations:

- Excluded (`disableddataauditoperations`) from auditing.
- Included (`enableddataauditoperations`) for auditing.

Example

The following example shows how to retrieve and verify the list of operations that are:

- Excluded from auditing
- Included for auditing

```
maprcli volume info -name test-volume -path /test/test-volume -json
```

Output

```
{
    "timestamp":1435182867317,
    "timeofday":"2016-01-10 02:54:27.317 GMT-0700",
    "status":"OK",
    "total":1,
    "data":[
        {
            "acl":{
                "Principal":"User mapr",
                "Allowed actions":[
                    "dump",
                    "restore",
                    "m",
                    "a",
                    "d",
                    "fc"
                ]
            },
            "creator":"mapr",
            "aename":"mapr",
            ...
```


last-updated: Apr 11, 2022

```
"enableddataauditoperations":"getattr,setattr,chown,chperm,chgrp,getxattr,listxattr,setxattr,removexattr,read,write,create,delete,readdir,rmdir,createsym,lookup,rename,createdev,truncate,tablecfcreate,tablecfdelete,tablecfmodify,tablecfScan,tableget,tableput,tablescan,tablecreate,tableinfo,tablemodify,getperm",
            "disableddataauditoperations":"mkdir",
            ...
            "ReplTypeConversionInProgress":"0"
        }
    ]
}
```

Enabling and Disabling Audit Streaming Using the CLI
Explains how to enable or disable audit streaming using the CLI.

Audit streaming is not enabled by default. You can enable or disable audit streaming using the CLI.

Run the following command to:

- Enable audit streaming:

  ```
  maprcli config save -values '{"mfs.enable.audit.as.stream":"1"}'
  ```

  Note: If you are re-enabling audit streaming after disabling it, the audit stream starts publishing to topics from where it left off processing audit logs.

- Disable audit streaming:

  ```
  maprcli config save -values '{"mfs.enable.audit.as.stream":"0"}'
  ```

Configuring Balancer Settings
Provides an overview of the MapR disk space and replication role balancers.

You can use the disk space balancer and the replication role balancer to redistribute data and containers in the MapR storage layer to ensure maximum performance and efficient use of space. The disk space balancer works to ensure that the percentage of space utilized on all storage pools in the cluster is similar and prevent nodes from being overloaded. The replication role balancer changes the replication roles of containers so that the replication process uses network bandwidth evenly.

You can pipe the `maprcli config load` command through `grep` to view the balancer configuration values.

Example:

```
maprcli config load -json | grep balancer
         "cldb.balancer.disk.deltaToRepopulateStoragePoolsBins":"5",
         "cldb.balancer.disk.deltatorepopulatestoragepoolsbins":"5",
         "cldb.balancer.disk.max.switches.in.nodes.percentage":"10",
         "cldb.balancer.disk.overused.threshold":"90",
         "cldb.balancer.disk.sleep.interval.sec":"120",
         "cldb.balancer.disk.threshold.percentage":"70",
         "cldb.balancer.logging":"0",
         "cldb.balancer.role.max.switches.in.nodes.percentage":"10",
         "cldb.balancer.role.paused":"1",
         "cldb.balancer.role.skip.container.active.sec":"600",
         "cldb.balancer.role.sleep.interval.sec":"900",
         "cldb.balancer.startup.interval.sec":"1800",
         "cldb.disk.balancer.enable":"0",
         "cldb.role.balancer.replicascount.tolerance":"1",
         "cldb.role.balancer.replicassize.tolerance":"5",
```

```
          "cldb.role.balancer.strategy":"BySize",
```

You can use the `config save` command to set the appropriate balancer configuration values.

Example:

```
maprcli config save -values
{"cldb.balancer.disk.max.switches.in.nodes.percentage":"20"}
```

Note: By default, the balancers are turned off.

- To turn on the disk space balancer, use `config save` to set `cldb.disk.balancer.enable` to `1`.
- To turn on the replication role balancer, use `config save` to set `cldb.balancer.role.paused` to `0`.

Disk Space Balancer
Describes the role of the disk space balancer.

The *disk space balancer* is a tool that balances disk space usage on a cluster by moving containers between nodes (subject to the constraints of the topology of the volume to which a container belongs). This movement of containers ensures that the percentage of space used on all the disks in the cluster is similar. The disk space balancer balances at the level of storage pools (SPs), keeping them at the same utilization level as the cluster average.

Note:

- Utilization Level of a SP = (Used space of the SP / Storage capacity of the SP)
- Cluster Average = (Used space across all SPs / Capacity of all SPs)

The disk space balancer distributes containers from highly utilized storage pools on one node in a cluster to less utilized storage pools on other nodes in the same cluster. It accomplishes this by first classifying storage pools into different bins (based on their utilization level). It checks every storage pool on a regular basis (every 2 minutes by default) and then classifies storage pools into bins based on their percentage utilization.

After classifying the storage pools into bins, the disk space balancer then moves containers (in two phases) out of the storage pools with more containers to storage pools with fewer containers. That is, it moves containers out of storage pools in higher bins to storage pools in lower bins in two phases:

- Phase 1 — storage pools in ‘Overused’ and ‘Above Average’ bins are balanced.
- Phase 2 — storage pools in ‘Average’ and ‘Below Average’ bins are balanced.

  Note: Movement of containers in phase 2 happens only when there are not many containers scheduled to be moved in phase 1, because movement of container at any point in time is throttled.

In both phases, the disk balancer attempts to move containers from storage pools in the highest utilized bin (the source SP) to suitable storage pools in the lowest utilized bin (the destination SP). If a suitable SP could not be found as the destination, the balancer attempts to move a container to the next least utilized bin. An SP is not deemed suitable as the destination if:

- Moving a container to that SP would cause the SP to move to the next bin.
- Data movement into the file server to which the SP belongs is blocked.

last-updated: Apr 11, 2022

- SP is not in the same topology as specified by the volume.
- Certain number of containers are currently being moved into the SP.

  Note: The number of simultaneous moves to a SP is capped at 2.
- SP is in the same bin as the source SP.
- Many containers of a container group associated with the same tiering-enabled volume reside on the SP.

  This feature ensures that containers of a container group associated with a tiering-enabled volume are distributed evenly across SPs and do not gather in a few SPs. The balancer accomplishes this by determining whether the destination SP has containers from the container group for the volume associated with the container that is identified for being moved to the destination SP.

An SP or its containers are not considered for balancing if:

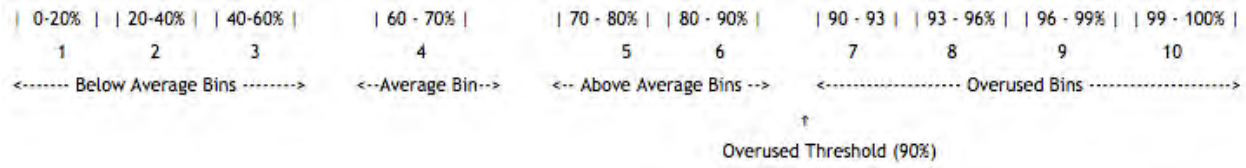
- Data movement from the file server to which the SP belongs is blocked.
- Container was active (that is, written to) in the previous 5 minutes.
- Containers were deleted in the previous minute.

  This is to allow space reclamation. If necessary, the bin will be balanced during the next iteration of the disk balancer.
- Percentage of data that is being transferred out of an SP is greater than or equal to 2% of the SP's storage capacity.
- Certain number of containers are currently being moved out of the SP.

  Note: The number of simultaneous moves out of an SP is capped at 2.

Sample Disk Balancer Settings for Organization of Bins

The following example illustrates disk balancer settings and the corresponding organization of bins to represent storage pool utilization:



In the preceding example, the:

1. Average cluster space utilization is 65%.
2. Average bin size is 10%. Hence, the average bin spans 5% on each side of the average space utilization.
3. Overused threshold is 90%.
4. Below average bin size is 20%.

   Below average bin utilization is partitioned into required number of bins (3). During division, the bin that is immediately to the left of the average bin might not span the default value. In other words, if the average bin spans from 50% to 60%, the below average bins will be cast as 0-20%, 20-40%, 40-50%.

last-updated: Apr 11, 2022

5. Above average bin size is 10%.

   Since the overused threshold is 90%, the above average bins span the utilization from the right boundary of the average bin up to the overused threshold. As with below average bins, as many possible bins are cast with the size of the above average bin, leaving the remaining utilization to be covered by the last (right-most) above average bin.

Enabling and Configuring Disk Balancer
Describes how to enable or disable the disk space balancer and other settings using the Control Panel and the CLI.

Enabling and Configuring Disk Balancer Using the Control System

To enable and configure disk space balancer using the Control System:

1. Log in to the Control System and click **Admin** > **Clusters Settings** > **Balancer**.
2. In the Disk Balancer section, move the **Enabled** slider to **Yes**.
   To enable disk space balancer from the command-line, see Configuring Disk Balancer Execution on page 768.
3. Select one of the presets or specify a custom percentage for **Threshold** and **Concurrent Disk Rebalancer** settings. You can :
   - **Disk Balancer Preset**:

     Moderate (Default)
     - **Threshold** — 70%
     - **Concurrent Disk Rebalancer** — 10%

     Rapid
     - **Threshold** — 50%
     - **Concurrent Disk Rebalancer** — 5%

     Relaxed
     - **Threshold** — 90%
     - **Concurrent Disk Rebalancer** — 25%

   - **Custom**, use the slider to set the threshold and concurrent disk rebalancer percentages.

   Here:

Threshold	Specifies the minimum utilization threshold for a storage pool to become eligible for rebalancing. Default value is 70%.
Concurrent Disk Rebalancer	Specifies the maximum percentage of data that can be rebalanced. The cluster will wait until the number of rebalancing operations affects less than this percentage of total data eligible for rebalancing. The default value is 10%.

4. Click **Save Changes** for the changes to take effect.

Enabling and Configuring Disk Balancer Using the CLI

The disk space balancer checks every storage pool on a regular basis and moves containers from a storage pool when that pool's utilization meets the following conditions:

- The storage pool is over 70% full.
- The storage pool's utilization exceeds the average utilization on the cluster by a specified threshold:
  - When the average cluster storage utilization is below 80%, the threshold is 10%.

last-updated: Apr 11, 2022

- When the average cluster storage utilization is below 90% but over 80%, the threshold is 3%.
- When the average cluster storage utilization is below 94% but over 90%, the threshold is 2%.

You can use the `config save` command to modify disk space balancer parameter values.

Example: `# maprcli config save -values {"cldb.balancer.disk.max.switches.in.nodes.percentage":"20"}`

The following list specifies the disk space balancer configuration parameters with their default values and descriptions:

cldb.disk.balancer.enable

Default Value: 1

Description: Specifies whether the disk space balancer runs:

- 0 - Disabled (does not perform any container moves)
- 1 - Enabled (normal operation)

By default, the disk balancer is disabled.

cldb.balancer.disk.threshold.percentage

Default Value: 70

Description: Threshold for moving containers out of a given storage pool, expressed as utilization percentage.

See also: Balancing Overused and Above Average Bins on page 769.

cldb.balancer.disk.max.switches.in.nodes.percentage

Default Value: 10

Description: This can be used to throttle the disk balancer. If it is set to 10, the balancer will throttle the number of concurrent container moves (minimum 1) to 10% of the total nodes in the cluster rounded up.

See Configuring Throttling on page 770.

Configuring Disk Balancer Execution
Explains how to tune the disk balancer execution parameters.

Even when the disk balancer is enabled, the disk balancer waits during cluster startup to give enough time for the cluster to settle down. The default wait time (specified in seconds) is 30 minutes and can be changed using the parameter `cldb.balancer.startup.interval.sec`. After every run, the disk balancer pauses for 2 minutes by default. The pause time (specified in seconds) can be increased through the parameter `cldb.balancer.disk.sleep.interval.sec`.

Run the `maprcli config save` command and set the following parameters to configure disk balancer:

Parameter	Default Value	Description
`cldb.balancer.startup.interval.sec`	1800 seconds	Wait time, in seconds, for cluster startup.
`cldb.balancer.disk.sleep.interval.sec`	120 seconds	Interval, in seconds, between disk balancer runs.

For example, to:

- Increase the wait time to an hour, change the default value of 1800 seconds to 3600 seconds:

```
maprcli config save -values {"cldb.balancer.startup.interval.sec":"3600"}
```

- Increase the pause time between disk balancer runs from 2 minutes to 5 minutes, change the default value of 120 seconds to 300 seconds:

```
maprcli config save -values
{"cldb.balancer.disk.sleep.interval.sec":"300"}
```

Preventing Volume Skew

By default, the disk balancer does not consider volume skew while moving containers out of a Storage Pool (SP). In the process, all containers of a volume may end up on only a few SPs. Such a volume skew is undesirable, specifically, for DB volumes. In such a skew, few servers handle all requests, resulting in reduced performance.

To prevent volume skew, set the `prevent.volume.skew.by.diskbalancer` parameter to `true` as follows:

```
maprcli config save -values {"prevent.volume.skew.by.diskbalancer":"true"}
```

This parameter checks whether moving a container out of a SP causes a volume skew elsewhere.

Note: Enabling this parameter may result in containers failing the volume underweight check and failing to be moved from a full SP. Consider this limitation and enable as per your need.

Configuring Bin Size
Explains how to configure the sizes of the various bins used by the disk balancer.

The number of bins used by the disk balancer is not constant and is determined by the sizes of different bins. You can configure the size of each of the bins (Below Average, Average, Above Average, and Overused) individually at run time. The larger the size of the bins, the greater the chance that two SPs that are in the vicinity of each other with respect to utilization fall in the same bin.

The default size of overused bins is only 3%, because SPs in this bin must be balanced at a finer granularity. The default size of above average, average, and below average bins is 20%. You can aggressively balance the storage pools across bins by reducing the size of each bin, forcing the SPs to fall into different bins. To reduce the size of each bin, specify the value for the following parameters using the `maprcli config save` command:

Parameter	Description
`dbal.above.avg.bin.size`	Specifies the bin size (%) of SPs whose usage is above the cluster average. The default is 20%.
`dbal.avg.bin.size`	Specifies the bin size (%) of SPs whose usage is in the average range. The default is 20%.
`dbal.below.avg.bin.size`	Specifies the bin size (%) of SPs whose usage is below the cluster average. The default is 20%.
`dbal.overused.bin.size`	Specifies the bin size (%) of SPs whose usage is in the overused range. The default is 3%.

For example, to reduce the size of the Below Average bin to 10%, run the following command:

```
maprcli config save -values {"dbal.below.avg.bin.size":"10"}
```

Balancing Overused and Above Average Bins
Describes how to balance highly utilized Overused and Above Average bins.

Storage pools in the Overused and Above Average bins are highly utilized. The default threshold of “overused” is 90%, which means that the first overused bin’s lower bound is 90%. You should in normal circumstances, never reset this threshold, but you can, if necessary, by setting the value for the parameter

last-updated: Apr 11, 2022

`cldb.balancer.disk.overused.threshold` using the `maprcli config save` command. For example:

```
maprcli config save -values {"cldb.balancer.disk.overused.threshold":"95"}
```

In scenarios where the cluster average is low, storage pools in Above Average bins too might not be highly utilized. In this case, to prevent unnecessary balancing activity, disk balancer uses an additional criterion to prevent wasted moves: only those storage pools whose utilization equals to or is greater than a certain threshold are considered for balancing. This threshold is controlled by the configurable parameter `cldb.balancer.disk.threshold.percentage`, whose default value is 70%.

Note: If the threshold of the Overused bin is set below the default value of 90%, the balancing threshold specified in the `cldb.balancer.disk.threshold.percentage` parameter is also applicable to Overused bins also.

Balancing Average and Below Average Bins
Provides an overview of how the disk balancer balances the Average and Below Average storage pool bins.

The primary task of the disk balancer is to balance highly utilized storage pools in Overused and Above Average bins. However, the disk balancer can also balance storage pools that are less utilized, for example, to distribute workload evenly across the nodes, or if a new node is added to the topology to add more storage. By default, the disk balancer performs this kind of balancing less frequently than the balancing of disk space utilization.

By default, the disk balancer balances storage pools in Average and Below Average bins every 6 hours. You can configure the interval by the setting the value (in minutes) for the parameter `dbal.below.avg.bins.balancing.frequency` using the `marpcli config save` command. For example:

```
maprcli config save -values
{"dbal.below.avg.bins.balancing.frequency":"360"}
```

Note: The disk balancer considers storage pools in these two bins only under the following conditions:

1. When there is not already too much SP balancing activity in the highly utilized bins.
2. If container movement out of SPs in the highly utilized bins is not possible.

Configuring Throttling
Explains how to throttle the number of container moves.

Although balancing storage pool disk space use is important, it requires network, computation, and disk bandwidth that must be shared with other functions. Hence, the number of container moves at any point of time is throttled.

The throttling factor controls the number of active container moves. If there are 100 nodes, and the throttling factor is 10, there can be 10 active moves. If however, the value is 12 (%), there can be 12 active moves.

Note: If you set the throttling factor to 0, the number of active moves is 1.

Therefore, effectively, the number of active moves is:

```
Max(1, throttling_factor x number_of_cluster_nodes)
```

By default, the maximum number of container moves is 10% of the number of nodes in the cluster. However, you can configure the throttling factor (percentage) by setting the parameter `cldb.balancer.disk.max.switches.in.nodes.percentage` using the `maprcli` command.

For example, to set the throttling factor to 12% of the number of nodes in the cluster, run the following command:

```
maprcli config save -values
{"cldb.balancer.disk.max.switches.in.nodes.percentage":"12"}
```

Retrieving Status of Storage Pools
Describes the CLI command to retrieve the status of Storage Pools.

You can use the `maprcli dump balancerinfo` command to view detailed information about the Storage Pools on a cluster.

Example:

```
# maprcli dump balancerinfo
usedMB  fsid                spid                                percentage
outTransitMB  inTransitMB  capacityMB
209     5567847133641152120  01f8625ba1d15db7004e52b9570a8ff3  1
0             0            15200
209     1009596296559861611  816709672a690c96004e52b95f09b58d  1
0             0            15200
```

If there are any active container moves when you run the command, `maprcli dump balancerinfo` returns information about the source and destination storage pools:

```
# maprcli dump balancerinfo -json
        ....
        {
        "containerid":7840,
        "sizeMB":15634,
        "From fsid":8081858704500413174,
        "From IP:Port":"10.50.60.64:5660-",
        "From SP":"9e649bf0ac6fb9f7004fa19d200abcde",
        "To fsid":3770844641152008527,
        "To IP:Port":"10.50.60.73:5660-",
        "To SP":"fefcc342475f0286004fad963f0fghij"
        }
```

Retrieving Balancer Status
Describes how to view the active container movement information from the Control Panel and the CLI.

You can view the active container moves information from the CLDB page on the Control Panel or use the `maprcli dump balancermetrics` command to see a cumulative count of container moves and MB of data moved between storage pools since the current CLDB became the primary CLDB.

Example:

```
# maprcli dump balancermetrics -json
{
  "timestamp":1337770325979,
  "status":"OK",
  "total":1,
  "data":[
    {
      "numContainersMoved":10090,
      "numMBMoved":3147147,
      "timeOfLastMove": "Wed May 23 03:51:44 PDT 2012"
    }
```

```
  ]
}
```

Viewing the List of Active Container Moves
Explains how to view the list of containers being moved, from the CLDB page.

- Log in to the Control System and go to the service information page for CLDB.
  The **Active Container Moves** section displays the following fields:

Container ID	The ID of the container being moved.
SizeMB	The size (in MB) of the container being moved.
From Location	The location from where the container is being moved.
From SP	The Storage Pool (SP) out of which the container is being moved.
To Location	The location to which the container is being moved.
To SP	The SP to which the container is being moved.

Volume Balancer
Describes the role of the volume balancer.

Volume balancer is used to distribute containers of a volume on all the storage pools that belong to the volume's topology. Although the disk balancer balances containers across storage pools, sometimes containers of a volume may accumulate on a few storage pools. For example:

- When the storage pools hosting a volume's containers are not highly utilized, the disk balancer might not spread the volume's containers across storage pools.
- When new storage pools are added to a topology and the storage pools on which the current containers reside are not highly utilized, although the disk balancer moves containers to new storage pools, it is not guaranteed that a specific volume's containers are evenly spread out.

In such cases, you can trigger the balancing of a volume using a `maprcli` command. Every time a volume gets out of balance, you can trigger the volume balancer (using the `maprcli` command) to balance the containers associated with the volume. The container moves triggered by disk and volume balancers do not cause other volumes to be imbalanced.

Note: If both disk balancer and volume balancer are triggered at the same time, volume balancer activity takes precedence.

Managing Volume Balancer
Explains the CLI commands that you can use to balance the containers of a volume.
Enabling and Disabling the Volume Balancer

The volume balancer is disabled by default. To enable or disable the volume balancer:

- Set the value for the `cldb.volume.balancing.enable` parameter using the `config save` on `page 1586` command. To:
  - Enable volume balancer, run the following command:

    ```
    maprcli config save -values {cldb.volume.balancing.enable:1}
    ```

  - Disable volume balancer, run the following command:

    ```
    maprcli config save -values {cldb.volume.balancing.enable:0}
    ```

After enabling the volume balancer feature, you must run the `volume balancecontainers` on `page 1925` command to balance the containers associated with a volume.

Balancing the Containers of a Volume

- Run the `maprcli volume balancecontainers` command to balance a volume.
  The basic command to balance a volume is:

```
/opt/mapr/bin/maprcli volume balancecontainers -name <vol_name>
```

  For more information, see `volume balancecontainers` on `page 1925`.

Stopping the Volume Balancer

- Run the `maprcli volume balancecontainers` command to stop or cancel a balancing activity.
  The command to cancel a volume balancer is:

```
/opt/mapr/bin/maprcli volume balancecontainers -name <vol_name> -cancel
true
```

  For more information, see `volume balancecontainers` on `page 1925`.

Related reference

config save on page 1586
Saves configuration information, specified as key/value pairs. Permissions required: `fc` or `a`.

volume balancecontainers on page 1925
Balances the containers, or stops the balancing of containers associated with the volume.

Retrieving Balancer Status
Retrieve the status of a balancer activity using the CLI.

Run the `maprcli volume balancinginfo` command to retrieve the status of a currently running or scheduled balancer activity.

The basic command to retrieve the status of volume balancer is:

```
/opt/mapr/bin/maprcli volume balancinginfo -name <vol_name>
```

For more information, see `volume balancinginfo` on `page 1926`. For example:

```
# /opt/mapr/bin/maprcli volume balancinginfo -name Volume1 -json
{
  "timestamp":1502529117881,
  "timeofday":"2017-08-12 09:11:57.881 GMT+0000",
  "status":"OK",
  "total":5,
  "data":[
    {
      "volumeName":"Volume1"
    },
    {
      "isBalancingInProgress":false
    },
    {
      "numContainers":15
    },
    {
      "volumeSize":384
    },
```

```
    {
      "spInfo":[
        {
          "spId":"f891ae9e6663fa2000598ec48808155c",
          "capacity":152969,
          "usedSize":96,
          "desiredSize":95,
          "isUnderweight":false,
          "isOverweight":false
        },
        {
          "spId":"bed92c0ecfaefc8b00598ec48b01cdfe",
          "capacity":152969,
          "usedSize":96,
          "desiredSize":95,
          "isUnderweight":false,
          "isOverweight":false
        },
        {
          "spId":"b61aa1b814fd8bbc00598ec48d0af1d2",
          "capacity":157065,
          "usedSize":96,
          "desiredSize":97,
          "isUnderweight":false,
          "isOverweight":false
        },
        {
          "spId":"7af11d5b9d223baa00598ec4850efb57",
          "capacity":152969,
          "usedSize":96,
          "desiredSize":95,
          "isUnderweight":false,
          "isOverweight":false
        }
      ]
    }
  ]
}
```


Related reference

volume balancinginfo on page 1926
Fetch currently running or scheduled balancer information for one or more volumes.

Replication Role Balancer

Describes the features of the replication role balancer.

The replication role balancer manages containers to optimize the following:

- Network bandwidth during the replication process
- Disk I/O and CPU when serving read requests

The replication role balancer switches the replication roles of name and data containers to balance them across each storage pool in a volume. You can modify the `cldb.role.balancer.strategy` parameter from the `maprcli` to change how the role balancer manages the containers, either by size or count. You can also run the `dump rolebalancerinfo` command to see the status of active role switches or how container roles are balanced across each storage pool for a particular volume.

Replicated Containers

The name container is the first container created in every volume. Name containers can have either a *primary* or a *replica* role. Data containers can have a *primary*, *intermediate*, or *tail* role. By default, each

name and data container is replicated across the cluster three times, with the primary being the first container written. The primary is sequentially replicated two more times, into a container with either an intermediate or a tail container role. If too many primary or intermediate containers exist on a storage pool or if the primary and intermediate containers are too large, the role balancer switches some of these containers to tail containers.

By default, the role balancer compares the size of the primary and tail containers to determine if containers within a storage pool are balanced. For the best performance, the size of the primary containers in a volume should be evenly distributed across storage pools. The role balancer maintains this balance by ensuring that each type of container (primary, intermediate, and tail) accounts for `1/ReplicationFactor` of the total container size in a volume.

If the role balancer is configured to manage containers by count, it compares the number of primary and tail containers and balances the roles such that each type of container accounts for `1/ReplicationFactor` of the total number of containers in a volume. For example, if the replication factor is set to 3, the role balancer maintains a balance of ⅓ primary, ⅓ intermediate, and ⅓ tail containers in each volume.

MapR Database Considerations

To optimize MapR Database performance, you should configure the role balancer to manage containers by size. As described at MapR Database and MapR XD on page 501, MapR Database shards tables into *tablets* and stores the tablets in data containers. Only primary data containers serve reads. Therefore, configuring the role balancer by size spreads read requests evenly across the storage pools for a volume. To ensure the most optimal balancing for your MapR Database tables, you should consider storing them on dedicated volumes.

Assign Cache

The assign cache is a list of reserved containers on a particular file server node that are allocated by the CLDB (container location database). The replication role balancer does not balance the containers in the assign cache and does not include them when balancing the roles. See the maprcli dump rolebalancerinfo command for assign cache values and details.

Enabling and Configuring Replication Role Balancer
Describes how to use the Control System or the CLI to enable and configure the Replication Role Balancer.

Enabling and Configuring the Replication Role Balancer Using the Control System

1. Log in to the Control System and click **Admin** > **Cluster Settings** > **Balancer**.
2. From the Role Balancer section, set the **Enabled** slider to **Yes** .
3. Select one of the presets or specify a custom value for the **Concurrent Role Rebalancer** and **Delay for Active Data in Seconds** settings. You can choose:
   - Presets:

Default	• **Concurrent Role Rebalancer** — 20% • **Delay for Active Data in Seconds** — 600 sec
Rapid	• **Concurrent Role Rebalancer** — 5% • **Delay for Active Data in Seconds** — 300 sec
Moderate	• **Concurrent Role Rebalancer** — 10% • **Delay for Active Data in Seconds** — 600 sec

last-updated: Apr 11, 2022

Relaxed	• **Concurrent Role Rebalancer** — 25% • **Delay for Active Data in Seconds** — 1800 sec

- **Custom**, use the slider to set the concurrent role rebalancer percentage and delay for active data in seconds.

Here:

Concurrent Role Rebalancer	Specifies the maximum percentage of data affected by concurrent role rebalancer operations. The cluster will wait until the number of rebalancing operations affects less than this percentage of total data eligible for rebalancing.
Delay for Active Data in Seconds	At the time of calculation, the role rebalancer will skip any data that is active within this time interval. This prevents unnecessary tampering with data used in recent or ongoing computations.

4. Click **Save Changes** for the changes to take effect.

Enabling and Configuring Replication Role Balancer Using the CLI

You can use the `config save` command to modify the replication role balancer parameter values.

Example: `# maprcli config save -values {"cldb.role.balancer.strategy":"BySize"}`

The following table lists the replication role balancer configuration parameters with their default values and descriptions:

Parameter	Value	Description
cldb.balancer.role.paused	1	Specifies whether the role balancer runs: • 0 - Not paused (normal operation) • 1 - Paused (does not perform any container replication role switches)
cldb.role.balancer.strategy	"BySize"	Specifies how the replication role balancer balances containers, either by size or count. Use `"BySize"` or `"ByCount"` to indicate how role balancer balances containers.
cldb.balancer.role.max.switches.in.nodes.percentage	10	This can be used to throttle the role balancer. If it is set to 10, the balancer will throttle the number of concurrent role switches to 10% of the total nodes in the cluster (minimum 2).

Retrieving Role Balancer Status Using the CLI
Lists the CLI command to view the number of active replication role switches.

You can use the `maprcli dump rolebalancerinfo` command to view the number of active replication role switches. During a replication role switch, the replication role balancer selects a primary or intermediate data container and switches its replication role to that of a tail data container.

last-updated: Apr 11, 2022

Example:

```
# maprcli dump rolebalancerinfo -json
{
    "timestamp":1335835436698,
    "status":"OK",
    "total":1,
    "data":[
            {
               "containerid": 36659,
               "Tail IP:Port":"10.50.60.123:5660-",
               "Updates blocked Since":"Wed May 23 05:48:15 PDT 2012"
            }

    ]
}
```

Managing Licenses
Provides a synopsis of adding licenses using the Control System and the CLI.

You can add and remove licenses on your cluster using the Control System or the CLI:

- In the Control System, go to **Admin** > **Cluster Settings** > **Licenses**.
- On the command line, use the `maprcli license` commands.

Warning: Remove old licenses from the cluster when you add a new license. If multiple licenses exist, the cluster activates only the license with the lowest node count.

Adding a License
Add a license through the Control System or the CLI.
Adding a License Using the Control System

Complete the following steps to add a license using the Control System:

1. On a machine that is connected to the cluster and to the Internet, perform the following steps to open the Control System:
   a) In a browser, view the Control System by navigating to the node that is running the Control System: `https:/<webserver>/:8443`.

      Your computer will not have a HTTPS certificate yet, so the browser will warn you that the connection is not trustworthy. You can ignore the warning this time.

   b) Log in to the Control System as the administrative user you designated earlier.

      Until a license is applied, the Control System dashboard might show some nodes in the amber "degraded" state. Do not worry if not all nodes are green and "healthy" at this stage.
2. In the Control System, go to **Admin** > **Cluster Settings** > **Licenses**.
3. Add a license using the following options:

Import License	Allows you to import a license from the server. You must enter your credentials in the **Import License** window to retrieve your license information.
Upload License	Allows you to upload your license file through a browser.
Copy/Paste License	Allows you to copy and paste a license key in the **Copy and Paste License** window.
Get a Free Trial License	Navigates to the MapR licensing form online to get a trial license.

last-updated: Apr 11, 2022

4. Click **Submit**.

   If the cluster is already registered, the license is applied automatically. Otherwise, go to hpe.com and follow the instructions there to register the cluster.

Adding a License Using the CLI or the REST API

To add a license from the CLI:

1. Obtain a valid license file from MapR.
2. Copy the license file to a cluster node.
3. Run the following command to add the license:

   ```
   maprcli license add [ -cluster <name> ] -license <filename> -is_file true
   ```

   See `license add` on page 1680 for complete reference information.

Related concepts

Upgrading and Your License on page 295
If you are upgrading from MapR version 5.0 or earlier, the Base License file must be manually updated on all nodes in your cluster.

Related tasks

Viewing the Licenses on the Cluster on page 778
List the licenses on the cluster using either the Control System or the CLI.

Removing a License on page 780
Describes how to remove a license using the Control System and the CLI.

Related reference

license add on page 1680
Adds a license. Permissions required: `fc` or `a`.

license addcrl on page 1681
Adds a certificate revocation list (CRL). Permissions required: `fc` or `a`.

license apps on page 1682
Displays the features authorized for the current license. Permissions required: `login`

license list on page 1684
Lists licenses on the cluster. Permissions required: `login`. For best results, use the `-json` option when running the command.

license listcrl on page 1686
Lists certificate revocation lists (CRLs) on the cluster. Permissions required: `login`.

license remove on page 1687
Removes a license. Permissions required: `fc` or `a`.

license showid on page 1689
Displays the cluster ID for use when creating a new license. Permissions required: `login`.

Viewing the Licenses on the Cluster

List the licenses on the cluster using either the Control System or the CLI.

Viewing the Licenses Using the Control System

To view licenses:

1. Log in to the Control System.
2. Click **Admin** > **Cluster Settings** > **Licenses**.

Under **LICENSES**, the pane displays the following information for each license:

Column Name	Column Description
Active	Indicates whether (✓) the license is active.
Grace	Denotes the remaining grace period (in days) before which you must renew the expired license.
Name	The name of the license.
Module/Type	The type of license.
Issued	The date the license was issued.
Expires	The license expiration date.
Nodes	The number of nodes to which the license applies.
Delete	The option to remove the license.

You can:

- Add a license
- Remove a license

Viewing the Licenses Using the CLI or REST API

The basic command to get a list of licenses on the cluster is:

```
maprcli license list -cluster <cluster>
```

For complete reference information, see `license list` on page 1684.

Related concepts

Upgrading and Your License on page 295
If you are upgrading from MapR version 5.0 or earlier, the Base License file must be manually updated on all nodes in your cluster.

Related tasks

Viewing the Licenses on the Cluster on page 778
List the licenses on the cluster using either the Control System or the CLI.

Adding a License on page 777
Add a license through the Control System or the CLI.

Removing a License on page 780
Describes how to remove a license using the Control System and the CLI.

Related reference

license add on page 1680
Adds a license. Permissions required: `fc` or `a`.

license addcrl on page 1681
Adds a certificate revocation list (CRL). Permissions required: `fc` or `a`.

license apps on page 1682
Displays the features authorized for the current license. Permissions required: `login`

license list on page 1684
Lists licenses on the cluster. Permissions required: `login`. For best results, use the `-json` option when running the command.

license listcrl on page 1686
Lists certificate revocation lists (CRLs) on the cluster. Permissions required: `login`.

last-updated: Apr 11, 2022

license remove on page 1687
Removes a license. Permissions required: `fc` or `a`.

license showid on page 1689
Displays the cluster ID for use when creating a new license. Permissions required: `login`.

Removing a License
Describes how to remove a license using the Control System and the CLI.
Removing a License Using the Control System

To remove a license:

1. Log in to the Control System and click **Admin** > **Cluster Settings** > **Licenses**.
2. Click the **Delete** link associated with the license to display the **Remove License** confirmation dialog.
3. Click **Submit** to remove the license.

Removing a License Using the CLI or REST API

To remove a license on a cluster:

1. From the command line, issue the `maprcli license list on page 1684` command. Example:
   ```
   maprcli license list
   ```
2. Look for the `id` parameter in the output from the license list command.
   This is the license ID. Example:
   ```
   grace id                          description       deletable license
   maxnodes
   true  5CTFWAeQQUIOc5Wm/onoOJqcCls MapR Base Edition false
        version: "1.0"
        customerid: "BaseLicenseUser"
        issuer: "MapR Technologies, Inc."
        licType: Base
        description: "MapR Base Edition"
        ...
   ```
3. Use the `maprcli license remove on page 1687` command to remove the license.
   Example:
   ```
   maprcli license remove -license_id 5CTFWAeQQUIOc5Wm/onoOJqcCls
   ```

Related concepts
Upgrading and Your License on page 295
If you are upgrading from MapR version 5.0 or earlier, the Base License file must be manually updated on all nodes in your cluster.

Related tasks
Viewing the Licenses on the Cluster on page 778
List the licenses on the cluster using either the Control System or the CLI.

Adding a License on page 777
Add a license through the Control System or the CLI.

Related reference
license add on page 1680
Adds a license. Permissions required: `fc` or `a`.

license addcrl on page 1681

Adds a certificate revocation list (CRL). Permissions required: `fc` or `a`.

license apps on page 1682
Displays the features authorized for the current license. Permissions required: `login`

license list on page 1684
Lists licenses on the cluster. Permissions required: `login`. For best results, use the `-json` option when running the command.

license listcrl on page 1686
Lists certificate revocation lists (CRLs) on the cluster. Permissions required: `login`.

license remove on page 1687
Removes a license. Permissions required: `fc` or `a`.

license showid on page 1689
Displays the cluster ID for use when creating a new license. Permissions required: `login`.

Setting Quota Defaults for Users and Groups
Explains how to set disk space quotas for users and groups.

Quotas limit the disk space used by a volume or an *entity* (user or group) on an Enterprise Edition-licensed cluster, by specifying the amount of disk space the volume or entity is allowed to use. A volume quota limits the space used by a volume. A user/group quota limits the space used by all volumes owned by a user or group. These quotas work on tenant volumes as well.

You can set hard quota and advisory quota defaults for users and groups. When a user or group is created, the default quota and advisory quota apply unless overridden by specific quotas. You can set an entity quota that differs from the default using the entity modify on page 1649 command or through the Control System.

The size of a disk space quota is expressed in terms of the actual data stored from the user's point of view. Only post-compression data blocks are counted, and snapshot and replica space do not count against quotas. For example, a 10G file that is compressed to 8G and has a replication factor of 3 consumes 24G (3*8G), but charges only 8G to the user or volume's quota.

You can set an entity quota through the Control System and using the CLI.

Setting Quotas Using the Control System

Complete the following steps to set the entity quota in the Control System:

1. Go to **Admin** > **Cluster Settings** > **Quotas**.
2. Set the advisory and hard quotas for the:
   - User under **USER QUOTA**.
   - Group under **GROUP QUOTA**.

   Hard quota prevents writes above the specified threshold. Advisory quota does not enforce disk usage limit, but raises an alarm when the specified threshold is exceeded:
   - AE_ALARM_AEADVISORY_QUOTA_EXCEEDED - an entity exceeded its advisory quota
   - VOLUME_ALARM_ADVISORY_QUOTA_EXCEEDED - a volume exceeded its advisory quota

   In most cases, it is useful to set the advisory quota a little lower than the hard quota, to give advance warning that disk usage is approaching the allowed limit.

3. Set the cluster reserve limit, which is the amount of disk space that you wish to allocate for all volumes on the cluster.

   When you set a reserve limit, you provision a certain amount of space to the volumes as a percentage of the cluster capacity. This allows you to free up space that could potentially be unused, or allocate more space for replication.

   As data is written to the volume, available space is automatically allocated. The volume reserve increases up to the reserve limit you set here.

4. After setting the quota, click **Save** to save the settings.

Setting Entity Quotas Using the CLI or the REST API

To set an entity (user or group) quota, run the following command:

```
maprcli entity modify -name <entityname> -advisoryquota <advisory
quota> -quota <quota>
```

To manage quotas, you must have a or `fc` permissions.

Quotas are expressed as an integer value plus a single letter to represent the unit:

- B - bytes
- K - kilobytes
- M - megabytes
- G - gigabytes
- T - terabytes
- P - petabytes

Example: 500G specifies a 500-gigabyte quota. Do not use two-letter abbreviations for units, such as MB or GB.

For complete reference information, see the `entity modify` on page 1649 command.

Setting the Cluster Reserve Limit Using the CLI or REST API

To set the resource usage limit for the cluster's disk resource, run the following command:

```
maprcli rlimit set -resource disk -cluster <cluster name> -value <limit>
```

For complete reference information, see the `rlimit set` on page 1736 command.

Related reference

entity modify on page 1649
Modifies a user or group quota or email address. Permissions required: `fc` or `a`.

rlimit set on page 1736
Sets the resource usage limit for the cluster's disk resource.

Specifying the Location of Gateways
Describes how to set the location of the MapR gateways using either the Control System or the CLI.

On every source MapR cluster, you can specify the location of the gateways by adding a DNS record to your DNS server's zone file for your domain. In your DNS server's zone file for your domain, add a record for the cluster where gateways are located, listing the nodes to use as gateways. You can use the Control System to create a record that you can copy into a DNS configuration file, run a `maprcli` command

to generate the record, or create a record manually. For more information on gateways, see Managing Gateways on page 1154.

Specifying the Location of Gateways Using the Control System

To create a record using the Control System, follow these steps:

1. Log in to the Control System on the cluster where the gateways are located.
2. Click **Admin** > **Cluster Settings** > **Gateway**.
3. Click **Copy to Clipboard** to copy the generated DNS entry.
4. Paste the record into your zone file.

Specifying the Location of Gateways Using the CLI

To generate a record by using the `maprcli` command, follow these steps:

1. On the cluster where the gateways are located, run the following command.

   ```
   maprcli cluster gateway local -format dns
   ```

   If you want to run the command from a different cluster and point to the cluster that hosts the gateways, use the `-cluster` parameter to provide the name of the latter cluster.

2. Copy and paste the output of this command into your zone file.

Creating a Record Manually

If you want to create a record manually, use this format:

```
gateway.<clustername> IN TXT "<space-delimited list of hostnames>"
```

You can also specify IP addresses, though using hostnames is recommended so that it is easier to locate gateways if their IP addresses change. Combinations of hostnames and IP addresses are also supported. The default port is `7660`. If a gateway is using a different port, append a colon to the address and then specify the port number. Here is an example entry:

```
gateway.newyork.bigcompany.com gw1ny.bigcompany.com gw2ny.bigcompany.com
```

Multi-homing is also supported. Simply separate the entries for a single node with semicolons, as in this example that uses IP addresses:

```
gateway.newyork.bigcompany.com 10.10.34.20 10.10.34.22
10.10.34.24;173.194.79.121
```

Related concepts

Administering MapR Gateways on page 1150
A MapR gateway mediates one-way communication between a source MapR cluster and a destination cluster. You can replicate MapR Database tables (binary and JSON) and MapR Event Store For Apache Kafka streams. MapR gateways also apply updates from JSON tables to their secondary indexes and propagate Change Data Capture (CDC) logs.

Configuring Gateways for Table and Stream Replication on page 1152
Configuring gateways involves installing the `mapr-gateway` package on nodes on a MapR destination cluster and then configuring the MapR source cluster to communicate with the destination cluster. The MapR source cluster is configured by specifying the destination cluster's CLDB node and gateway nodes.

gateway.conf on page 2197
Gateways for Replicating MapR Database Tables on page 621

last-updated: Apr 11, 2022

In MapR Database table replication, MapR Database replicates updates to tables (binary and JSON) on source MapR clusters to replicas of those tables on destination MapR clusters. Gateways are services that receive these updates and apply them to the replicas. These gateways also propagate updates from JSON tables to their secondary indexes.

Related reference

cluster gateway delete on page 1564
Deletes the list of MapR gateways from a source MapR cluster.

cluster gateway get on page 1565
Lists the MapR gateways that a source MapR cluster is using.

cluster gateway list on page 1567
Lists all of the gateways that a source MapR cluster is using.

cluster gateway local on page 1569
Lists the gateways that are configured on the MapR cluster where this command is run.

cluster gateway resolve on page 1571
Lists the gateways configured on a MapR cluster that are up and running at the time that the command is issued.

cluster gateway set on page 1573
Specifies the locations of the MapR gateways that a source MapR cluster can use for table replication to a destination MapR cluster, or for indexing table data in an Elasticsearch cluster.

Related information

Managing Gateways on page 1154
Describes the commands for listing gateways, checking status of gateways, managing gateways if they fail, and troubleshooting gateways.

Managing Alarms

You can view all the alarms and configure settings, including severity and notifications, on the Control System and using the CLI.

Viewing the List of Alarms

Specifies how to view the list of alarms raised, using either the Control System or the CLI.

You can view all the alarms on the Control System and using the CLI.

Viewing the List of Alarms in the Control System

1. Log in to the Control System and click **Admin** > **Cluster Settings** > **Alarms**.
   The **ALL ALARMS** pane displays all the alarms on the cluster.
2. Select:
   - **Cluster Alarms** — indicate problems that affect the cluster as a whole
   - **Node Alarms** — indicate problems on individual nodes
   - **Table Alarms** — indicate table replication-related problems
   - **User/Group Alarms** — indicate problems with user or group quotas
   - **Volume Alarms** — indicate problems in individual volumes

   For the selected view, the pane displays the following for each alarm:

Column Name	Column Description
Alarm Name	The name of the alarm.
Severity	The user-defined severity for the alarm.

Column Name	Column Description
Description	A one-line description of the alarm.
Info	Information on the alarm including alarm name, description, and recommended action to address the alarm.

You can click the alarm name to configure alarm settings.

Retrieving the List of Alarms Using the CLI or REST API

The basic command to list all alarms by type (Cluster, Node, User, or Volume) is:

```
maprcli alarm list -type (cluster|node|volume|AE)
```

For complete reference information, see `alarm list` on page 1545.

Configuring Alarm Settings

Set the severity and notifications for each alarm using either the Control System or the CLI.

Configuring Alarm Settings Using the Control System

To configure alarm setting from the Control System:

1. Log in to the Control System and click **Admin** > **Cluster Settings** > **Alarms**.
2. Select:
   - **Cluster Alarms** to configure settings for the alarms that affect the cluster as a whole
   - **Node Alarms** to configure settings for the alarms that indicate problems on individual nodes
   - **Table Alarms** to configure settings for the alarms that indicate table replication-related problems
   - **User Alarms** to configure settings for the alarms that indicate problems with user or group quotas
   - **Volume Alarms** to configure settings for the alarms that indicate problems in individual volumes
3. Click the name of the alarm to display the **Alarm Settings** window.
4. Specify a description of the alarm under **GENERAL** settings.
5. Configure alarm notifications (under **NOTIFICATIONS**) to allow MapR to send an email notification when the alarm is raised.
   a) Select (to enable) or deselect (to disable) the **Email Notifications** checkbox.
   b) If notifications are enabled, enter the user name or group name to which to send email when the alarm is raised, and click **Add**.
   See Setting Up SMTP on page 754 for additional information.
6. Click **Save Changes** to save your settings.

Configuring Alarm Settings Using the CLI or REST API

To set up alarm notifications, run the `alarm config save` command from the command line.

Warning: You must have `fc` (full control) or a (admin) permissions to run this command.

last-updated: Apr 11, 2022

The format of the command is:

```
maprcli alarm config save -cluster <cluster_name> -values
"<alarm>,<enableEmail>,<email>"
```

Assign values as follows:

Value	Description	Example
`alarm`	Name of the alarm	`DISK_FAILURE_ALARM`
`enableEmail`	Specifies whether individual alarm notifications are sent to any email address (including the default email address) for the alarm type. • 0 - do not send notifications to any email address for the alarm type • 1 - send notifications to all email addresses for the alarm type	`1`
`email`	One or more email addresses other than the default email address. If specified, alarm notifications are sent to these addresses as well, if *enableEmail* is set to *1*. Multiple email addresses must be separated by spaces only. You cannot use commas or other delimiters. For example, `user1@mycorp.com user2@mycorp.com` is valid.	`user1@mycorp.com`

Configuring the Alarm Threshold Using the CLI
You can configure the alarm threshold for certain alarms. For the alarms that support threshold configuration, this topic describes the command to run to set the threshold.

VOLUME_ALARM_INODES_EXCEEDED

Threshold is configurable at both the cluster and the volume level.

If configured at both the cluster and volume levels, the volume level threshold overrides cluster-level threshold.

To configure at the cluster-level, run the following commands:

```
# /opt/mapr/bin/maprcli config
save -values
'{"cldb.max.inodes.volume.alarm.thresh
":"<value>"}'
# /opt/mapr/bin/maprcli config
save -values
'{"cldb.default.max.namespace.size.mb.
alarm.thresh":"<value>"}'
```

To configure at the volume-level, run the following commands:

```
# /opt/mapr/bin/maprcli
volume modify -name
<volname> -maxinodesalarmthreshold
<threshold>
# /opt/mapr/bin/maprcli
```


last-updated: Apr 11, 2022

	`volume modify -name <volname> -maxnssizembalarmthreshold <threshold>`
VOLUME_ALARM_TOPOLOGY_ALMOST_FULL	Threshold is configurable at cluster level. To configure, run the following command: `# /opt/mapr/bin/maprcli config save -values '{"cldb.topology.almost.full.percentage":"<value>"}'`
VOLUME_ALARM_QUOTA_EXCEEDED	Threshold is configurable in volume properties. To configure, run the following command: `# /opt/mapr/bin/maprcli volume modify -name <volname> -quota <value>`
VOLUME_ALARM_TABLE_INDEX_LAG_HIGH	Threshold is configurable in volume properties. To configure, run the following command: `# /opt/mapr/bin/maprcli volume create -name <volname> -dbindexlagsecalarmthresh <value in seconds>`
VOLUME_ALARM_TABLE_REPL_LAG_HIGH	Threshold is configurable in volume properties. To configure, run the following command: `# /opt/mapr/bin/maprcli volume create -name <volname> -dbrepllagsecalarmthresh <value in seconds>`
VOLUME_ALARM_ADVISORY_QUOTA_EXCEEDED	Threshold is configurable in volume properties. To configure, run the following command: `# /opt/mapr/bin/maprcli volume modify -name <volname> -advisoryquota <value>`
AE_ALARM_AEQUOTA_EXCEEDED	Threshold is configurable in ae properties. `# /opt/mapr/bin/maprcli entity modify -name <entityname> -type <type> -quota <value>`
AE_ALARM_AEADVISORY_QUOTA_EXCEEDED	Threshold is configurable in ae properties. `# /opt/mapr/bin/maprcli entity modify -name <entityname> -type <type> -advisoryquota <value>`
NODE_ALARM_TOO_MANY_CONTAINERS	Threshold is configurable at cluster level.

last-updated: Apr 11, 2022

	This alarm is also raised when total number of containers (including snap containers) exceed 10 times the value of `pernode.numcntrs.alarm.thr`. `# /opt/mapr/bin/maprcli config save -values '{"pernode.numcntrs.alarm.thr":"<value>"}'`
NODE_ALARM_NO_HEARTBEAT	Threshold is configurable at cluster level. `# /opt/mapr/bin/maprcli config save -values '{"cldb.fs.mark.inactive.sec":"<value>"}'`
NODE_ALARM_HIGH_MFS_MEMORY	Threshold is configurable at cluster level. This alarm is raised when MapR File System memory consumption exceeds the threshold. `# /opt/mapr/bin/maprcli config save -values '{"mfs.high.memory.alarm.threshold":"<value>"}'`
CLUSTER_ALARM_CLUSTER_ALMOST_FULL	Threshold is configurable at cluster level. `# /opt/mapr/bin/maprcli config save -values '{"cldb.cluster.almost.full.percentage":"<value>"}'`
CLUSTER_ALARM_LICENSE_NEAR_EXPIRATION	Threshold is configurable at cluster level. `# /opt/mapr/bin/maprcli config save -values '{"mapr.license.exipry.notificationdays":"<value>"}'`
CLUSTER_ALARM_TOO_MANY_SNAPSHOT_CONTAINERS	Threshold is configurable at the cluster level by setting the value for the `cldb.snap.cntr.count.alarm.threshold` property in the `cldb.conf` file. See `cldb.conf` on page 2182 for more information. **Note:** The default value is `100000000`.

Viewing Alarm Information

Describes how to view alarm information using the Control System.

You can view notes for an alarm in the Control System.

Viewing Alarm Information in the Alarms Page

1. Log in to the Control System and click **Admin** > **Cluster Settings** > **Alarms**.
2. Select:
   - **Cluster Alarms** to view notes for the alarms that affect the cluster as a whole

last-updated: Apr 11, 2022

- **Node Alarms** to view notes for the alarms that indicate problems on individual nodes
- **Table Alarms** to view notes for the alarms that indicate table replication-related problems
- **User Alarms** to view notes for the alarms that indicate problems with user or group quotas
- **Volume Alarms** to view notes for the alarms that indicate problems in individual volumes

3. Click  associated with the alarm in the **Info** column.
   The **Alarm Information** window displays a description of the alarm and the recommended action to address the alarm.

Viewing Alarm Information in the Active Alarms Pane

- Click  associated with the alarm in the **Active Alarms** pane to view information on the alarm.
  The **Alarm Information** window displays a description of the alarm, type of alarm, information on the alarm, and recommended action to address the alarm.

Dismissing an Alarm
Describes how to dismiss an alarm, either manually or automatically, using either the Control System or the CLI.

You can dismiss an alarm using the Control System or the CLI. When you dismiss an alarm, it will be cleared. You can raise the alarm again:

- Manually using the CLI
- Automatically when the conditions for raising the alarm again are met

Dismissing Alarm(s) Using the Control System

The **Dismiss** action is available in all the **Active Alarms** pane and in the **Alarms Summary** page. To dismiss alarm(s):

1. Click **Dismiss**.
   The **Dismiss Alarms** dialog displays.
2. Verify the alarm(s) to dismiss and click **Dismiss** to dismiss the alarm(s).

Dismissing Alarm(s) Using the CLI

The basic command to dismiss an alarm is:

```
maprcli alarm clear -alarm <alarm>
```

For complete reference, see `alarm clear` on page 1536.

Muting and Unmuting Alarms
Describes how to mute and unmute alarms using either the Control System or the CLI.

You can mute (silence) one or more (non-critical) alarms for a specific period of time using either the Control System or the CLI. The alarm will be silenced for the duration of the mute period and CLDB will raise the alarm again after the mute period **only** if the conditions for raising the alarm instance again are met.

Muting Alarm(s) Using the Control System

The **Mute** action is available in all the **Active Alarms** pane and in the **Alarms Summary** page. To mute alarm(s):

1. Click **Mute** to display the **Mute Alarms** dialog.

last-updated: Apr 11, 2022

2. Verify the alarm(s) to mute.
3. Select the period of time (1 hour, 6 hours, or 24 hours) to mute the alarm for from the **Mute Alarms for** drop-down list.
4. Click **Mute Alarms** to mute the alarm(s).
   The alarm will be raised again if the associated issue is not resolved within the specified period of time.

Muting Alarm(s) Using the CLI

The basic command to mute an alarm is:

```
maprcli alarm mute -alarm <alarm name>[:<entity>]:<mute_period>
```

For complete reference, see alarm mute on page 1548.

Unmuting Alarm(s) Using the CLI

The basic command to mute an alarm is:

```
maprcli alarm unmute
```

For complete reference, see alarm unmute on page 1552.

Working with Multiple Instances of the File System
The Multi-MapR File System feature allows multiple instances of the file server to run on a single node in a single process.

Multiple instances of the MapR file server can run on a single node in a single process with the installation of the MapR XD Distributed File and Object Store, or the MapR Database software. On servers with SSDs with at least 2 storage pools (SP), two instances (per node) are configured by default. On servers without SSDs, a single instance is configured by default. Each instance runs as a separate library that is dynamically loaded into a single process. In this mode, each instance has a separate host ID; however, all the instances share the same hostname.

The maximum number of supported instances is 32. Instances should be configured based on the available CPU, memory, disks, and SPs. Each instance needs a minimum of 2GB, and instances should not exceed:

- Number of CPUs / 2
- Number of SPs (enforced)

File Server Instances

On MapR XD and MapR Database installations, nodes with SSDs can run multiple file server instances. To determine whether a node has SSDs, MapR uses the value of `mfs.disk.is.ssd` in the mfs.conf on page 2204 file, which must be set to `1`. Add this parameter to mfs.conf on page 2204, on a node that has SSDs.

For clusters with MapR XD or MapR Database license, if you set the `mfs.disk.is.ssd` in the mfs.conf on page 2204 file to 1, CLDB configures nodes with SSDs to have 2 file server instances by default. On homogeneous clusters, you can modify the number of instances by changing the value of the `multimfs.numsps.perinstance` parameter.

Ports for Multiple Instances of the MapR File System

Each instance listens on its own set of ports. Ensure that the appropriate ports are open for this feature. For example, instance 0 will use four ports from 5660, 5692 (5660+32), 5724 (5660+64), and 5756

(5660+96), instance 1 will use four ports from 5661, 5693, 5725, 5757, and so on for every additional instance. The topology of all instances is the same.

The total number of instances depends on the number of MFS threads, as indicated by the parameter `mfs.numrpcthreads` in mfs.conf on page 2204.

To verify that these ports are open, run the following command from a remote machine:

```
mrconfig -i -h <ip> -p <port number> info threads
```

An error indicates that the port is not open. If a port (for example, port 5661) is blocked, this command prints something similar to the following:

```
------------------
|From Instance 5661::|
------------------
<...> rpc failed <...>
```

Host IDs for Multiple Instances of the MapR File System

For multiple instances of the MapR File System, do NOT add IDs manually, but let the system handle the host ID numbering.

Log Files

Each instance has its own set of log files in `$MAPR_HOME/logs`. When multiple instances are configured, the log files have the same name with a different instance ID; for example, `mfs.log.<N>-3` where `N` is the instance number.

Note: For the primary instance, the log file name does not include the instance number.

The RPC and security trace information are in a separate file per instance, `mfs-<N>.err`, where `N` is the instance number. For the primary instance, the file name does not include the instance number.

For example, suppose there are 2 instances running on ports 5660 and 5661. There are 2 sets of log files, one for each instance:

- `mfs.log-3` for the primary instance
- `mfs.log.1-3` for the second instance

The RPC and security trace information are present in the following files :

- `mfs.err` for the primary instance
- `mfs-1.err` for the second instance

Configuring the Number of Storage Pools per Instance

Describes how to set the number of storage pools for each filesystem instance, from the CLI.

As you add MapR File System instances, MapR assigns SPs to them. If MapR File System instances are removed, the SPs assigned to those instances are re-allocated among the remaining live MapR File System instances. By default, the value is 1, which implies that there is only 1 SP for all instances. You can re-configure the number of SPs per instance globally or at the node-level.

last-updated: Apr 11, 2022

Note: If the number of MapR File System instances is not as configured, the Instance Mismatch Alarm will be raised. If the alarm is raised on a:

- CLDB node, restart warden by running the following command:

```
service mapr-warden restart
```

- Non-CLDB node, restart file server by running the following command:

```
maprcli node services -nodes <node-ip> -fileserver restart
```

Global Configuration

If you configure globally, the configuration will be applied to all the nodes in the cluster. Make the following changes only on homogeneous clusters (that is, when all nodes in the cluster have the same type of disks and the stripe width of the disks is the same):

1. Run the following commands:

```
maprcli config save -values {multimfs.numsps.perinstance:3}
maprcli config save -values {multimfs.numinstances.pernode:2}
```

   The default value of the `multimfs.numsps.perinstance` parameter is 0. Suppose a node reports 9 SPs:

   - For a value of 3, the node would need to start 3 instances.
   - For a value of 5, the node would need to start 2 instances.

   For clusters with fast SSDs, this can be set to 1.

   Note: On AWS nodes with HDD, set the `multimfs.numsps.perinstance` parameter value to 50 to use a single instance.

2. Restart Warden in every node for the configuration change to take effect.

Node-level Configuration

At the node level, you can configure different number of instances for each node in the cluster. To change the number of SPs per instance:

1. Run the following command:

```
maprcli node modify -nodes <nodename> -numSpsPerInstance <n>
```

   The number of instances changes automatically when new SPs are created.

2. Restart Warden on the nodes where the configuration has changed.

Monitoring Multiple Instances of MapR File System
Describes how to monitor the health and performance of your cluster.
Determining the Number of Running Instances

- Run the following command to determine the number of instances actually running:

```
/opt/mapr/server/mrconfig info instances
```

Your output will look similar to the following. This output shows that two File Server instances are running on ports `5660` and `5661`.

```
/opt/mapr/server/mrconfig info instances
    2
    5660 5661
```

Alternatively, on large clusters, run the following command to:

- Determine the number of configured instances:

```
maprcli node list -columns numInstances
    hostname           numInstances  ip
    atsqa4-161.qa.lab  1             10.10.88.161
    atsqa4-162.qa.lab  1             10.10.88.162
    atsqa4-163.qa.lab  1             10.10.88.163
    atsqa4-164.qa.lab  1             10.10.88.164
```

- Determine the number of running instances reported by MapR File System to CLDB:

```
maprcli node list -columns numReportedInstances
    numReportedInstances  hostname           ip
    2                     atsqa4-161.qa.lab  10.10.88.161
    1                     atsqa4-162.qa.lab  10.10.88.162
    1                     atsqa4-163.qa.lab  10.10.88.163
    2                     atsqa4-164.qa.lab  10.10.88.164
```

Determining the Number of MapR File System Threads

You can run the mrconfig info threads on page 2163 command to view MapR File System threads from all the instances. The output is tagged to identify the instance.

Converting a Cluster from Root to Non-Root User from the Command-Line
Provides a synopsis of changing the running user from `root` to a non-root user on a cluster.

You can change a MapR cluster that runs as `root` to a non-root user. In addition to converting the MapR user to a non-root user, you can also disable superuser privileges to the cluster for the `root` user for additional security.

Warning: You must perform these steps on all nodes on a stable cluster. Do not perform this procedure concurrently while upgrading packages.

Converting a MapR Cluster from Root to Non-Root User from the Command-Line
Lists the process to change the running user from `root` to a non-root user on a cluster.

1. Create a user with the same UID/GID across the cluster. Assign that user to the `MAPR_USER` environment variable.
2. On each node:
   a) Stop the warden and the ZooKeeper (if present).

```
service mapr-warden stop
service mapr-zookeeper stop
```

   b) Run the `config-mapr-user.sh` script to configure the cluster to start as the non-root user.

```
/opt/mapr/server/config-mapr-user.sh -u <MapR user> [-g <MapR group>]
```


last-updated: Apr 11, 2022

c) Start the ZooKeeper (if present) and the warden.

```
service mapr-zookeeper start
service mapr-warden start
```

3. After the previous step is complete on all nodes in the cluster, run the `upgrade2mapruser.sh` script on all nodes.

```
/opt/mapr/server/upgrade2mapruser.sh
```

This command may take several minutes to return. The script waits ten minutes for the process to complete across the entire cluster. If the cluster-wide operation takes longer than ten minutes, the script fails. Re-run the script on all nodes where the script failed.

Warning: The `MAPR_UID_MISMATCH` alarm may be raised during this process. The alarm will be cleared when this process is complete on all nodes.

Disabling Superuser Access for the Root User from the Command-Line
Describes how to disable superuser access for the `root` user.

Note: Enabling the `cldb.squash.root` **OR** `cldb.reject.root` configuration values can cause instability with ecosystem open source components if they are running as `root`. [On MapR clusters, services are running as the admin cluster user, which is `mapr` (by default).] Root squash applies only to files, not tables or streams. Ensure that `root` is not running any services before performing this procedure.

Important: You can enable either of the following parameters, but NOT both.

1. To disable root user (UID 0) access to the MapR filesystem on a cluster that is running as a non-root user, use either of the following commands:
   - The *squash root* configuration value treats all requests from UID 0 as coming from UID -2 (nobody):

```
/opt/mapr/bin/maprcli config save -values {"cldb.squash.root":"1"}
```

   - The *reject root* configuration value automatically fails all filesystem requests from UID 0.

```
/opt/mapr/bin/maprcli config save -values {"cldb.reject.root":"1"}
```

2. You can verify that these commands worked, as shown in the following example.

```
/opt/mapr/bin/maprcli config load -keys cldb.squash.root,cldb.reject.root
cldb.reject.root cldb.squash.root
0 1
```

Starting Up a Cluster
Lists the steps to start a cluster that was previously shut down.

1. If the cluster nodes are not running, start them.
2. Change to the `root` user (or use `sudo` for the following commands).
3. Start ZooKeeper on the nodes where it is installed:

```
service mapr-zookeeper start
```

4. On all nodes, start Warden:

```
service mapr-warden start
```

5. Over a period of time (depending on the cluster size and other factors), the cluster comes up automatically. After the CLDB restarts, there is a 15-minute delay before replication resumes. This delay allows all nodes to register and begin heartbeat processing. You can configure this delay by using the `config save` command to set the `cldb.replication.manager.start.mins` parameter.

Shutting Down a Cluster

Lists the considerations to note and the procedure to shutdown a cluster.

Verify that MapReduce processes are not active, and that no data is being loaded to the cluster or being persisted within the cluster.

When you shut down a cluster, follow this sequence to preserve your data and replication:

1. Verify that recent data has finished processing.
2. Shut down any NFS servers.
3. Shut down any ecosystem components that are running.
4. Shut down ResourceManager and NodeManager services if you are using YARN
5. Shut down Warden on all nodes that are not running CLDB.
6. Shut down Warden on the CLDB nodes.
7. Shut down ZooKeeper on the ZooKeeper nodes.

Complete the following steps to shut down the cluster:

1. Change to the `root` user (or use `sudo` for the following commands).
2. Before shutting down the cluster, you will need a list of NFS nodes, CLDB nodes, and all remaining nodes. Once the CLDB is shut down, you cannot retrieve a list of nodes; it is important to obtain this information at the beginning of the process. Use the node list on page 1705 command as follows:
   - Determine which nodes are running the NFS gateway. Example:

```
/opt/mapr/bin/maprcli node list -filter "[rp==/
*]and[svc==nfs]" -columns id,h,hn,svc,rp
id
service
hostname             health  ip
6475182753920016590
fileserver,nodemanager,nfs,hoststats
node-252.cluster.us 0       10.10.50.252
8077173244974255917
nodemanager,cldb,fileserver,nfs,hoststats
node-253.cluster.us 0       10.10.50.253
5323478955232132984
webserver,cldb,fileserver,nfs,hoststats,resourcemanager
node-254.cluster.us 0  10.10.50.254
```

- Determine which nodes are running the CLDB. Example:

```
/opt/mapr/bin/maprcli node list -filter "[rp==/
*]and[svc==cldb]" -columns id,h,hn,svc, rp
```

- List all non-CLDB nodes. Example:

```
/opt/mapr/bin/maprcli node list -filter "[rp==/*]and[svc!
=cldb]" -columns id,h,hn,svc, rp
```

3. Shut down all NFS instances. Example:

```
/opt/mapr/bin/maprcli node services -nfs stop -filter [svc=="nfs"]
```

4. If your cluster is running any ecosystem components, shut down those components on all nodes.
5. Shut down all ResourceManager and NodeManager services on all nodes. To shut down ResourceManager and NodeManager services specify the maprcli node services command with the `name` parameter and either the `filter` or the `node` parameter . Example:

```
maprcli node services -name resourcemanager -filter <filter> -action stop
maprcli node services -name nodemanager -nodes <node> -action stop
```

6. SSH into each node that is not running CLDB and stop Warden with the command:

```
service mapr-warden stop
```

7. SSH into each CLDB node and stop Warden with the command:

```
service mapr-warden stop
```

8. SSH into each Zookeeper node and stop Zookeeper with the command:

```
service mapr-zookeeper stop
```

9. (Optional) Shut down the nodes using the Linux `halt` command.

Managing Another Unsecure Cluster Using the Control System
When logged into the Control System of one cluster, you can navigate to the Control System of another unsecure cluster.

To navigate to the Control System of another unsecure cluster:

- Select the name of the cluster to go to from the drop-down list associated with the name of the cluster into which you are currently logged.

  After a few minutes, the Control System login page for the other cluster displays.

Allocating Cluster Resource from the Command-Line
Provides a general overview on allocating cluster resources for a MapR Hadoop cluster.

In a MapR Hadoop cluster, the warden sets the default resource allocation for the operating system, MapR File System, MapR Hadoop services, and YARN applications. Warden allocates resources to MapR Hadoop services and applications based on the roles installed on a node. For example, warden allocates resources for YARN applications on nodes with NodeManager role installed.

In general, you should not need to override the values set in the default configuration files and by warden. However, you can provide updated values by adding or updating parameters in the Hadoop site configuration files or warden files. To override parameter values for a single job, the option can be overridden in the command line when submitting a YARN application to the cluster.

To determine the current value of a hadoop parameter, run `hadoop conf | grep <ParameterName>`. In the following example, the `hadoop conf` command was used to get the value of `mapreduce.map.memory.md`:

```
hadoop conf | grep mapreduce.map.memory.mb
<property><name>mapreduce.map.memory.mb</name><value>1024</
value><source>mapred-site.xml</source></property>
```

Alternatively, run `hadoop conf` without the `grep` command to get a full list of the current parameter values. To determine the current value of a warden parameter, open the warden files located in the following directories: `/opt/mapr/conf/conf.d` and `/opt/mapr/conf`.

Note: In some cases, the current value of the parameter can only be seen in the Control System or in ResourceManager.

Refer to Allocating Memory for Nodes on page 818 to allocate memory and resources in a MapR cluster.

Administering Nodes

Provides a synopsis of managing nodes in a cluster.

This section provides information about managing nodes in a MapR cluster. Topics include how to add nodes to the cluster and/or remove nodes from the cluster, manage the services installed on the nodes, and manage disks. You can manage nodes, disks, and services in the MapR cluster using the Control System and the CLI.

Managing Nodes

Describes the Nodes page on the Control System.

The **Nodes** page contains panes that display:

- Node Health — the health of the nodes organized by topology (by default) or service.
- Current Resource Utilization — the nodes that utilize the most (in percentage) CPU and memory.
- Active Node Alarms — the list of active node alarms on the cluster.
- List of nodes — the list of nodes on the cluster.

Note: The **Nodes** page is not available in the Kubernetes version of the Control System.

You can perform the following procedures to manage and monitor nodes using the Control System and the CLI:

Viewing the list of Nodes
Explains how to view the list of Nodes using either the Control System or the CLI.
Viewing the list of Nodes on the Control System

- Log in to the Control System and click **Nodes**.

  Note: The **Nodes** page is not available in the Kubernetes version of the Control System.

  The page contains the following panes:

Node Health	Displays each node's health.
Active Alarms	Displays active alerts for nodes in the cluster.
Current Resource Utilization	Plots the current CPU and memory utilization for each node as a graph. This helps visualize the nodes that utilize the most CPU and memory.
Nodes	Displays all the nodes in the cluster.

For each node in the cluster, the **Nodes** pane displays the following:

Column Name	Column Description
Health	The health of the node. Value can be: • — Healthy • — Degraded • — Critical • — Maintenance
Hostname	The hostname of the node.
Physical IPs	The physical IP address or addresses associated with the node.
Last FS Heartbeat	The time since the node's last heartbeat to the CLDB.
Memory Utilized	The amount of memory used by the node.
CPU Utilized	The CPU usage metric for the node.
Disk Utilized	The amount of disk space utilized on the node.
Physical Topology	The rack path to the node.
Running Services	The number of services running on the node.

Selecting the checkbox beside a node makes the following buttons available:

- Change Topology
- Remove Nodes
- Manage Services

Retrieving the list of Nodes Using the CLI or REST API

The basic command to view all the nodes on a cluster is:

```
maprcli node list -cluster <cluster>
```

For complete reference information, see node list on page 1705.

Customizing the List of Columns/Fields
Explains how to customize the columns that are displayed in the Control System, and the fields that are returned in the CLI.
Customizing the Columns in the Control System

1. Log in to the Control System and go to:

last-updated: Apr 11, 2022

- **Data** > **Volumes** page to customize columns displayed in the **Volumes** pane.

  Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

- **Nodes** page to customize columns displayed in the **Nodes** pane.

  Note: The **Nodes** page is not available on the Kubernetes version of the Control Panel.

2. Click the **Customize Columns** icon ( ).
   In the **Customize Columns** dialog, the:
   - **Available** list displays the columns that are available for display.
   - **Selected** list displays the columns currently displayed in the pane.
3. Select the columns from the:
   a) Available list of columns and click to move selection to **Selected** columns (for display).
   b) Selected list of columns and click to remove selected columns from displaying.
4. (Optional) Click and/or down arrows to sort the order of columns.
5. Click **Save Changes** for the customization to take effect.

   Tip: To reset the display to its default columns, click **Reset to default columns**.

Customizing the Fields Using the CLI or REST API

Use the `-column` parameter with the `maprcli` command to view specific fields in the list. For example:

- To view the health of the nodes and services installed on the nodes being retrieved, run the following command:

  ```
  maprcli node list -columns service,health
  ```

  For complete reference information, see the `node list` on page 1705 command.

- To view the volume name for the list of volumes being retrieved, run the following command:

  ```
  maprcli volume list -columns volumename
  ```

  For complete reference information, see `volume list` on page 1979 command.

Reverting to Default List of Columns
Describes how to revert to the default list of columns on the Control System

1. Log in to the Control System and click:
   - **Data** > **Volumes** to revert to the default list of columns in the **Volumes** pane.

     Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

   - **Nodes** to revert to the default list of columns in the **Nodes** pane.

     Note: The **Nodes** page is not available in the Kubernetes version of the Control System.
2. Click the **Customize Columns** icon ( ) .

last-updated: Apr 11, 2022

3. Click **Reset to default columns**,
4. Click **Save Changes**.
   The pane displays the default list of columns.

Filtering the List of Nodes
Describes how to setup search expressions and filter nodes based on specific criteria.

The filter lets you build search expressions to provide sophisticated filtering capabilities for locating specific data on views that display a large number of nodes. Expressions are implicitly connected by the AND operator.

Filtering the Node List in the Control System

1. Log in to the Control System and click **Nodes** to filter the list of nodes in the **Nodes** pane.

   Note: The **Nodes** menu is not available on the Kubernetes version of the Control System.

2. Click and select one of the following from the **Add Filter** drop down menu.
   - Health — to filter the list by node health
   - Hostname — to filter the list by hostname of node
   - Physical IP — to filter the list by IP address of node
   - Last FS Heartbeat — to filter the list by number of heartbeats sent to FS
   - CPU Utilized — to filter the list by number of cores utilized
   - Physical Topology — to filter the list by rack path
   - Configured Services — to filter the list by configured services
   - Memory Used — to filter the list by amount of memory used
   - Running Services — to filter the list by installed services
   - Used Disk Space — to filter the list by amount of disk space used
   - Virtual IP — to filter the list by virtual IP address
3. Specify the value in the drop-down field for the selected filter (to filter the list of nodes by) and click **Filter**.

   As you make selections and specify the filtering criteria, the pane displays only the nodes that match the specified filtering criteria.
4. Click:
   - **Add Filter** to add another filtering criteria.
   - to remove a filtering criteria.
   - to clear all filter settings.

Filtering the List Using the CLI

Use the `node list on page 1705` command with the `-filter` option, to specify large numbers of nodes by matching specified values in specified fields rather than by typing the name of each node explicitly. For example, you can retrieve all nodes on a specific subnet as follows:

```
maprcli node list -filter [ip==20.30.40.*]
```

For more information, see Filters on page 1526.

Monitoring Nodes
Explains how to monitor nodes using either the Control System or the CLI.

You can check the health of the nodes on the cluster in the Control System, organized by service or by topology, or by using the CLI.

Note: The metrics collection infrastructure must be installed during installation to visualize the graphs and charts. If the metrics collection infrastructure is not installed, perform an Incremental Install to install the metrics collection infrastructure.

Note: The **Nodes** page is not available on the Kubernetes version of the Control System.

Monitoring Node Health Using the Control System

To monitor the health of nodes:

1. Log in to the Control System and click:
   - **Overview** to view the health of the nodes in the **Node Health** pane.
   - **Nodes** to view the health of the nodes in the **Node Health** pane.
2. Select one of the following from the drop-down menu in the **Node Health** pane.
   - **By Service** to organize the display of nodes by services.

     This is the default view in the **Overview** page. This view contains the list of services and the nodes on which the service is running ( ) and is down ( ).

     Note: The color of the node (which reflects the status of the service) is even when a service is stopped (not running) on the node.
   - **By Topology** to view the display of nodes by topology.

     This is the default view in the **Nodes** page. This view contains the list of topologies and the health of the nodes (as shown in the following table) in the topology.

	Indicates the node is healthy.
	Indicates the node is degraded and/or may need attention. A node is considered to be in degraded state if: • There is no heartbeat from the MapR filesystem/NFS node for over 60 seconds. • One or more services are down on the node. • One or more alarms are raised on the node.
	Indicates the node is in maintenance mode.
	Indicates critical issue(s) on the node. A node is considered to be in critical state if: • There is no heartbeat from the node for more than 5 minutes.

last-updated: Apr 11, 2022

- All MapR files system disks on the node are dead or are offline.
- All containers on the node are being re-replicated because either the node was removed, unregistered, or there was no heartbeat from the node for more than 1 hour.
- File server is dead/inactive because there is no heartbeat for a long time.
- NFS server on node is dead.
- MapR install directory is full.
- Node reported high MapR filesystem memory usage.

Monitoring Node Resource Utilization from the Control System

- Log in to the Control System and click **Nodes** to view the nodes that consumed the most CPU and memory (in percentage) in the **Current Resource Utilization** pane. The shade of the bubble indicates node resource utilization with the darker shade indicating the nodes that are nearing disk capacity.

Monitoring Active Node Alarms from the Control System

See Viewing Active Node Alarms on page 1317 for more information.

Monitoring Node Health Using the CLI or REST API

You can check general health of the nodes with the following command:

```
maprcli node heatmap -cluster <cluster>
```

This command displays a heatmap for the nodes on the specified cluster; a subset of the output can also be visualized on the Control System. For complete reference information, see `node heatmap` on page 1702.

Viewing Node Details

Describes how to view node details using either the Control System or the CLI.

Viewing Node Details Using the Control System

1. Log in to the Control System and click **Nodes**.

   Note: The **Nodes** page is not available in the Kubernetes version of the Control System.

2. Click the hostname of the node.

   The information page for the node displays.

   - Summary tab (default view)
   - Metrics tab

   You can:

   - Change the topology of the node
   - Remove the node

Viewing Node Details Using the CLI or REST API

The basic command to retrieve information on a node is:

```
maprcli node metrics -nodes <hostname> -columns <column names>
```

For complete reference information, see `node metrics` on page 1723.

Viewing Node Summary
View a summary of the alarms, services, and disks on a node using either the Control System or the CLI.
Viewing Node Summary Using the Control System

- Log in to the Control System and go to the node information page.

  The page with information on the node displays and the **Summary** tab displays by default. The **Summary** tab contains panes for node-specific:

  - **Alarms** — displays active and recent alarms on the node. See Viewing Active Node Alarms on page 1317 for more information.
  - **Services** — displays services running on the node. See Viewing the Services Running on a Node Using the Control System on page 828 for more information.
  - **Disks** — displays information on the disks on the node. See Viewing the List of Disks on page 834 for more information.

Viewing Node Summary Using the CLI

The basic command to retrieve a summary of the disks on a node is:

```
maprcli node metrics -nodes <hostname> -columns DISKS
```

For complete reference information, see `node metrics` on page 1723.

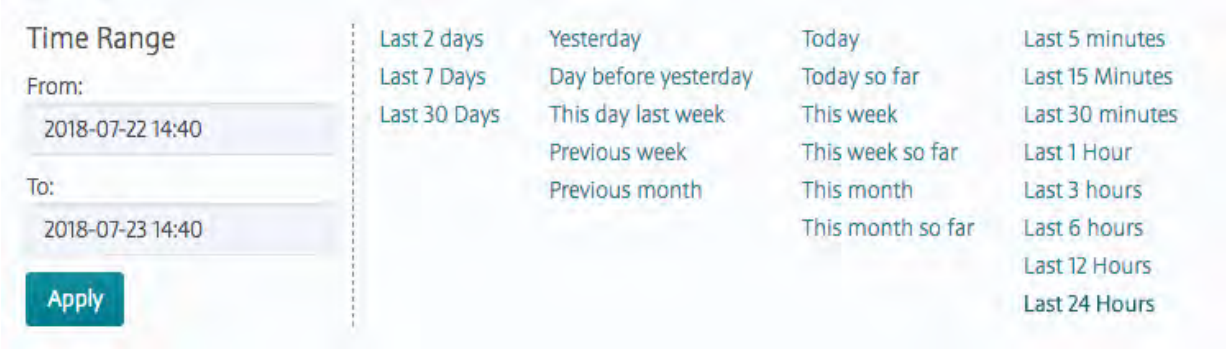
Viewing Node Metrics
Explains how to view node metrics using the Control System.

Note: The metrics collection infrastructure must be installed during installation to visualize the metrics in the various panes. If the metrics collection infrastructure is not installed, perform an Incremental Install to visualize the metrics that are desribed in the following section.

Monitoring Node Metrics Using the Control System

- Log in to the Control System and go to the **Metrics** tab in the node information page.

  By default, the page displays charts that show metrics for the last 24 hours. You can select a preset or specify a custom time range.



  You can also zoom in (by clicking and dragging the cursor in the pane) for a more granular view. Click **Zoom Out** to expand time window or click:

  - › to shift time window forwards.
  - ‹ to shift time window backwards.

  Click ⓘ associated with the chart to view information about the graph. Click ✎ to display the **Customize Active Charts** window. You can select charts to display and remove from the **Available** and **Selected** lists in the **Customize Active Charts** window. You can view up to 6 charts at a time in the page.

last-updated: Apr 11, 2022

Use the following table when selecting the charts to view in the page. In the following table, the Charts column lists the charts that are available and the Metric column describes that type of metric that can be visualized in the chart:

Metric	Charts
CPU Usage	• Node Active CPU Usage • Node CPU Usage** • Node CPU Usage IDLE • Node CPU Usage NICE • Node CPU Usage SYSTEM • Node CPU Usage USER • Node CPU Usage WAIT • MFS CPU Usage • Allocated vs Available CPU Cores • MapR Process CPU Usage • MAST Gateway CPU Usage • DB Gateway CPU Usage • Data Access Gateway CPU Usage
Memory Usage	• Node Free Memory • Node Utilized Memory*** • Node Memory Free vs Used* • MFS Process Memory Usage • MapR Process Memory Usage
SWAP Space	• Node Swap Free • Node Swap Used • Node Swap Space Available vs Used* • Node Swap IO
Node IOs	• Node Network IO* • Node Network Interface Input • Node Network Interface Output • Node Network Interface Error Input • Node Network Interface Error Output

last-updated: Apr 11, 2022

Metric	Charts
System Disk Throughput	• Disk Read Ops • Disk Write Ops • Disk Reads and Writes*
System Disk Latency	• Disk Avg Read Latency • Disk Avg Write Latency • Disk Read and Write Times
MFS Throughput	• MFS Read Throughput • MFS Write Throughput • MFS Read and Write Throughput • MFS System Disk Activity in Bytes*

* This metric is diplayed in the default chart view for a node.

** This metric is diplayed in the default chart view for a node and in the default list view for a table.

*** This metric is diplayed in the default list view for a table.

For information on viewing metrics for:

- All table activities on a node, see Viewing Per Node Metrics for Table Activities on page 1297.
- All stream activities on a node, see Monitoring Streams Operations Using the Control System on page 1313.

Setting Up Node Topology
Define node topologies for every node in the cluster.

Define your cluster's topology by specifying a topology for each node in the cluster. You can use topology to group nodes by rack or switch, depending on how the physical cluster is arranged and how you want MapR to place replicated data.

Topology paths can be as simple or complex as needed to correspond to your cluster layout. In a simple cluster, each topology path might consist of the rack only (for example, `/rack-1`). In a deployment consisting of multiple large datacenters, each topology path can be much longer (for example, `/europe/uk/london/datacenter2/room4/row22/rack5/`). MapR uses topology paths to spread out replicated copies of data, placing each copy on a separate path. By setting each path to correspond to a physical rack, you can ensure that replicated data is distributed across racks to improve fault tolerance.

Changing the Topology of one or more Nodes
Describes how to move nodes from one topology to the other using either the Control System or the CLI.
Changing the Topology of Multiple Nodes Using the Control System

To change the rack or switch path for one or more nodes, under **Nodes**:

Note: The **Nodes** menu is not available on the Kubernetes version of the Control System.

1. Select the nodes from the list of nodes in the **Nodes** pane and click **Change Topology**.
   The **Change Node Topology** dialog displays.
2. Choose one of the following:

last-updated: Apr 11, 2022

- **Select Existing Topology** to select a topology from the list of existing topologies.
- **Create New Topology** to specify a new topology for the selected nodes.

3. Click **Change Topology** for the changes to take effect.

Changing the Topology of a Node Using the Control System

To change the rack or switch path for a node:

1. Go to the node information page and click **Change Topology**.
   The **Change Node Topology** dialog displays.
2. Choose one of the following:
   - **Select Existing Topology** to select a topology from the list of existing topologies.
   - **Create New Topology** to specify a new topology for the node.
3. Click **Change Topology** for the changes to take effect.

Changing the Topology Using the CLI or REST API

The basic command to move nodes to a different topology is:

```
/opt/mapr/bin/maprcli node move -serverids <server IDs> -topology <topology>
```

For complete reference information, see `node move` on page 1728.

The move will fail if the server ID is negative. To fix this issue, perform one of the following:

- If you are moving only a single server ID that is negative, or a bunch of server IDs that are all negative, prefix 0 as an additional server ID. For example:

```
/opt/mapr/bin/maprcli node move -serverids
0,-6151492882499457449,-2668056288676628812 -topology /data/mytopo -json
```

- If you are moving a bunch of server IDs with a mix of positive and negative server IDs, place a positive ID as the first ID. For example:

```
/opt/mapr/bin/maprcli node move -serverids
1507661865183706279,-6151492882499457449,-2668056288676628812 -topology /
data/mytopo -json
```

Setting Node Topology with a Script
Provides an overview of how to script setting up node topology.

For large clusters, you can specify complex topologies in a text file or by using a script. Each line in the text file or script output specifies a single node and the full topology path for that node in the following format:

```
<ip or hostname> <topology>
```

The text file or script must be specified and available on the local filesystem on all CLDB nodes:

- To set topology with a text file, set `net.topology.table.file.name` in `/opt/mapr/conf/cldb.conf` to the text file name
- To set topology with a script, set `net.topology.script.file.name` in `/opt/mapr/conf/cldb.conf` to the script file name

If you specify a script and a text file, the MapR system uses the topology specified by the script.

Adding Nodes to a Cluster
Describes how to add nodes to a cluster.

You can add nodes to a cluster using the web-based MapR Installer (version 1.6 or later), the MapR Installer Stanzas, or manually. To add nodes to your cluster using the MapR Installer or MapR Installer Stanzas, see Extending a Cluster by Adding Nodes on page 5437. Complete the following steps to add nodes manually to a cluster:

1. Prepare all nodes.

   If you do not use the Domain Name System (DNS), ping the new node from an existing node and vice versa. Use the host name instead of an IP address. If you do not get a response, and if you rule out a network problem, a possible fix is to edit the `/etc/hosts` files of all nodes in the cluster. All nodes need to be listed in all `/etc/hosts` files.

2. Plan which packages to install based on services you want to run on the new nodes.

   See Select Services on page 107 and MapR Repositories and Packages on page 128 for more information.

3. Install MapR Software.

   - On all new nodes, add the MapR Repository.
   - On each new node, install the planned packages.

   See Step 2: Prepare Packages and Repositories on page 142 and Step 3: Install Cluster Service Packages on page 150 for more information.

4. Configure all new nodes by running `configure.sh`.

   If you added a ZooKeeper role to a node, run the following command on all nodes with the new ZooKeeper list: `configure.sh -no-autostart`. See `configure.sh` on page 2053 for more information.

5. On all new nodes, format disks for use by MapR if you plan to re-use a node from another cluster.

   Format the disks from a re-used node to remove data from the old cluster.

   Note: All the disks (for use by MapR) on a node must be of the same type. That is, all the disks on a node must either be rotational or SSDs; node with disks of both types is not supported.

   See Formatting Disks on a Node From the Command-line on page 840 for more information.

6. If you manually modified configuration files on the existing nodes and those changes apply to the new nodes, copy only those changes to the respective files on the new nodes.

7. Perform the following steps if you added the node(s) to any secure cluster that is configured for cross-cluster operations.

   a) Copy the `/opt/mapr/conf/mapr-clusters.conf` file and `/opt/mapr/conf/ssl_truststore` file from another node to the new node(s).

   b) Copy the `/opt/mapr/conf/maprserverticket` file from:

      - A CLDB node if the new node is a CLDB node.
      - A non-CLDB node if the new node is not a CLDB node.

      The `/opt/mapr/conf/maprserverticket` file contains additional entry for cross-cluster tickets. See Configuring Secure Clusters for Cross-Cluster NFS Access on page 1490 for more information.

last-updated: Apr 11, 2022

8. Start ZooKeeper on all new nodes that have ZooKeeper installed:
   `service mapr-zookeeper start`
9. Start Warden on all new nodes:
   `service mapr-warden start`
10. Restart services that you reconfigured.
    Running `configure.sh` alone does not reconfigure services, such as ZooKeeper. Reconfigured services also require a restart. For example, restart ZooKeeper on each node, one at a time after running `configure.sh`. Restart the lead ZooKeeper last. Restarting ZooKeeper adds the new nodes into the existing ZooKeeper quorum. Services that need to connect to CLDB do not always discover a newly added CLDB node without restarting warden.
11. Set up node topology for the new nodes.
12. On any new nodes running NFS, set up NFS for HA.

Isolating CLDB Nodes
Lists the pros of creating CLDB-only nodes.

In a large cluster (100 nodes or more) create CLDB-only nodes to ensure high performance. This configuration also provides additional control over the placement of the CLDB data, for load balancing, fault tolerance, or high availability (HA). Setting up CLDB-only nodes involves restricting the CLDB volume to its own topology and making sure that all other volumes are on a separate topology. As both the CLDB-only path and the non-CLDB path are children of the root topology path, new non-CLDB volumes are not guaranteed to keep off the CLDB-only nodes. To avoid this problem, set a default volume topology. See Setting Default Volume Topology Using the CLI on page 917.

Setting Up a CLDB-Only Node
Describes how to setup a node for CLDB alone.

1. SET UP the node as usual:
   a) PREPARE the node, making sure it meets the requirements.
   b) ADD the MapR repository.
2. INSTALL the following packages to the node.
   - `mapr-cldb`
   - `mapr-webserver`
   - `mapr-core`
   - `mapr-fileserver`

Setting Up Volume Topology to Restrict the CLDB Volume to Specific Nodes
Explains how to permit access to CLDB volumes only from specific nodes.

1. Move all CLDB nodes to a CLDB-only topology (e. g. `/cldbonly`) using the MapR Control System or the following command:
   ```
   maprcli node move -serverids <CLDB nodes> -topology /cldbonly
   ```
2. Restrict the CLDB volume to the CLDB-only topology using the MapR Control System or the following command:
   ```
   maprcli volume move -name mapr.cldb.internal -topology /cldbonly
   ```

Moving Volumes to a Separate Topology from the CLDB-Only Nodes
Explains how to move non-CLDB volumes to a separate topology.

1. Move all non-CLDB nodes to a non-CLDB topology (e. g. `/defaultRack`) using the MapR Control System or the following command: `maprcli node move -serverids <all non-CLDB nodes> -topology /defaultRack`
2. Restrict all existing volumes to the topology `/defaultRack` using the MapR Control System or the following command: `maprcli volume move -name <volume> -topology /defaultRack`

   All volumes except `mapr.cluster.root` are re-replicated to the changed topology automatically.

   Warning: To prevent subsequently created volumes from encroaching on the CLDB-only nodes, set a default topology that excludes the CLDB-only topology.

Isolating ZooKeeper Nodes
Provides an overview on how to install a ZooKeeper-only node.

For large clusters (100 nodes or more), isolate the ZooKeeper on nodes that do not perform any other function. Isolating the ZooKeeper node enables the node to perform its functions without competing for resources with other processes. Installing a ZooKeeper-only node is similar to any typical node installation, but with a specific subset of packages.

Warning: Do not install the FileServer package on an isolated ZooKeeper node in order to prevent MapR from using this node for data storage.

Setting Up a ZooKeeper-Only Node
Explains how to install a ZooKeeper-only node.

1. SET UP the node as usual:
   a) PREPARE the node, making sure it meets the requirements.
   b) ADD the MapR Repository.
2. INSTALL the following packages to the node.
   - `mapr-zookeeper`
   - `mapr-zk-internal`
   - `mapr-core`

Configuration Example
This example assumes you are adding a new node to a cluster that is running the CLDB and ZooKeeper on three other nodes: `node_a`, `node_b`, and `node_c`. To configure a `new node_d`, which is not a CLDB or ZooKeeper node, run the following command:

```
$ /opt/mapr/server/configure.sh -N my.cluster.com -C
node_a,node_b,node_c -Z node_a,node_b,node_c
```

To configure a ZooKeeper node, use the `-no-autostart` option and the `-Z` option followed by the list of ZooKeeper nodes.

Removing Nodes from a Cluster
Provides an overview of how to remove nodes from a cluster.

You can remove a node using the `node remove` command, or using the Control System. Removing a node detaches the node from the cluster, but does not remove the MapR software from the cluster.

The following sections provide information about removing nodes from a cluster:

last-updated: Apr 11, 2022

Removing One or More Nodes
Describes how to decommission a node from service.

Perform the following prerequisite steps before removing a node using the Control System or CLI or REST API:

1. Drain the node of data by moving the node to the `/decommissioned` physical topology. All the data on a node in the `/decommissioned` topology is migrated to other volumes and nodes in the appropriate topologies.

2. Run the following command to check if a given volume is present on the node:

   ```
   maprcli dump volumenodes -volumename <volume> -json | grep IP:Port
   ```

   As an example, consider the volume `rocky` that is present on a node with IP `10.163.167.212`. To check whether this volume exists on this node, run the command:

   ```
    maprcli dump volumenodes -volumename rocky -json
   {
           "timestamp":1606879372378,
           "timeofday":"2020-12-01 07:22:52.378 GMT-0800 PM",
           "status":"OK",
           "total":1,
           "data":[
                   {
                           "Servers":{

   "IP:Port":"10.163.167.212:5660-192.168.122.1:5660--3-VALID"
                           }
                   }
           ]
   }
   ```

   The output shows that the volume `rocky` exists on node `10.163.167.212` that is accessible on port `5660`.

   To return just the IP and the Port alone, pipe the output through the `grep` command as follows:

   ```
    maprcli dump volumenodes -volumename test -json | grep IP:Port
    "IP:Port":"10.163.167.212:5660-192.168.122.1:5660--3-VALID"
   ```

   Run this command for each non-local volume in your cluster to verify that the node being removed is not storing any volume data.

3. Install CLDB or ZooKeeper on another node (only) if the node you are removing is a CLDB or ZooKeeper node and run `configure.sh` with `-C` and `-Z` options.

   This is to ensure that ZooKeeper quorum is maintained and that an optimal number of CLDB is available for high availability.

You can remove one or more nodes using the Control System and the CLI.

Removing Multiple Nodes Using the Control System

To remove one or more nodes:

1. Log in to the Control System and click **Nodes**.

   Note: The **Nodes** menu is not available on the Kubernetes version of the Control System.

2. Select the nodes from the list of nodes in the **Nodes** pane and click **Remove Node(s)**.
   The **Remove Node(s)** dialog displays.
3. Verify the list of nodes to remove and click **Remove Nodes**.

Removing a Node Using the Control System

To remove a node:

1. Go to the Viewing Node Details on page 802 page and click **Remove Node**.
   The **Remove Node(s)** confirmation dialog displays.
2. Click **Remove Node**.

Removing one or more Nodes Using the CLI or REST API

Use the node remove on page 1729 command to remove one or more server nodes from the cluster. To run this command, you must have full control (`fc`) or administrator (a) permission. The syntax is:

```
maprcli node remove -nodes <node names> ]
```

If the following error is generated, you must wait for the state duration of the CLDB master node to reach 15 minutes or more. Otherwise, the node remove fails:

```
node remove failed for node <node_name>, Error: Resource temporarily
unavailable; CLDB just became master, node removed not allowed until
sometime
```

To check the state duration value, use this command:

```
maprcli dump cldbstate -json
```

After you issue the node remove on page 1729 command, wait several minutes to ensure that the nodes have been completely removed.

> **Tip:** To ensure that a node that is removed does not rejoin the cluster on reboot, either remove all MapR packages from the node, or remove the cluster configuration that is present in the `/opt/mapr/conf/mapr-clusters.conf` file on the node.

Related concepts

Migrating a Volume off a Node Using the CLI on page 915

Decommissioning a Node and Uninstalling MapR Software from the Command-line

Use the following procedure to remove a node and uninstall MapR software. This procedure detaches the node from the cluster and removes the MapR packages, log files, and configuration files, but does not format the disks.

Note: Before decommissioning a node, make sure any data on the node is replicated and any needed services are running elsewhere. If the node you are decommissioning runs a critical service, such as CLDB or ZooKeeper, verify that enough instances of that service are running on other nodes in the cluster. See Planning the Cluster for recommendations about service assignment to nodes.

Complete the following steps to permanently decommission a node:

1. Drain the node of data by moving the node to the `/decommissioned` physical topology. All data on a node in the `/decommissioned` topology is migrated to volumes and nodes in the `data` topology.

last-updated: Apr 11, 2022

2. Run the following command to check if a given volume is present on the node:

```
maprcli dump volumenodes -volumename <volume> -json | grep <ip:port>
```

Note: Run this command for each non-local volume in your cluster to verify that the node being decommissioned is not storing any volume data.

3. Change to the `root` user (or use `sudo` for the following commands).
4. Stop Warden on the node by running the following command:

```
service mapr-warden stop
```

5. If ZooKeeper is installed on the node, stop it: `service mapr-zookeeper stop`
6. Determine which MapR packages are installed on the node:
   - `dpkg --list | grep mapr` (Ubuntu)
   - `rpm -qa | grep mapr` (Red Hat or CentOS)
7. Remove the packages by issuing the appropriate command for the operating system, followed by the list of services. Examples:
   - `apt-get purge mapr-core mapr-cldb mapr-fileserver` (Ubuntu)
   - `yum erase mapr-core mapr-cldb mapr-fileserver` (Red Hat or CentOS)
8. Remove the `/opt/mapr` directory to remove any instances of `hostid`, `hostname`, `zkdata`, and `zookeeper` left behind by the package manager.
9. Remove the `/mapr` directory to delete any NFS/POSIX client mount points.
10. Remove any MapR cores in the `/opt/cores` directory.
11. If the node you have decommissioned is a CLDB node or a ZooKeeper node, then run `configure.sh` on all other nodes in the cluster. See Configuring the Node.
12. Remove the node by running the following command:

```
maprcli node remove -nodes <node-name>
```

Reconfiguring a Node from the Command-Line
Provides an overview of the procedure to reconfigure a node using the CLI.

This procedure is designed to make changes to existing MapR software on a machine that has already been set up as a MapR cluster node. If you need to install software for the first time on a machine to create a new node, see Adding Nodes to a Cluster.

Complete the following steps to reconfigure a node:

1. Stopping the Node
Describes how to stop a node.

1. Change to the `root` user (or use `sudo` for the following commands).
2. Stop Warden: `service mapr-warden stop`

last-updated: Apr 11, 2022

3. Stop ZooKeeper, if installed on the node: `service mapr-zookeeper stop`

2. Formatting the Disks (Optional)
Provides an overview of how to format the disks.

If you are re-using a node that was used previously in another cluster, be sure to format the disks by using the `disksetup` script to remove any traces of data from the old cluster. Refer to the previous section, Formatting Disks on a Node, for instructions.

3. Installing or Removing Software or Hardware
Lists the considerations to install or remove software or hardware.

When the node is stopped, you can add, upgrade or remove software or hardware. At some point after adding or removing services, you should restart Warden, to re-optimize memory allocation among all the services on the node. It is not crucial to perform this step immediately; you can restart Warden any time when the cluster is not busy.

To add or remove individual MapR packages, use the standard package management commands for your Linux distribution:

- `apt-get` (Ubuntu)
- `yum` (Red Hat or CentOS)

For information about the packages to install, see Planning the Cluster.

The following sections provide information about adding or removing services from a node after it has been deployed in a cluster:

Adding a Service to an Existing Node
Explains how to add a service to a node.

The process of adding a service to a node is similar to the initial installation process for nodes. For further detail see Installing MapR Software.

1. Install the package(s) corresponding to the new role(s) using `apt-get` or `yum`.
2. Run `configure.sh` with a list of the CLDB nodes and ZooKeeper nodes in the cluster.
3. If you added the CLDB or ZooKeeper role, you must run `configure.sh` on all other nodes in the cluster.
4. If you added the fileserver role, run `disksetup` to format and prepare disks for use as storage.
5. Restart Warden.

```
service mapr-warden restart
```

When Warden restarts, it picks up the new configuration and starts the new services, making them active in the cluster.
Removing a Service from an Existing Node
Explains how to remove a service from a node.

1. Stop the service you want to remove by using the Control System or the `maprcli` command-line tool. The following example stops the Fileserver service:

```
maprcli node services -fileserver stop -nodes mapr-node1
```

2. Purge the service packages with the `apt-get`, `yum`, or `zypper` commands, as suitable for your operating system.

last-updated: Apr 11, 2022

3. Run the `configure.sh` script with the `-R` option.
4. When you remove the CLDB or ZooKeeper role from a node, run `configure.sh -R` on all nodes in the cluster.

4. Configuring the Node
Provides an overview of the `configure.sh` script to use to configure a node.

The script `configure.sh` configures a node to be part of a MapR cluster, or modifies services running on an existing node in the cluster. The script creates (or updates) configuration files related to the cluster and the services running on the node.

Before you run `configure.sh`, make sure you have a list of the hostnames of the CLDB and ZooKeeper nodes. You can optionally specify the ports for the CLDB and ZooKeeper nodes as well. The default ports are:

Service	Default Port #
CLDB	7222
ZooKeeper	5181

The script `configure.sh` takes an optional cluster name and log file, and comma-separated lists of CLDB and ZooKeeper host names or IP addresses (and optionally ports), using the following syntax:

```
/opt/mapr/server/configure.sh -C <host>[:<port>][,<host>[:<port>]...] -Z
<host>[:<port>][,<host>[:<port>]...] [-L <logfile>][-N <cluster name>]
```

Note:

Each time you specify the `-Z <host>[:<port>]` option, you must use the *same order* for the ZooKeeper node list. If you change the order for any node, the ZooKeeper leader election process will fail.

Example:

```
/opt/mapr/server/configure.sh -C
r1n1.sj.us:7222,r3n1.sj.us:7222,r5n1.sj.us:7222 -Z
r1n1.sj.us:5181,r2n1.sj.us:5181,r3n1.sj.us:5181,r4n1.sj.us:5181,r5n1.sj.us:5
181 -N MyCluster
```

5. Starting the Node
Explains how to start a node.

1. If ZooKeeper is installed on the node, start it: `service mapr-zookeeper start`
2. Start Warden: `service mapr-warden start`

Renaming a Node from the Command-Line
Provides distribution-specific instructions for renaming a node.

Attention: Ensure that the host name you set is resolvable. Add the host name to the `/etc/hosts` file. For example: `10.10.19.22 host.qa.net`. MapR installation and commands fail if the host name is not resolvable.

To rename a node:

1. Stop Warden on the node. Example:

```
service mapr-warden stop
```


last-updated: Apr 11, 2022

2. If the node is a ZooKeeper node, stop ZooKeeper on the node. Example:

```
service mapr-zookeeper stop
```

3. Rename the host:
   - Red Hat 6.x and CentOS 6.x: To preserve the new host name after reboot, edit the `HOSTNAME` parameter in the `/etc/sysconfig/network` file and restart the `xinetd` service or reboot the node. To change the host name temporarily without a reboot, run:

     ```
     hostname desired-host-name
     ```

   - Red Hat 7.x and CentOS 7.x: Run the command:

     ```
     hostnamectl set-hostname desired-host-name --static
     ```

     Alternatively, enter the host name in the `/etc/hostname` file, and run:

     ```
     hostname -F /etc/hostname
     ```

     Both these methods preserve the host name across reboots.

   - On Ubuntu, first install *dbus* if it is not installed.

     ```
     apt-get install dbus
     ```

     Next, run the command:

     ```
     hostnamectl set-hostname desired-host-name --static
     ```

     Alternatively, edit the host name in the `/etc/hostname` file, and run:

     ```
     hostname -F /etc/hostname
     ```

     Both these methods preserve the host name across reboots.

4. If the node is a ZooKeeper node or a CLDB node, run `configure.sh` on page 2053 with a list of CLDB and ZooKeeper nodes.
5. If the node is a ZooKeeper node, start ZooKeeper on the node. Run:

```
service mapr-zookeeper start
```

6. Start Warden on the node. Run:

```
service mapr-warden start
```

After you rename a:

- CLDB or ZooKeeper node, run `configure.sh` on page 2053 on all the nodes with the new host name, to update the `mapr-clusters.conf` file with the new host name. Ensure that there are no duplicate entries in the file. Also, verify that the new host is accessible from all the nodes.

last-updated: Apr 11, 2022

- Node, some local volumes (such as for audit, and metrics) may exist with both the old and new host names. If you want, you can remove the local volumes with the old host name, use the existing local volume path, or remount to the new path.

Changing the IP Address of a Node
Describes how to change the IP address of any node in the cluster using the CLI.
Changing the IP Address of a Data Node

Complete the following steps to change the IP address of a data node:

1. Shut down Warden and ZooKeeper on the node to be changed.

   ```
   service mapr-zookeeper stop
   service mapr-warden stop
   ```

2. Change the IP address of the node.
3. Edit the `/etc/hosts` file on all nodes to reflect the IP address change, or ensure that the IP addresses are resolvable through a DNS search.
4. On the node where you changed the IP address, restart the network interface. The interface shuts down, so you lose the connection.
5. Log into the node using the new IP address.
6. Check the IP address.
   For example, run `ifconfig`.
7. If the `MAPR_SUBNETS` environment variable is set, edit the value for the `MAPR_SUBNETS` environment variable in the `/opt/mapr/conf/env.sh` file and make sure that the new IP address is part of it.
   See Setting Environment Variables for NIC Segregation on page 850 for more information.
8. Restart Warden on the node(s) where the IP address has changed.
9. Check that all nodes appear in the output of the node list command.

   ```
   /opt/mapr/bin/maprcli node list -columns ip
   ```

   You might have to wait a few minutes until all nodes are registered before you get the output from this command.

Changing the IP Address of CLDB Node

Complete the following steps to change an IP address of a CLDB node:

1. Shut down Warden and ZooKeeper on the node to be changed.

   ```
   service mapr-zookeeper stop
   service mapr-warden stop
   ```

2. Change the IP address of the node.
3. Edit the `/etc/hosts` file on all nodes to reflect the IP address change, or ensure that the IP addresses are resolvable through a DNS search.
4. On the node where you changed the IP address, restart the network interface. The interface shuts down, so you lose the connection.

5. Log into the node using the new IP address.
6. Check the IP address. For example, run `ifconfig`.
7. Run `configure.sh`.
   Use the `-C` option to provide a list of CLDB nodes.

   Note:

   If the initial setting was based on the IP address, run `configure.sh` on all nodes in the cluster.

   If the initial setting was based on the hostname, there is no need to run `configure.sh` on any nodes when you change the IP address.

8. Perform a rolling restart of Warden on all the nodes.
9. Check that all nodes appear in the output of the node list command. You might have to wait a few minutes until all nodes are registered before you get output from this command.

```
/opt/mapr/bin/maprcli node list -columns ip
```

Changing the IP Address of ZooKeeper Node

Complete the following steps to change an IP address of a ZooKeeper node:

1. Shut down Warden and ZooKeeper on the node to be changed.

```
service mapr-zookeeper stop
service mapr-warden stop
```

2. Change the IP address of the node.
3. Edit the `/etc/hosts` file on all nodes to reflect the IP address change, or ensure that the IP addresses are resolvable through a DNS search.
4. On the node where you changed the IP address, restart the network interface. The interface shuts down, so you lose the connection.
5. Log into the node using the new IP address.
6. Check the IP address.
   For example, run `ifconfig`.
7. Run `configure.sh`.
   Use the `-Z` option to provide the list of ZooKeeper nodes.

   Note:

   If the initial setting was based on the IP address, run `configure.sh` on all the ZooKeeper, CLDB, and Data nodes in the cluster.

   If the initial setting was based on the hostname, there is no need to run `configure.sh` on any nodes when you change the IP address.

8. If you run the Drillbit service on any nodes in the cluster:
   a) Change the ZooKeeper address in the `conf/drill-override.conf` file on the Drill nodes.

last-updated: Apr 11, 2022

b) Start ZooKeeper on the ZooKeeper node, and then perform a rolling restart of ZooKeeper on all other ZooKeeper nodes.

A rolling restart of ZooKeeper means restart ZooKeeper on each node, one at a time, pausing until the last restart finishes before beginning the next. Restart the ZooKeeper leader last.

9. Verify that the new node joined the ZooKeeper quorum.

```
service mapr-zookeeper status
```

10. Perform a rolling restart of Warden on all the nodes.

11. Check that all nodes appear in the output of the node list on page 1705 command. You might have to wait a few minutes until all nodes are registered before you get output from this command.

```
/opt/mapr/bin/maprcli node list -columns ip
```

Viewing Active Node Alarms
Describes how to view active node alarms using the Control System and the CLI.
Viewing Active Node Alarms in the Control System

- **Note:** The **Nodes** page is not available in the Kubernetes version of the Control System.

Log in to the Control System and:

- Click **Nodes** to view all the node alarms on the cluster in the **Active Alarms** pane.
- Go to the node information page to view alarms in the **Alarms** pane for the selected node.
- Click (in the top navigation bar) to display the **Alarm Summary** page and select **Node Alarms** from the drop-down menu in the **All** alarms pane.
- Click **Overview** and select **Node Alarms** from the drop-down menu in the **Active Alarms** pane to view all the node alarms on the cluster.

You can:

- View alarm notes
- Mute an alarm
- Dismiss an alarm

Retrieving Active Node Alarms Using the CLI or REST API

The basic command to retrieve node alarms is:

```
maprcli alarm list -cluster <cluster name> -type node
```

For complete reference information, see `alarm list` on page 1545.

Allocating Memory for Nodes
Describes how the Warden allocates memory for nodes.

When you run `configure.sh` on a node, Warden allocates memory for the operating system, `mfs service`, MapRHadoop services, and applications using the settings in the `warden.conf` and the `warden.<servicename>.conf` file.

Warden allocates memory to the following components in the following order:

1. Operating system
2. `mfs service`
3. MapR Hadoop services
4. Applications, such as YARN applications
5. If the node only runs MapR File System, NFS, and gateway, then 85% of all memory is allocated to MapR File System.

Note: In general, modify the settings in the warden files only under certain circumstances. If you modify the values in `warden.conf` or `warden.<servicename>.conf` file, you must restart Warden. Otherwise, updated parameters will not be used to allocate resources.

Allocating Memory for the OS, MapR File System, and Hadoop Services
Lists the parameters that control how Warden allocates memory for the OS, filesystem and the MapR Hadoop services.

Warden allocates memory to the operating system, MapR File System, and MapR Hadoop services based on the following parameters:

Parameter	Default OS	MapR File System	Hadoop Service(s)	Description
`service.command.<os|mfs|servicename>.heapsize.percent`	10	varies	varies	Defines the heap size percentage.
`service.command.<os|mfs|servicename>.heapsize.maxpercent`	Not Applicable	85	Not Applicable	Defines the heap size maximum percentage
`service.command.<os|mfs|servicename>.heapsize.max`	4000	Not Applicable	5000	Defines the maximum heap size in MB.
`service.command.<os|mfs|servicename>.heapsize.min`	256	512	256	Defines the minimum heap size in MB.

Memory settings for the operating system and the MapR File System are configured in the `warden.conf` file. The `warden.conf` file is located in `/opt/mapr/conf`. Other services, such as NodeManager and ResourceManager, have their own `warden.<servicename>.conf` file within `/opt/mapr/conf/conf.d`. For more information about the Warden files, see `warden.conf` and `warden.<servicename>.conf`.

Note: Warden allocates resources only for the MapR Hadoop services associated with roles that are installed on the node.

Allocating Memory for the MapR File System Service
Describes how Warden allocates memory for the filesystem service.

By default, Warden adds up the total memory consumed by all services and the OS, and then allocates 85% of the remainder to the MapR File System. If you do not intend to use MapR Database, you can set the `-noDB` option in `configure.sh` to specify that 20% of the memory available should be allocated to the MapR File System service. Allocating more memory to the MapR File System improves performance due to greater data caching. Data caching is especially vital when your main constraint is disk I/O. For the

last-updated: Apr 11, 2022

parameters that you can configure to give Warden more memory, see Allocating Memory for the OS, MapR File System, and Hadoop Services on page 819.

Performing Maintenance on a Node from the Command-Line
Describes how to maintain the performance of a node using the CLI.

You can place a node into a maintenance mode for a specified timeout duration. For the duration of the timeout, the cluster CLDBs do not consider this node's data as lost and do not trigger a resync of the data on the node.

The following sections describe how to perform maintenance on a node.

Putting a Node into Maintenance Mode
Describes how to put a node into maintenance mode.

When you put a node in maintenance mode, the node is marked unserviceable, but is still attached to the cluster.

Before putting a node into maintenance mode, ensure that:

- All copies of the CLDB volume exist if the node is a CLDB node. You can shut down the CLDB service on a node only if the CLDB is a secondary CLDB, or if you have enabled high availability for CLDB.

  You *cannot* put a node that is running both the CLDB and MapR File System services into maintenance mode.

- All running processing tasks (such as NodeManager, Spark, for example) that depend on the MapR File System have been stopped.

Warning: Do not put a node under maintenance if there are any volume under-replicated alarms because doing so may make some data completely offline.

To put a node into maintenance mode, perform the following actions:

1. From a terminal, issue the `node maintenance on page 1722` command:

   ```
   /opt/mapr/bin/maprcli node maintenance -nodes <IP|
   hostname> -timeoutminutes <minutes>
   ```

   When you run this command, specify a timeout (in minutes) long enough for you to perform necessary maintenance on the node.

   Note: For the duration of the timeout, the cluster's CLDB considers this node as available which implies CLDB does not trigger replication of data of this node until it is in maintenance mode. However, if a node is put under maintenance for more than 5 minutes, the MapR File System is shut down on that node so that clients who access containers on this node receive the appropriate error and retry other container copies. This value of 5 minutes is hard coded and cannot be changed. Even if you reboot the node, the maintenance mode persists till the timeout is reached.

2. Stop warden on the node.

   To bring the node back online, do one of the following::

   - Run the `node maintenance on page 1722` command with a timeout of `0`
   - Wait till the timeout is over. The node is automatically brought back online after the specified timeout is complete.

Taking a Node Out of Maintenance Mode
Describes how to bring a node back online from maintenance mode.

To take a node out of maintenance mode before the timeout expires, follow this process:

1. From a terminal, issue the following command:

```
maprcli node maintenance -nodes <IP address> -timeoutminutes 0
```

2. Restart Warden:

```
service mapr-warden restart
```

Managing Roles

Describes how to manage roles on a node.

You must install *role*s on nodes in a cluster before their corresponding *services* can be launched. For information on how to install roles on nodes, see Adding Roles to a Node. Refer to the following topics for managing the roles using the CLI.

Adding Roles to a Node

Lists the process to add a role.

You can add *role*s on a node after you deploy the node in a cluster. The process of adding a *role* to a node involves installing a package on the node and updating the cluster to recognize the new *role*. The process of adding a *role* depends on the *role* type.

Once you have added a *role* to a node, you must restart Warden. Observe the following best practices when restarting Warden on ZooKeeper and CLDB nodes:

- Perform a rolling restart of Warden to ensure that all services are up. A rolling restart of Warden means restart Warden on each node, one at a time, pausing until the previous restart finishes, before beginning the next.
- To avoid a failover from occurring, identify nodes running critical services, such as ResourceManager, and restart Warden last on those nodes.
- Restart Warden on nodes that run critical cluster services, such as ResourceManager, during periods of low activity.

Adding a Role

Describes how to add various roles to a MapR node.

Do not use these steps to add the CLDB or ZooKeeper role.

Note: When collectd is installed on a node with YARN ResourceManager or NodeManager, running configure.sh –R, to add or remove roles on the node, triggers these services to restart. During a restart, the NodeManager and ResourceManager are temporarily unavailable for new application submission. A patch is available to resolve this behavior. See Applying Patches.

The following steps describe how to add a role to a node:

1. Install the package corresponding to the new role using `apt-get, yum,` or `zypper,` depending on your platform.
2. Run `configure.sh -R` on the node where you added the role.

   If Warden is running, the new service starts automatically.
3. If you added the File server role, run `disksetup` to format and prepare disks for use as storage.

last-updated: Apr 11, 2022

4. Issue the following command to restart Warden on the node where you installed the role:

```
% service mapr-warden restart
```

Adding a CLDB Role Using the CLI
Describes how to add a CLDB role to a MapR node using the CLI.

Complete the following steps to add the CLDB role to a node in the cluster:
Adding a CLDB Role to a Node on an Unsecure Cluster

1. Install the CLDB package, `mapr-cldb`, on the node with the `apt-get`, `yum`, or `zypper` commands, depending on your operating system.
2. Run `configure.sh` on page 2053 with the `-C` and `-R` options on the node where you added the new CLDB role.
   Use the `-C` option to provide the list of CLDB nodes. If Warden is running, the CLDB service starts automatically.
3. Run `configure.sh` on page 2053 with the `-R` options on all other nodes in the cluster.
   Use the `-C` option and provide the list of CLDB nodes, including the new node.
4. Verify that the node has a CLDB role by running the following command:

```
maprcli node listcldbs
```

   The output should show all the CLDB nodes including the node where the role was added.

Adding a CLDB Role to a Node on a Secure Cluster

1. Install the CLDB package, `mapr-cldb`, on the node with the `apt-get`, `yum`, or `zypper` commands, depending on your operating system.
2. Copy the following files from the `/opt/mapr/conf` directory on any existing CLDB node on the cluster to the `/opt/mapr/conf` directory on this node.
   - `cldb.key`
   - `dare.master.key`
   - `maprserverticket`
   - `ssl_keystore`
   - `ssl_keystore.p12`
   - `ssl_keystore.pem`
   - `ssl_truststore`
   - `ssl_truststore.p12`
   - `ssl_truststore.pem`

   Note: This `dare.master.key` will be present only if the cluster is enabled for encryption of data at rest.

3. Run `configure.sh` on page 2053 with the following options on the node where you added the new CLDB role.
   - `-secure`: Use this option to enable the node for security.
   - `-C`: Use this option to include this node in the list of CLDB nodes. If Warden is running, the CLDB service starts automatically.
   - `-dare`: Use this option only if the cluster is enabled for data at rest encryption. Do not specify this if the cluster is not enabled for data at rest encryption.
   - Use one of the following to configure the list of ZooKeeper nodes:
     - `-R`: Use this option if the node is an existing cluster node. This option uses the previously configured list of ZooKeeper nodes. When `-R` is specified, the ZooKeeper credentials are read from `warden.conf` file.
     - `-Z`: Use this option if the node is a new node on the cluster. This option specifies the list of ZooKeeper nodes.
4. Run `configure.sh` on page 2053 with the following options on all other nodes in the cluster.
   - `-C`: Use this option to include the new CLDB node in the list of CLDB nodes.
   - `-R`: Use this option to use the previously configured list of ZooKeeper nodes.
5. Verify that the node has a CLDB role by running the following command:

   ```
   maprcli node listcldbs
   ```

   The output should show all the CLDB nodes including the node where the role was added.

Adding a ZooKeeper Role
Describes how to add a ZooKeeper role to a MapR node using the CLI.

If you are increasing the number of ZooKeeper roles in the cluster, for example from one to three, shut down the cluster before you add the role to the nodes to prevent any problems and then start the cluster upon completion.

Complete the following steps to add the ZooKeeper role to a node:

1. Install the ZooKeeper package corresponding to the new role.
2. Run `configure.sh -Z` on the node where you added the new ZooKeeper role.
   Using the `-Z` option provides the list of ZooKeeper nodes that includes the new node.
3. Run `configure.sh -Z` on all other nodes in the cluster.
   Using the `-Z` option to provide the list of ZooKeeper nodes that includes the new node.
4. Issue the following command to start or restart ZooKeeper on the node where you added the ZooKeeper role:

   ```
   % service mapr-zookeeper restart
   ```

5. Perform a rolling restart of ZooKeeper on all other ZooKeeper nodes.
   A rolling restart of ZooKeeper means restart ZooKeeper on each node, one at a time, pausing until the last restart finishes before beginning the next. Restart the ZooKeeper leader last.

6. Issue the following command to verify that the new node joined the ZooKeeper quorum:

```
% service mapr-zookeeper qstatus
```

7. Perform a rolling restart of Warden on all other nodes.

   Warden picks up the new ZooKeeper node. Issue the following command to restart Warden on the node where you installed the role:

```
% service mapr-warden restart
```

Removing Roles from a Node
Describes how to remove a role from a node.

You can remove *role*s from nodes in the MapR cluster. The process of removing a *role* from a node depends on the *role* type.

Once you remove a *role* from a node, you must restart Warden. Observe the following best practices when restarting Warden on nodes that were ZooKeeper or CLDB nodes:

- Perform a rolling restart of Warden to ensure that all services are up. A rolling restart of Warden means restart Warden on each node, one at a time, pausing until the previous restart finishes, before beginning the next.
- To avoid a failover from occurring, identify nodes running critical services, such as ResourceManager, and restart Warden last on those nodes.
- Restart Warden on nodes that run critical cluster services, such as ResourceManager, during periods of low activity.

Removing a Role
Describes how to remove a role from a node, using the CLI.

Do not use these steps to remove the CLDB, ZooKeeper, or Fileserver role from a node.

Note: When collectd is installed on a node with YARN ResourceManager or NodeManager, running configure.sh –R, to add or remove roles on the node, triggers these services to restart. During a restart, the NodeManager and ResourceManager are temporarily unavailable for new application submission. A patch is available to resolve this behavior. See Applying Patches.

The following steps describe how to remove a role from a node:

1. If you are removing the NFS role, unmount any existing client mounts.

   Removing the NFS role from a node affects any Virtual IP (VIP) pools that include this node.

2. If the cluster has only one CLDB, run `configure.sh` with the `-C` option on all the nodes.

3. Stop the service for the role you want to remove, either through the Control System or by issuing a `maprcli` command:

```
% maprcli node services -name <service_name> -action stop -nodes
<node-name>
```

   The following example stops the webserver role on node "my-node":

```
% maprcli node services -name webserver -action stop -nodes my-node
```

4. Purge the role packages with the `apt-get`, `yum`, or `zypper` commands, depending on your operating system.
5. Run `configure.sh -R` on the node where you removed the role.

   Warden picks up the new configuration automatically.
6. Issue the following command to restart Warden on the node where you removed the role:

```
% service mapr-warden restart
```

Removing a CLDB Role
Describes how to remove the CLDB role from a node.

1. If you have only one CLDB node in the cluster, add the CLDB role to another node.

   When failover occurs after removal of the CLDB node, the new CLDB node becomes the primary CLDB.
2. Issue the following command to stop the CLDB service on the node:

```
/opt/mapr/bin/maprcli node services -name cldb -action stop -nodes
mapr-<node>
```

3. Purge the CLDB package, `mapr-cldb`, with the `apt-get`, `yum`, or `zypper` commands, depending on your operating system.
4. Run `configure.sh` on page 2053 with the `-C`, `-N` and `-Z` options on the node where you removed the role.

   Use the `-C` option to provide the list of CLDB nodes, excluding the node where you removed the role, `-N` to pass the name of the cluster, and `-Z` to specfy the list of ZooKeeper nodes.
5. Run `configure.sh` on page 2053 with the `-C`, `-N` and `-Z` options on all other nodes in the cluster.

   Use the `-C` option to provide the list of CLDB nodes, excluding the node where you removed the role, `-N` to pass the name of the cluster, and `-Z` to specify the list of ZooKeeper nodes.

Removing a ZooKeeper Role
Describes how to remove the ZooKeeper role from a node.

The following steps describe how to remove the ZooKeeper role from a node:

1. Issue the following command to stop ZooKeeper on the node:

```
% service mapr-zookeeper stop
```

2. Purge the ZooKeeper package `mapr-zookeeper`.
3. Run `configure.sh`.

   Use the `-Z` option and provide the list of ZooKeeper nodes that excludes the node where you removed the role.
4. Perform a rolling restart of ZooKeeper on all other ZooKeeper nodes.

   A rolling restart of ZooKeeper means restart ZooKeeper on each node, one at a time, pausing until the last restart finishes before beginning the next. Restart the ZooKeeper leader last.

5. Issue the following command to verify that ZooKeeper is healthy and that the expected nodes adopted the ZooKeeper node:

```
% service mapr-zookeeper qstatus
```

6. Perform a rolling restart of Warden on all other nodes.

   Warden picks up the revised quorum.

Removing a Fileserver Role
Describes how to remove the Fileserver role from a node.

Removing the fileserver role from a node is more complex than removing other roles. The CLDB tracks data precisely on all fileserver nodes, and therefore you should direct the cluster CLDB to stop tracking the node before removing the fileserver role. For a planned decommissioning of a node, use node topologies to migrate data off the node before removing the fileserver role. For example, you could move the node out of a live `/data` topology into a `/decommissioned` topology that has no volumes assigned to it, in order to force data off the node. Otherwise, some data will be under-replicated as soon as the node is removed. Refer to Node Topology.

Note: The following procedure involves halting all MapR services on the node temporarily. If this might disrupt critical services on your cluster, such as CLDB, migrate those services to a different node first, and then proceed.

The following steps describe how to remove the fileserver role from a node:

1. Stop the warden, which will halt all MapR services on the node. Wait 5 minutes, after which the CLDB will mark the node as critical.
2. Remove the node from the cluster, to direct the CLDB to stop tracking this node.

   You can do this in the Control System GUI or with the `maprcli node remove` command.
3. Remove the fileserver role by deleting the file `/opt/mapr/roles/fileserver` on the node.
4. Run `configure.sh` on the node to reconfigure the node without the fileserver role.
5. Issue the following command to restart Warden on the node:

```
% service mapr-warden restart
```

6. Remove any volumes that were stored locally on the node.

   You can do this in the Control System or with the `maprcli volume remove` command.

Assigning Roles to Nodes for Best Performance
Guidelines to optimise the cluster's service layout for best performance.

The architecture of MapR software allows virtually any service to run on any node, or nodes, to provide a high-availability, high-performance cluster. The following guidelines help plan your cluster's service layout.

Do not Overload ZooKeeper

High latency on a ZooKeeper node can lead to an increased incidence of ZooKeeper quorum failures. A ZooKeeper quorum failure occurs when the cluster finds too few copies of the ZooKeeper service running. If the ZooKeeper node is also running other services, competition for computing resources can lead to increased latency for that node. If your cluster experiences issues relating to ZooKeeper quorum failures, consider reducing or eliminating the number of other services running on the ZooKeeper node.

last-updated: Apr 11, 2022

Separate High-Demand Services

The following guideline states the services to separate on large clusters:

- **ResourceManager on ZooKeeper nodes**: Avoid running the ResourceManager service on nodes that are running the ZooKeeper service. On large clusters, the ResourceManager service can consume significant resources.

Managing Services

Synopsis on managing services.

Once a role is installed on a node and the warden has been restarted, MapR recognizes the role for that node. You can then start the service. Refer to the following topics for information on managing services on a node using the Control System and the CLI.

Viewing the List of Services

Explains how to view the list of services using either the Control System or the CLI.

Viewing the Services Installed on the Cluster Using the Control System

- Log in to the Control System and click **Services**.

  The **Services** pane displays all the services installed on the cluster. On the non-Kubernetes version of the Control System, the pane displays the following:

Column Name	Column Description
Service	The name of the installed service.
Running Nodes	The number of nodes on which the associated service is running. The service can be stopped ( ) or restarted ( ). Click the number in this column to view the nodes on which the service is running.
Standby Nodes	The number of nodes on which the associated service is in standby (available, but not running) state. The service can be started ( ) or restarted ( ). Click the number in this column to view the nodes on which the service is in standby state.
Failed Nodes	The number of nodes on which the service has failed. The service can be started ( ) or restarted ( ). Click the number in this column to view the nodes on which the service has failed.
Stopped Nodes	The number of nodes on which the associated service is stopped (and not running). The service can be started ( ) or restarted ( ). Click the number in this column to view the nodes on which the service has been stopped.
Config Path	The path to the configuration file for the service.
Log Viewer	(Displays only if Kibana is installed on a node) The link ( ) to the Kibana UI.

  You can filter the list of services displayed by:

  - **EEP**, which includes services such as Hive, Drill, etc.
  - **Core**, which includes services such as CLDB, Hoststats, File server, etc.
  - **Monitoring**, which includes services such as Grafana, Kibana, etc.

last-updated: Apr 11, 2022

Viewing the Services Running on a Node Using the Control System

1. Log in to the Control System and click **Nodes**.

   Note: The **Nodes** menu is not available in the Kubernetes version of the Control System.

2. You can:
   - Hover the cursor over the number listed in the **Running Services** column in the **Nodes** pane to view the list of services installed on that node.
   - Go to the **Summary** tab in the node information page to view detailed information on the services installed on a node.

   In the **Summary** tab, for each service running on the node, the **Services** pane displays the following:

Column Name	Column Description
Service	The name of the service.
State	The current state of the service. Value can be: • Running • Stopped
Memory Allocated	The amount of system memory allocated to the service.
System Memory Utilized	The percentage of memory utilized by the service.
CPU Usage	The CPU used by the service.
Log Path	The path to the service log file.
Log Viewer	The link to the Kibana UI (only if Kibana is installed).

   You can select the checkbox beside one or more services to take the following actions:
   - Start Services
   - Stop Services
   - Restart Services

   Note: If Kibana is installed, you can click to view the logs. See Kibana User Guide for more information.

Retrieving the Services Running on a Node Using the CLI or REST API

The command to list all the services on a node is:

```
maprcli service list -node <node name>
```

For complete reference information, see service list on page 1746.

Enabling and Disabling a Service Using the CLI and REST API

Describes how to enable or disable a service using either the REST API or the CLI.

You can disable a service to prevent it from starting or restarting when Warden starts or restarts, and enable a service to allow it to start or restart when Warden starts or restarts.

last-updated: Apr 11, 2022

Disabling a Service Using the CLI or REST API

CLI	Run the following command: `maprcli node services -nodes <hostName> -name <serviceName> -action disable`
REST	Send a request of type POST. For example: `curl -k -X POST 'https://<host>:8443/rest/node/services?nodes=<hostName>&name=<serviceName>&action=disable' --user mapr:mapr`

Note: When you disable a service, the service is stopped and the service is not automatically starrted/restarted when Warden is started/restarted.

See `node services` on page 1730 for more information.

Enabling a Service Using the CLI or REST API

CLI	Run the following command: `maprcli node services -nodes <hostName> -name <serviceName> -action enable`
REST	Send a request of type POST. For example: `curl -k -X POST 'https://<host>:8443/rest/node/services?nodes=<hostName>&name=<serviceName>&action=enable' --user mapr:mapr`

Note: When you enable a service, the service is started/restarted when Warden is started/restarted.

See `node services` on page 1730 for more information.

Related tasks

Restarting the Services on page 831
Describes how to restart a service using either the Control System, the CLI or the REST API.

Starting the Services
Explains how to start services using either the Control System, the CLI or the REST API.

You can start one or more services using the Control System or the CLI if the service is not disabled. If the service is disabled, you must enable the service first, in order to start the service. See Enabling and Disabling a Service Using the CLI and REST API on page 828 for more information.

Starting the Services Running on the Nodes Using the Control System

To start the services running on the nodes:

1. Log in to the Control System and click **Nodes** to display the **Nodes** page.

   Note: The **Nodes** page is not available on the Kubernetes version of the Control System.

2. Select the nodes from the list of nodes in the **Nodes** pane and click **Manage Services** to display the **Manage Services** window.

last-updated: Apr 11, 2022

3. Choose the **Start** radio button for the services you wish to start on the selected nodes and click **Save**.

Starting the Services Running on a Node Using the Control System

To start one or more services running on a node:

1. Go to the **Summary** tab in the node information page.
2. Select the services to start in the **Services** pane.
3. Click **Start Services**.
   The **Start Services** confirmation dialog displays.
4. Verify the list of services to start and click **Start Service**.

Starting the Services on the Cluster Using the Control System

1. Log in to the Control System and click **Services** to display the list of services on the cluster.
2. On the non-Kubernetes version of the Control System, click  for the service to start.
   The **Start Service** confirmation dialog displays.
3. Verify the list of nodes on which to start the service and click **Start Service**.

Starting a Service Using the CLI or REST API

The basic command to start a service on a node is:

```
maprcli node services -nodes <node name> -name <service> -action start
```

For complete reference information, see `node services` on page 1730.

Stopping the Services
Describes how to stop services using either the Control System, the CLI or the REST API.
Stopping the Services Running on the Nodes Using the Control System

To stop the services running on the nodes:

1. Log in to the Control System and click **Nodes** to display the **Nodes** page.

   Note: The **Nodes** page is not available in the Kubernetes version of the Control System.
2. Select the nodes from the list of nodes in the **Nodes** pane and click **Manage Services** to display the **Manage Services** window.
3. Choose the **Stop** radio button for the services you wish to stop on the selected nodes and click **Save**.

Stopping the Services on a Node Using the Control System

To stop one or more services running on a node:

1. Go to the **Summary** tab in the node information page.
2. Select the services to stop in the **Services** pane.
3. Click **Stop Services**.
   The **Stop Services** confirmation dialog displays.
4. Verify the list of services to stop and click **Stop Service**.

Stopping a Service on the Cluster Using the Control System

1. Log in to the Control System and click **Services**.
2. On the non-Kubernetes version, click  associated with the service to stop.
   The **Stop Service** confirmation dialog displays.
3. Verify the list of nodes on which to stop the service and click **Stop Service**.

Stopping the Services Using the CLI or REST API

The basic command to stop a service on a node is:

```
maprcli node services -nodes <node name> -name <service> -action stop
```

For complete reference information, see `node services` on page 1730.

Restarting the Services

Describes how to restart a service using either the Control System, the CLI or the REST API.

When a MapR system is rebooted, the following services are automatically restarted:

- mapr-warden
- mapr-zookeeper
- mapr-loopbacknfs
- mapr-posix-client-*

These services are also automatically restarted if they are shut down externally (as opposed to being shut down explicitly via `service` or `sysctl` commands).

Note: This feature is implemented with `systemd` and is only supported on the following operating systems:

- RHEL 7.0, 7.1
- CentOS 7.0, 7.1
- SLES 12

This feature is not supported on any of the Ubuntu versions that MapR currently supports.

You can restart one or more services using the Control System and the CLI if the services are not disabled. However, if a service is disabled, the service cannot be restarted. To restart a service, make sure the service is enabled. See Enabling and Disabling a Service Using the CLI and REST API on page 1159 for more information.

Restarting the Services Running on the Nodes Using the Control System

To restart the services running on the nodes:

1. Log in to the Control System and click **Nodes** to display the **Nodes** page.

   Note: The **Nodes** page is not available on the Kubernetes version of the Control System.
2. Select the nodes from the list of nodes in the **Nodes** pane and click **Manage Services** to display the **Manage Services** window.
3. Choose the **Restart** radio button for the services you wish to restart on the selected nodes and click **Save**.

last-updated: Apr 11, 2022

Restarting one or more Services on a Node Using the Control System

1. Go to the **Summary** tab in the node information page.
2. Select the services to restart in the **Services** pane.
3. Click **Restart Service(s)**.
   The **Restart Service(s)** confirmation dialog displays.
4. Verify the list of services to restart and click **Restart Service**.

Restarting the Services on the Cluster Using the Control System

1. Log in to the Control System and navigate to **Services**.
2. On the non-Kubernetes version, click  associated with the service to restart.
   The **Restart Service** confirmation dialog displays.
3. Verify the list of nodes on which to restart the service and click **Restart Service**.

Restarting a Service Using the CLI or REST API

The basic command to restart a service on a node is:

```
maprcli node services -nodes <node name> -name <service> -action restart
```

For complete reference information, see `node services` on page 1730.

Related tasks

Enabling and Disabling a Service Using the CLI and REST API on page 1159
Describes how to enable or disable a service using either the REST API or the CLI.

Changing the User for MapR Services from the Command-Line

Explains how use the CLI to change the user that MapR services run as.

All services should run with the same uid/gid on all nodes in the cluster.

Running MapR Services as the Root User

1. Stop Warden.

   ```
   service mapr-warden stop
   ```

2. If ZooKeeper is installed on the node, stop it.

   ```
   service mapr-zookeeper stop
   ```

3. Run the script `$INSTALL_DIR/server/config-mapr-user.sh -u root`
4. If Zookeeper is installed, start it.

   ```
   service mapr-zookeeper start
   ```

5. Start Warden.

   ```
   service mapr-warden start
   ```


last-updated: Apr 11, 2022

Running MapR Services as a Non-Root User

1. Stop Warden.

```
service mapr-warden stop
```

2. If ZooKeeper is installed on the node, stop it.

```
service mapr-zookeeper stop
```

3. If the MAPR_USER does not exist, create the user/group with the same UID and GID.
4. If the MAPR_USER exists, verify that the uid of MAPR_USER is the same as the value on the CLDB node.
5. Run `$INSTALL_DIR/server/config-mapr-user.sh -u MAPR_USER`.
6. If Zookeeper is installed, start it.

```
service mapr-zookeeper start
```

7. Start Warden.

```
service mapr-warden start
```

8. After clearing NODE_ALARM_MAPRUSER_MISMATCH alarms on all nodes, run `$INSTALL_DIR/server/upgrade2mapruser.sh` on all nodes wherever the alarm is raised.

Running Data-Fabric Services as the Root User

1. Stop Warden:

```
service mapr-warden stop
```

2. If ZooKeeper is installed on the node, stop it:

```
service mapr-zookeeper stop
```

3. Run the script `$INSTALL_DIR/server/config-mapr-user.sh -u root`
4. If ZooKeeper is installed, start it:

```
service mapr-zookeeper start
```

5. Start Warden:

```
service mapr-warden start
```

Running Data-Fabric Services as a Non-Root User

1. Stop Warden:

```
service mapr-warden stop
```

2. If ZooKeeper is installed on the node, stop it:

```
service mapr-zookeeper stop
```

3. If the MAPR_USER does not exist, create the user/group with the same UID and GID.
4. If the MAPR_USER exists, verify that the uid of MAPR_USER is the same same as the value on the CLDB node.
5. Run `$INSTALL_DIR/server/config-mapr-user.sh -u MAPR_USER`
6. If Zookeeper is installed, start it:

```
service mapr-zookeeper start
```

7. Start Warden:

```
service mapr-warden start
```

8. After clearing NODE_ALARM_MAPRUSER_MISMATCH alarms on all nodes, run `$INSTALL_DIR/server/upgrade2mapruser.sh` on all nodes wherever the alarm is raised.

Managing Disks

Provides a brief overview of adding and removing disks from the MapR filesystem.

You can add and remove disks in the MapR File System from the MapR Control System or using the `diskadd` and `diskremove` commands. MapR File System groups disks into *storage pools*, usually made up of two or three disks. When adding disks to the MapR File System, it is a good idea to add at least two or three at a time so that MapR can create properly-sized storage pools. Each node in a MapR cluster can support up to 36 storage pools.

To see which disks are used by MapR File System, check the `disktab` file that MapR maintains on each node.

Warning: For instructions on performing disk maintenance on a node, see Performing Maintenance on a Node. If a disk failure alarm is raised (`NODE_ALARM_DISK_FAILURE`), see Handling Disk Failures for instructions.

Refer to the following procedures to manage disks using the Control System and the CLI.

Viewing the List of Disks
Explains how to view the disks on a node using either the Control System or the CLI.
Viewing the List of Disks Using the Control System

To view both the system and filesystem disks:

1. Log in to the Control System and go to the node information page.
2. Go to the **Summary** tab.
   On this page, the **Disks** and **System Disks** panes display the following for each disk:

last-updated: Apr 11, 2022

Column Name	Column Description
Status	The status of the disk. Value can be one of the following: • — indicates disk is active or good. • — indicates disk is on standby. • — indicates disk is offline.
Device	The disk partition(s).
Mnt	Indicates whether (✓) or not the disk is mounted.
MapR File System	(Displayed in the **Disks** pane only) The disks available for MapR File System. A ✓ indicates that the disk was added to MapR File System.
File System	(Displayed in the **System Disks** pane only) The disks for system use.
Allocated	The amount of space allocated, in gigabytes.
Used	The percentage of disk space used.
Model#	The disk model number.
Firmware Version	The disk firmware version.
Storage Label	The label assigned to the disk.
Storage Pool	The ID of the storage pool associated with the disk.

Select the checkbox beside a disk in the **Disks** pane to:

- Add Disk(s) to MapR File System
- Remove Disk(s) from MapR File System

Note: Disks in the **System Disks** pane cannot be selected.

Retrieving the List of Disks Using the CLI or REST API

The basic command to list the disks on a node is:

```
maprcli disk list -host <host>
```

For complete reference information, see `disk list on page 1604`.

Setting Up Disks for MapR

This section describes how to set up disks during the normal installation process. Go to the `disksetup on page 2092` command page for information about other uses of this command.

MapR formats and uses disks for the Lockless Storage Services layer (MapR File System), and records these disks in the `disktab on page 2189` file. In a production environment, or when testing performance, MapR should be configured to use physical hard drives and partitions. In some cases, it is necessary to reinstall the operating system on a node so that the physical hard drives are available for direct use by MapR. Reinstalling the operating system provides an unrestricted opportunity to configure the hard drives. If the installation procedure assigns hard drives to be managed by the Linux Logical Volume Manager(LVM) by default, you should explicitly remove the drives you plan to use with MapRMapR Data Platform from the LVM configuration. It is common to let LVM manage one physical drive containing the operating system partition(s) and to leave the rest unmanaged by LVM for use with MapRMapR Data Platform.

last-updated: Apr 11, 2022

Note: It is not necessary to set up RAID (Redundant Array of Independent Disks) on disks used by MapR File System. MapRMapR Data Platform uses the disksetup script to set up storage pools. In most cases, you should let MapRMapR Data Platform calculate storage pools using the default stripe width of two or three disks. If you anticipate a high volume of random-access I/O, you can use the `-W` option with disksetup to specify larger storage pools of up to 8 disks each.

Notice: For more information on setting up disks, see Drive Configuration.

The following procedures are intended for use on physical clusters or Amazon EC2 instances. On EC2 instances, EBS volumes can be used as MapRMapR Data Platform storage, although performance will be slow.

Note: If you are using MapR on Amazon EMR, you do not have to use this procedure; the disks are set up for you automatically.

Determine if a disk or partition is ready for use by MapR
Explains the procedure to determine whether a disk or partition is ready for use by MapR.

Any disk or partition that passes the following testing procedure can be added to the list of disks and partitions passed to the `disksetup` command.

1. Run the command `sudo lsof <partition>` to determine whether any processes are already using the disk or partition.

   There should be no output when running `sudo fuser <partition>`, indicating there is no process accessing the specific disk or partition.

2. The disk or partition should not be mounted, as checked via the output of the `mount` command. If the disk or partition is mounted, unmount it using the `umount` command.

3. The disk or partition should not have an entry in the `/etc/fstab file`; comment out or delete any such entries.

4. The disk or partition should be accessible to standard Linux tools such as `mkfs`. You should be able to successfully format the partition using a command like `sudo mkfs.ext3 <partition>` as this is similar to the operations that MapR performs during installation. If `mkfs` fails to access and format the partition, then it is highly likely that MapR will encounter the same problem.

Specify disks or partitions for use by MapR
Describes the use of the `disksetup` script to format disks.

The disksetup script is used to format disks for use by the MapR cluster. Create a text file `/tmp/disks.txt` listing the disks and partitions for use by MapR on the node. Each line lists either a single disk or all applicable partitions on a single disk. When listing multiple partitions on a line, separate by spaces. For example:

```
/dev/sdb
/dev/sdc1 /dev/sdc2 /dev/sdc4
/dev/sdd
```

Later, when you run the disksetup script to format the disks, specify the `disks.txt` file. For example:

```
/opt/mapr/server/disksetup -F /tmp/disks.txt
```

Note: The disksetup script removes all data from the specified disks. Make sure you specify the disks correctly, and that any data you wish to keep has been backed up elsewhere.

If you are re-using a node that was used previously in another cluster, be sure to format the disks to remove any traces of data from the old cluster.

Note: Run the disksetup script only after running the configure.sh script.

Evaluate MapR using a flat storage file instead of formatting disks
For evaluation, you can use a flat storage file instead of formatting disks.

When setting up a small cluster for evaluation purposes, if a particular node does not have physical disks or partitions available to dedicate to the cluster, you can use a flat file on an existing disk partition as the node's storage. Create at least a 16GB file, and include a path to the file in the disk list file for the `disksetup` script.

The following example creates a 20 GB flat file (`bs=1G` specifies 1 gigabyte blocks, multiplied by `count=20`) at `/root/storagefile`:

```
dd if=/dev/zero of=/root/storagefile bs=1G count=20
```

Add the created flat file to the disk list file `/tmp/disks.txt` to be used by `disksetup`:

```
/root/storagefile
```

Adding Disks to MapR File System
Describes how to add disks using either the Control System or the CLI.

You can add disks to MapR File System using the Control System and the CLI. Before adding the disks to MapR File System, add the physical disks to the node or nodes according to the correct hardware procedure.

- If you are removing and replacing failed disks, you must install the replacements, then re-add the replacement disks to MapR File System, along with the other disks that were in the same storage pool(s) as the failed disks. See Handling Disk Failures for more details.
- If you are removing disks but not replacing them, you can just re-add the other disks that were in the same storage pool(s) as the failed disks.

Note: Disks must be added on CLDB nodes one node at a time when Warden and MapR services are running.

Attention: Disable write caching on all MapR disks if the disks are not battery backed.

Adding Disks Using the Control System

Complete the following steps to add disks of type MapR File System using the Control System:

1. Log in to the Control System and go to the **Summary** tab in the node information page.

   Note: The **Nodes** page is not available on the Kubernetes version of the Control System.

2. Select the disks not yet added to MapR File System in the **Disks** pane and click **Add Disks to File System**.

   The **Add Disks to File System** confirmation dialog displays.

   Note: You cannot select disks of type *System* to add.

3. Review the list and click **Add Disk**.

   The disks are automatically formatted and properly-sized storage pools are automatically allocated.

last-updated: Apr 11, 2022

Adding Disks Using the CLI or REST API

The basic command to add disks to a node is:

```
maprcli disk add -disks <disk names> -host <host>
```

Note: This step reformats the disks. Any data on these disks will be lost.

For complete reference information, see `disk add` on page 1602.

Removing Disks from the File System

Explains how to remove disks using either the Control System or the CLI.

When you remove a disk from the MapR File System, other disks in the storage pool are also removed automatically from the MapR File System and are no longer in use (they are available but off-line). Their disk storage goes to 0%, and they are eligible to be added again to the MapR File System to build a new storage pool. You can either replace the disk and re-add it along with the other disks that were in the storage pool, or just re-add the other disks if you do not plan to replace the disk you removed. See Adding Disks to MapR File System on page 837 for more information.

Warning: Removing a disk in the storage pool that contains Container ID 1 shuts down CLDB, triggering a CLDB failover. Container ID 1 contains CLDB data for the primary CLDB. From the command-line, run the `maprcli disk remove` command without the `-force 1` option first and examine the warning messages to make sure you are not removing the disk with Container ID 1. To safely remove such a disk, perform a CLDB Failover on page 1500 to make one of the other CLDB nodes the primary CLDB, then remove the disk as normal with addition of the `-force 1` option.

Before removing or replacing disks, make sure the Replication Alarm (VOLUME_ALARM_DATA_UNDER_REPLICATED), Data Alarm (VOLUME_ALARM_DATA_UNAVAILABLE), Warm-Tier Data Node Down (VOLUME_ALARM_DEGRADED_EC_STRIPES), and EC Degraded Alarm (VOLUME_ALARM_CRITICALLY_DEGRADED_EC_STRIPES) are not raised. These alarms can indicate potential or actual data loss. If either alarm is raised, you might be able to repair the problem using the `/opt/mapr/server/fsck` utility before removing or replacing disks.

Note: Using the `/opt/mapr/server/fsck` utility with the `-r` flag to repair a MapR File System risks data loss. Call MapR support before using `/opt/mapr/server/fsck -r`.

Removing Disks from MapR File System Using the Control System

Complete the following steps to remove disks using the Control System:

1. Log in to the Control System and go to the **Summary** tab in the node information page.
2. Select the disks to remove in the **Disks** pane and click **Remove Disk(s) from File System**.
   The **Remove Disk(s) from File System** confirmation dialog displays.

   Warning: One or more disks you selected may have unreplicated data on it and this action will forcefully remove the disks.
3. Review the list and click **Remove Disk**.
   Wait several minutes while the removal process completes. After you remove the disks, any other disks in the same storage pools are taken offline and marked as *available* (not in use by MapR).
4. Remove the physical disks from the node or nodes according to the correct hardware procedure.

last-updated: Apr 11, 2022

5. From a command line terminal, remove the failed disk log file from the `/opt/mapr/logs` directory.
   These log files are typically named like this:

   ```
   diskname.failed.info
   ```

Removing Disks from MapR File System Using the CLI or REST API

1. On the node, determine which disk to remove/replace by examining **Disk** entries in the `/opt/mapr/logs/faileddisk.log` file.
2. Run the following command, substituting the hostname or IP address for `<host>` and a list of disks for `<disks>`

   ```
   maprcli disk remove -disks <disk names> -host <host>
   ```

   Note: This command does not remove a disk containing unreplicated data unless forced.

   For complete reference information, see `disk remove` on page 1608.
3. Examine the screen output in response to the command you ran in step 2.
   For example:

   ```
   maprcli disk remove -host `hostname -f` -disks /dev/sdd
   message   host   disk
   removed.  host1  /dev/sdd
   removed.  host1  /dev/sde
   removed.  host1  /dev/sdf
   ```

   Make a note of the *additional* disks removed when the disk is removed. For example, the disks `/dev/sde` and `/dev/sdf` are part of the same storage pool and therefore removed along with the disk (`/dev/sdd`).
4. Confirm that the removed disks do not appear in the `disktab` file.
5. Remove the disk log file from the `/opt/mapr/logs` directory.
   For failed disks, these log files are typically named in the pattern *`diskname`*`.failed.info`.

When you replace a failed disk, add it back to the MapR File System along with the other disks from the same storage pool that were previously removed. Adding only the replacement disk to the MapR File System, results in a non-optimal storage pool layout, which can lead to degraded performance.

Once you add the disks to the MapR File System, the cluster automatically allocates properly-sized storage pools. For example, if you add ten disks, MapR allocates two storage pools of three disks each and two storage pools of two disks each.

Determining the Amount of Free Disk From the Command-Line
Lists the command to display the amount of free disk space.

To determine the amount of used and available disk space on the filesystem, run `df -h`. When running this command, if:

last-updated: Apr 11, 2022

- The given path points to the mount point, the output will display used and available disk space for the entire cluster. For example:

```
[root@atsqa6c69 ~]df -h /mapr/clus.posix/
Filesystem            Size  Used  Avail  Use%  Mounted on
posix-client-basic    4.4T  9.7G  4.4T   1%    /mapr
```

- The given path points to a volume with no (hard) quota, the output will display used and available disk space for the entire cluster. For example:

```
[root@atsqa6c69 ~]df -h /mapr/clus.posix/vol3
Filesystem          Size  Used  Avail  Use%  Mounted on
posix-client-basic  4.4T  9.7G  4.4T   1%    /mapr
```

- The given path points to a volume with (hard) quota set, the output will display the used and available disk space for the specific volume based on the allocated quota. For example:

```
[root@atsqa6c69 ~]df -h /mapr/clus.posix/vol2/
Filesystem          Size  Used  Avail  Use%  Mounted on
posix-client-basic  5.0G  2.5G  2.6G   49%   /mapr
```

Tolerating Slow Disks
Explains how to tune disk response timeouts.

The parameter `mfs.io.disk.timeout` in `mfs.conf` determines how long MapR waits for a disk to respond before assuming it has failed. If healthy disks are too slow, and are erroneously marked as failed, you can increase the value of this parameter.

Formatting Disks on a Node From the Command-line
Provides an overview of the disksetup script to format disks from the command line.

The disksetup script is used to format disks for use by the MapR cluster. Create a text file `/tmp/disks.txt` listing the disks and partitions for use by MapR on the node. Each line lists either a single disk or all applicable partitions on a single disk. When listing multiple partitions on a line, separate by spaces. For example:

```
/dev/sdb
/dev/sdc1 /dev/sdc2 /dev/sdc4
/dev/sdd
```

Later, when you run disksetup to format the disks, specify the `disks.txt` file. For example:

```
/opt/mapr/server/disksetup -F /tmp/disks.txt
```

Note: The disksetup script removes all data from the specified disks. Make sure you specify the disks correctly, and that any data you wish to keep has been backed up elsewhere.

If you are re-using a node that was used previously in another cluster, be sure to format the disks to remove any traces of data from the old cluster.

Warning: Run disksetup only after running configure.sh.

Handling Disk Failures
Explains how to handle disk failures.

When a disk fails, MapR raises the node-level alarm `NODE_ALARM_DISK_FAILURE` on the node with the failed disk (or disks). At the same time, other disks in the same storage pool as the failed disk are taken

last-updated: Apr 11, 2022

offline. You can look at the Control System **Overview** page to view the health of the nodes and a list of alarms.

When you see a disk failure alarm, examine the log file at `/opt/mapr/logs/faileddisk.log` and check the **Failure Reason** field.

Examining the Cause of Failure
Names the log file that contains the cause of disk failures.

In the `faileddisk.log` file, you will see information on the cause of failure. In the following sample log output, the failure reason is *I/O error*. Notice that the log file also provides instructions for removing disks and adding them back to the MapR File System.

```
############################ Disk Failure Report
############################
Disk                :    /dev/sdd
Vendor              :    [vendor]
Model Number        :    [model]
Serial Number       :    [serial]
Firmware Revision   :    [firmware]
Size                :    [total]
Failure Reason      :    I/O error
Time of Failure     :    Fri Jan 31 12:48:00 GMT 2014
Resolution          :   Please refer to MapR's online documentation at
https://docs.datafabric.hpe.com on how to handle
        disk failures.
In summary, run the following steps:
a. If this appears to be a software failure, go to step
b. Otherwise, physically remove the disk /dev/sdd. Optionally, replace it
with a new disk.
b. Run the command "maprcli disk remove -host 127.0.0.1 -disks /dev/sdd" to
remove
        /dev/sdd from MapR-FS.
c. In addition to /dev/sdd, the above command removes all the disks that
belong to the same
        storage pool, from MapR-FS.      Note down the names of all removed
disks.
d. Add all the above removed disks (exclude /dev/sdd) and the new disk to
MapR-FS by
   running the command:
   "maprcli disk add -host 127.0.0.1 -disks <comma separated list of
      disks>"
   For example, If /dev/sdx is the new replaced disk, and /dev/
sdy, /dev/sdz were removed in
        step c), the command would be:
   "maprcli disk add -host 127.0.0.1 -disks /dev/sdx,/dev/sdy,/dev/
sdz"      If there is no new disk, the command would just
be:                   "maprcli disk add -host 127.0.0.1 -disks
    /dev/sdy,/dev/sdz"
```

Recovering from Disk Failure
Lists the disk errors and their resolution.

Most software failures can be remedied by running the fsck utility, which scans the storage pool to which the disk belongs and reports errors. For hardware failures, remove the failed disk and replace it according to the procedure in Removing and Replacing Disks.

The following are the types of failures and the recommended courses of action:

I/OTimeOut Error	*Failure Reason*: The default value for `mfs.disk.io.timeout` parameter is 60 seconds. The time to declare an IO as stuck is 3 times the value of this parameter (3 x `mfs.disk.io.timeout`). The

	disk will be taken offline even if a single IO has not completed. *Action*: **1.** Check if the disks are good and still reliable. **2.** If disks are good, increase the value of the `mfs.io.disk.timeout` parameter in the `/opt/mapr/conf/mfs.conf` file. Otherwise, replace the disks.
No Such Device	*Failure Reason*: The `$INSTALL_DIR/conf/disktab` file contains `"/MissingDisk"` or references a disk path not found in `/proc/partitions` file. *Action*: Run `mrdisk <device path>` to determine whether a disk is formatted for MapR File System. Also, check the device paths in `$INSTALL_DIR/conf/disktab` file. The `disktab` file contains the disk path and disk GUID that is used to load the disks in the MapR File System. If the disk paths have been renamed, fix them or run `disksetup -X` command to regenerate the disktab from `/proc/partitions`. Alternatively, restart the MapR File System to resolve disk name changes. If the problem still persists, contact MapR support.
ENODEV: MissingDisk# Error: disktab file contains a /MissingDisk# entry	*Failure Reason*: A disk corresponding to a GUID is missing and the corresponding disk path in the `disktab` file belongs to another disk. When an attempt is made to automatically fix the `disktab` file, this entry is replaced with `/MissingDisk# path`. *Action*: If a disk corresponding to a GUID is permanently lost, remove the line corresponding to it in the `disktab` file. Alternatively, run `maprcli disk remove _MissingDisk#` command, where # corresponds to the disk number, and restart the MapR File System.
EIO Error	*Failure Reason*: I/O error. This could be due to a bad block or disk. The system will offline the SP after one final attempt to complete the IO. *Action*: Check `/var/log/messages` for errors from the disk drivers.

CRC Error	*Failure Reason*: This could be due to a bad block or bit flip on the disk. The SP will be taken offline immediately.
	Action: Run `fsck -n <sp> -d` to perform a CRC (Cyclic Redundancy Check) on the data blocks in the storage pool, then bring it back online.
	To load all the SPs to the list of SPs, run:
	`mrconfig disk load or mrconfig sp load`
	To bring back all SPs online, run:
	`mrconfig sp refresh`

last-updated: Apr 11, 2022

To bring specific SPs back online, run:

```
mrconfig sp online <sp path>
```

SlowDisk Error

Failure Reason: The default value for the `mfs.disk.io.timeout` parameter is 60 seconds. The time to declare an IO as slow is equal to the value of this parameter (1 x `mfs.disk.io.timeout`). Thirty or more slow IO completions in a short span of time (5 seconds) on the same disk is recorded as a slow event. The SP will be taken offline if 3 such events are recorded within an hour.

Note: After an hour, MapR filesystem will reset tracking (to 0).

Action:

1. Check if the disks are good and still reliable.
2. If disks are good, increase the value of the `mfs.io.disk.timeout` parameter in the `/opt/mapr/conf/mfs.conf` file. Otherwise, replace the disks.

GUID of disk mismatches with the one in `$INSTALL_DIR/conf/disktab`

Failure Reason: Possible that disk names have changed.

Action: After a node restart, the operating system can reassign the drive labels (for example, `/sda`), resulting in drive labels no longer matching the entries in the `disktab` file. The `disktab` file contains the disk path and disk GUID that is used to load the disks in the MapR File System. Run `$INSTALL_DIR/server/disksetup -X` to update the `disktab` file by looking up the disks in `/proc/partitions` and make the disk paths match the GUIDs.

Unknown Error

Failure Reason: Any reason

Action: Contact MapR support.

Addressing Data Alarms

Lists all the data alarms and their mitigation.

When a disk fails, data on that disk becomes unavailable. As a result, you will probably see one of these two data alarms along with a **Disk Failure** alarm:

- **Data Unavailable** (`VOLUME_ALARM_DATA_UNAVAILABLE`) - if there was only one copy of data and it was on the failed disk; or if data was replicated more than once, but all disks with that data failed
- **Data Under Replicated** (`VOLUME_ALARM_DATA_UNDER_REPLICATED`) - if data on the failed disk is replicated elsewhere, but the minimum replication factor is not met as a result of the failed disk

If you see a **Data Unavailable** volume alarm in the cluster, follow these steps to run the /opt/mapr/server/fsck utility on all the offline storage pools. On each node in the cluster that has raised a disk failure alarm:

1. Run the following command to identify which storage pools are offline:

   ```
   [user@host] /opt/mapr/server/mrconfig sp list | grep Offline
   ```


last-updated: Apr 11, 2022

2. For each storage pool reported by the previous command, run the following command, where <sp> specifies the name of an offline storage pool:

   ```
   [user@host] /opt/mapr/server/fsck -n <sp> -r
   ```

   When you run fsck with the `-r` option, it identifies corrupt blocks and removes them. If there are no corrupt blocks, fsck clears the error condition so you can bring the storage pool back online.

   Note: Using the /opt/mapr/server/fsck utility with the `-r` flag to repair a filesystem risks data loss. Call MapR support before using `/opt/mapr/server/fsck -r`.

3. Verify that all **Data Unavailable** volume alarms are cleared. If **Data Unavailable** volume alarms persist, contact MapR support or post on answers.mapr.com.

If there are any **Data Under Replicated** volume alarms in the cluster, MapR can repair the problem by automatically replicating data and putting it on another disk. After you allow a reasonable amount of time for re-replication, verify that the under-replication alarms are cleared.

Using the /opt/mapr/server/fsck utility with the `-r` option produces different results depending on the scenario. The fsck utility does not interpret the scenario nor does it have a safe mode.

- If a disk is offline because of an imbalanced b-tree, using `fsck -r` may result in data loss from bad containers and data loss if additional replicas are unavailable.
- If a disk is offline because of an I/O error, using `fsck -r` produces indeterminate results. A disk that is throwing I/O errors is questionable in terms of data content and reliability. For example, an operation that completed on the disk but was never returned may have partial data remaining on the disk. Using `fsck -r` retains any partial data.
- If a disk is offline because of a slow I/O, using `fsck -r` does not produce data loss.

The most conservative usage of `fsck -r` is to run `fsck` without the `-r` option (verification mode) and check the output. If the output is ok, then run `fsck` with the `-r` option.

Note: Disk Failure node alarms that persist require disk replacement. If **Data Under Replicated** volume alarms persist, contact MapR support or post on answers.mapr.com.

Removing and Replacing Disks

If a disk fails due to a hardware problem, you will need to remove the disk. You can replace it, and then add that disk back to MapR File System along with the other disks that were automatically removed at the same time.

Refer to the following for information on how to remove and replace disks using the MapR command-line interface and the MapR Control System:

- Removing Disks from the File System on page 838
- Adding Disks to MapR File System on page 837

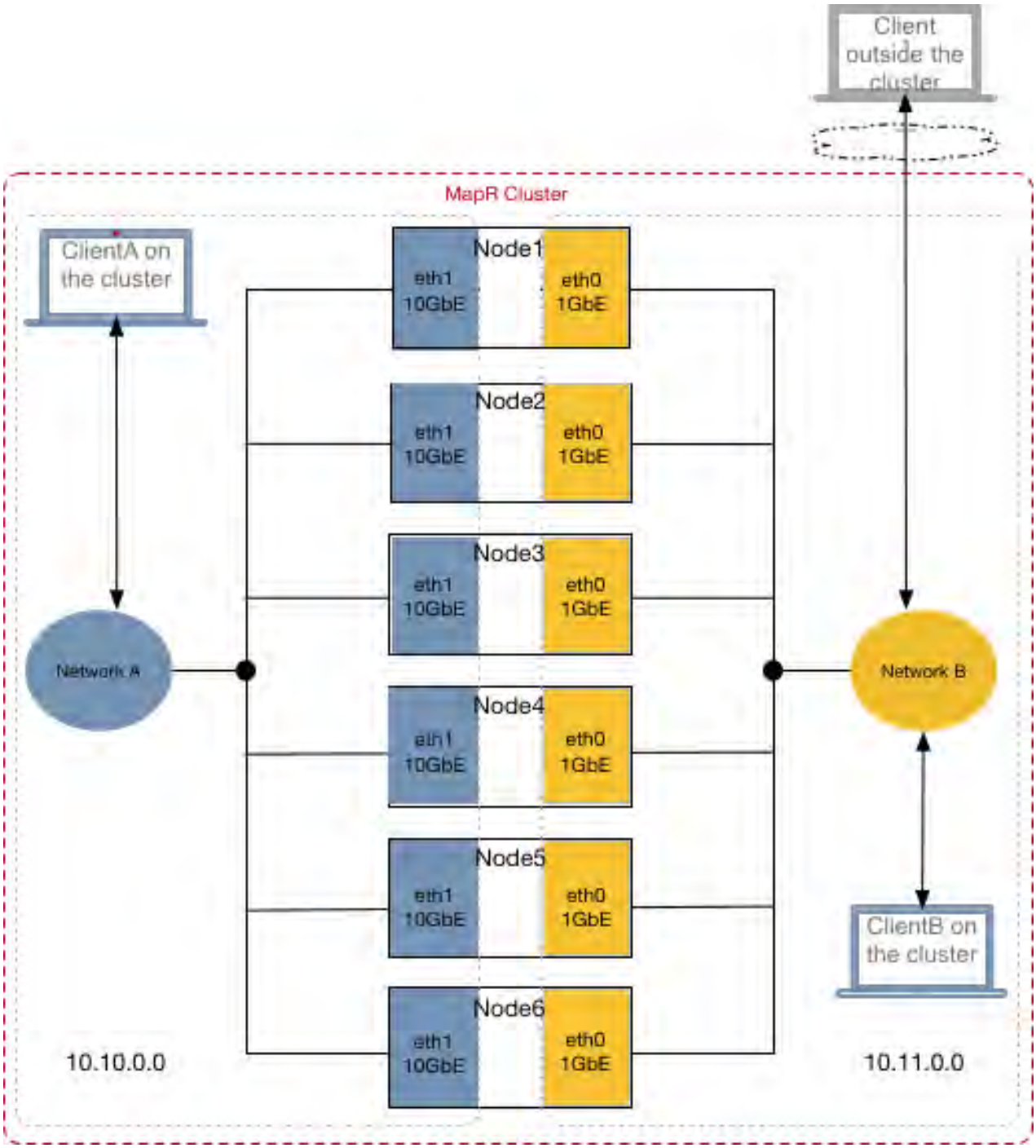
Designating NICs for MapR
Explains how to assign IP address blocks for MapR.

By default, MapR File System instances and the CLDB nodes advertise all the available IP addresses, and MapR automatically uses all available network interface cards (NICs) on each node for all communication. For nodes that have multiple NICs, MapR supports segregation of NICs such that certain IPs can be used for clients/communication within the cluster and certain IPs can be used for clients/communication from/to outside the cluster. Also, NICs can be segregated for specific (high-performance and/or low-performance) clients within the cluster.

For example, if you use multiple NICs of mixed speeds (such as 1GbE and 10GbE) on each node, you might want to separate them to two different networks depending on the Ethernet card speeds. You can

assign IP addresses in the same network to the NICs of 1GbE and assign IP addresses in another network to the NICs of 10GbE. That way, you can use the faster NICs for communication within the cluster or for certain high-performance clients (for example, FUSE-based POSIX client) and the slower NICs for external communication or for low-performance clients/jobs.

To illustrate this arrangement, the following diagram shows six nodes on a MapR cluster, each with a 1GbE NIC (eth0) and a 10GbE NIC (eth1). All the 1GbE NICs are networked together and connected to Network B. Likewise, all the 10GbE NICs are networked together (as part of a subnet written as 10.10.10.0/24 in CIDR notation) and connected to Network A, where peak performance is required. ClientA, which is within the cluster, communicates with cluster nodes over Network A and clients outside the cluster communicate with cluster nodes over Network B. The illustration also shows ClientB, which is a low-performance client inside the cluster, communicating with cluster nodes over Network B.



last-updated: Apr 11, 2022

MapR provides two environment variables, MAPR_SUBNETS and MAPR_EXTERNAL, which can be used to segregate NICs for internal and external clients or to segregate NICs for high-performance and low-performance clients.

About MAPR_SUBNETS Environment Variable

The MAPR_SUBNETS environment variable can be used to restrict MapR to a subset of NICs. If MAPR_SUBNETS is not set, all IPs are available for all communication. When MAPR_SUBNETS is set on:

- MapR File System nodes, MapR File System registers these IP addresses with CLDB as internal IP addresses on which MapR File System nodes can be reached.
- CLDB nodes, CLDB advertises the IP address to clients on the cluster.

You can set the MAPR_SUBNETS environment variable in the /opt/mapr/conf/env_override.sh file on all the nodes. On the cluster nodes, the value for this environment variable is a comma-separated list of subnet masks. For example:

```
export MAPR_SUBNETS=10.10.15.0/24,10.10.16.0/24
```

You can specify up to four NICs in the MAPR_SUBNETS environment variable. If your system has more than four NICs, MapR advertises the first four it finds or if MAPR_SUBNETS environment variable is set, MapR restricts the networks/IPs that are advertised based on the subnets specified therein.

The MAPR_SUBNETS environment variable can be set on the client if there is a NAT between the server and client. On the client, the value for this environment variable is an IP address of the client. For example:

```
export MAPR_SUBNETS=10.11.12.13/32
```

When specifying the IP address in the MAPR_SUBNETS environment variable on the client, use /32 to specify a single IP address.

For more information on the MAPR_SUBNETS environment variable, see Environment Variables on page 2289.

About MAPR_EXTERNAL Environment Variable

If all the IP addresses on the servers are public and can be accessed from an external system, the MAPR_EXTERNAL environment variable need not be set. However, if your cluster nodes have private IP addresses, to allow clients outside the cluster to reach the cluster nodes (such as when MapR is installed on the cloud or Docker container), specify the public IP addresses in the MAPR_EXTERNAL environment variable.

On the cluster nodes, you can set this variable in the /opt/mapr/conf/env_override.sh file. When MAPR_EXTERNAL is set on:

- MapR File System nodes, MapR File System registers these IP addresses with CLDB as the IP addresses on which external clients can reach MapR File System nodes. Communication between MapR File System nodes on the cluster still occurs over the internal IP addresses.
- CLDB nodes, CLDB advertises these IPs addresses to clients outside the cluster or data center.
- MAST Gateway nodes, the gateway registers these IP addresses with CLDB as the IP addresses on which external clients can reach the MAST Gateway.

Note: Do not set the MAPR_EXTERNAL environment variable on the client(s).

last-updated: Apr 11, 2022

The value for this environment variable is a comma-separated list of IP addresses; you cannot specify the hostname as value. For example:

```
ip1,ip2,ip3;
```

For example, you can specify the IP addresses of the 1GbE NICs (shown in the previous illustration) as the value for this environment variable, to allow external or low-performance clients to communicate with the cluster nodes.

```
export MAPR_EXTERNAL=10.11.0.0
```

For more information on the MAPR_EXTERNAL environment variable, see Environment Variables on page 2289.

About IP Addresses for ZooKeeper Nodes

You can specify the IP addresses of ZooKeeper nodes by running the `configure.sh` utility with both the `-Z` and `-EZ` options during cluster configuration and list the:

- Internal IP addresses with the `-Z` option
- External IP addresses with the `-EZ` option

When you specify the IP addresses using the `-Z` and `-EZ` options, these IP addresses are registered with CLDB and included in the `cldb.conf` file. In the `cldb.conf` file, the internal IP addresses set using the `-Z` option are the values for the `cldb.zookeeper.servers` parameter. The external IP addresses set using the `-EZ` option are the values for the `cldb.external.zookeeper.servers` parameter.

Note: You do not need to run the `configure.sh` command with the `-EZ` option during client configuration.

For more information, see `configure.sh` on page 2053.

If all the ZooKeepers have different IP addresses, port forwarding is not required and, optionally, you can specify the same port with all the IP addresses. However, in some cases, such as when there is a single external IP address being used by multiple ZooKeepers (as in a Docker container), you can specify ports for ZooKeepers when you run the `configure.sh` utility with the `-Z` and `-EZ` options. For more information, see Specifying Ports on page 852.

About Internal and External Clients

Clients communicating with CLDB using internal IP address (of CLDB) are considered internal clients (or clients within the cluster) and clients communicating with CLDB using external IP address (of CLDB) are considered external clients (or clients outside the cluster).

To configure a client as an internal or high-performance client, include CLDB's internal IP address in the `mapr-clusters.conf` file on the client host. Similarly, to configure a client as an external or low-performance client, include CLDB's external IP address in the `mapr-clusters.conf` file on the client host.

The `mapr-clusters.conf` file on the client host should not contain both internal and external IP addresses of the server on a cluster. The `mapr-clusters.conf` file can contain internal and external IP addresses only when the entries in the file on the client host are for multiple clusters. For example, suppose a client, which is an internal client on one cluster and external client on another cluster. The `mapr-clusters.conf` file on the client host can contain CLDB's internal IP address for the cluster on which the client is considered an internal client and CLDB's external IP address for the cluster on which the client is considered an external client.

Note: The `mapr-clusters.conf` file on the cluster nodes should not contain any external IP address.

Limitations

If:

- Both `MAPR_SUBNETS` and `MAPR_EXTERNAL` environment variables are set, segregation of NICs for internal and external communication is possible. Internal communication happens over the IP addresses listed in the `MAPR_SUBNETS` environment variable and external communication happens over the IP addresses listed in the `MAPR_EXTERNAL` environment variable.
- Only `MAPR_SUBNETS` environment variable is set, MapR File System registers the IP addresses (in the `MAPR_SUBNETS` environment variable) with CLDB as internal IPs.

Note: Set both the environment variables in the `/opt/mapr/conf/env_override.sh` file to segregate internal or high performance clients, and external or low-performance clients.

You can specify up to 4 IP addresses in the `MAPR_SUBNETS` environment variable, and 4 IP addresses in the `MAPR_EXTERNAL` environment variable.

Note: You must configure ZooKeeper with an IP address that is reachable by both internal and external clients.

Summary

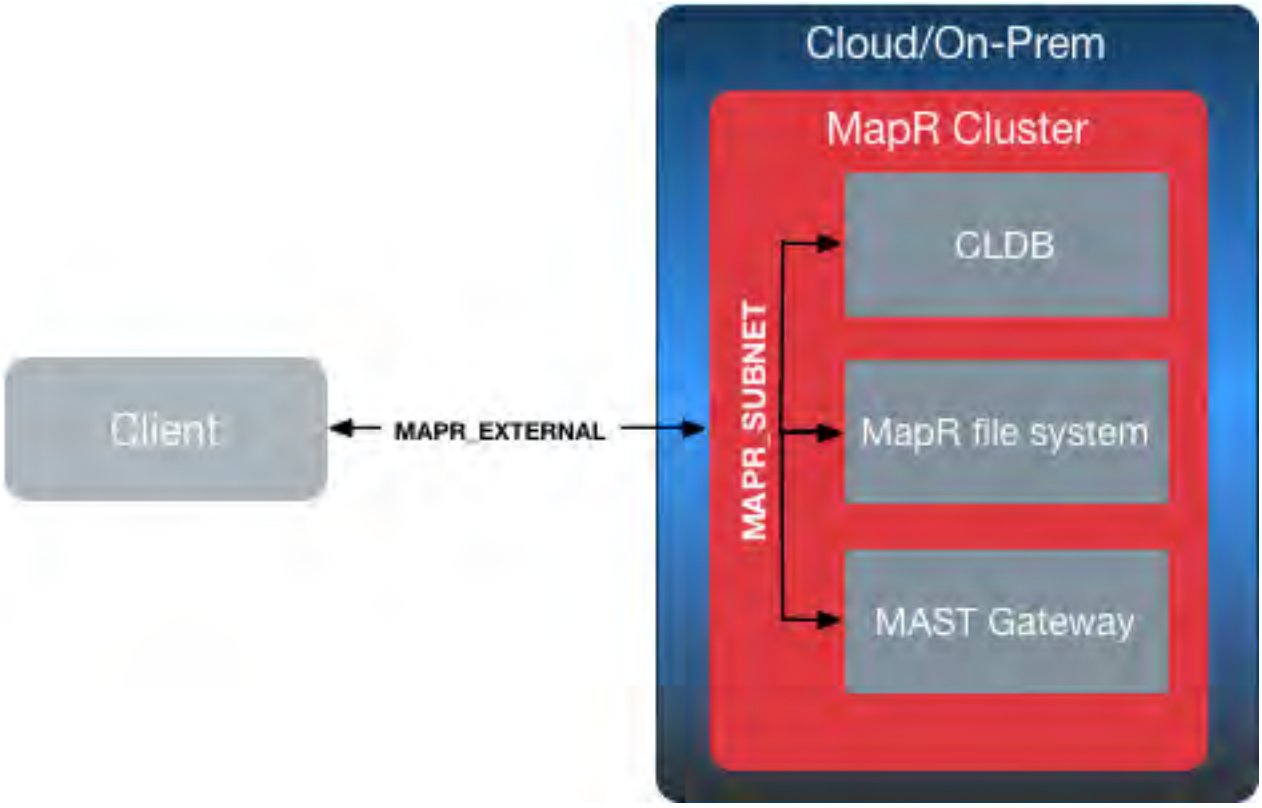
The following table describes the environment variables to set for the various services that use non-default ports and that support public IP address(es) for communication with external clients and remote clusters:

Service	Environment variable to set...	
	Public IP Address for External Clients/ Remote Clusters	**Non-default Port**
CLDB	`MAPR_EXTERNAL`	`CLDB_EXTERNAL_RPC_PORT`
MapR File System	`MAPR_EXTERNAL`	`MAPR_EXTERNAL`
MAST Gateway	`MAPR_EXTERNAL`	`MASTGATEWAY_EXTERNAL_RPC_PORT`

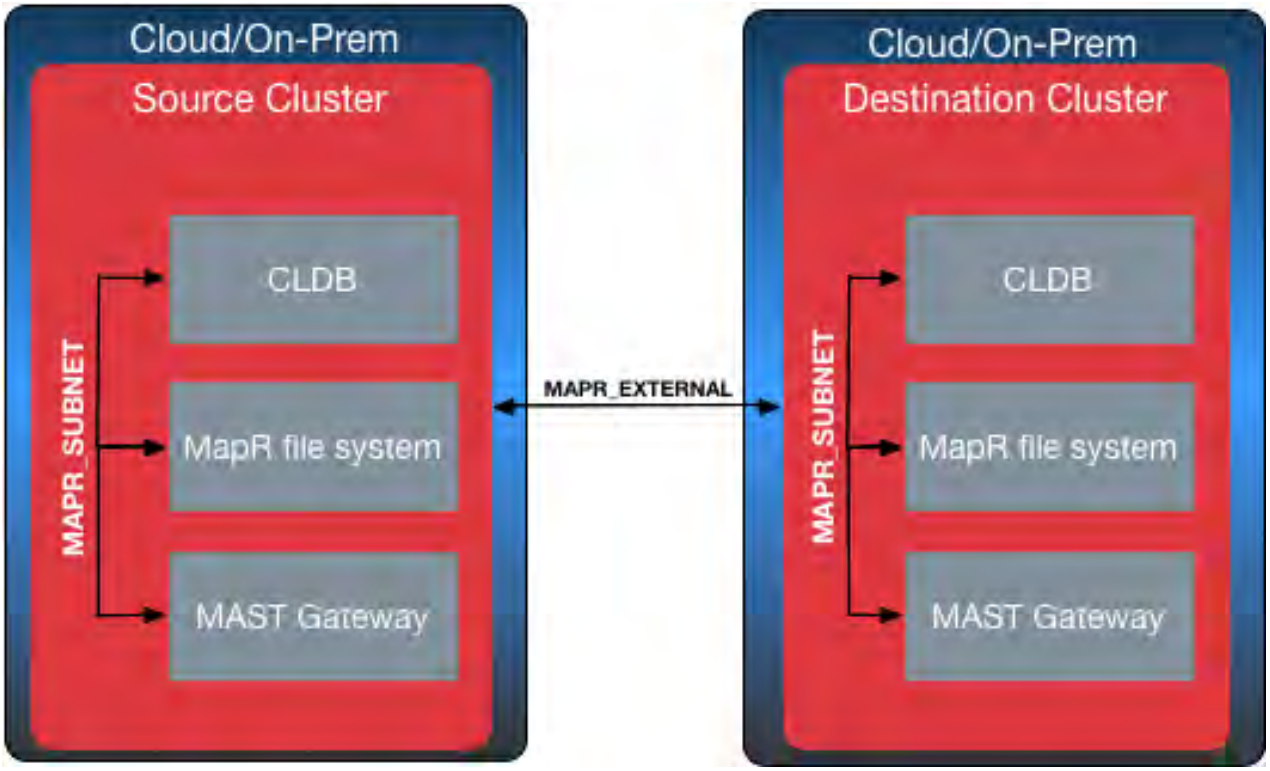
The following illustration shows the client communicating with CLDB, MapR filesystem, and MAST Gateway using the IP address(es) defined in the `MAPR_EXTERNAL` environment variable because all the IP addresses on the servers are not public and accessible outside the cluster. All communication between CLDB, MapR File System, and MAST Gateway on the same cluster happen over the IP address specified in the `MAPR_SUBNETS` environment variable because communication between the services and clients on the cluster is restricted to a subset of the available NICs.

When the client connects to the MapR filesystem from outside the cluster, the client uses either the default port (5660) or the port specified for the MapR filesystem in the `MAPR_EXTERNAL` environment variable. When communicating with CLDB, if the `CLDB_EXTERNAL_RPC_PORT` environment variable is set, the client communicates with CLDB over the port specified in this environment variable. Similarly for MAST Gateway, if the `MASTGATEWAY_EXTERNAL_RPC_PORT` environment variable is set, the client communicates with MAST Gateway over the port specified in this environment variable. For both CLDB and MAST Gateway, if the ports are not set in the `CLDB_EXTERNAL_RPC_PORT` and `MASTGATEWAY_EXTERNAL_RPC_PORT` environment variables respectively, the client communicates over the default port.

last-updated: Apr 11, 2022



The following illustration shows that during mirroring and other cross-cluster activities, the services on the destination cluster communicate with the services on the source cluster using the IP address defined in the `MAPR_EXTERNAL` environment variable. Similar to the external client, the services and clients in the remote destination cluster communicate with the services in the source cluster over the default ports or the port specified in the environment variable for the service.



last-updated: Apr 11, 2022

Setting Environment Variables for NIC Segregation
Describes how to set environment variables to segregate NICs.

Use the `MAPR_SUBNETS` and `MAPR_EXTERNAL` environment variables to segregate NICs for internal and external clients, or to segregate NICs for high-performance and low-performance clients.

Setting the MAPR_SUBNETS Environment Variable

To specify the internal IP addresses of CLDB and MapR File System nodes:

1. Stop warden on all the nodes on the cluster.
2. Set the IP address range to use in the `MAPR_SUBNETS` environment variable in the `/opt/mapr/conf/env_override.sh` file. For more information about this file, see About env_override.sh on page 2290.

   For example:

   ```
   export MAPR_SUBNETS=10.10.0.0/24
   ```

   To specify multiple subnets for MapRMapR Data Platform, use comma to separate the IP addresses.

   Before specifying the IP address, make sure the client and cluster nodes can communicate using that IP address. That is, ensure that the client can send packets and that they can be routed to all the interfaces of the cluster nodes, and the cluster nodes all have a route that reaches back to the client IP address.

   Note: For standalone programs (not using the `mapr` classpath), which do not pick up the settings in the `/opt/mapr/conf/env.sh` file, set `MAPR_SUBNETS` explicitly before the start of program.
3. Perform a rolling restart of warden on all the nodes for the changes to take effect.
4. Add CLDB's internal IP address (or IP address specified in the `MAPR_SUBNETS` environment variable on the CLDB host) to the `mapr-clusters.conf` file on the (internal or high-performance) client host(s).

   The `mapr-clusters.conf` file specifies IP addresses, on which the CLDB nodes (for one or more clusters) can be reached. For more information, see mapr-clusters.conf on page 2200.

When you restrict MapRMapR Data Platform to certain subnets, MapRMapR Data Platform clients have full access to the MapRMapR Data Platform cluster on the designated subnets.
Setting the MAPR_EXTERNAL Environment Variable

To specify the external IP addresses of CLDB, MapR File System, and/or MAST Gateway nodes:

1. Stop warden on all the nodes on the cluster.
2. Set the IP addresses to use for external communication or for low-performance clients in the `MAPR_EXTERNAL` environment variable in the `/opt/mapr/conf/env_override.sh` file. For more information about this file, see About env_override.sh on page 2290.

   For example:

   ```
   export MAPR_EXTERNAL=10.11.0.0;
   ```

   To specify multiple subnets for MapRMapR Data Platform, use a comma to separate the IP addresses.
3. Perform a rolling restart of warden on all the nodes for the changes to take effect.
4. Add the following in the mapr-clusters.conf file on the (external or low-performance) client host(s):

- CLDB's external IP addresses, which is the IP addresses specified in the `MAPR_EXTERNAL` environment variable on the CLDB hosts.
- CLDB's external port, which is the value of the `CLDB_EXTERNAL_RPC_PORT` environment variable if this is set on the CLDB hosts. See Specifying Ports for CLDB for more information.

The mapr-clusters.conf file contains the IP addresses, on which the CLDB nodes (for one or more clusters) can be reached. For more information, see mapr-clusters.conf on page 2200.

Examples

Suppose the value for the `MAPR_EXTERNAL` environment variable on MapR File System node is the following:

```
10.10.103.80,10.10.30.205
```

External clients can connect to MapR File System on IPs 10.10.103.80, 10.10.30.205 and the ports on which the MapR File System is reachable are the default ports. If MapR File System is running 2 instances, then:

- Instance 1 is reachable on 10.10.103.80:<5660>, 10.10.30.205:<5660>
- Instance 2 is reachable on 10.10.103.80:<5661>, 10.10.30.205:<5661>

If MapR File System is running 3 instances:

- Instance 1 is reachable on 10.10.103.80:<5660>, 10.10.30.205:<5660>
- Instance 2 is reachable on 10.10.103.80:<5661>, 10.10.30.205:<5661>
- Instance 3 is reachable on 10.10.103.80:<5662>, 10.10.30.205:<5662>

Suppose the value for the `MAPR_EXTERNAL` environment variable on a MAST Gateway node is the following:

```
10.20.30.100
```

External clients can connect to MAST Gateway on IP 10.20.30.100 and the port on which MAST Gateway is reachable is the default port (8660). If MapR File System is also running on this node, then both MapR File System and MAST Gateway are reachable on the IP 10.20.30.100 and the ports on which they are reachable are the default ports.

Specifying External IP Address of ZooKeeper Nodes

To specify the external IP addresses of ZooKeeper nodes, during cluster configuration:

- Run the configure.sh utility as follows:

```
/opt/mapr/server/configure.sh -C <hostname|IP>[,<hostname|IP>,..] -Z
<IP>[,<IP>..] \
-EZ <IP>[:<port>][,<IP>[:<port>]..] [-F <disk_list_file>] [-N
<cluster_name>]
```

In the preceding command:

- When each ZooKeeper node has a different external IP address, use the `-EZ` option to specify the IP address of each ZooKeeper node, and optionally the port as well (separated by a colon); the IP address can be different while the port number must be the same for every node.

last-updated: Apr 11, 2022

- When there are multiple ZooKeeper nodes listening on the same external IP (such as in a Docker container), use the `-EZ` option to specify IP address and port (separated by a colon); the port can be different while the IP address is the same for every node.

For more information, see Specifying Ports on page 852.

Specifying Ports

On installations where the MapR File System instances, CLDB, and/or MAST Gateway must be reached on non-standard ports, you can specify the ports to advertise in the `MAPR_EXTERNAL`, `CLDB_EXTERNAL_RPC_PORT`, and `MASTGATEWAY_EXTERNAL_RPC_PORT` environment variables respectively. This setting does not change the ports used by the servers, but changes the ports advertised to clients (to support port forwarding).

If the cluster nodes are no longer reachable on the standard ports, you can specify ports for MapR File System using the `MAPR_EXTERNAL` environment variable. `MAPR_EXTERNAL` allows the specification of the advertised ports for the MapR File System instances only; this environment variable cannot be used to specify ports for CLDB or the MAST Gateway. Instead, use `CLDB_EXTERNAL_RPC_PORT` environment variable to specify port for CLDB and `MASTGATEWAY_EXTERNAL_RPC_PORT` environment variable to specify port for MAST Gateway. If ZooKeeper is not available on the default port or if there are multiple ZooKeepers listening on the same external IP address, you can specify ports for each ZooKeeper using the configure.sh utility.

See the following sections for more information on setting the ports.

Specifying Ports for MapR File System

If the port forwarding table is set up, ports must be configured for every MapR File System node on every MapR File System instance. For more information on the number of ports used by MapR File System instance(s), see Ports Used by MapR Software on page 2290. To specify the ports for MapR File System:

1. Open the `$MAPR_HOME/conf/env_override.sh` file. If the `env_override.sh` file is not present, you might have to create it. See About env_override.sh on page 2290.
2. Set the value for the `MAPR_EXTERNAL` environment variable.

   The value for the `MAPR_EXTERNAL` environment variable is a comma-separated list of IP addresses and colon-separated list of ports (to use for port forwarding).

   For example:

   ```
   export MAPR_EXTERNAL=10.11.0.0;9000,9001,9002,9003
   ```

   The following example shows 3 MapR File System instances with 4 ports:

   ```
   export
   MAPR_EXTERNAL=10.11.0.0;9000,9001,9002,9003:10000,10001,10002,10003:11000
   ,11001,11002,11003
   ```

   To specify:

   - Multiple IP addresses, use comma to separate the IP addresses.
   - Ports for multiple instances, use:
     - comma (`,`) to separate the ports for an instance
     - colon (`:`) to separate the set of ports for each instance

   If ports are not specified, MapR File System is assumed to be reachable on the default ports.
3. Save and close the `$MAPR_HOME/conf/env_override.sh` file.

last-updated: Apr 11, 2022

Specifying Ports for CLDB

The default port for CLDB is 7222. If you want to use another port:

1. Open the `$MAPR_HOME/conf/env_override.sh` file on the CLDB host(s). If the env_override.sh file is not present, you might have to create it. See About env_override.sh on page 2290.
2. Set the value for the `CLDB_EXTERNAL_RPC_PORT` environment variable in the file.

   The value for this environment variable is the port to use for CLDB.

   For example:

   ```
   export CLDB_EXTERNAL_RPC_PORT=5000
   ```

   This is especially useful if MapRMapR Data Platform is installed in a Docker container or other guest hosts. If this is not set, CLDB must be reachable on the default port 7222.
3. Save and close the `$MAPR_HOME/conf/env_override.sh` file.
4. Ensure that the mapr-clusters.conf file on the client host(s) contains the correct port number for CLDB.

Note: After setting this environment variable, make sure that `cldb.feature.external.ip` is enabled if you upgraded from a prior version of MapR to v6.0. For more information on enabling this feature, see Step 4: Enable New Features on page 330.

Specifying Port for MAST Gateway

The default port for MAST Gateway is *8660*. If you want to use another port:

1. Open the `$MAPR_HOME/conf/env_override.sh` file on the MAST Gateway host(s). If the env_override.sh file is not present, you might have to create it. See About env_override.sh on page 2290.
2. Set the value for the `MASTGATEWAY_EXTERNAL_RPC_PORT` environment variable in the file.

   The value for this environment variable is the port to use for MAST Gateway.

   For example:

   ```
   export MASTGATEWAY_EXTERNAL_RPC_PORT=15000
   ```

   If this is not set, MAST Gateway must be reachable on the default port *8660*.
3. Save and close the `$MAPR_HOME/conf/env_override.sh` file.

Specifying Ports for ZooKeeper

If ZooKeeper is not available on the default port or if all the ZooKeepers are listening on the same external IP address (such as in a Docker container), you can specify the port on which to reach each ZooKeeper. To specify the port on which to reach each ZooKeeper, during cluster configuration:

- Run the configure.sh utility with the `-EZ` option.

  The value for the `-EZ` option is a comma-separated list of external IP addresses of the ZooKeeper nodes and the port (for each IP address), separated by a colon, on which ZooKeeper can be reached. For example:

  ```
  /opt/mapr/server/configure.sh -C <IP|Hostname>[,<IP|Hostname>,...] -Z <IP|
  Hostname>[,<IP|Hostname>,...] \
  -EZ <IP|Hostname>:<Port>[,<IP|Hostname>:<Port>,...]
  ```

For example, you can specify:

- Different ports when the same external IP address is used for all ZooKeeper nodes as shown below:

```
/opt/mapr/server/configure.sh -C 172.17.0.2,172,17.0.3,172.17.0.4 -Z
172.17.0.2,172,17.0.3,172.17.0.4 \
-EZ 10.10.104.34:5181,10.10.104.34:5182,10.10.104.34:5183 -N
my.cluster.com
```

- Same ports when different IP addresses are specified for ZooKeeper nodes:

```
/opt/mapr/server/configure.sh -C 172.17.0.2,172,17.0.3,172.17.0.4 -Z
172.17.0.2,172,17.0.3,172.17.0.4 \
-EZ 10.10.104.34:5181,10.20.105.34:5181,10.30.106.34:5181 -N
my.cluster.com
```

Configuring MR AppMaster Port Mapping

1. Set the `yarn.app.mapreduce.am.job.client.port-range` parameter in the yarn-site.xml file to specific range of free ports in all the NodeManager nodes.

   Specify the range of ports that the MapReduce AppMaster can use when binding. Do not specify a value for this parameter if you want all possible ports. For example:

   ```
   50000-50050,50100-50200
   ```

   Note: Each Docker instance where NodeManager is running should have different range and the range should be different across all NodeManager nodes.

   For example:

   ```
   The yarn-site.xml file in docker container 1:
   <property>
           <name>yarn.app.mapreduce.am.job.client.port-range</name>
           <value>50000-50050</value>
   </property>
   The yarn-site.xml file in docker container 2:
   <property>
           <name>yarn.app.mapreduce.am.job.client.port-range</name>
           <value>50100-50150</value>
   </property>
   The yarn-site.xml file in docker container 3:
   <property>
           <name>yarn.app.mapreduce.am.job.client.port-range</name>
           <value>50151-50200,50250-50300</value>
   </property>
   ```

2. Set the port forwarding rules in host machine for these specific ranges.

   For example, if NM1 contains ranges from 50000 to 50050, then set the IP table rules such that when requests come on these ports, it is forwarded to NM1.

3. Specify the AWS or Docker host name for the IP address in the `/etc/hosts` file on the client system so that external clients can resolve Docker or AWS hostname properly when running the jobs.

   For example, your entry in the `/etc/hosts` file should look similar to the following:

   ```
   54.208.145.112 ip-10-10-0-103.ec2.internal
   ```

Working with a Logical Volume Manager
Explains the role and usage of a Logical Volume Manager.

The Logical Volume Manager creates symbolic links to each logical volume's block device, from a directory path in the form:

```
/dev/<volume group>/<volume name>
```

MapR needs the actual block location, which you can find by using the `ls -l` command to list the symbolic links.

1. Make sure you have free, unmounted logical volumes for use by MapR:
   - Unmount any mounted logical volumes that can be erased and used for MapR.
   - Allocate any free space in an existing logical volume group to new logical volumes.
2. Make a note of the volume group and volume name of each logical volume.
3. Use `ls -l` with the volume group and volume name to determine the path of each logical volume's block device. Each logical volume is a symbolic link to a logical block device from a directory path that uses the volume group and volume name:

   ```
   /dev/<volume group>/<volume name>
   ```

   The following example shows output that represents a volume group named `mapr` containing logical volumes named `mapr1, mapr2, mapr3,` and `mapr4`:

   ```
   ls -l /dev/mapr/mapr*
    lrwxrwxrwx 1 root root 22 Apr 12 21:48 /dev/mapr/mapr1 -> /dev/mapper/
   mapr-mapr1
    lrwxrwxrwx 1 root root 22 Apr 12 21:48 /dev/mapr/mapr2 -> /dev/mapper/
   mapr-mapr2
    lrwxrwxrwx 1 root root 22 Apr 12 21:48 /dev/mapr/mapr3 -> /dev/mapper/
   mapr-mapr3
    lrwxrwxrwx 1 root root 22 Apr 12 21:48 /dev/mapr/mapr4 -> /dev/mapper/
   mapr-mapr4
   ```

4. Create a text file `/tmp/disks.txt` containing the paths to the block devices for the logical volumes (one path on each line). Example:

   ```
   cat /tmp/disks.txt
    /dev/mapper/mapr-mapr1
    /dev/mapper/mapr-mapr2
    /dev/mapper/mapr-mapr3
    /dev/mapper/mapr-mapr4
   ```

5. Pass `disks.txt` to disksetup.

Tuning for SSDs
Lists the parameters to tune for optimal SSD performance.

On servers with SSDs:

1. Enable TRIM operation in the `mfs.conf` file, if recommended by the SSD vendor.

   By default, TRIM is disabled. To enable, set the value for `mfs.ssd.trim.enabled` to 1 in the `mfs.conf` file. For example:

   ```
   mfs.ssd.trim.enabled=1
   ```

2. Disable IO throttling in the `mfs.conf` file.

   To disable, set the value for `mfs.disk.iothrottle.count` to 50000. The default value for `mfs.disk.iothrottle.count` is 100. For example:

   ```
   mfs.disk.iothrottle.count=50000
   ```

3. Create storage pool with multiple SSDs (so that the throughput is less than 2GB/sec).

   Note: Create one storage pool per SSD only if the device is high-end.

   To create, run `disksetup`:

   ```
   /opt/mapr/server/disksetup -W <n> disks.txt
   ```

   For example, to create a storage pool with 2 SSDs, run the following command:

   ```
   /opt/mapr/server/disksetup -W 2 disks.txt
   ```

   For more information, see `disksetup` on page 2092.

Administering Volumes

This section provide information about how to organize and manage data using volumes, a unique feature of MapR clusters.

MapR provides volumes as a way to organize data and manage cluster performance. A volume is a logical unit that allows you to apply policies to a set of files, directories, and sub-volumes. You can use volumes to enforce disk usage limits, set replication levels, establish ownership and accountability, and measure the cost generated by different projects or departments. Create a volume for each user, department, or project.

You can mount volumes under other volumes to build a structure that reflects the needs of your organization. Sub-volumes are created by mounting a volume in a sub-directory of an already mounted volume. This establishes a parent-child relationship between the volumes whereas the parent volume is mounted in top-level directory and the child volume is mounted in the sub-directory. The volume structure defines how data is distributed across the nodes in your cluster. Create multiple small volumes with shallow paths at the top of your cluster's volume hierarchy to spread the load of access requests across the nodes.

A well-structured volume hierarchy is an essential aspect of your cluster's performance. As your cluster grows, keeping your volume hierarchy efficient maximizes data availability. Cluster performance is negatively affected when a volume structure is not in place.

The Control System **Volumes** page under **Data** contains the following tabs:

- Summary
- Snapshots
- User Disk Usage

last-updated: Apr 11, 2022

- Schedules
- Storage Policies
- Remote Targets

Managing Data with Volumes

Provides an overview of how to manage volume data.

The **Summary** tab in the **Volumes** page displays the following panes:

- Top Volume Utilization — Volumes that use the most amount of disk space.
- Active Alarms — List of active volume alarms on the cluster.
- Local and Remote Tier Storage utilization.
- Volumes — List of volumes.

The page includes the **Create Volume** button to create standard or (local and/or remote) mirror volumes. You can perform the following procedures to manage volumes on the MapR cluster using the Control System and the CLI.

Viewing the List of Volumes
Explains how to view the list of volumes either through the Control Panel or the CLI.
Viewing All the Volumes Using the Control System

- Log in to the Control Panel and click **Data** > **Volumes** > **Summary.**.

  Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

  The **Volumes** pane in the **Summary** tab (under **Data** > **Volumes**) displays the volumes in the cluster.

  Note: By default, system volumes are not displayed. If you wish to view system volumes also, select the **Include System Volumes** checkbox.

  If you are in another view, select **All Volumes** from the drop-down menu in the **Volumes** pane. Up to 10 volumes, sorted by name, are displayed in each page. For each volume, the pane displays the following information, by default:

Column Name	Column Description
Volumes	The name of the volume (used for default sorting).
Data Tiering	Whether volume is enabled or disabled for data tiering.
Alarms	The number of alarms associated with the volume. Hover the cursor over the number for information on the alarm including alarm name, severity, and time when the alarm was raised.
Mnt	Specifies whether (✓) volume is mounted.
Type	The type of volume. Both standard and mirror volumes are displayed.
Mount Path	The path where the volume is mounted.
Creator	The user or group that owns the volume.

last-updated: Apr 11, 2022

Column Name	Column Description
Quota	The amount of disk space allocated and utilized by the volume and associated snapshots and the cluster reserve limit (in red). If quota is not set, displays option to **Set Quota**.
Total Size	The total physical size of the volume and associated snapshots. When you move the cursor over the value, the popover shows the total logical size (**Data Size**), the logical size of the volume (**Volume Size**), and the logical size of the snapshots (**Snapshot Size**).
RF	The replication factor that specifies the number of copies for the volume.
Physical Topology	The rack path to the volume.

You can sort the list by volume name, mount, mount path, or creator.

Selecting the checkbox beside a volume makes the **Actions** drop-down menu available. From the **Actions** drop-down menu, you can:

- Edit the selected volume(s)
- Remove the selected volume(s)
- Create snapshot(s) of the selected volume
- Change the selected mirror volume(s) to standard volume(s)
- Start the mirroring operation(s) for the selected mirror volume(s)
- Stop the mirroring operation(s) for the selected mirror volume(s)
- Mount the selected volume(s)
- Unmount the selected volume(s)
- Offload selected volume(s)
- Recall selected volume(s)
- Abort currently running tiering job for selected volume(s)

Viewing the List of Standard Volumes Using the Control System

The **Volumes** pane in the **Summary** tab under **Data** > **Volumes** (under **Volumes** in the Kubernetes version of the Control System) displays all the volumes in the cluster by default. To view a list of only the standard volumes:

- Select **Standard Volumes** from the drop-down menu in the **Volumes** pane.

  The list of standard volumes in the cluster displays. By default, the list is sorted by volume name. For each volume, the page displays the following information by default:

Column Name	Description
Volumes	The name of the volume (used for default sorting).
Alarms	The number of alarms associated with the volume. You can view the **Active Alarms** pane for more information on the alarms associated with a volume.
Mnt	Specifies whether the volume is mounted.

last-updated: Apr 11, 2022

Column Name	Description
Type	The type of volume. Only standard volumes are displayed.
Mount Path	The path where the volume is mounted.
Data Tiering	Whether volume is enabled or disabled for data tiering.
Creator	The user or group that owns the volume.
Quota	The amount of disk space allocated and utilized by the volume and associated snapshots and the reserve limit (in red). If quota is not set, displays option to **Set Quota**.
Total Size	The size of the volume and associated snapshots.
RF	The replication factor that specifies the number of copies for the volume.
Physical Topology	The rack path to the volume.

Selecting the checkbox beside a volume makes the **Actions** drop-down menu available. From the **Actions** drop-down menu, you can:

- Edit the selected volume(s)
- Remove the selected volume(s)
- Create snapshot(s) of the selected volume
- Mount the selected volume(s)
- Unmount the selected volume(s)

Viewing the List of Mirror Volumes Using the Control System

The **Volumes** pane in the **Summary** tab under **Data** > **Volumes** (under **Volumes** in the Kubernetes version of the Control System) displays the list of (both) standard and mirror volumes in the cluster by default. To view a list of only the mirror volumes:

- Select **Mirror Volumes** from the drop-down menu in the **Volumes** pane.

  The list of mirror volumes in the cluster displays. By default, the list is sorted by volume name and the following columns are displayed:

Column Name	Description
Volumes	The name of the volume (used for default sorting).
Alarms	The number of alarms associated with the volume. You can view the **Active Alarms** pane (above) for more information on the alarms associated with a volume.
Mnt	Specifies whether the volume is mounted.
Source Volume	The source volume name for the mirror volume.
Source Cluster	The source cluster name for the mirror volume.
Originating Cluster	The originating cluster for the data being mirrored.
Originating Volume	The originating volume for the data being mirrored.
Mirror Status	The status of the last mirroring operation.
Percentage Complete	The percentage of the in-progress mirroring operation that has been completed.

last-updated: Apr 11, 2022

Column Name	Description
Data Tiering	Whether or not the volume is enabled or disabled for data tiering.

Selecting the checkbox beside a volume makes the **Actions** drop-down menu available. From the **Actions** drop-down menu, you can:

- Edit the selected volume(s)
- Remove the selected volume(s)
- Create snapshot(s) of the selected volume
- Change the mirror volume to a standard volume
- Start the mirroring operation(s) for the selected mirror volume(s)
- Stop the mirroring operation(s) for the selected mirror volume(s)
- Mount the selected volume(s)
- Unmount the selected volume(s)

Viewing the List of Tiered Volumes Using the Control System

The **Volumes** pane in the **Summary** tab of the **Data** > **Volumes** page (under **Volumes** in the Kubernetes version of the Control System) displays the list of (both) standard and mirror volumes in the cluster by default. To view a list of tiered standard and mirror volumes:

- Select **Tiered Volumes** from the drop-down menu in the **Volumes** pane.

  The list of tiered standard and mirror volumes in the cluster displays. By default, the list is sorted by volume name. For each volume, the pane displays the following information, by default:

Column Name	Description
Volumes	The name of the volume (used for default sorting).
Offload Tier	The name of the tier where the volume data is stored.
Job	The tiering operation that is currently running or was last performed on the volume. Value can be one of the following: • Offload — if volume data was offloaded or is currently being offloaded • Abort — if volume data offload or recall operation was aborted or is being aborted • Recall — if volume data was recalled or is being recalled from the tier
State	The status of the job.
Progress	The job completion percentage.
Start Date/Time	The date and time when the job was started.
End Date/Time	The date and time when the job completed.
Offload Speed	The amount of data (in MB) offloaded or being offloaded per second.

last-updated: Apr 11, 2022

Column Name	Description
Recall Speed	The amount of data (in MB) recalled or being recalled per second.

Selecting the checkbox beside a volume makes the **Actions** drop-down menu available. From the **Actions** drop-down menu, you can:

- Edit selected volume(s)
- Remove selected volume(s)
- Create snapshot(s) of selected volume(s)
- Change the mirror volume to a standard volume
- Start the mirroring operation(s) for the selected mirror volume(s)
- Stop the mirroring operation(s) for the selected mirror volume(s)
- Mount selected volume(s), if they are not already mounted
- Unmount selected volume(s), if they are currently mounted
- Offload selected volume(s)
- Recall selected volume(s)
- Abort currently running tiering job for selected volume(s)

Retrieving the List of Volumes Using the CLI and REST API

The basic command to retrieve the list of volumes is:

```
maprcli volume list
```

For complete reference information, see `volume list` on page 1979.

Customizing the List of Columns/Fields
Explains how to customize the columns that are displayed in the Control System, and the fields that are returned in the CLI.
Customizing the Columns in the Control System

1. Log in to the Control System and go to:
   - **Data** > **Volumes** page to customize columns displayed in the **Volumes** pane.

     Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

   - **Nodes** page to customize columns displayed in the **Nodes** pane.

     Note: The **Nodes** page is not available on the Kubernetes version of the Control Panel.

2. Click the **Customize Columns** icon ( ).
   In the **Customize Columns** dialog, the:
   - **Available** list displays the columns that are available for display.
   - **Selected** list displays the columns currently displayed in the pane.

last-updated: Apr 11, 2022

3. Select the columns from the:
   a) Available list of columns and click ▶ to move selection to **Selected** columns (for display).
   b) Selected list of columns and click ◀ to remove selected columns from displaying.
4. (Optional) Click ^ and/or down ⌄ arrows to sort the order of columns.
5. Click **Save Changes** for the customization to take effect.

   Tip: To reset the display to its default columns, click **Reset to default columns**.

Customizing the Fields Using the CLI or REST API

Use the `-column` parameter with the `maprcli` command to view specific fields in the list. For example:

- To view the health of the nodes and services installed on the nodes being retrieved, run the following command:

  ```
  maprcli node list -columns service,health
  ```

  For complete reference information, see the `node list` `on page 1705` command.

- To view the volume name for the list of volumes being retrieved, run the following command:

  ```
  maprcli volume list -columns volumename
  ```

  For complete reference information, see `volume list on page 1979` command.

Reverting to Default List of Columns
Describes how to revert to the default list of columns on the Control System

1. Log in to the Control System and click:
   - **Data** > **Volumes** to revert to the default list of columns in the **Volumes** pane.

     Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

   - **Nodes** to revert to the default list of columns in the **Nodes** pane.

     Note: The **Nodes** page is not available in the Kubernetes version of the Control System.
2. Click the **Customize Columns** icon ( ) .
3. Click **Reset to default columns**,
4. Click **Save Changes**.
   The pane displays the default list of columns.

Filtering the List of Volumes
Explains how to filter the list of volumes using either the Control System or the CLI.

The filter lets you build search expressions to provide sophisticated filtering capabilities for locating specific data on views that display a large number of volumes. Expressions are implicitly connected by the AND operator.

Filtering on the Control System

1. Log in to the Control System and click **Data** > **Volumes** to filter volumes in the **Volumes** pane.

   Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

2. Select one of the following options from the **Add Filter** drop-down menu:
   - Volume — to filter the list by volume name
   - Usage — to filter the list by amount of disk used
   - Mount Path — to filter the list by mount path
   - Creator — to filter the list by entity or volume owner
   - Total Size — to filter the list by size of volume
   - Replication Factor — to filter the list by replication factor
   - Physical Topology — to filter by the rack path
   - Tier Type - to filter by a type of tier
   - Quota — to filter by hard quota
   - Data on Wire Encryption — to filter by volumes enabled (**On**) or disabled (**Off**) for on-wire encryption
   - Data at Rest Encryption — to filter by volumes enabled (**On**) or disabled (**Off**) for data-at-rest encryption (DARE)
   - Last Access Time - to filter by the Last Access Time
   - Coalesce Interval - to filter on the coalesce interval
3. Specify the value in the drop-down field for the selected filter (by which to filter the list of volumes) and click **Filter**.

   As you make selections and specify the filtering criteria, the pane displays only the volumes that match the specified filtering criteria.
4. Click:
   - **Add Filter** to add another filtering criteria.
   - ✕ to remove a filtering criteria.
   - 🗑 to clear all filter settings.

Filtering Using the CLI

The `volume list on page 1979` command can be used with the `-filter` option, which let you specify large numbers of volumes by matching specified values in specified fields rather than by typing each name explicitly. For example, you can display all volumes whose owner is *root* and whose name begins with *test* as follows:

```
maprcli volume list -filter [n=="test*"]and[on=="root"]
```

For more information, see Filters on page 1526.

last-updated: Apr 11, 2022

Creating a Volume
Describes volume types, and illustrates how to create volumes using either the Control System, CLI, or the REST API.

You can create a new (Standard or Mirror) volume using the Control System, the CLI or the REST API.

Creating a Volume Using the Control System

To create a new (Standard or Mirror) volume using the Control System:

1. Go to **Data** > **Volumes** and click **Create Volume**.

   Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

   The **Create New Volume** page displays.

2. Select the **Volume Type** under **SETTINGS** in the **Properties** tab. Choose:
   - **Standard Volume** to create a read-write volume.
   - **Mirror Volume** to create a volume that is a read-only copy of a source volume.

     Tip: See also: Mirror Types on page 464.

3. Specify the following required settings in the **Properties** tab:

Standard Volume

Volume Name	Enter a name for the volume. The name should contain only the following characters: `A-Z a-z 0-9 _ - .` **Note:** • The volume name should not begin with `mapr.` because `mapr.` is used for system volumes. If you use `mapr.` at the start of the volume name, the volume may not display in the default view of the list of volumes on the Control System; you must select the **Include System Volumes** checkbox in the **Volumes** pane to view volumes with names beginning with `mapr`. • For tiering-enabled volumes, volume name should not exceed ninety-eight characters.
Accountable Entity	Select the type of *accountable entity (AE)* and enter the name of the entity in the associated text field.

Steps 4 to 8 are optional and allow you to define optional volume properties (for auditing, replication, and data tiering), volume access, and volume quota.

last-updated: Apr 11, 2022

If these are not defined, default values, where available, are used. You can skip to:

- (Optional) Step 9 to associate a snapshot schedule and/or an offload schedule with the volume.
- Step 10 to create the volume with basic settings.

Mirror Volume

Volume Name	Enter a name for the volume. The name should contain only the following characters: `A-Z a-z 0-9 _ - .`  **Note:** Do not begin the volume name with `mapr.` as `mapr.` is used for system volumes. If you use `mapr.` at the start of the volume name, the volume may not display in the default view of the list of volumes on the Control System; you must select the **Include System Volumes** checkbox in the **Volumes** pane to view volumes with names beginning with `mapr.`.

Source Volume Cluster	Enter the name of the cluster on which the source volume exists. The name should contain only the following characters: `A-Z a-z 0-9 _ - .` Mirroring only works between two secure clusters or between two non-secure clusters. Mirroring does not work when one cluster is secure and the other is non-secure.  **Note:** When setting up mirror volumes for mirroring between clusters, for the mirroring operation to run successfully, servers in one cluster cannot use the same IP addresses as servers in the other cluster. For example, if node A in cluster A has a private IP address of 10.10.20.29, no server in cluster B can have the same private IP address. Also, all the servers in destination cluster must be able to reach all the servers in the source cluster and vice versa. For example, suppose 10.10.20.29 is the only IP address used by node A in cluster A; then all servers in cluster B should be able to reach the IP address 10.10.20.29.

last-updated: Apr 11, 2022

Source Volume Name	Enter the name of the source volume, from which the mirror volume pulls data, (after selecting the source volume cluster). The name should contain only the following characters: `A-Z a-z 0-9 _ - .` If the source volume is on: • The same cluster, you must create a local mirror volume, which is useful for load balancing or for providing a read-only copy of a data set. See Local Mirroring on page 465 for more information. • Another cluster, you must create a remote mirror volume, which is useful for offsite backup, for data transfer to remote facilities, and for load and latency balancing. See Creating Remote Mirrors on page 880 for more information. **Note:** If you plan to enable tiering for the mirror volume, ensure that the selected source volume is also tiering-enabled. You cannot create tiering-enabled mirror volumes to mirror data in standard volumes that are not tiering-enabled. For information on setting up mirror cascades, see Mirror Cascades on page 466.

last-updated: Apr 11, 2022

Accountable Entity	Select the type of *accountable entity (AE)* and enter the name of the entity in the associated text field.

Steps 4 to 8 are optional and allow you to define optional volume properties (for auditing and replication), volume access, and volume quota. If these are not defined, default values, where available, are used. You can skip to:

- (Optional) Step 9 to associate a mirroring schedule, a snapshot schedule, and/or an offload schedule with the mirror volume.

  The mirroring schedule specifies how often to run the mirroring operation, which copies data (that has changed at the file-block level since the last data transfer) directly from the source, even if the files in the source volume are being written to or deleted.

  The snapshot schedule specifies how often to take a snapshot of the mirror volume for the purpose of preserving the state of the mirror before a subsequent mirroring operation.

  The offload schedule specified how often to offload data in the volume to the configured tier based on the criteria in the storage policy.

- Step 10 to create the mirror volume with basic settings.

4. (Optional) Specify the following general and auditing settings under **Properties**:

Mount	Specifies whether to automatically mount (**Yes**) or not mount (**No**) the volume after creation. By default, volumes are mounted immediately after creation. If this is set to **Yes**, you must also specify the mount path.
Mount Path	The path to mount the volume. This is required if the value for **Mount** is **Yes**. **Note:** The path must be relative to `/` and cannot be in the form of a global namespace path (for example, `/mapr/<cluster-name>/`).
Data Tier	Specifies whether (**Yes**) or not (**No**) to enable data tiering for volume.
Collect Metrics	Specifies whether (**Yes**) or not (**No**) to enable metrics collection for this volume. For more information, see Collecting Volume Metrics on page 1300 and Enabling Volume Metric Collection on page 1301.
Volume Access	Specifies whether the volume is read-only or read/write volume. Data on a read-only volume can only be read and data can both be read from and written to a read/write volume. By default, a standard volume is created with read/write access. A mirror volume can only be a read-only volume.

last-updated: Apr 11, 2022

Enable Auditing	Select one of the following: • Disable auditing • Enable auditing for volume so that auditing can selectively be enabled for directories, files, tables, and streams in the volume • Enable auditing of operations on all files, tables, and streams in the volume whether or not auditing is enabled for files, tables, and streams in the volume.
Coalesce Interval	The interval of time (in minutes) to use when logging multiple READ, WRITE, or GETATTR operations on one file from one client IP address, if auditing is enabled. The default value is 60 minutes.
Data on Wire Encryption	Specifies whether (**Yes**) or not (**No**) to encrypt data in the volume during transmission. By default, the value is **Yes** for all new volumes on a secure cluster. This parameter is not supported on unsecure clusters.
Data at Rest Encryption	Specifies whether (**Yes**) or not (**No**) to encrypt data-at-rest. This is not supported on unsecure clusters.

5. (Optional) Specify the following under **Properties** for volume (hot) data replication.

Topology	The rack path to the volume. The default topology is `/data`.
Optimize Replication for	The basis for the replication factor: • High throughput, or cascading replication, where volumes are replicated sequentially on intermediate and tail containers. • Low latency, or star replication, where volumes are replicated on multiple containers in parallel. The default value is high throughput. See Selecting a Replication Type for High Availability on page 909.
Replication	The minimum (**Minimum Replication**) and desired (**Target Replication**) number of copies for the volume data. The default minimum is 2, and the default target is 3.
Name Container Replication	The (**Minimum Replication**) and desired (**Target Replication**) number of copies for the name container associated with the volume. The default minimum is 2, and the default target is 3.
Guarantee Min Replication	Specifies whether (**Yes**) or not (**No**) to enforce minimum number of copies. If this is enabled (**Yes**), writes succeed only when the minimum number of copies exist. If this is enabled (**Yes**) and minimum number of copies are not available, the client is asked to retry. For more information, see Understanding Replication on page 454.

6. (Optional) If data tier is enabled (**Yes**), select one of the following, and set values for associated properties.

Erasure Coding (Warm)	For offloading data to an erasure coded volume, specify values for the following properties. If values are not specified, default values are applied.

Topology	The topology of the erasure coded volume from the drop-down list.
Storage Policy	The rule for offloading data in this volume. You can click: • **Browse** to select an existing rule. • **Create** to create a new rule for offloading data. See steps 3 - 5 in the Creating a Storage Tier Policy on page 972 topic for more information. If you do not select a storage policy, the default policy named `default.ectier.rule`, which is all files (p), is associated with the volume.

Erasure Coding	The erasure coding scheme, which is the number of data chunks and number of parity chunks. You can select one of the following from the **Scheme** drop-down menu: • 3,2 — for 3 data chunks and 2 parity chunks. You must have 5 or more nodes on the cluster for this option. If selected, this has 60% storage overhead and can tolerate failure of up to 2 nodes. • 4,2 — for 4 data chunks and 2 parity chunks. You must have 6 or more nodes on the cluster for this option. If selected, this has 50% storage overhead and can tolerate failure of up to 2 nodes. • 5,2 — for 5 data chunks and 2 parity chunks. You must have 7 or more nodes on the cluster for this option. If selected, this has 40% storage overhead and can tolerate failure of up to 2 nodes. • 6,3 — for 6 data chunks and 3 parity chunks. You must have 9 or more nodes on the cluster for this option. If selected, this has 50% storage overhead and can tolerate failure of up to 3 nodes.  **Note:** Although you can create a volume even if the required number of nodes are not present, offload operation fails if the required number of nodes are not present. See Erasure Coding Scheme for Data Protection and Recovery on page 926 for more information.

last-updated: Apr 11, 2022

Remote Archiving (Cold)

For offloading data to a low cost storage alternative on the cloud, specify values for the following properties.

Remote Target	The location to offload data to. You can click: • **Browse** to select an existing tier. • **Create** to create a new tier. See Creating a Storage Tier on page 958 for more information.
Storage Policy	The rule for offloading data in this volume. You can click: • **Browse** to select an existing rule. • **Create** to create a new rule for offloading data. See Creating a Storage Tier Policy on page 972 for more information.
Retention Duration after Recall	The number of days to retain data recalled from the tier to the MapR cluster. Once the number of days is reached, recalled data on the MapR cluster is purged (if there are no changes), or offloaded (if there are changes).
Tier Encryption	Specifies whether (**Yes**) or not (**No**) to enable encryption of data on the tier. This cannot be modified once it is set. The default value is **No** (disabled).

7. (Optional) Specify the following volume (administrative and user) access control settings in the **Authorization** tab:

<table>
<tr>
<td>ADMINISTRATIVE CONTROLS</td>
<td>
The users and/or groups that have one or more of the following permissions:
<table>
<tr><td>Dump & Backup</td><td>Transport large amount of data or copies of the volume on physical media to a remote cluster using backup files.</td></tr>
<tr><td>Restore & Mirror</td><td>Restore a volume from a dump file and create mirror volumes, which is a read-only copy of the source volume.</td></tr>
<tr><td>Edit</td><td>Edit volume properties, create and delete snapshots.</td></tr>
<tr><td>Delete</td><td>Delete the volume.</td></tr>
<tr><td>Admin</td><td>View and edit access control settings (but cannot perform volume operations).</td></tr>
<tr><td>Full Control</td><td>Perform all volume-related administrative operations except changing access control settings.</td></tr>
</table>
To define administrative access control settings for another user or group, click Add Another.
Note: To perform this action from the command line, refer to acl set on page 1531.
By default, the root user and the volume creator have full control permissions on the volume.
<table>
<tr><td>Root Directory Permission</td><td>Set read, write, and/or execute permissions on the root directory for users, groups and others.</td></tr>
</table>
</td>
</tr>
</table>

USER ACCESS CONTROLS	The users, groups, and/or roles that have and/or do not have access to read and/or write to the volume. To grant or block access to users, groups, and/or roles, from the: • Basic settings, select the type — public, (OR) user, group, or role — from the drop-down menu, specify the name of the user, group, or role, and select one or more checkbox to grant permissions. **Tip:** Click to create a copy of the associated access control setting. Click to remove the associated access control expression. To add *ACE*s for another user, group, or role, click **Add Another** and repeat this step. • Advanced settings, specify public (p) or user (u), group (g), and/or role (r) who have or do not have the type of access using the following boolean expressions and subexpressions: • ! — Negation operator. • & — AND operation. • | — OR operation. Use ( ), parentheses, for subexpressions. **Note:** You *cannot* specify user, group, or role individually if access is granted to all users (public). Alternatively, click associated with the type of access to use the **Access Control Expression** window to define access for public or users, group, and/or role. See Defining ACEs on page 1473 for more information. **Note:** If you switch from **Basic** to **Advanced**, the basic settings, if any, are carried over to the advanced settings. If you switch from **Advanced** to **Basic**, all the settings are lost because the subexpressions and AND (&) and negation (!) operations that are supported by advanced settings are not supported in the basic settings. By default, access is granted to all users (public). If access is granted to all users (public), access *cannot* also be granted individually to users, groups, and/or roles.

8. Define the amount of disk space (optional) in megabytes (MB), gigabytes (GB), or terabytes (TB) to allocate to the volume in the **Usage** tab:

   By default, there is no limit on the disk space.

   Warning: If the quota set for the mirror volume is less than the quota set for its source volume, the CLDB raises an alarm. The mirroring operation does not fail, but you must decide whether to add space and increase the mirror volume quota, or remove unwanted space from the source volume and decrease its quota.

Advisory Quota	An alarm is raised when this limit is reached.

last-updated: Apr 11, 2022

Hard Quota	Writes are not allowed when this limit is reached. When you set a hard quota for a tiering-enabled volume, the quota is the total space allocated for the volume irrespective of the location (cluster or tier) where the volume data is stored. For example, if you allocate 1GB of hard quota for a tiering-enabled volume, writes fail after you write 1GB of data whether or not the volume data is local (on the cluster), or offloaded (to the tier).

Note: The value for advisory quota cannot be greater than the value for hard quota.

The graph shows the currently used (across volumes), reserve limit (which is 90% of the total disk space on the cluster), and total amount of disk space at the cluster level.

9. (Optional) From the **Schedule** tab, select the schedule(s) to associate with the volume.

Standard Volume

You can click **Browse** to select an existing schedule, or click **Create** to create a schedule, for the following:

Snapshot Schedule	The snapshot schedule specifies when snapshots should be automatically created. If you do not define this schedule, snapshots are not created for this volume. Each snapshot created by a schedule has an expiration date that determines how long the snapshot is retained.
Tier Offload	The tier offload schedule specifies the frequency for automatically offloading data in the volume to the configured tier. This is only available for tiering enabled volumes.

Mirror Volume

You can click **Browse** to select an existing schedule, or click **Create** to create a schedule for the following:

last-updated: Apr 11, 2022

Snapshot Schedule	The snapshot schedule specifies how often to take a snapshot of the mirror volume to preserve the state of the mirror before a subsequent mirror operation. If corrupt data is copied from the source volume's snapshot into the mirror volume, you can roll back the mirror contents to the snapshot. Each snapshot created by a schedule has an expiration date that determines how long the snapshot is retained.
Mirror Schedule	The mirror schedule specifies how frequently the mirror volume must be synchronized with the source volume. If a disaster (or any type of data loss on a read-write source volume) occurs, the data can be recovered from the mirror volume, but any data written to the source volume since the last successful mirror operation is not on the mirror volume. Therefore, you should set the mirror schedule such that it meets your Recovery Point Objective (RPO). **Tip:** For best performance, select a mirroring schedule according to the anticipated rate of data changes and the available bandwidth for mirroring. See also: Guidelines for Setting Mirror Schedules on page 955.

last-updated: Apr 11, 2022

Tier Offload	The tier offload schedule specifies the frequency for automatically offloading data in the volume to the configured tier. This schedule is available only for tiering enabled volumes.

When selecting a schedule, you can choose your custom schedule if it exists, or one of the (following) defaults:

- Critical data
- Important data
- Normal data
- Automatic Tiering Scheduler (available only for tiering-enabled volumes)

10. Click **Create Volume** to create the volume.

- If you created a remote mirror, edit the `mapr-clusters.conf` file so that each cluster can resolve the nodes in the other cluster. For more information, see Setting Up Cross-Mirroring Between Unsecure Clusters on page 880.
- If the source volume at the primary datacenter is in a secure cluster, the destination cluster needs authorization to pull data from the source cluster in order to create a mirror volume. Authorization can be granted by means of a cross-cluster ticket generated by the source cluster administrator. For more information, see Configuring Secure Clusters for Cross-Cluster Mirroring and Replication on page 1486.

Creating a Volume Using the CLI or the REST API

The basic command to create a volume is:

```
/opt/mapr/bin/maprcli volume create -name <name>
```

You can create a standard read-write volume, a (local or remote) mirror volume, and/or a tiering enabled standard read-write, and/or a (local or remote) mirror volume.

The name should contain only the following characters:

```
A-Z a-z 0-9 _ - .
```

Standard Volume

The basic command to create a standard volume is:

```
/opt/mapr/bin/maprcli volume
create -name <volume name> -path
<volume path> -type rw
```

When creating and mounting a volume, the location of the mount point can be specified using the `path` parameter. The volume that is last in the `path` parameter is referred to as the parent volume. (The parent volume is the volume on which the volume link is created.)

For complete reference information, see `volume create` on page 1931.

Local Mirror Volume

1. Log in to the node on the cluster where you want to create the mirror.
2. Use the `volume create` command to create the mirror volume.

   Specify the `source` volume name, provide a `name` for the mirror volume, and specify the `type` as `mirror`. Optionally, associate a schedule with the volume. For example:

   ```
   /opt/mapr/bin/maprcli volume
   create -name volume-a -source
   volume-a -type mirror -schedule 2
   ```

   For complete reference information, see `volume create` on page 1931.

Remote Mirror Volume

1. Log in to the node on the cluster where you wish to create the mirror.
2. Use the `volume create` command to create the mirror volume.

   Specify the source volume and cluster in the format `<volume>@<cluster>`, provide a `name` for the mirror volume, and specify the `type` as `mirror`. Optionally, associate a schedule with the volume. For example:

   ```
   /opt/mapr/bin/maprcli volume
   create -name volume-a -source
   volume-a@cluster-1 -type
   mirror -schedule 2
   ```

   When creating and mounting a volume, the location of the mount point can be specified using the `path` parameter. The volume that is last in the `path` parameter is referred to as the parent volume. (The parent volume is the volume on which the volume link is created.)

   For complete reference information, see `volume create` on page 1931.
3. Ensure that each cluster can resolve the nodes in the other cluster.

   See:

   - Setting Up Cross-Mirroring Between Unsecure Clusters on page 880 for more information, if both the clusters are non-secure.
   - Configuring Secure Clusters for Cross-Cluster Mirroring and Replication on page 1486 for more information, if both clusters are secure.

Tiering Enabled Volume	The basic command to create a tiering-enabled volume is the following: `/opt/mapr/bin/maprcli volume create -name <volName> -path <volmountpath> -tieringenable true`

For the complete list of supported parameters, see `volume create` on page 1931.

Creating Remote Mirrors
Describes the use of remote mirror volumes. The remote mirror volume is present on a different cluster from the source volume.

Creating remote mirrors is similar to creating local mirrors, except that the mirror volume resides in a different cluster from the source volume. To properly identify the source volume, you must specify the source cluster name when the mirror volume is created. In addition, you must edit the `mapr-clusters.conf` file so that each cluster can resolve the nodes in the other cluster.

To create a mirror on a remote cluster, you must have the same UID for the `MAPR_USER` (the cluster owner) for both the primary cluster (where the source volume resides) and the remote clusters (where the mirror volumes reside; also known as the destination clusters). You also need to have the following volume permissions:

- `dump` permission on the source volumes
- `restore` permission on the mirror volumes at the destination clusters

When a mirror volume is created on a remote cluster (according to the entries in the `mapr-clusters.conf` file), the CLDB checks that the local volume exists in the local cluster. If both clusters are not set up and running, the remote mirror volume cannot be created.

To summarize:

- Each cluster must be already set up and running.
- Each cluster must have a unique name.
- Every node in each cluster must be able to resolve all nodes in remote clusters, either through DNS or entries in `/etc/hosts`.
- The UID for the `MAPR_USER` (cluster owner) must be the same for the source and destination clusters, irrespective of which user account triggers the mirror operation.
- Volume permission must be set to `dump` on the source volumes.
- Volume permission must be set to `restore` on the mirror volumes.

See also: Remote Mirroring on page 465.

Setting Up Cross-Mirroring Between Unsecure Clusters
Describes how to edit the `mapr-clusters.conf` file to mirror volumes between unsecure clusters.

To mirror volumes between unsecure clusters, start by editing the `mapr-clusters.conf` on page 2200 file on the cluster of the source volume and create an entry for each additional cluster that hosts a mirror of the volume. The entry must list the name of the cluster, followed by a space-separated list of hostnames and ports for the CLDB nodes of the cluster. In addition, use the `secure` parameter to specify whether the clusters are secure or non-secure. See Configuring Secure Clusters for Cross-Cluster Mirroring and Replication on page 1486 for more information about generating tickets for secure clusters.

Note: When mirroring between clusters, servers in one cluster cannot use the same IP addresses as servers in the other cluster. For example, if node A in cluster A has a private IP address 10.10.20.29, no server in cluster B can have the same private IP address. Also, all the servers in one cluster must be able to reach all the servers in the other cluster and vice versa. For example, suppose 10.10.20.29 is the only IP address used by node A in cluster A; then all servers in cluster B should be able to reach the IP address 10.10.20.29.

To set up cross-mirroring between unsecure clusters:

1. Review the requirements for Creating Remote Mirrors on page 880 before you begin.
2. On each cluster, make a note of the cluster name and CLDB nodes (the first line in `mapr-clusters.conf` on page 2200)
3. On each webserver and CLDB node, add the CLDB nodes of the remote cluster to `/opt/mapr/conf/mapr-clusters.conf`, using the following format:

   ```
   clustername1  <CLDB> <CLDB> <CLDB>
   [ clustername2  <CLDB> <CLDB> <CLDB> ]
   [ ... ]
   ```

   For example, suppose you have a cluster, `clusterA`, with two CLDB nodes, `nodeA` and `nodeB`. Now you want to add a second cluster called `clusterB` with CLDB nodes `nodeC` and `nodeD`. Edit the `mapr-clusters.conf` file and add the line for `clusterB` as shown:

   ```
   clusterA nodeA:7222 nodeB:7222
   clusterB nodeC:7222 nodeD:7222
   ```

   Warning: You must include the port number in the CLDB hostname notation.

   Here:

   a. First line contains cluster name and CLDB nodes of cluster A (the local cluster)

   b. Second line contains cluster name and CLDB nodes of cluster B (the remote cluster)

   The `mapr-clusters.conf` file for cluster B with 2 CLDB nodes (`nodeC` and `nodeD`) looks as follows:

   ```
   clusterB nodeC:7222 nodeD:7222
   clusterA nodeA:7222 nodeB:7222
   ```

   By creating additional entries in the `mapr-clusters.conf` file, you can mirror from one cluster to several others.
4. Set `secure=false` if both clusters are not secure. If both the clusters are secure, see Configuring Secure Clusters for Cross-Cluster Mirroring and Replication on page 1486 for more information.

   Warning: Mirroring only works between two secure clusters or between two non-secure clusters. Mirroring does not work when one cluster is secure and the other is non-secure.
5. On each cluster, restart the `mapr-apiserver` service on all nodes where it is running.

Creating a Volume for a Tenant
Provides an overview of how to create a volume for a tenant using the CLI.

last-updated: Apr 11, 2022

To create a volume for a tenant (when you have Multitenancy on File System on page 488), do the following:

1. Log in to the cluster as administrator and ensure that the ticket for the tenant has already been generated and copied on to the tenant host.
   If necessary, follow steps in Generating a Ticket for a Tenant on page 1429 to set up the ticket for the tenant.
2. Create a volume for the tenant by running the following command:

```
$ maprcli volume create -name <volumeName> -path
<path_to_volume> -tenantuser <tenant_user>
```

Note: For more information, see the `maprcli volume create` command.

Viewing Volume Information
Describes how to view volume information using either the Control System or the CLI.

You can retrieve and view volume information using either the Control System or the CLI.

1. Log in to the Control System and go to the **Summary** tab in the **Data** > **Volumes** page.

   Note: The **Summary** tab is under the **Volumes** menu in the Kubernetes version of the Control System.
2. Click the volume name in the **Volumes** pane.
   The volume information page displays. On this page, you can perform the following actions based on the type of the volume.

Standard Volume	Click the **Select Action** drop-down menu to: • Edit the volume • Remove the volume • Snapshot a volume • Change the mount path and mount or unmount the volume • Convert the volume to a mirror volume • Rename the volume
Mirror Volume	Click the **Select Action** drop-down menu to: • Edit the Volume • Remove the volume • Snapshot a volume • Start/stop mirroring • Promote the volume to a standard (read/write) volume • Change volume mount information • Rename the volume
Tiering Volume	Click the **Select Action** drop-down menu to:

- Edit the Volume
- Remove the volume
- Snapshot a volume
- Change volume mount information
- Rename the volume
- Offload a volume to a tier
- Recall a volume from a tier
- Abort a volume tiering job

The page displays tabs for viewing:

- Summary of the volume including:
  - All recent and active volume alarms
  - Metrics (such as disk usage) for the volume
  - Volume details such as the general, auditing, and tiering settings for the volume, ACLs and ACEs on the volume, and volume quotas and schedules
- Snapshots associated with the volume

Viewing Volume Details

Describes how to retrieve and view volume information using either the Control System or the CLI.

Viewing Volume Details in the Control System

- Log in to the Control System and go to the **Summary** tab in the volume information page for the volume.

  You can view the following in the **Detail** pane:

Properties

Table

Created By	Indicates the user who created the volume.
Volume Type	Indicates the type of volume. Value can be one of the following: • Standard • Mirror
Volume Name	Indicates the name of the volume.
Accountable Entity	Indicates the name of the accountable entity.
Volume Access	Indicates the type of access allowed on the volume. Value can be one of the following: • Read/Write • Read-only

last-updated: Apr 11, 2022

Table (Continued)

Mounted	Indicates whether or not volume is mounted.
Mount Path	Indicates the mount path of the volume.
Data on Wire Encryption	Indicates whether (**Yes**) or not (**No**) on-wire encryption is enabled for the volume.
Data at Rest Encryption	Indicates whether (**Yes**) or not (**No**) data-at-rest encryption (DARE) is enabled for the volume.
Topology	Indicates the rack path to the volume.
Partially out of Topology	Indicates whether containers associated with the volume are outside the volume's main topology.
Last accessed	Indicates the date when the volume was last accessed.
Enable Auditing	Indicates whether auditing is enabled at the volume level.
Coalesce Interval	Indicates the interval of time (in minutes) used for logging multiple READ, WRITE, or GETATTR operations on one file from one client IP address, if auditing is enabled.

Table

Replication	Indicates the minimum (Min Target) and desired (Max Target) number of copies for the volume data.
Name Container Replication	Indicates the minimum (Min Target) and desired (Max Target) number of copies for the name container.
Optimize Replication	Indicates the basis for replication. Value can be one of the following: • High throughput, or cascading replication • Low latency, or star replication

Table (Continued)

Guarantee Min Replication	Indicates whether or not to enforce minimum number of copies.

Note: The following is visible only if warm tiering is enabled for the volume.

Table

Topology	Indicates the topology of the erasure-coded volume.
Remote Target	Indicates the name of the tier.
Storage Policy Name	Indicates the name of the storage policy.
Policy Detail	Indicates the rules in the policy for offloading data.
Retention Duration after Recall	Indicates the period of time to keep recalled data in the hot tier.
Erasure Coding Scheme	Indicates the number of data and parity chunks.
Tier Purged	Indicates the amount of offloaded data.

Note: The following is visible only if cold tiering is enabled for the volume.

Table

Remote Target	Indicates the name of the remote storage.
Storage Policy Name	Indicates the name of the storage policy.
Policy Detail	Indicates the rules in the policy for offloading data.
Retention Duration after Recall	Indicates the number of days to keep recalled data in the MapR cluster (hot tier).
Tier Purged	Indicates the amount of offloaded data.

last-updated: Apr 11, 2022

Authorization

ADMINISTRATIVE CONTROLS	Indicates the users and/or groups who: • Have admin access and/or full control on the volume. • Can perform one or more administrative operations such as create a dump file and backup the volume, restore and mirror the volume, edit the volume properties, and delete the volume.
USER ACCESS CONTROLS	Indicates the type of users — public, user, group, and/or role — that have and do not have access to volume data.

Usage

Quota	Indicates the amount of (advisory and hard) disk space allocated to the volume.
Graph	The graphs show: • **DISKUSAGE**: The total amount of disk space allocated for the volume and the amount of disk space used by the volume and associated snapshots. • **CLUSTER DISKSPACE**: The total amount of disk space on the cluster, the amount of disk space allocated and utilized by the volumes in the cluster, and the reserve limit.

Schedule

Snapshot Schedule	Indicates the schedule for automatically creating a snapshot of the volume for the purpose of preserving the state of the volume.

last-updated: Apr 11, 2022

Mirror Schedule	(Available only for mirror volumes) Indicates the schedule for automatically copying data (that has changed at the file-block level since the last data transfer) directly from the source volume to the mirror volume.
Offload Schedule	Indicates the schedule for automatically offloading data to the tier.

Viewing Volume Information Using the CLI or the REST API

The basic command to retrieve volume information is:

```
maprcli volume info (-name <volume_name> | -path <volume_path>)
```

You must specify either `name` or `path`, but not both. For complete reference for this command, see `volume info` on page 1965.

Viewing the list of Snapshots
Describes how to view the list of snapshots that are present on a cluster, using the Control System or the CLI.

You can view the snapshots on the cluster using the Control System or the CLI.

Viewing All the Snapshots Using the Control System

- The **Snapshots** tab under the **Data** > **Volumes** page displays all the snapshots on the cluster.

  For each snapshot, you can view the following:

Column Name	Column Description
Snapshot Name	The name of the snapshot.
Volume	The volume with which the snapshot is associated.
Created On	The date when the snapshot was created.
Expires On	The date when the snapshot expires.
Reclaim Size	TBD

  You can select one or more snapshots to:

  - Preserve
  - Remove

Viewing the Snapshots Associated with a Volume Using the Control System

- Log in to the Control System and go to the **Snapshots** tab in the volume information page.

  The list of snapshots associated with the volume displays in this tab. For each snapshot, the pane displays the following:

Column Name	Column Description
Snapshot Name	The name of the snapshot.

last-updated: Apr 11, 2022

Column Name	Column Description
Volume	The volume with which the snapshot is associated.
Created On	The date when the snapshot was created.
Expires On	The date when the snapshot expires.
Reclaim Size	TBD

You can create a snapshot of the volume or select one or more snapshots to:

- Preserve
- Remove

Viewing Snapshots Using the CLI or REST API

The basic command to retrieve a list of snapshots is:

```
maprcli volume snapshot list
```

For complete reference information, see `volume snapshot list` on page 2030.

Removing Volumes
Explains how to remove a volume using either the Control System or the CLI.

Note: When you remove a volume enabled for:

- Cold tiering, the directory/folder for the volume in the metadata volume associated with the tier is also removed asynchronously.
- Warm tiering, the directory/folder for the volume in the metadata volume associated with the tier and the erasure-coded volume are also removed asynchronously.

Removing Volumes Using the Control System

To remove one or more volumes, in the **Summary** tab under **Data** > **Volumes**:

Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

1. Select the volumes to remove from the list of volumes in the **Volumes** pane.

   Note: Alternatively, you can click the name of the volume to traverse to the volume details page.

2. Click **Remove Volume(s)** from the **Actions** drop-down menu.
   The **Remove Volume(s)** confirmation dialog displays.

3. Choose one of the following:

   - **Remove only if there are no dependent artifacts** — Specifies whether to check for dependencies before removing the volume. If the volume has dependencies, such as associated snapshots, the volume will not be deleted, but an alarm will be raised.

     Note: Volume will not be deleted if mirroring is in progress as the mirror volume is synchronized from a hidden internal snapshot of the source volume.

   - **Force remove even if there are dependent artifacts** — Indicates that volume must be removed forcefully, even if the volume has dependencies.

last-updated: Apr 11, 2022

4. Click **Remove Volume**.
   The selected volumes are removed.

Removing Volumes Using the CLI or REST API

The basic command to remove a volume is:

```
maprcli volume remove -name <volume name>
```

For complete reference information, see `volume remove` on page 2026.

Note: When a volume is removed, the data present in that volume is not purged from the filesystem immediately. The following parameters control when the deleted volumes are purged.

- `cldb.purge.delay.hours` — Time to wait (in hours) to trigger purge of any volume after CLDB becomes primary. The default value is 1 hour.
- `cldb.volumes.purge.frequency` — The frequency (in minutes) for purging the data of deleted volumes. The default value is 60 minutes.

  Use the `maprcli` command to set these parameters. For example:

  ```
  maprcli config save -values {"cldb.purge.delay.hours":"2"}
  maprcli config save -values {"cldb.volumes.purge.frequency":"120"}
  ```

Modifying Multiple Volumes

Explains how to modify volumes using either the Control System or the CLI.

Modifying Volumes Using the Control System

To edit multiple volumes, in the **Summary** tab under **Data** > **Volumes**:

Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

1. Select the volumes to edit in the **Volumes** pane and click **Edit Volume(s)** from the **Actions** drop-down menu.
   The **Edit Volumes** page displays. You can edit certain volume properties, authorization settings, usage settings, and schedule settings for the selected volumes.
2. Modify one or more of the following settings.

   Note: If the selected volumes share the same settings, the fields are pre-populated with the current value. If the selected volumes contain different settings, only the fields that have been set in all the selected volumes display.

last-updated: Apr 11, 2022

Properties

SETTINGS AND AUDITING	**Accountable Entity**	The *entity* accountable for this volume's usage.
	Collect Metrics	Specifies whether (**Yes**) or not (**No**) to enable metrics collection for this volume. For more information, see Collecting Volume Metrics on page 1300 and Enabling Volume Metric Collection on page 1301.
	Volume Access	Specifies whether the volume is read-only or a read/write volume. By default, a standard volume is created with read/write access. A mirror volume can only be a read-only volume.
	Enable Auditing	Select one of the following: • Disable auditing • Enable auditing for volume so that auditing can selectively be enabled for directories, files, tables, and streams in the volume • Enable auditing of operations on all files, tables, streams in the volume whether or not auditing is enabled for files, tables, and streams in the volume.

REPLICATION (HOT)	**Replication**	Toggle Minimum Replication and set the desired and minimum number of copies of the volumes.
	Name Container Replication	Set the desired and minimum number of copies of the name container associated with the volumes.

Authorization

ADMINISTRATIVE CONTROLS	Users and groups that have permissions to perform administrative operations on this volume.

Usage

Volume Advisory Quota	Allocated disk space, which when exceeded raises an alarm but does not prevent writes.
Volume Hard Quota	Allocated disk space, which when exceeded prevents further writes.

Schedule

Snapshot Schedule	Schedule for creating snapshots of this volume.
Mirror Schedule	Schedule for running mirroring operation for the volumes.  **Note:** This is available only if all the selected volumes are mirror volumes.

 **Note:** To undo a change to a specific setting, click the associated . This action will revert the value to the current setting.

3. Click **Save Changes** for the changes to take effect.

Modifying Volumes Using the CLI or REST API

The basic command to modify a volume is:

```
maprcli volume modify -name <volume name>
```

For complete reference information including supported options, see `volume modify` on page 2005.

last-updated: Apr 11, 2022

Modifying a Volume
Describes how to edit a volume using either the Control System or the CLI.
Modifying a Volume Using the Control System

To modify a volume:

1. Log in to the Control System and go to the volume information page.
2. Click **Edit Volume** to display the **Edit Volume** page.
3. Modify one or more of the following general, auditing, and replication settings:

Standard Volume

SETTINGS AND AUDITING	**Accountable Entity**	The *entity* accountable for the volume's usage.
	Collect Metrics	Whether (**Yes**) or not (**No**) to enable metrics collection for the volume.
	Volume Access	Whether volume is a read-only or read/write volume.
	Enable Auditing	Select one of the following: • Disable auditing • Enable auditing for volume so that auditing can be enabled by directories, files, tables, and streams in the volume • Enable auditing of operations on all files, tables, streams in the volume whether or not auditing is enabled by files, tables, and streams in the volume.
	Coalesce Interval	Interval of time used when logging multiple instance of an operation on a node from a client IP, if auditing is enabled.
	Data on Wire Encryption	Whether (**Yes**) or not (**No**) to encrypt data in the volume during transmission. This is not supported on insecure clusters.

last-updated: Apr 11, 2022

HOT (REPLICATION)	**Topology**	The location of the volume in the cluster rack topology.
	Replication	Desired and minimum number of copies of the volumes.
	Name Container Replication	Desired and minimum number of copies of the name container associated with the volumes.

Mirror Volume

SETTINGS AND AUDITING	**Source Cluster Name**	The name of the cluster on which the source volume exists.
	Source Volume Name	The name of the source volume to mirror.
	Accountable Entity	The *entity* accountable for the volume's usage.
	Collect Metrics	Whether (**Yes**) or not (**No**) to enable metrics collection for the volume.
	Enable Auditing	Select one of the following: • Disable auditing • Enable auditing for volume so that auditing can be enabled by directories, files, tables, and streams in the volume • Enable auditing of operations on all files, tables, streams in the volume whether or not auditing is enabled by files, tables, and streams in the volume.
	Coalesce Interval	The interval of time to use when logging multiple instance of an operation on a node from a client IP, if auditing is enabled.
	Data on Wire Encryption	Whether (**Yes**) or not (**No**) to encrypt data in the volume during transmission. This is not

last-updated: Apr 11, 2022

REPLICATION (HOT)	**Topology**	The location of this volume in the cluster rack topology.
	Replication	Desired and minimum number of copies of the volume.
	Name Container Replication	Desired and minimum number of copies of the name container associated with the volume.

4. Set new (if **Data Tier** was enabled (**On**) during volume creation) or modify existing **DATA TIERING** settings.

Erasure Coding (Warm)

For offloading data to a warm tier, set new or modify existing values for the following properties.

Topology	The topology of the erasure coded volume from the drop-down list only if it is already not set.
Storage Policy	The rule (or criteria) for automatically offloading data in this volume. You can click: • **Browse** to select an existing rule. • **Create** to create a new rule for offloading data. See Creating a Storage Tier Policy Using the Control System on page 972 for more information.
Erasure Coding	The erasure coding scheme, which is the number of data chunks and number of parity chunks, only if it is already not set; you cannot modify existing scheme. See Erasure Coding Scheme for Data Protection and Recovery on page 926 for more information.

Remote Archiving (Cold)

For offloading data to a low cost storage alternative on the cloud, set new or modify existing values for the following properties.

Remote Target	The location to offload data to only if it is already not set; you cannot modify if it is already set. You can click: • **Browse** to select an existing tier. • **Create** to create a new tier. See Creating a Cold Tier Using the Control System on page 959 for more information.
Storage Policy	The rule (or criteria) for offloading data in this volume. You can click: • **Browse** to select an existing rule. • **Create** to create a new rule for offloading data. See Creating a Storage Tier Policy Using the Control System on page 972 for more information.
Retention Duration after Recall	The number of days to retain data recalled from the tier to the MapR cluster. Once the number of days is reached, recalled data on the MapR cluster is purged (if there are no changes) or offloaded (if there are changes).
Tier Encryption	The setting to enable (**Yes**) or disable (**No**) encryption of data on the tier only if it is already not set.

5. Make changes as needed to volume access.

ADMINISTRATIVE CONTROLS	Users and groups that have permissions to perform administrative operations on this volume. To: • Modify access to existing users, select or deselect permissions. • Grant access to another user, click **Add Another**, enter the user name, and select permissions to grant to the user.

last-updated: Apr 11, 2022

USER ACCESS CONTROLS	Specifies public or the users, groups, and/or roles that have and/or do not have access to read from and/or write to this volume. To grant or block access to users, groups, and/or roles, from the: • Basic settings, select the type — public, (OR) user, group, or role — from the drop-down menu, specify the name of the user, group, or role, and select one or more checkbox to grant permissions. **Tip:** Click to create a copy of the associated access control setting. Click to remove the associated access control expression. To add *ACE*s for another user, group, or role, click **Add Another** and repeat this step. • Advanced settings, specify public (p) or user (u), group (g), and/or role (r) who have or do not have the type of access using the following boolean expressions and subexpressions: • ! — Negation operator. • & — AND operation. • | — OR operation. Use ( ), parentheses, for subexpressions. **Note:** You *cannot* specify user, group, or role individually if access is granted to all users (public). Alternatively, click associated with the type of access to use the **Access Control Expression** window to define access for public or users, group, and/or role. See Defining ACEs on page 1473 for more information. **Note:** If you switch from **Basic** to **Advanced**, the basic settings, if any, are carried over to the advanced settings. If you switch from **Advanced** to **Basic**, all the settings are lost because the subexpressions and AND (&) and negation (!) operations that are supported by advanced settings are not supported in the basic settings.

6. (Optional) Modify volume quota as needed.

Volume Advisory Quota	Allocated disk space, which when exceeded raises an alarm but does not prevent writes.
Volume Hard Quota	Allocated disk space, which when exceeded prevents further writes.

7. (Optional) Add or modify schedule(s) as needed.

Standard Volume

Snapshot Schedule	Schedule for creating snapshots of the volume.

last-updated: Apr 11, 2022

Tier Offload Schedule	Schedule for offloading data to the storage tier. This is available only if volume is enabled for storage tiering.

Mirror Volume

Snapshot Schedule	Schedule for creating snapshots of this volume.
Mirror Schedule	Schedule for automatically synchronizing (mirroring) this volume with the source volume.
Tier Offload Schedule	Schedule for offloading data to the storage tier. This is available only if volume is enabled for storage tiering.

8. Click **Save Changes** for the changes to take effect.

Modifying a Volume Using the CLI or REST API

CLI

The basic command to modify a volume is:

```
maprcli volume modify -name <volume
name>
```

REST

Send a request of type POST. For example:

```
curl -k -X
POST 'https://<hostname>:8443/
rest/volume/modify?name=<volume
name>' --user mapr:mapr
```

For the complete list of editable parameters, see `volume modify` on page 2005.

Renaming a Volume

Describes how to rename a volume using the Control System, CLI or the REST API.

If you rename a volume, you must unmount and re-mount the volume to allow applications and/or users to continue accessing the volume.

Renaming a Volume Using the Control System

1. Log in to the Control System and go the volume information page.
2. Click **Rename Volume** from the **Select Action** drop-down menu to display the **Rename Volume** window.
3. Enter the new name for the volume in the **New Volume Name** field and click **Save Changes** for the changes to take effect.

   Note: For tiering-enabled volumes, volume name cannot exceed ninety-eight characters.

last-updated: Apr 11, 2022

Renaming a Volume Using the CLI and REST API

CLI

The basic command to rename a volume is:

```
maprcli volume rename -name
<oldName> -newname <newName>
```

REST

Send a request of type POST. For example:

```
curl -k -X POST 'https://<host>:8443/
rest/volume/rename?
name=<oldName>&newname=<newName>' --us
er mapr:mapr
```

See `volume rename` on page 2026 for more information.

Specifying Volume Inheritance Using the CLI

Lists volume properties that can be inherited from other volumes.

Volumes can be mounted using the web console, the `maprcli` commands (volume create or volume mount), or the REST commands. If the mount point is specified while creating a volume, new volumes can inherit properties from the parent volume. Mirror volumes can also inherit properties from the source volume of the mirror.

Volume inheritance is a convenience that can only be used during volume creation.

The maprcli volume modify command can be used to change the volume inheritance settings of a volume. That is, you can toggle the flag (associated with `allowgrant`) that indicates whether a volume, as a parent volume, wants its properties to be inherited by default or not. When creating and mounting a volume, the location of the mount point can be specified using the `path` parameter. The volume that is last in the `path` parameter is referred to as the parent volume. (The parent volume is the volume on which the volume link is created.)

Inheritance applies during volume creation only. If the settings in the parent volume is modified after the child volumes are created, these modified properties do not propagate to the child volumes.

Inheritance

The following table shows the list of inheritable parameters that are (Yes) and are not (No) inherited by a:

- Mirror volume from the source volume on the same cluster
- Mirror volume from the source volume on a different cluster

Note: All (non-mirror) volumes inherit all the inheritable properties from the parent volume. For more information on the properties, refer to `volume create` parameters.

Inheritable Properties (which are inherited by non-mirror volumes by default)	Inherited by Mirror Volume on the same cluster as the source volume?	Inherited by Mirror Volume on a different cluster from the source volume?
`advisoryquota`	Yes	Yes
`ae`	Yes	No
`aetype`	Yes	No
`allowgrant`	Yes	Yes
`allowinherit`	Yes	Yes
`auditenabled`	Yes	Yes

last-updated: Apr 11, 2022

Inheritable Properties (which are inherited by non-mirror volumes by default)	Inherited by Mirror Volume on the same cluster as the source volume?	Inherited by Mirror Volume on a different cluster from the source volume?
coalesce	Yes	Yes
dare	Yes	Yes[1], No[2]
dataauditops	Yes	Yes
dbindexlagsecalarmthresh	Yes	Yes
dbrepllagsecalarmthresh	Yes	Yes
ecscheme	Yes	No
ectopology	Yes	No
group	Yes	Yes
inherit	Yes	Yes
localvolumehost	No	No
localvolumeport	No	No
maxinodesalarmthreshold	Yes	Yes
minreplication	Yes	Yes
mirrorschedule	Yes	No
mirrorthrottle	Yes	Yes
nsminreplication	Yes	Yes
nsreplication	Yes	Yes
ofloadschedule	Yes	No
quota	Yes	Yes
readonly	Yes	Yes
recallexpirytime	Yes	No
replication	Yes	Yes
replicationtype	Yes	Yes
rereplicationtimeoutsec	Yes	Yes
rootdirperms	Yes	Yes
schedule	Yes[3]	No
source	Yes	Yes
tierencryption	Yes	No
tieringenable	Yes	No
tieringrule	Yes	No
tierkey	Yes	No
tiername[4]	Yes	No
topology	Yes	No
type	Yes	Yes
user	Yes	Yes

last-updated: Apr 11, 2022

Inheritable Properties (which are inherited by non-mirror volumes by default)	Inherited by Mirror Volume on the same cluster as the source volume?	Inherited by Mirror Volume on a different cluster from the source volume?
`wiresecurityenabled`	Yes	Yes

- [1] If destination cluster is also enabled for data-at-rest encryption, `dare` setting is inherited by the mirror volume on the destination cluster.
- [2] If destination cluster is not enabled for data-at-rest encryption, `dare` setting is not inherited by the mirror volume on the destination cluster.
- [3] If `schedule` keyword is specified with the `skipinherit` parameter, schedule(s) are not inherited while inheriting volume properties from the source volume.
- [4] If `tiername` keyword is specified with the `skipinherit` parameter:
  - The tiering properties are not inherited by the mirror volume while inheriting volume properties from the tiering-enabled source volume.
  - For volumes enabled for warm-tier, the backend erasure-coded volume is not created.

Setting Data ACEs
Describes how to set ACE expressions using both the GUI and the CLI.

To set data *ACE* using the **Add Access Permission** window in the MapR Control System:

1. Specify the entities to set permissions for by doing one of the following:
   - Move the slider associated with **Public** to **Yes** to grant access to all users or to **No** to set permissions for individual users, groups, and/or roles.
   - Specify the users, groups, and/or roles to set permissions for in the associated fields.
   - Select the **Custom ACE** checkbox and enter the access control expression in the field.
2. Click **Add** to set permissions for all or for the specified users, groups, and/or roles.
3. Select the permissions to grant the specified users, groups, and/or roles from the **Permissions** column associated with the entities.
4. Click **Save Changes** to save the *ACE* settings.

Setting Whole Volume ACEs Using the CLI

See Setting Whole Volume ACEs on page 1459.
Setting Table ACEs Using the CLI

See Enabling Table and Stream Authorizations with ACEs on page 1461.
Setting Stream ACEs Using the CLI

See Enabling Table and Stream Authorizations with ACEs on page 1461.

Changing or Setting Mount Information for a Volume
Describes how to set the mount path for a volume using either the Control System, the CLI, or the REST API.

You can set or change volume mount settings using the Control System, the CLI (`volume create` or `volume mount`), or the REST commands. To mount or unmount volumes under a directory, the user must have read/write permissions on the directory.

Changing or Setting Mount Information for a Volume Using the Control System

1. Log in to the Control System and go to the volume information page.
2. Select **Change Mount Information** from the **Select Action** drop-down menu.
   The **Change Mount Information** dialog displays.
3. Make the necessary changes.
   a) Specify whether (**Yes**) or not (**No**) to mount the volume.
   b) Enter the path in the **Mount Path** field.
   The path must be relative to `/` and cannot be in the form of a global namespace path (for example, `/mapr/<cluster-name>/`).

   Restriction:

   The path should contain only the following characters:

   ```
   A-Z a-z 0-9 _ - .
   ```
4. Click **Save Changes** for the changes to take effect.

Note: After changing the mount point, run `maprcli volume fixmountpath` command to notify CLDB of the change in the volume mount path.

Changing or Setting Mount Information Using the CLI or REST API

The basic command to mount the volume is:

```
maprcli volume mount -name <volume name> -path <mount path>
```

Restriction: The volume name and the path should contain only the following characters:

```
A-Z a-z 0-9 _ - .
```

For complete reference information including all available options, see `volume mount` on page 2020.

Mounting one or more Volumes

Explains how to set or change mount settings for volumes using the Control System or the CLI.

To mount volumes under a directory, you must have read/write permissions on the directory.

Changing or Setting Mount Information Using the Control System

To mount one or more volumes, in the **Summary** tab under **Data** > **Volumes**:

1. Select the volumes to mount from the list of volumes in the **Volumes** pane.
2. Select **Mount Volume(s)** from the **Actions** drop-down menu.
   The **Mount Volume(s)** confirmation dialog displays.

   Note: Only unmounted volumes with mount paths can be mounted. Volumes with no mount paths and volumes that are currently mounted, if selected, cannot be mounted.
3. Verify list of selected volumes.
   If necessary, click ✕ to remove a volume from the list of volumes to mount.
4. Click **Mount Volumes** to mount the selected volumes.

last-updated: Apr 11, 2022

Changing or Setting Mount Information Using the CLI or REST API

The basic command to mount the volume is:

```
maprcli volume mount -name <volume name> -path <mount path>
```

For complete reference information including all available options, see `volume mount` on page 2020.

Unmounting one or more Volumes
Describes how to unmount a volume using either the Control System or the CLI.

You can unmount multiple volumes using the Control System or the CLI. To unmount volumes under a directory, you must have read/write permissions on the directory.

Unmounting one or more Volumes Using the Control System

To unmount one or more volumes, in the **Summary** tab under **Data** > **Volumes**:

Note: When running on a Kubernetes cluster, the **Summary** tab is in the **Volumes** page under the **Volumes** menu.

1. Select the volumes to unmount from the list of volumes in the **Volumes** pane.
2. Select **Unmount Volume** from the **Actions** drop-down menu.
   The **Unmount Volume** confirmation dialog displays.

   Note: Only mounted volumes can be unmounted. Volumes that are currently not mounted, if selected, cannot be unmounted.
3. Verify list of selected volumes.
4. Click **Unmount Volumes** to unmount the selected volumes.

Unmounting one or more Volumes Using the CLI or REST API

The basic command to unmount the volume is:

```
maprcli volume unmount -name <volume name>
```

For complete reference information including all available options, see `volume unmount` on page 2050.

Mounting a Tenant Volume
Describes the steps to mount and access a tenant volume.

After the tenant volume is created on the cluster (for a multi-tenant environment), access the tenant volume on the tenant host with the following steps:

1. Log in to the tenant host as the tenant admin (`root`) and verify that a valid tenant ticket is available on the tenant host.
   For example, run the following command:

   ```
   ~tenantAdmin@tenantHost: maprlogin print -ticketfile /user/tenantAdmin/
   tenant_sample_ticket.txt
   Opening keyfile /user/tenantAdmin/tenant_sample_ticket.txt
   cHost: user = sampleTenant, created = 'Mon Jul 11 07:14:53 UTC 2016',
   expires = 'Mon Jul 11 07:14:53 UTC 12016', RenewalTill = 'Mon Jul 11
   07:14:53 UTC 12016',
   uid = 500, gids = 500, 42, CanImpersonate = true, tenant = true
   ```
2. Perform one of the following:

- Set up the FUSE-based POSIX client configuration parameters (see Configuring MapR FUSE-based POSIX Client for Tenant Environment on page 1253) and mount the volume (see Mounting the MapR File System on page 1257).
- Set up an alias for NFS exports to export the tenant volume path (see Setting Up Aliases for NFS Exports on page 1184) and mount the volume for loopbacknfs service (see Starting the mapr-loopbacknfs Service to Access a Cluster).

3. Create accessible directories within the provisioned space for the tenant users.

Unmounting a Tenant Volume
Explains how unmount a tenant volume using the CLI.

To unmount a tenant volume and:

- Kill the FUSE process, run the following command:

```
service mapr-posix-client-* stop
```

When you run the command, replace * with basic or platinum, which corresponds with the POSIX client package that is installed on the system.

Note: For more information, see Unmounting the FUSE Mount.

- Stop the loopbacknfs service, run the following command:

```
service mapr-loopbacknfs stop
```

Note: For more information, see Managing the mapr-loopbacknfs Service on page 1236

Changing Volume Type

You can convert a standard volume to a mirror volume and promote a mirror volume to a standard volume.

A standard volume with one or more associated mirror volumes can be converted to a mirror volume that mirrors one of the its associated mirror volumes. The mirror volume that the converted standard volume is set to mirror must then be promoted to a standard volume. The converted standard volume then becomes a read-only copy of the promoted mirror volume.

Note: Standard volumes that do not have one or more associated mirror volumes cannot be converted to mirror volumes.

A mirror volume is a read-only physical copy of a standard volume. In general, mirror volumes are created for the purpose of preventing or minimizing data loss. Mirror volumes are also used to improve performance or to make copies of data for use in other clusters without impacting production. Mirror volumes can be changed to read-write volumes by converting the mirror volumes to standard volumes.

The following topics include procedures for converting a standard volume to a mirror volume and vice versa.

Changing a Standard Volume to a Mirror Volume
Describes how to convert a standard volume to a mirror volume.

You can convert a standard volume to a mirror volume and set it up to mirror one of its associated mirror volumes using the Control System, the CLI, or the REST API.

Changing a Standard Volume to a Mirror Volume Using the Control System

1. Log in to the Control System and go to the Viewing Volume Details on page 883 page for the standard volume.

2. Select **Make Mirror Volumes** from the **Select Action** drop-down menu.
   The **Mirror Volume** confirmation dialog displays.
3. Select the:
   - Name of the source cluster where the associated mirror volume that the converted volume will mirror, exists.
   - Name of the source volume that the converted volume will mirror, from the list of associated mirror volumes.

   The standard volume, when converted, can only be a mirror of one of its associated mirror volumes.
4. Click **Save Changes** for the changes to take effect.

   Note: It might take some time (approximately 10 minutes or so) to convert a standard volume to a mirror volume. You need to wait until the operation is complete before performing other actions.

After converting the standard volume to a mirror volume:

1. Convert the source (mirror) volume to a standard (read/write) volume to prevent a deadlock and to allow writes to continue on the volume.
2. Associate a mirroring schedule with this volume to ensure that data on this volume is in sync with the source volume.

Changing a Standard Volume to a Mirror Volume Using the CLI or the REST API

CLI

To convert a standard volume to a mirror volume from the CLI, run the `maprcli volume modify` command with the `-type` option value set to `mirror`.

```
maprcli volume modify -name <volume
name> -type mirror
```

REST

To convert a standard volume to a mirror volume, send a request of type POST. For example:

```
curl -k -X POST 'https://
<hostname>:8443/rest/volume/modify?
name=<volName>&type=mirror' --user
mapr:mapr
```

For more information, see `volume modify` on page 2005.

Note: It might take some time (approximately 10 minutes or so) to convert a standard volume to a mirror volume. You need to wait until the operation is complete before performing other actions.

After converting the standard volume to a mirror volume:

1. Convert the source (mirror) volume to a standard (read/write) volume to prevent a deadlock and to allow writes to continue on the volume.
2. Associate a mirroring schedule with this volume to ensure that data on this volume is in sync with the source volume.

Changing Mirror Volumes to Standard Volumes
Describes how to convert a mirror volume to a standard volume.

To change read-only mirror volumes to read-write (standard) volumes, you must promote the mirror volumes to standard volumes. After the mirror volume is promoted, the snapshot schedule specified for the mirror is used for the promoted read-write volume and the mirror schedule is disabled. You can promote a mirror volume to a standard volume using the Control System or the CLI.

Note: When you use promotable mirrors, the volumes on the destination cluster must be set up in the same way as on the primary site. This means that volume names are the same and mount points are the same. If a hierarchical mounting structure (such as /A/B) is used on the primary site, the same structure must be recreated once mirror volumes are promoted at the secondary site.

Mirror promotion time is typically negligible, but is dependent on the number of containers in the volume being promoted. For a volume with thousands of containers and a few terabytes of data, it may take a few seconds. For enormous volumes with tens of thousands of containers and hundreds of terabytes of data, promotion could take a few minutes.

After the promotion is complete, a status message is logged in the `cldb.log` file with the time taken. For example:

```
2021-06-27 22:47:00,563 INFO  VolumeMirrorInfo [VolumeMirrorThread0]:
    Volume conversion successfully completed for volume
v100k.m@c.228toMirror false.
    Time taken : 142395ms
```

Changing Multiple Mirror Volumes to Standard Volumes Using the Control System

To change one or more mirror volumes to standard volumes, in the **Summary** tab under **Data** > **Volumes**:

Note: The **Summary** tab is under the **Volumes** menu in the Kubernetes version of the Control System.

1. Select the mirror volumes to promote from the list of volumes in **Volumes** pane.
2. Select **Change to Standard Volume(s)** from the **Actions** drop-down menu.
   The **Change to Standard Volume(s)** confirmation dialog displays.
3. Review the list of mirror volumes to promote and click **Change Volume(s)** to change the mirror volumes to standard volumes.

Note: Mirror volumes that are promoted to standard (read-write) volumes are not available for write operations until they are mounted explicitly. For more information, see Handling Mount Points in Promoted Mirror Volumes on page 908.

Changing a Mirror Volume to a Standard Volume Using the Control System

1. Log in to the Control System and go to the volume information page for the mirror volume.
2. Select **Change to Standard Volume(s)** from the **Select Action** drop-down menu.
   The **Change to Standard Volume(s)** confirmation dialog displays.
3. Click **Change Volume(s)** to change the mirror volume to a standard volume.

Note: Mirror volumes that are promoted to standard (read-write) volumes are not available for write operations until they are mounted explicitly. For more information, see Handling Mount Points in Promoted Mirror Volumes on page 908.

Changing a Mirror Volume to a Standard Volume Using the CLI or REST API

For changing a mirror volume to a standard (read/write) volume using the CLI, run the `maprcli volume modify` command with the `-type` option value set to `rw` on the cluster where the mirror volume resides

last-updated: Apr 11, 2022

and specify the name of the mirror volume that is being promoted. In the following example, the mirror volume is named volA:

```
Cluster2> maprcli volume modify -name volA -type rw
```

For more information, see `volume modify` on page 2005.

Note: Mirror volumes that are promoted to standard (read-write) volumes are not available for write operations until they are mounted explicitly. For more information, see Handling Mount Points in Promoted Mirror Volumes on page 908.

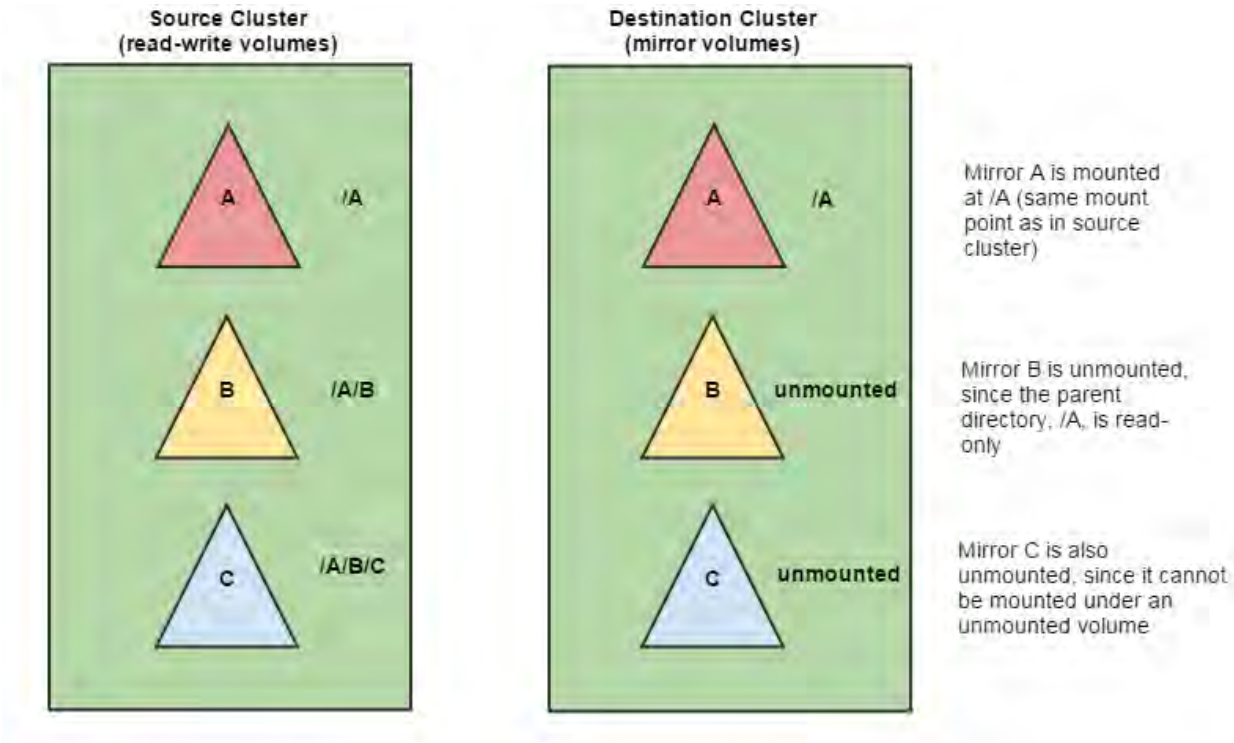
Handling Mount Points in Promoted Mirror Volumes
Explains how to use mount points in promoted mirror volumes.

After you promote read-only mirror volumes to read-write standard volumes, you must re-establish the mount points that were set up in the source cluster. To understand the steps in this process, consider the following scenario:

A source cluster has volumes A, B, and C, which are mounted at /A, /A/B, and /A/B/C respectively. Each source volume is mirrored to a volume in another cluster (the destination cluster). The names of the corresponding mirror volumes are also A, B, and C.

Note: When you use promotable mirrors, the volumes on the destination cluster must be set up in the same way as on the primary site. This means that volume names are the same and mount points are the same. If a hierarchical mounting structure (such as /A/B) is used on the primary site, the same structure must be recreated once mirror volumes are promoted at the secondary site.

Mirror volume A is mounted at /A, but since the mirror is read-only, no mount point can be created beneath it for mirror B or mirror C.



Warning: Mirror volumes that are promoted to standard (read-write) volumes are not available for write operations until they are mounted explicitly.

last-updated: Apr 11, 2022

Now suppose that all three mirror volumes are promoted to read-write volumes. Before any data can be written to these volumes, the volume links must be removed and the volumes must be remounted. The commands for each step are as follows:

1. Promote A, B, and C to read-write volumes.

```
Cluster2> maprcli volume modify -name A -type rw
Cluster2> maprcli volume modify -name B -type rw
Cluster2> maprcli volume modify -name C -type rw
```

   To promote using the Control System, see Changing Mirror Volumes to Standard Volumes on page 906.

2. Remove the volume links located at /A/B and /A/B/C. Since mirror A was already mounted, its volume links do not need to be removed.

```
maprcli volume link remove -path /A/B
maprcli volume link remove -path /A/B/C
```

3. Mount the promoted read-write volumes B and C at the same mount points used in the primary (source) cluster, in order to maintain an exact replica in the destination cluster.

```
Cluster2> maprcli volume mount -name B -path /A/B
Cluster2> maprcli volume mount -name C -path /A/B/C
```

   To mount using the Control System, see Mounting one or more Volumes on page 903.

Now the promoted volumes are accessible for write operations.

Selecting a Replication Type for High Availability
Describes replication types for high-availability clusters, and the tradeoffs of using them.

MapR volumes, stored as pieces called containers, are replicated, typically three times, on separate nodes to protect data and provide uninterrupted access to data in the event of a node failure. Since all form of data is replicated, in the event of a node failure, after a brief delay while the failure is being detected, clients are simply directed to a replica, which serves as an alternative location for a data object, to continue normally. The latency as a result of the failure being detected can be reduced by adjusting the number of TCP retries. Furthermore, selecting a container replication type that is appropriate for your cluster layout allows for faster replication of container state, which in turn allows for retrieval of the most current data in the event of a node failure.

MapR supports two types of container replication -- high-throughput or cascading replication, where volumes are replicated sequentially on intermediate and tail containers and low-latency or star replication, where volumes are replicated on multiple containers in parallel.

Note: For more information, see How MapR File System and Associated Services Work.

The tradeoffs between the replication types is one of latency and throughput. While the low-latency replication delivers relatively lower throughput than the high-throughput replication, the high-throughput replication suffers from relatively higher latencies than the low-latency replication. Another advantage of low-latency replication is that since the primary container is connected to all other replica containers, there is no need to failover a replica container in the event of a failure thus reducing the duration of recovery. However, with high-throughput replication, in the event of a failure of an intermediate container, clients may experience increased latency while CLDB, after detecting the failure, attempts to update the replication chain by making the next or tail container (whichever comes immediately after the failed container) as the next container in the chain.

last-updated: Apr 11, 2022

Converting Volume Replication Type (Low Latency to High Throughput) Using the CLI
Lists the process to convert a volume's replication type using the CLI.

A high throughput replication type allows for volumes to be replicated sequentially on intermediate and tail containers from a primary container.

To convert from a low-latency to a high throughput-replication type:

1. Change the permissions on the volume from read-write to read only.
   For example:

   ```
   maprcli volume modify -name mvol1,mvol2 -readonly true
   ```

   Wait for the running operations to complete before proceeding to the next step.

2. Convert the volume from `low_latency` replication type to `high_throughput` replication type using the `maprcli` command.
   For example:

   ```
   maprcli volume modify -name mvol1,mvol2 -replicationtype high_throughput
   ```

   Wait till replication type conversion is complete and the first container of the volume acquires a primary container. If necessary, run the following command to see if replication type has been converted:

   ```
   maprcli volume list -columns ReplTypeConversionInProgress,volumename
   ```

   If the conversion is complete, the `ReplTypeConversionInProgress` flag will be set to false (`0`). For example, the `0` in the `ReplTypeConversionInProgress` column in the following sample output indicates successful conversion of corresponding volume in the volumename column:

   ```
   maprcli volume list -columns ReplTypeConversionInProgress,volumename
   ReplTypeConversionInProgress
   volumename

   0
   mapr.apps

   0
   mapr.cldb.internal

   0
   mapr.cluster.root

   0
   mapr.configuration

   0
   mapr.hbase


   0
   mapr.metrics

   0
   mapr.node-20.lab.local.audit

   0
   mapr.node-20.lab.local.logs

   0
   mapr.node-20.lab.local.mapred

   0
   mapr.node-20.lab.local.metrics

   0
   mapr.node-20.local.audit

   0
   mapr.node-20.local.logs

   0
   mapr.node-20.local.metrics

   0
   mapr.node-21.lab.local.audit
   ```


last-updated: Apr 11, 2022

```
0
mapr.node-21.lab.local.logs

0
mapr.node-21.lab.local.mapred

0
mapr.node-21.lab.local.metrics

0
mapr.node-22.lab.local.audit

0
mapr.node-22.lab.local.logs

0
mapr.node-22.lab.local.mapred

0
mapr.node-22.lab.local.metrics

0
mapr.node-23.lab.local.audit

0
mapr.node-23.lab.local.logs

0
mapr.node-23.lab.local.mapred

0
mapr.node-23.lab.local.metrics


0
mapr.opt

0
mapr.resourcemanager.volume

0
mapr.tmp

0
mapr.var

0
mvol1

0
mvol2

0
mvol3

0
users

0                               vol3
```

3. Reset the permissions on the volume to read-write.
   For example, to reset, run the following command:

```
maprcli volume modify -name vol1,vol2 -readonly false
```

Converting Volume Replication Type (High Throughput to Low Latency) Using the CLI
Lists the process to convert the replication type of a volume using the CLI.

A low latency replication type allows for volumes to be replicated on multiple containers (in parallel) from the primary container.

Note: Contact MapR support before converting volumes to the `low_latency` replication type.

To convert from a high-throughput to a low-latency replication type:

1. Change the permissions on the volume from read-write to read only.
   For example:

```
maprcli volume modify -name mvol1,mvol2 -readonly true
```

   Wait for the running operations to complete before proceeding to the next step.

2. Convert the volume from `high_throughput` replication type to `low_latency` replication type using the `maprcli` command.
   For example:

```
maprcli volume modify -name mvol1,mvol2 -replicationtype low_latency
```

   Wait till replication type conversion is complete and all the containers of the volume acquire a primary container. If necessary, run the following command to see if replication type has been converted:

```
maprcli volume list -columns ReplTypeConversionInProgress,volumename
```

   If the conversion is complete, the `ReplTypeConversionInProgress` flag will be set to false (`0`). For example, the `0` in the `ReplTypeConversionInProgress` column in the following sample output indicates successful conversion of corresponding volume in the volumename column:

```
maprcli volume list -columns ReplTypeConversionInProgress,volumename
                        volumename
ReplTypeConversionInProgress
                        mapr.apps                     0
                        mapr.cldb.internal            0
                        mapr.cluster.root             0
                        mapr.configuration            0
                        mapr.doc22.lab.local.audit    0
                        mapr.doc22.lab.local.logs     0
                        mapr.doc22.lab.local.mapred   0
                        mapr.doc22.lab.local.metrics  0
                        mapr.doc23.lab.local.audit    0
                        mapr.doc23.lab.local.logs     0
                        mapr.doc23.lab.local.mapred   0
                        mapr.doc23.lab.local.metrics  0
                        mapr.hbase                    0
                        mapr.metrics                  0
                        mapr.opt                      0
                        mapr.resourcemanager.volume   0
                        mapr.tmp                      0
                        mapr.var                      0
                        users                         0
```

3. Reset the permissions on the volume to read-write.
   For example, to reset, run the following command:

```
maprcli volume modify -name vol1,vol2 -readonly false
```

Setting Quota for a Volume
Describes how to set disk quotas for a volume using either the Control System, the CLI or the REST API.
Setting Quota for a Volume Using the Control System

To set volume quotas, in the **Summary** tab under **Data** > **Volumes**:

Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

1. Ensure that the **Quota** column is displayed in the **Volumes** pane.
   If necessary, customize the columns to see the **Quota** column.
2. Click the **Set Quota** link associated with the volume for which you want to set quotas to display the **Set Quota** window.
3. Specify the following in the **Set Quota** window:
   a) Hard quota, which raises an alarm when the threshold is reached and prevents further writes.

      Note: When you set a hard quota for a tiering-enabled volume, the quota is the total space allocated for the volume irrespective of the location (cluster or tier) where the volume data is stored. For example, if you allocate 1GB of hard quota for a tiering-enabled volume, writes will fail after you write 1GB of data whether or not the volume data is local (on the cluster) or offloaded (to the tier).

   b) Advisory quota, which raises an alarm when the threshold is reached, but does not prevent further writes.

   Note: Both, advisory and hard, quotas can be expressed in megabytes (MB), gigabytes (GB), which is the default, or terabytes (TB).
4. Click **Save Changes** for the changes to take effect.

Setting Quota for a Volume Using the CLI or the REST API

You can set quotas for a volume when creating a new or modifying an existing volume.

CLI

The basic command to set quota for a volume is:

```
maprcli volume
create -name <volName> -path
<mountPath> -advisoryquota
<advisoryQuota> -quota <hardQuota>
```

```
maprcli volume
modify -name <volName> -advisoryquota
<advisoryQuota> -quota <hardQuota>
```

REST

Send a request of type POST. For example:

```
curl -k -X POST 'https://
<hostname>:8443/rest/volume/create?
name=<volName>&path=<mountPath>&adviso
```

```
ryquota=<advisoryQuota>&quota=<hardQuo
ta>' --user mapr:mapr
```

```
curl -k -X POST 'https://
<hostname>:8443/rest/volume/modify?
name=<volName>&advisoryquota=<advisory
Quota>&quota=<hardQuota>' --user
mapr:mapr
```

For the complete list of required and optional parameters, see `volume create` on page 1931 and `volume modify` on page 2005.

Migrating a Volume off a Node Using the CLI

When you need to migrate a data volume off a particular node, move that node from the `/data` path to the `/decommissioned` path to avoid data under-replication.

- Establish a `/data` topology path to serve as the default topology path for the volumes in that cluster.
- Establish a `/decommissioned` topology path that is not assigned to any volumes.

> **Tip:** It is recommended that CLDB and ZooKeeper nodes are not in the same topology as the data nodes to ensure fast failover of the failed data node in the event of a data node failure.

Since no data volumes are assigned to the `/decommissioned` topology path , standard data replication will migrate the data off that node to other nodes that are still in the `/data` topology path.

You can run the following command to check if a given volume is present on a specified node:

```
maprcli dump volumenodes -volumename <volume> -json | grep <ip:port>
```

Run this command for each non-local volume in your cluster. Once all the data has migrated off the node, you can decommission the node or place it in maintenance mode.

If you need to segregate CLDB data, create a `/cldb` topology node and move the CLDB nodes under `/cldb`. Point the topology for the CLDB volume (`mapr.cldb.internal`) to `/cldb`. See Isolating CLDB Nodes for details.

Note: To move an existing volume to another topology, you must have the Converged Enterprise Edition. Without the Converged Enterprise Edition, when you run the `maprcli volume move` command to move a volume to another topology, the following error message is returned:

```
ERROR (10010) - Volume Move: No license for requested operation
```

Setting Up Volume Topology

Specifies how to use volume topology to place volumes on specific racks, nodes, or groups of nodes.

After you define the node topology for the nodes in your cluster, you can use volume topology to place volumes on specific racks, nodes, or groups of nodes.

This section describes the process of setting up:

- Default volume topology
- Local volume topology
- Custom topology for local volume replicas

last-updated: Apr 11, 2022

Setting Up Volume Topology
Explains how to setup Volume Topology using either the Control System or the CLI.

MapRMapR Data Platform supports data placement control, in which you can place a volume on specific racks, nodes, or groups of nodes by setting its topology to an existing node topology. You can set volume topology using the Control System or with the volume move on page 2021 command.

To move an existing volume to another topology, you must have the Converged Enterprise Edition installed on your system. Without the Converged Enterprise Edition, when you try to move a volume to another topology, the following error message is returned:

```
ERROR (10010) - Volume Move: No license for requested operation
```

Setting Up Volume Topology Using the Control System

You can set up volume topology at the time of volume creation or change the volume topology after volume creation. To:

- Set up volume topology at the time of volume creation, see Creating a Volume on page 864.
- Modify volume topology, see Modifying a Volume on page 892.

Setting Up Volume Topology Using the CLI or REST API

To move a volume to a different topology, run the following command:

```
maprcli volume move -name <volume name> -topology <path>
```

For complete reference information, see volume move on page 2021.

Setting the Topology for Local Volume Replicas
Explains how to set the topology for local volume replicas using the CLI.

The primary copies for containers of local volumes are placed on the local node (specified with parameter `-localhost` in the `volume create` command). The nodes for the replica copies for containers of local volumes are chosen as follows:

1. If a topology is explicitly specified for replicas during `volume create` or `volume edit`, that topology will be used.

   ```
   maprcli volume create -name egLocalVol -path /test/local/volumes/
   examples/sample1 \
   -localvolumehost 10.20.30.40 -topology /rack1/test
   ```

   In the above example, the primaries for the volume are placed on node 10.20.30.40, and replicas will be placed on nodes in the topology /rack1/test.

   The `-topology` parameter is optional, and if it is not specified, CLDB will fall back to 2 or 3 below.

2. If the configuration parameter that specifies a relative path for replicas of local volumes is set, that will be used.

   ```
   maprcli config save -values
   {"cldb.local.volume.topology.trim.index":"-1"}
   ```

   This will trim the topology of the node specified by the localhost parameter and restrict the replicas to the resultant topology.

   By default, this configuration parameter is not set. To set this configuration parameter, see Creating Replicas of Local Volumes in Custom Topology Using the CLI on page 917. If the configuration parameter is not set, CLDB will fall back to 3 below.

3. The default volume topology will be used.

   The default volume topology is the value specified by the configuration parameter `cldb.default.volume.topology`. The default value for this parameter is `/data`. See Setting Default Volume Topology Using the CLI on page 917.

Creating Replicas of Local Volumes in Custom Topology Using the CLI

To set the configuration parameter for placing replicas of volumes in a topology relative to the local node, run the `maprcli config save` command. The value can be a:

- Positive number to indicate the number of paths to keep from the initial root (of the topology path). For example:

```
maprcli config save -values {"cldb.local.volume.topology.trim.index":"1"}
```

- Negative number to indicate the number of paths to skip from the end of the topology path. For example:

```
maprcli config save -values {"cldb.local.volume.topology.trim.index":"-2"}
```

For example, suppose the local volume is created on a node that is under the topology `/data-center1/lab2/rack3/shelf4/10.10.20.30`. To create a local volume where the replicas are restricted to `/data-center1/lab2/rack3` topology, run the following command:

```
maprcli config save -values {"cldb.local.volume.topology.trim.index":"3"}
maprcli volume create -name egLocalVol -path /data-center1/lab2/rack3/
shelf4/10.10.20.30 -localvolumehost 10.10.20.30
```

Alternatively, you can run the following command to specify the path for the volume from the end of the topology path:

```
maprcli config save -values {"cldb.local.volume.topology.trim.index":"-2"}
maprcli volume create -name egLocalVol -path /data-center1/lab2/rack3/
shelf4/10.10.20.30 -localvolumehost 10.20.30.40
```

The replicas for containers of the volume, egLocalVol, will be created on nodes under `/data-center1/lab2/rack3`.

Setting Default Volume Topology Using the CLI

Use the `config save` command to set the default topology for volumes.

By default, new volumes are created with a topology of `/data`. To change the default topology, use the `config save` on page 1586 command to change the `cldb.default.volume.topology` configuration parameter. Example:

```
maprcli config save -values "{\"cldb.default.volume.topology\":\"/data/
rack02\"}"
```

After this command is run, new volumes have the volume topology `/data/rack02` by default, which could be useful to restrict new volume data to a subset of the cluster.

Viewing Active Volume Alarms

Describes how to view volume alarms using the Control System and the CLI.

You can view volume alarms in the Control System and using the CLI.

Viewing Active Volume Alarms in the Control System

- Log in to the Control System and:

last-updated: Apr 11, 2022

- Click **Data** > **Volumes** to view all active volume alarms in the **Active Alarms** pane.

  Note: The **Volumes** page is under the **Volumes** menu on the Kubernetes version of the Control System.

- Go to the **Summary** tab in the volume information page to view the recent and active alarms for the selected volume in the **Alarms** pane.

- Click (in the top navigation bar) and select **Volume Alarms** from the drop-down menu in the **All** alarms pane to view all the active volume alarms.

- Click **Overview** and select **Volume Alarms** from the drop-down menu in the **Active Alarms** pane to view all active volume alarms.

You can:

- View information related to the alarm.
- Dismiss an alarm.
- Mute an alarm.

See Volume Alarms on page 2240 for more information on the volume alarms.

Retrieving Active Volume Alarms Using the CLI or REST API

The basic command to retrieve node alarms is:

```
maprcli alarm list -cluster <cluster name> -type volume
```

For complete reference information, see `alarm list` on page 1545.

Working with Mirror Volumes

The mirroring process transmits the differences between the source volume and the mirror. The initial mirroring operation copies the entire source volume, but subsequent mirroring operations are generally very fast. The following sections describe how to manage the mirroring operation.

Changing the Limit for Concurrent Mirror Operations Using the CLI

The system allows a maximum of 50 concurrent mirroring operations by default. Mirroring operations include both mirroring and promoting from read-only mirrors to read-write standard volumes. The system parameter that controls this limit is `mapr.mirror.concurrent.ops`.

For large-scale mirror operations involving many volumes, a script automates the process. For example, if a script queues 100 volumes for mirroring operations, and the `mapr.mirror.concurrent.ops` limit is set to 50, the mirroring operations start on the first 50 volumes in the queue. As soon as one volume completes, another volume is processed from the queue until all 100 are completed. Since volumes are processed from the queue in first-in first-out (FIFO) order, the script should specify the most critical volumes first.

If you want to process more volumes at a time, you can raise the limit of the `mapr.mirror.concurrent.ops` parameter. To tune this parameter for maximum efficiency, consider the number of containers per volume. A higher number of containers per volume requires a lower limit than a lower number of containers per volume. To raise the limit to 500 for example, run the following command:

```
maprcli config save -values {"mapr.mirror.concurrent.ops":"500"}
```


last-updated: Apr 11, 2022

Pushing Changes to Mirrors Using the CLI

To *push* a mirror means to start pushing data from the source volume to all of its local mirrors. You can push source volume changes out to all mirrors using the `volume mirror push` command, which returns after the data has been pushed.

Moving Large Amounts of Data to a Remote Cluster Using the CLI

You can use the `volume dump create` command to create volume copies for transport on physical media. The `volume dump create` command creates backup files containing the volumes, which can be reconstituted into mirrors at the remote cluster with the volume dump restore command. Associate these mirrors with their source volumes with the `volume modify` command to re-establish synchronization.

Another way to transfer large amounts of data to a remote cluster is to create a small cluster locally and mirror to that local cluster. Then move that cluster to a remote location and enlarge it by adding more nodes.

Disabling Mirror Throttling Using the CLI

By default, mirror throttling is enabled, which means that the server that sends mirror data, restricts itself to about 30% of the available bandwidth, as measured in MapR's internal environment, with the default settings of the following parameters. Mirror throttling is based on the number of outstanding requests on the network, and outstanding I/O requests on disk, and can be tuned using the parameters, `mfs.disk.iothrottle.count`, `mfs.disk.resynciothrottle.factor`, and `mfs.network.resynciothrottle.factor`, in the `mfs.conf` file. When other processes need more network bandwidth, the server throttles back to slow down the rate of data transfer.

By disabling throttling, the mirror operation completes faster. To disable mirror throttling from the command line, run the `volume modify` command on the *source* volume and set the `-mirrorthrottle` option to `false`, as shown in this example:

```
/opt/mapr/bin/maprcli volume modify -name volA -mirrorthrottle false
```

This command disables throttling for all mirror volumes whose source is volA. Note that the `-mirrorthrottle` option only applies to volumes that have mirrors.

Recovering Volumes from Mirrors Using the CLI

Lists the process to recover mirror volumes, using the CLI.

1. Use the `volume dump create` command to create a full volume dump for each mirror volume you want to restore. Example: `maprcli volume create -e statefile1 -dumpfile fulldump1 -name volume@cluster`
2. Transport the volume dump to the rebuilt cluster.
3. For each volume on the mirror cluster, set up a corresponding volume on the rebuilt cluster.
   a. Restore the volume using the `volume dump restore` command. Example: `maprcli volume dump restore -name volume@cluster -dumpfile fulldump1`
   b. Copy the files to a standard (non-mirror) volume.

Starting the Mirror

Explains how to start a mirror operation using either the Control System or the CLI.

When a mirror starts, all the data in the source volume is copied into the mirror volume. Starting a mirroring operation requires the mirror volume to exist and be associated with a source. After you start a mirror, synchronize it with the source volume regularly to keep the mirror current.

Note: The `getIPTypeForCluster` method in `CLDBRpcCommonUtils` is unable to determine whether the IP type is internal or external, if `mapr-clusters.conf` contains both internal and external IPs. The fix is to only put in the internal IP in `mapr-clusters.conf` and keep the external IP in the `env.sh` file, before starting the mirror.

Starting the Mirror for Multiple Mirror Volumes Using the Control System

To start mirroring, in the **Summary** tab under **Data** > **Volumes**:

Note: The **Summary** tab is under the **Volumes** tab in the Kubernetes version of the Control System.

1. Select the mirror volume(s) to synchronize.
   You *cannot* start mirror for:
   - Standard volumes
   - Mirror volumes currently mirroring
2. Select **Start Mirroring** from the **Actions** drop-down menu.
   The **Start Mirroring Volume(s)** confirmation dialog displays.
3. Verify the list of volumes to synchronize and click **Start Mirroring**.
   When a mirror is started, the mirror volume is synchronized from a hidden internal snapshot so that the mirroring process is not affected by any concurrent changes to the source volume. The changes to the mirror volume occur atomically at the end of the mirroring process; deltas transmitted from the source volume do not appear until mirroring is complete.

Starting the Mirror for a Mirror Volume Using the Control System

To start mirroring:

1. Go to the volume information page for the mirror volume to synchronize.
2. Select **Start Mirroring** from the **Select Action** drop-down menu.
   The **Start Mirroring Volume** confirmation dialog displays.
3. Click **Start Mirroring**.
   When a mirror is started, the mirror volume is synchronized from a hidden internal snapshot so that the mirroring process is not affected by any concurrent changes to the source volume. The changes to the mirror volume occur atomically at the end of the mirroring process; deltas transmitted from the source volume do not appear until mirroring is complete.

Starting the Mirror Using the CLI or REST API

The basic command to start a mirror is:

```
maprcli volume mirror start -name <volume name>
```

For complete reference information, see `volume mirror start` on page 1990.

Stopping the Mirror
Explains how to stop a mirror operation using either the Control System or the CLI.

Stopping a mirror halts any replication or synchronization process currently in progress. Stopping a mirror does not delete or remove the mirror volume.

Stopping the Mirror for Multiple Mirror Volumes Using the Control System

To stop mirroring, in the **Summary** tab under **Data** > **Volumes**:

Note: The **Summary** tab is under the **Volumes** tab in the Kubernetes version of the Control System.

1. Select the mirror volume(s) to stop.
   You *cannot* stop the mirror operation for:
   - Standard volumes
   - Mirror volumes that are currently not mirroring
2. Select **Stop Mirroring** from the **Actions** drop-down menu.
   The **Stop Mirroring Volume(s)** confirmation dialog displays.
3. Verify the list of volumes to stop and click **Stop Mirroring**.
   When a mirroring operation is stopped, replication or synchronization processes currently in progress will halt.

Stopping the Mirror for a Mirror Volume Using the Control System

To stop mirroring:

1. Go to the volume information page for the mirror volume to stop.
2. Select **Stop Mirroring** from the **Select Action** drop-down menu.
   The **Stop Mirroring Volume** confirmation dialog displays.
3. Click **Stop Mirroring**.
   When a mirroring operation is stopped, replication or synchronization processes currently in progress will halt.

Stopping the Mirror Using the CLI or REST API

The basic command to stop a mirror is:

```
maprcli volume mirror stop -name <volume name>
```

For complete reference information, see `volume mirror stop` on page 2004.

Viewing Mirror Status
List mirror volumes and their status using the Control System and the CLI.

You can see a list of all mirror volumes and their current status in the **Mirror Volumes** view (in the Control System, select **Mirror Volumes** from the drop-down menu in the **Volumes** page under **Data** > **Volumes**) or using the `volume list` on page 1979 command. Use the volume mirror status on page 1991 command to view the details of the mirroring operation that is in progress. The volume mirror status on page 1991 command helps in troubleshooting the mirroring operation. For more information on trouble shooting mirroring, see the support article titled Monitor and Understand Volume Mirroring in MapR.

Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

To use the volume list command to explicitly list mirror volumes, you must define a filter. For example:

```
maprcli volume list -filter [n==<mirror name>] -columns \
         n,p,mirror-percent-complete,mrt -cluster <target cluster>
```

Mirrors and Performance

Completion time for a mirroring operation is affected by the available network bandwidth, and the amount of data to transmit. For best performance, set the mirroring schedule according to the anticipated rate of data changes, and the available bandwidth for mirroring.

Using Promotable Mirrors for Disaster Recovery

The concept of promoting a mirror refers to the ability to make a read-only mirror volume into a read-write volume. The main use case for this feature is to support disaster-recovery scenarios in which a read-only mirror needs to be promoted to a read-write volume so that it can become the primary volume for data storage.

A MapR administrator can perform the following tasks from a remote datacenter before, during, and after a disaster:

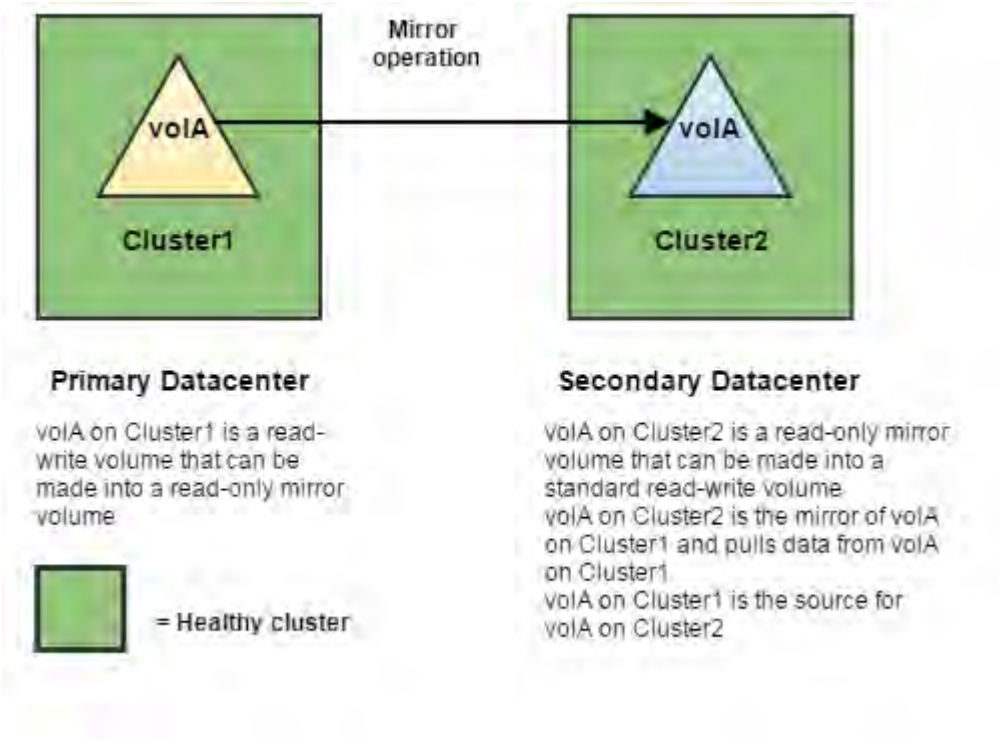
- Set up mirroring to a remote cluster
- Fail over to a mirror volume
- Restore the mirror relationship

For a brief overview of the terminology used to describe volume types, along with some basic commands, see the Types of Volumes on page 460.

The following sections provide information about how to use promotable mirrors for disaster recovery:

Setting up Mirroring to a Remote Cluster

Once data volumes are created in a primary datacenter, the MapR administrator creates mirror volumes in a remote secondary datacenter. The following diagram illustrates the mirror relationship between these two volumes:



Note: When you use promotable mirrors, the volumes on the destination cluster must be set up in the same way as on the primary site. This means that volume names are the same and mount points are the same. If a hierarchical mounting structure (such as /A/B) is used on the primary site, the same structure must be recreated once mirror volumes are promoted at the secondary site.

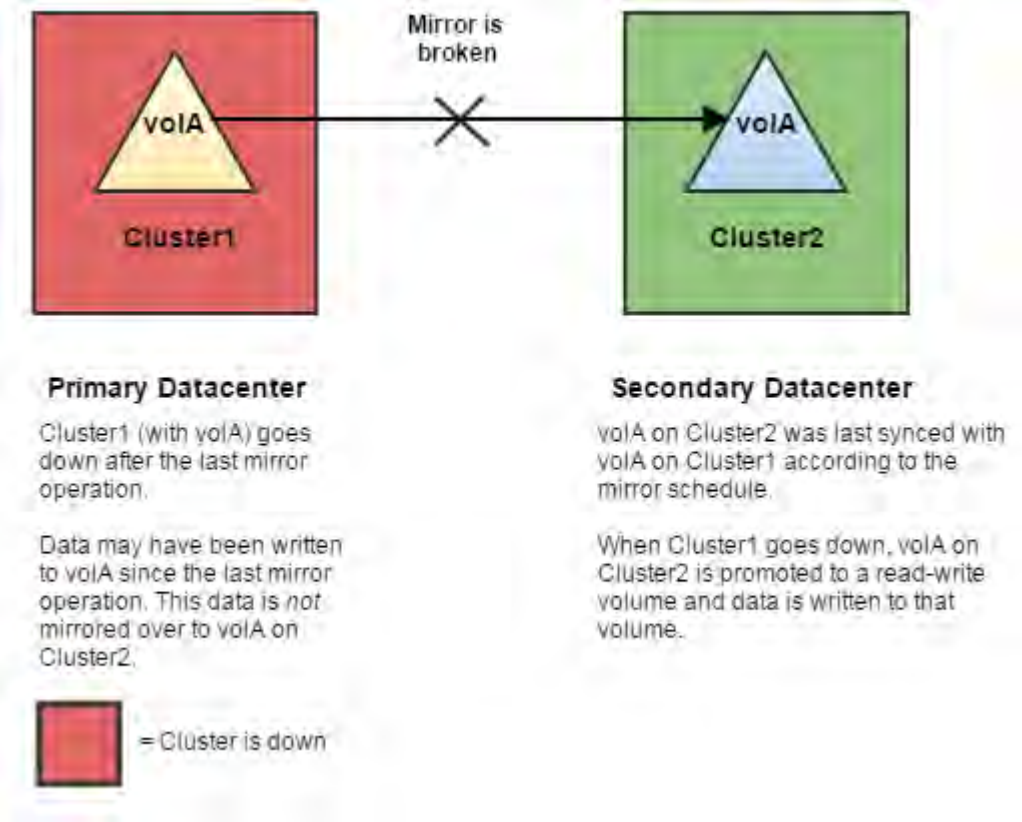
last-updated: Apr 11, 2022

The following sections provides information about how to set up mirroring to a remote cluster:

1. Creating Remote Mirrors on page 880
2. Configuring Secure Clusters for Cross-Cluster Mirroring and Replication on page 1486
3. Creating a Mirror Volume
4. Creating a Mirroring Schedule

Failing Over to a Mirror Volume

When a disaster occurs at a primary datacenter, data can no longer be written to the volumes in that location, and the mirror operation cannot be performed. In order to maintain business continuity, the administrator at the secondary datacenter promotes the read-only mirror volume to a read-write volume, which breaks the mirror relationship. At this point, the promoted mirror volume contains all the data that was on the source volume at the time of the most recent successful mirror operation.



The following sections provide information about how to fail over to a mirror volume:

- Changing Mirror Volumes to Standard Volumes on page 906
- Handling Mount Points in Promoted Mirror Volumes on page 908
- Changing the Limit for Concurrent Mirror Operations Using the CLI on page 918

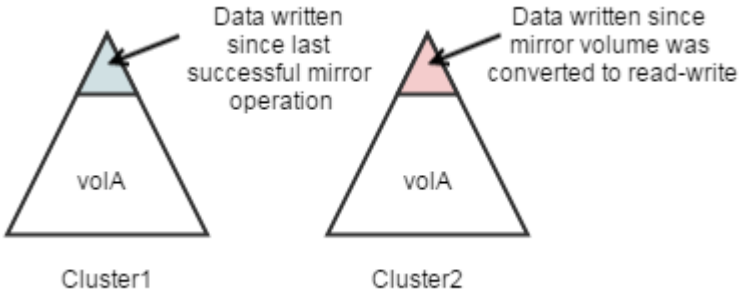
Restoring the Mirror Relationship

Explains how to restore the mirror relationship between the original read-write volume in the primary datacenter and the promoted read-write volume in the secondary datacenter.

If the primary datacenter comes back online, the administrator can re-establish the mirror relationship between the original read-write volume in the primary datacenter and the promoted read-write volume in the secondary datacenter.

last-updated: Apr 11, 2022

Note that the two read-write volumes will have different data, since data was written to the promoted mirror while the original source volume was down. The original source volume might also have different data that was written after the last mirror operation, but before the cluster went down. The administrator must decide which data to keep and use as the source.



Warning: Some data loss is inevitable in a disaster recovery scenario. To minimize potential data loss, use mirrors to provide a synchronized copy of each volume with critical data, and in the event of discrepancies, decide which data to preserve based on your company's policies.

The following sections provide information about how to restore the mirror relationship:

Preserving volA/Cluster1's Data

Suppose that volA in the primary datacenter contains crucial data that must be preserved, and you want to mirror its data to volA in the secondary datacenter (the same mirror relationship that was established originally). To recreate the original mirror relationship, convert the promoted volume, volA/Cluster2, from a read-write volume to a mirror of volA/Cluster1 by running the following command:

```
Cluster2> maprcli volume modify -name
volA -type mirror -source
volA@Cluster1
```

To use the Control System to convert volA/Cluster2 from a read-write volume to a mirror of volA/Cluster1, follow steps for Changing a Standard Volume to a Mirror Volume on page 905.

Preserving volA/Cluster2's Data on volA/Cluster1

Now suppose you want to preserve the data on volA/Cluster2 (in the remote datacenter) but you still want volA/Cluster1 to be the primary volume with volA/Cluster2 as its mirror. From the command line or the Control System, you can save volA/Cluster2's data on volA/Cluster1 and reestablish the original mirror relationship from volA/Cluster1 to volA/Cluster2.

You can use either of the following methods to preserve the data:

From the Control System

Complete the following steps from the Control System to save volA/Cluster2's data on volA/Cluster1 and reestablish the original mirror relationship from volA/Cluster1 to volA/Cluster2.

1. Stop writing new data to volA/Cluster2 by making this volume read-only:
   For detailed steps, see Modifying a Volume on page 892.
2. Make volA/Cluster1 a mirror of volA/Cluster2.
   For detailed steps, see Changing a Standard Volume to a Mirror Volume on page 905.

last-updated: Apr 11, 2022

3. Start mirroring.
   For detailed steps, see Starting the Mirror on page 919.
4. Promote volA/Cluster1 to a read-write volume.
   For detailed steps, see Changing Mirror Volumes to Standard Volumes on page 906.
5. Make volA/Cluster2 a mirror of volA/Cluster1.
   For detailed steps, see Changing a Standard Volume to a Mirror Volume on page 905.

From the Command Line

1. Stop writing new data to volA/Cluster2. To be sure no data is written to this volume, make it read-only by running this command:

```
Cluster2> maprcli volume modify -name volA -readonly true
```

2. Pull the data from volA/Cluster2 to volA/Cluster1 by making volA/Cluster1 a mirror of volA/Cluster2.

```
Cluster1> maprcli volume modify -name volA -type mirror -source
volA@Cluster2
```

3. Start the mirror operation.

```
Cluster1> maprcli volume mirror start -name volA
```

4. Once mirroring is done, promote volA/Cluster1 to a read-write volume. Note that the mirror relationship breaks at this point.

```
Cluster1> maprcli volume modify -name volA -type rw
```

5. Make volA/Cluster2 a mirror of volA/Cluster1.

```
Cluster2> maprcli volume modify -name volA -type mirror -source
volA@Cluster1
```

Enabling and Restricting Access to Tenant Volume and Data
Describes how to restrict access to tenant volumes in a multi-tenant environment.

In a multi-tenant environment, the tenant volume (share) can be accessed by all users on the tenant instance by default. To restrict access to specific users and/or groups:

1. Log in to the cluster as the cluster administrator and set *ACE*s on the volume using the volume commands.
   For example:

```
/opt/mapr/bin/maprcli volume modify -name <volumename> -readAce
"u:<user>|g:<group>" -writeAce "u:<user>|g:<group>"
```

   Here, value for <user> must be the UID of the user and value of <group> must be GID of the group on the tenant host.

   Tip: For more information, see `maprcli volume modify` command.

2. Log in as the tenant admin and set permissions for data access.

last-updated: Apr 11, 2022

You can set permissions using:

- Linux commands such as `chmod`, `chown`, and so on.
- *ACE*s, which can be set on files and directories in the volume. For more information, see Enabling Volume, Directory, and File Authorizations with ACEs on page 1452.

Working with Tiered Volumes

This section describes how to create tiered volumes and manage automatic and manual tiering jobs on the tiered volume.

Erasure Coding Scheme for Data Protection and Recovery

Describes the erasure coding (EC) schemes for data protection and recovery.

Erasure coding (EC) is a data protection method in which data is broken into fragments, expanded and encoded with redundant data pieces, and stored across a set of different nodes or storage media.

EC ensures that if data becomes corrupted, it can be reconstructed using other data and parity fragments.

The time required to reconstruct data depends on the number of data fragments in the chosen EC scheme, and the number of failures that have occurred. For example, reconstruction of EC scheme *10+2* takes longer compared to the reconstruction of EC scheme *3+2*, as a larger number of data and parity fragments must be read.

Considerations When Selecting an EC scheme

As an administrator, consider the following points when selecting an EC scheme:

- How many nodes can you afford to have?
- How many failures might occur? Do you anticipate a single failure, or multiple failures?
- Consider the following when determining how long you are willing to wait for a node to be rebuilt:
  - Rebuilds overhead - For 4+2, need to read data from 4 nodes to rebuild, for 6+3, need to read from 6 nodes to rebuild.
  - Reads overhead - For 4+2, need to read data from 4 nodes , for 6+3, need to read from 6 nodes. For degraded states, parity calculation overhead is an add-on.
  - Data classification - High ecschemes 10+2 , 12+4, are ideal for archived data, since data mostly remains untouched.

EC Schemes

In an erasure coded volume, an erasure coding scheme has the stripe layout *m+n*. The stripe is an array of *m* data fragments and *n* parity fragments.

Each fragment is called a stripelet. Each stripelet is present on a container and one stripe is across different containers on different nodes. The default stripelet size is 4MB. For an EC scheme *4+2* for example, the stripe size is 24MB.

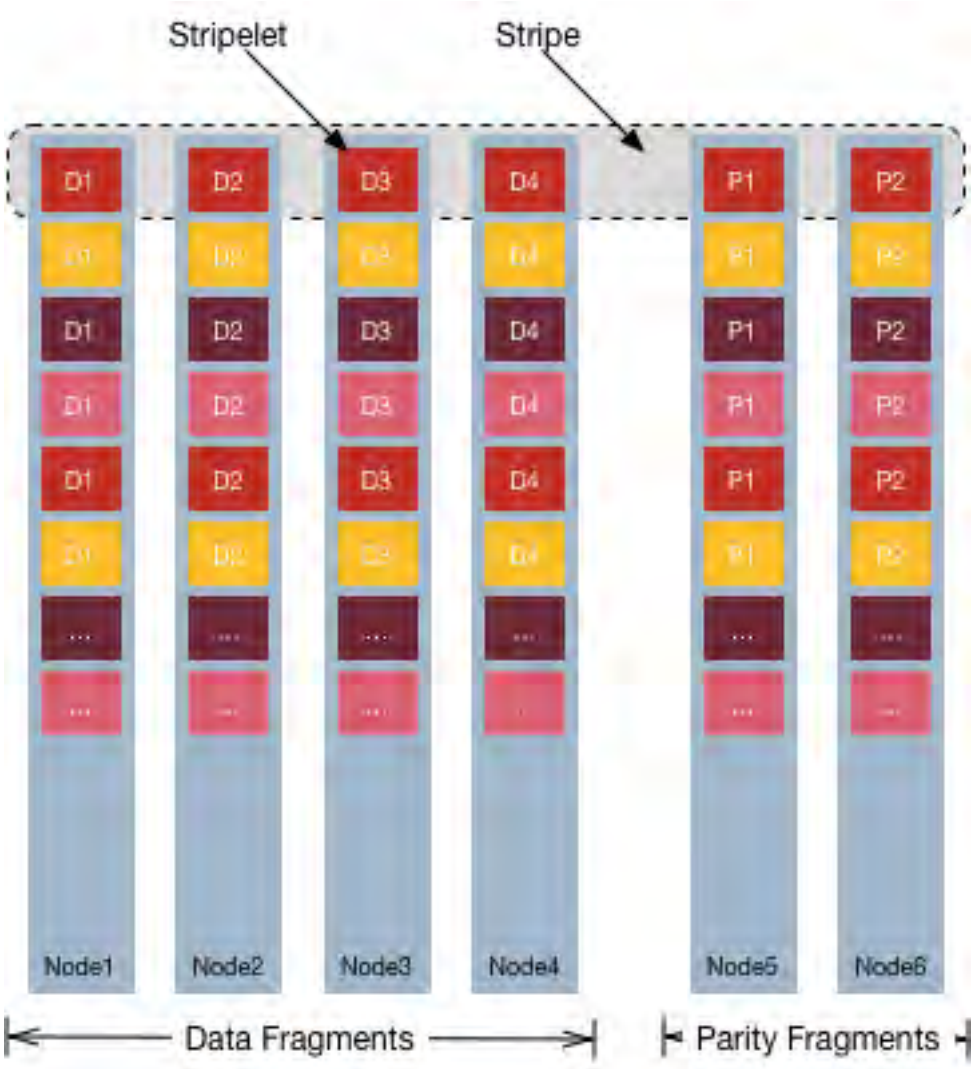
A Container Group (CG) is collection of such stripes. Based on the maximum size of a container (32 GB), the maximum number of stripes in a CG is 8K.

Each stripe is created by the same number of data fragments from all containers in the group of EC containers. Each container is placed on a different physical node.

- *m* is the number of data fragments.
- *n* is the number of redundant fragments (referred to as parity fragments).
- The parity is calculated using data from all data fragments.

- *m/(m+n)* is the encoding rate.
- *m+n* is the number of encoded fragments.
- You need to read a minimum of *m* blocks to recover data.
- You can recover data from a maximum of *n* failures.

For example, assume m=4, n=2, and stripe depth=4 MB.



- The number of data fragments is four (4), and while the number of parity fragments is two (2).
- The number of encoded fragments is six (6).
- The stripe size is 16 MB (4x4 MB) of user data, and 8 MB (2x4 MB) of parity fragments.
- The system can handle two (2) failures, and any fragment can be recovered from four (4) other fragments.

Requirements for using an erasure coding scheme

Requirements when using the Control System

- The number of data fragments must be 3 to 10.

last-updated: Apr 11, 2022

- The number of parity fragments must be greater than 1 and less than the number of data fragments.
- The number of data fragments must be greater than the number of parity fragments.
- The number of nodes must be greater than or equal to the sum of data and parity fragments.

Requirements when using the CLI

- The number of data fragments must be 2 to 10.
- The number of parity fragments must be greater than or equal to 1 and less than or equal to the number of data fragments.
- The number of data fragments must be greater than or equal to the number of parity fragments.

Select from the following schemes for erasure-coded volumes:

EC Scheme		Number of Data Nodes	Number of Parity Nodes	Total Number of Nodes Needed	Number of Failures Recoverable	Number of Nodes to Read to Recover Data
CLI	Control System					
3+2	3, 2	3	2	5	2	3
4+2	4, 2	4	2	6	2	4
5+2	5, 2	5	2	7	2	5
6+3	6, 3	6	3	9	3	6
10+<x> where x is a value from 1 to 9	N.A	10	x	10+x	x	10

Although you can create a volume without the required number of nodes for a specific scheme, volume offload fails if the required number of nodes are not present.

When choosing the scheme, note that more nodes leads to longer recovery time, resulting in degraded performance, network expense, and lengthy time to rebuild.

For example, consider a *12 + 4* EC scheme represented as `D0 + D1 + D2 +....+D10 + D11 + P0 +…+P3`

Suppose node D4 goes down, now to rebuild, a total of 12 stripelets must be read. This leads to huge performance degradation in network bandwidth, CPU cycles, and Disk IO .

To reduce the reconstruction cost, use EC Local Parity, where the number of stripelets to be read reduces to 6 for a single failure in the *12+2+2* scheme.

Important: The recommended number of nodes required for erasure coding is M+2N (rather than M+N) to ensure MapR self-healing and proper operation after N failures. N failures with only M+N nodes allows you to continue reading the data, but with significantly reduced performance because each read requires rebuilding data fragments. Also, manual intervention is required to protect the data from further failures. Currently, data will not be erasure coded if only M nodes are available. With M+2N nodes, N failures will self-heal with no operator intervention.

Specifying a Schedule for Offloading Data
Explains how to create a schedule for automatic offloading of data, using the Control System, the CLI and the REST API.

You can create a schedule using the Control System, the CLI, and REST API. After creating a schedule, you can associate it with the tiering-enabled volume when you create or modify the volume. If a schedule

for offloading data is associated with the volume, data is offloaded automatically as scheduled based on the rules associated with the volume for offloading data. You can also manually trigger the `maprcli` command to offload data.

The following schedules are available out-of-the-box for offloading data:

Schedule Name	Schedule ID
Critical data	1
Important data	2
Normal data	3
Automatic Tiering Scheduler	4*

* The Automatic Tiering Scheduler ID might be different on different clusters. To retrieve the correct ID, run the `schedule list` on page 1740 command.

For volumes enabled for warm tiering, the Automatic Tiering Scheduler is used by default for offloading data if you do not explicitly assign a schedule. The frequency of the Automatic Tiering Scheduler run is based on two the following:

time	The frequency. The `cldb.auto.offload.frequency.minutes` property stores the default value of 24 * 60 minutes. This can be configured using the `config save` on page 1586 command. The value for this property must be in minutes.
size	The amount of data (that has not yet been offloaded) in the volume. The `cldb.auto.offload.threshold.gb` property stores the global value for the size threshold. The default value for this property is 1024GB, which cannot be modified. However, you can override the global value at the volume-level using the `autooffloadthresholdgb` parameter with the `Creating a Volume` on page 864 and `volume modify` on page 2005 commands.

The Automatic Tiering Scheduler run is based on the time setting. However, it runs sooner if the size of the volume in the hot tier reaches or exceeds the size threshold.

For volumes enabled for cold tiering, you must assign a schedule to automatically offload data; if you do not assign a schedule, data is not offloaded automatically and you must manually run the offload command to offload data. You can associate the Automatic Tiering Scheduler with the cold-tier enabled volume or create a custom schedule and associate it with the volume to automatically offload data.

To:

- Create a schedule before creating the volume, see Creating a Schedule on page 954.
- Create a schedule when you are creating the volume, see step 9 in Creating a Volume on page 864.

Specifying a Schedule Using the Control System

You can associate a schedule with a tiering-enabled volume when you are:

- Creating a volume by clicking **Create Volume** button in the **Data** > **Volumes** page.
- Editing the tiering-enabled volume by clicking **Edit Volume** button in the volume information page.

To associate an offload schedule with the volume, in the **Schedule** tab of the **Create Volume** or **Edit Volume** page:

1. Click the **Browse** link associated with the **Offload Schedule** field to display the **Browse Schedules** window.
2. Review the name and detail of each schedule and choose a schedule from the list.
3. Click **Save** to associate the schedule with the volume.
4. Complete the steps for creating or editing the volume.

Specifying a Schedule Using the CLI and REST API

You can associate a schedule with a tiering-enabled volume by specifying the `offloadschedule` parameter with the `volume create` on page 1931 or `volume modify` on page 2005 command.

CLI

Run a command similar to the following to associate a schedule when:

- Creating a volume:

```
maprcli volume
create -name <volName> -path
<mountPath> -tieringenable
true -tiername
<tierName> -offloadschedule
<scheduleID> -json
```

  For the list of all other required and optional parameters, see `volume create` on page 1931.

- Editing the volume:

```
maprcli volume modify -name
<volName> -offloadschedule
<scheduleID> -json
```

  For the list of all other required and optional parameters, see `volume modify` on page 2005.

REST

Send a request of type POST. For example, to associate a schedule when:

- Creating a volume:

```
curl -k -X POST 'https://
<host>:8443/rest/volume/create?
name=<volName>&path=<mountPath>&tie
ringenable=true&tiername=<tierName>
&offloadschedule=<scheduleID>' --us
er mapr:mapr
```

  For the list of all other required and optional parameters, see `volume create` on page 1931.

- Editing the volume:

```
curl -k -X POST 'https://
<host>:8443/rest/volume/modify?
name=<volName>&offloadschedule=<sch
eduleID>' --user mapr:mapr
```

For the list of all other required and optional parameters, see `volume modify` on page 2005.

To disable automatic schedule-based offload of data, set the value for the `offloadschedule` parameter to `0`.

Creating a Tiering-Enabled Volume

You can create a tiering-enabled volume using the Control System, the CLI, and the REST API.

Creating a Tiering-Enabled Volume Using the Control System

1. Go to **Data** > **Volumes** and click **Create Volume**.
   The **Create New Volume** page displays.
2. Select the **Volume Type**, specify values for required/optional settings and auditing, and move the slider to **Yes** for **Data Tier** in the **Properties** tab.
   For information on all other properties and settings, see Creating a Volume on page 864.

   Note: The source volume for a tiering-enabled mirror volume must also be tiering-enabled. You cannot set up a tiering-enabled mirror volume to mirror a volume that is not tiering-enabled.
3. Click **Create Volume** to create the tiering-enabled volume.

You can proceed to associate a tier, tiering rule, and/or schedule with the volume.

Creating a Tiering-Enabled Volume Using the CLI and REST API

CLI

Run the following command to create a tiering-enabled volume:

```
$maprcli volume
create -name <volName> -path
<volmountpath> -tieringenable true
```

REST

Send a request of type POST. For example:

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=<volName>&path=<volmountpath>&tie
ringenable=true' --user mapr:mapr
```

For more information, see `volume create` on page 1931.

Associating a Tier, Tiering Rule, and/or Schedule with a Volume

You can associate a tier, tiering rule, and/or schedule with a new volume or with an already tiering-enabled volume using the Control System, CLI, and the REST API.

Associating a Tier, Tiering Rule, and/or Schedule with a Volume Using the Control System

- Perform the steps in the following topics to associate a tier, tiering rule, and/or schedule with:
  - A new volume

- A tiering-enabled volume

Associating a Tier, Tiering Rule, and/or Schedule with a Volume Using the CLI

- Run the following command to associate a tier, tiering rule, and/or schedule with:
  - A new volume:

    ```
    $maprcli volume create -name <vol_name> -path
    <vol_mount_path> -tieringenable true -tiername <tier_name> -tieringrule
    <rule_name> -offloadschedule <schedule_ID>
    ```

    For more information, see volume create on page 1931.

  - An already tiering-enabled volume:

    ```
    $maprcli volume modify -name <vol_name> -tiername
    <tier_name> -tieringrule <rule_name> -offloadschedule <schedule_ID>
    ```

    For more information, see volume modify on page 2005.

Note: You cannot change the tier type or the tier for a volume after it is set.

Associating a Tier, Tiering Rule, and/or Schedule with a Volume Using the REST API

- Send a request of type POST to associate a tier, tiering rule, and/or schedule with:
  - A new volume. For example:

    ```
    curl -k -X POST 'https://<host>:8443/rest/volume/create?
    name=<vol_name>&path=<vol_mount_path>&tieringenable=true&tiername=<tier_
    name>&tieringrule=<rule_name>&offloadschedule=<schedule_ID>' --user
    mapr:mapr
    ```

    For more information, see volume create on page 1931.

  - An already tiering-enabled volume. For example:

    ```
    curl -k -X POST 'https://<host>:8443/rest/volume/modify?
    name=<vol_name>&tiername=<tier_name>&tieringrule=<rule_name>&offloadsche
    dule=<schedule_ID>' --user mapr:mapr
    ```

    For more information, see volume modify on page 2005.

Note: Volume's data tiering properties like tiername, ectopology, ecscheme, etc. cannot be modified after they are set.

Determining if a Volume is Enabled for Tiering

You can determine if a volume is enabled for tiering and if rules, schedules, and/or settings for recalled data are associated with the volume using the Control System and the CLI.

Determining if a Volume is Enabled for Tiering Using the Control System

- Log in to the Control System and click **Data** > **Volumes**.

  In the list of volumes displayed in the **Volumes** pane, the **Data Tiering** column contains the value **Enabled** for a volume if the volume is enabled for tiering. If you do not see the column, you can see the column by selecting the columns to display in the Control System.

last-updated: Apr 11, 2022

Determining if a Volume is Enabled for Tiering Using the CLI

- Run one of the following commands to determine if a volume is enabled for tiering:
  - `maprcli volume list -json`
  - `maprcli volume info -name <volName> -json`

  The output, if the volume is tiering-enabled (and has associated settings), should look similar to the following:

```
{
        "timestamp":1533959507772,
        "timeofday":"2018-08-10 08:51:47.772 GMT-0700 PM",
        "status":"OK",
        "total":1,
        "data":[
                {
                        "acl":{
                                "Principal":"User root",
                                "Allowed actions":"[dump, restore, m, a,
d, fc]"
                        },
                        ...
                        ...
                        "tierenable":"true",
                        "tierid":"169211273",
                        "tierruleid":"2",
                        "tieroffloadscheduleid":"6",
                        "tierencryption":"false",
                        "tierrecallexpirytime":"1",
                        "tiercompactionscheduleid":"4",
                        "tiercompactionoverheadthresh":"30",
                        "gateway":"10.10.108.120:8660",
                        "cvtotalused":0
                }
        ]
}
```

Offloading a Volume to a Tier

Explains how to offload a volume to a tier using either the Control System, the CLI, or the REST API.

At the volume level, data can be offloaded automatically by creating and associating a schedule with the tiering enabled volume or manually by triggering the offload operation. See Data Offload and Purge on page 475 for more information. The following sections describe how to set up an automatic offload of data and how to trigger a one-time manual offload data at the volume level using either the Control System, the CLI, or the REST API.

Note: For a tiered volume, there can be only one running job at any given time. For example, suppose another job is running on the tiered volume, if you trigger an offload operation, the offload operation will fail.

To offload volume data, you must have one of the following permissions:

- Cluster level fc permissions
- Volume level fc permissions
- Volume modify permissions

last-updated: Apr 11, 2022

Note: You can also offload individual files in a tiering-enabled volume to the associated tier. See Offloading a File to a Tier Using the CLI and REST API on page 1012 for more information.

Important: EC volumes are automatically offloaded once they crossed the size of `autooffloadthresholdgb` even if they are not using the Automatic scheduler. The default size is 1024 GB (1 TB). You can modify this size as specified in Offloading a Volume to a Tier on page 933.

Setting up Automatic Offload of Data Using the Control System

1. Create a tier.

   See Creating a Cold Tier Using the Control System on page 959 or Creating a Warm Tier Using the Control System on page 959 for more information.

2. Create a storage policy to associate with the volume.

   See Creating a Storage Tier Policy Using the Control System on page 972 for more information.

3. Create an offload schedule.

   See Specifying a Schedule for Offloading Data on page 928 for more information.

4. Create a tiering enabled volume and associate the tier, the storage policy, and schedule with the volume.

   See Creating a Tiering-Enabled Volume Using the Control System on page 931 for more information.

Triggering an Offload of all Data in a Volume to a Tier Using the Control System

1. Log in to the Control System and click **Data** > **Volumes**.

   Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

2. Select the tiered volumes to offload from the list of volumes in the **Volumes** pane.

   Selecting a volume makes the **Actions** drop-down menu available.

3. Click **Offload Data** from the **Actions** drop-down menu to display the **Offload Volume Data** confirmation window.

4. Review the list of volumes to offload and click **Offload**.

If the offload fails, CLDB retries the operation after some time. See Retrying Failed Operation on page 939 for more information.

Setting up Automatic Offload of Data Using the CLI and REST API

To automatically offload data:

1. Create a tier, a rule that contains the criteria for offloading data to the tier, and a schedule to automatically offload data to the tier.

   For example:

CLI

```
/opt/mapr/bin/maprcli tier
create -name <tier_name> -type
ectier
/opt/mapr/bin/maprcli tier
create -name <tier_name> -type
cold -url <tier_url> -credential
<credentials>.txt -json
/opt/mapr/bin/maprcli tier rule
create -name <rule_name> -expr
```

```
<expressions>
/opt/mapr/bin/maprcli schedule
create -schedule <JSON>
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/create?
name=<tier_name>&type=ectier' --user
 mapr:mapr
curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/create?
name=<tier_name>&type=cold&url=<tier
_url>&credential=<credential_str>'
 --user mapr:mapr
curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/rule/
create?
name=<rule_name>&expr=<expressions>'
 --user mapr:mapr
curl -k -X POST 'https://
abc.sj.us:8443/rest/schedule/create?
schedule=<JSON>' --user mapr:mapr
```

For more information, see Managing Tiers on page 957 and Managing Storage Policies on page 970.

2. Create a tiering-enabled volume and associate the tier, rule, and schedule (that you created in step 1) with the volume.

   For example, to create a volume and enable it for:

   - Warm tier:

     CLI

     ```
     /opt/mapr/bin/maprcli volume
     create -name <vol_name> -path
     <vol_mount_path> -tieringenable
     true -tiername
     <tier_name> -ecscheme
     <coding_scheme> -ectopology
     <ec_vol_topo> -tieringrule
     <rule_name> -offloadschedule
     <schedule_ID>
     ```

     REST

     ```
     curl -k -X POST 'https://
     abc.sj.us:8443/rest/volume/create?
     name=<vol_name>&path=<vol_mount_pat
     h>&tieringenable=true&tiername=<tie
     r_name>&ecscheme=<coding_scheme>&ec
     topology=<ec_vol_topo>&tieringrule=
     <rule_name>&offloadschedule=<schedu
     le_ID>' --user mapr:mapr
     ```

   - Cold tier:

     CLI

     ```
     /opt/mapr/bin/maprcli volume
     create -name <vol_name> -path
     <vol_mount_path> -tieringenable
     true -tiername
     ```


last-updated: Apr 11, 2022

```
<tier_name> -tieringrule
<rule_name> -offloadschedule
<schedule_ID> -recallexpirytime
2 -json
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=<vol_name>&path=<vol_mount_pat
h>&tieringenable=true&tiername=<tie
r_name>&tieringrule=<rule_name>&off
loadschedule=<schedule_ID>&recallex
pirytime=2' --user mapr:mapr
```

You can also specify the maximum amount of data (in GB) to offload automatically for warm-tier volumes using the `autooffloadthresholdgb` parameter. For more information, see Working with Tiered Volumes on page 926.

Triggering an Offload of all Data in a Volume to a Tier Using the CLI and REST API

CLI

Run the following command to manually trigger an offload of all data in the volume:

```
maprcli volume offload -name
<volume-name>
```

If you run the command with the `ignorerule` option value set to `true`, rules for the volume where the data resides is ignored and data is offloaded immediately. If the `ignorerule` option value is not specified or is set to `false` (default), data is offloaded based on the rules associated with the volume where the data resides.

REST

Submit a request of type POST. For example:

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/offload?
name=sampleVol' --user mapr:mapr
{"timestamp":1519947659597,"timeofday"
:"2018-03-01 03:40:59.597 GMT-0800
PM","status":"OK","total":0,"data":
[],"messages":["Successfully started
offload."]}
```

For more information, see `volume offload` on page 2023.

If the offload fails, CLDB retries the operation after some time. See Retrying Failed Operation on page 939 for more information.

Recalling a Volume to MapR File System

Explains how to recall offloaded data to the filesystem.

When you:

- Read data offloaded to a remote target (or cold tier), data is automatically recalled to the MapR File System to allow the read to succeed.
- Modify data offloaded to an erasure coded volume (or warm tier) or a remote target (or cold tier), data is automatically recalled to the MapR File System to allow the write to succeed.

The recalled data is automatically:

- Purged based on the expiration time period set at the volume level for recalled data if there are no changes (for example, read operation).
- Offloaded based on the rule and the expiration time period set at the volume level for recalled data if there are changes (for example, overwrite operation).

See Data Reads, Writes, and Recalls on page 481 for more information. If the recall fails, CLDB retries the operation after some time. See Retrying Failed Operation on page 939 for more information.

You can manually recall all data in a volume using the Control System, CLI, or the REST API.

Note: For a tiered volume, there can be only one running job at any given time. If you trigger a recall operation when another job is running on the tiered volume, the recall operation will fail.

You can also recall individual files from the tier. See Recalling a File to MapR File System Using the CLI and REST API on page 1012 for more information.

Recalling Offloaded Volume Using the Control System

1. Log in to the Control System and click **Data** > **Volumes**.

   Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

2. Select the tiered volumes to recall from the list of volumes in the **Volumes** pane.
   Selecting a volume makes the **Actions** drop-down menu available.
3. Click **Recall Data** from the **Actions** drop-down menu to display the **Recall Tiered Data** confirmation window.
4. Review the list of volumes to recall and click **Recall Volumes**.
   For more information, see Recall of Tiered Data on page 484.

Recalling Offloaded Volume Using the CLI and REST API

CLI	Run the following command to recall volume data: `/opt/mapr/bin/maprcli volume recall -name <volName>`
REST API	Send a request of type POST to URL. For example: `curl -k -X POST 'https://abc.sj.us:8443/rest/volume/recall?name=volName' --user mapr:mapr`

For more information, see `volume recall` on page 2025.

Viewing the List of Running Jobs

You can view the tiering jobs currently running for a volume using the CLI and REST API. For a tiered volume, at any given time, there can be only one running job.

- Run the following command or send a request of type GET to retrieve the list of currently running tiering operations for a volume:

CLI	`maprcli volume tierjobstatus -name <volName>`
REST	`curl -k -X GET 'https://<host>:8443/rest/volume/tierjobstatus?name=<volName>' --user mapr:mapr`

For more information, see `volume tierjobstatus` on page 2038.

Terminating a Running Volume-Level Tiering Job
Describes how to terminate a volume-level tiering job using either the Control System or the CLI.

You can terminate an ongoing offload or recall of a volume using the Control System or the CLI. Terminating a running:

- Offload operation does not prevent future offloads; if a schedule is associated with the volume, data that is still on the cluster is automatically offloaded based on the rules as per schedule. You can also manually offload data again at any time by running the `volume offload` on page 2023 command.
- Recall operation does not prevent future recalls; you can run the recall command again to recall the remaining data on the tier. Based on the expiry time set on the volume (associated with the recalled data), recalled data is offloaded if there are changes or purged if there are no changes. See Recalling a Volume to MapR File System on page 936 for more information.

You can check the status of an abort operation using the `volume tierjobstatus` on page 2038 command.

Note: If you terminate a job that CLDB was retrying after a prior failed attempt (FailureRetriable job), CLDB will stop trying to run the job again. For more information on FailureRetriable job, see Retrying Failed Operation on page 939.

For information on terminating a file-level job, see Terminating a Running File-Level Tiering Job on page 1013 and Running Tiering Commands when maprcli and hadoop Commands are not Available on page 1014.

Terminating a Volume Offload or Recall Operation Using the Control System

1. Log in to the Control System and click **Data** > **Volumes**.

   Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

2. Select the tiered volumes to offload from the list of volumes in the **Volumes** pane.

   Selecting a volume makes the **Actions** drop-down menu available.

3. Click **Abort Tiering Job** from the **Actions** drop-down menu to display the **Abort Tiering Job** confirmation window.

4. Review the list of volumes and click **Abort Job**.

Aborting Volume Offload or Recall Operation Using the CLI and REST API

CLI

Run the following command to abort a running offload or recall operation:

```
maprcli volume tierjobabort -name
<volName>
```

For more information, see volume tierjobabort on page 2037.

REST API

Send a request of type POST. For example:

```
curl -k -X
POST 'https://<host>:8443/rest/volume/
tierjobabort?name=<volName>' --user
mapr:mapr
```

Retrying Failed Operation

Volume-level

If an offload, recall, or abort operation for a volume fails, the volume tierjobstatus on page 2038 command shows one of the following statuses:

FailureFatal	Indicates failure is fatal and CLDB cannot retry the operation again.
FailureRetriable	Indicates failure to offload; however, CLDB will try again if the job is not manually restarted again or aborted.

CLDB tries the operation again (up to 5 times by default) after a specific wait time (of 30 minutes by default) for the following errors:

- EAGAIN
- ETIMEDOUT
- ENETUNREACH
- ENETDOWN
- ECONNRESET

The `RetryCount` field value in the volume tierjobstatus on page 2038 command output shows the number of times CLDB has retried so far. For example:

```
# maprcli volume tierjobstatus -name
testvol -json
{
 "timestamp":1503308792266,
 "timeofday":"2017-08-21 09:46:32.266
GMT+0000",
 "status":"OK",
 "total":1,
 "data":[
  {
   "offload":{
    "state":"FailureRetriable,
```


last-updated: Apr 11, 2022

```
RetryCount: 5",
    "startTime":"2017-08-21
09:07:17.506 GMT+0000",
    "endTime":"2017-08-21
09:08:49.799 GMT+0000",
    "gateway":"10.10.102.68:8660"
   }
  }
 ]
}
```

File-level

If the offload or recall operation for an individual file fails, the `file tierstatus` on page 1676 or `hadoop mfs` on page 5373 command returns one of the following:

Code	Message	Description
0	HAS_LOCAL_DATA	Indicates that the file is not yet fully offloaded.
1	NO_LOCAL_DATA	Indicates that the file was completely offloaded.
2	OP_FAIL	Indicates that the operation to retrieve the status failed.
3	INVALID_FILE	Indicates that the file does not exist.
4	FILE_NOT_TIERED	Indicates that the file is not in a tiered volume.
5	FILE_EMPTY	Indicates that the file specified for offload is an empty file.
6	NO_GATEWAY	Indicates that no MAST Gateways are available for offload operation.
7	OP_TIMEOUT	Indicates that there was no response from the MAST Gateway (maybe as a result of an error) during the offload or recall operation.
8	FTOS_SUCCESS	Indicates that the file was successfully offloaded or recalled.

last-updated: Apr 11, 2022

Code	Message	Description
9	FTOS_ABORTED	Indicates that the file offload or recall operation was aborted.
10	FTOS_ABORT_IN_PROGRESS	Indicates that the file offload or recall job is being aborted.
11	FTOS_TRANSFER_IN_PROGRESS	Indicates that the file offload is in progress.
12	FTOS_REQ_QUEUED	Indicates that the file offload is scheduled, but has not yet started.
13	FTOS_JOB_NOT_AVAILABLE	Indicates that the job ID specified with the tierjobstatus command is not available.

When a file-level offload or recall operation fails, CLDB **does not** retry the operation. For failed file-level:

- Offload operation, you can run the command to retry the operation. For more information, see Offloading a File to a Tier Using the CLI and REST API on page 1012. Alternatively, if the volume that the file is associated with has a data offload schedule, the file data is automatically offloaded based on the rules associated with the volume.
- Recall or abort operation, you can run the command again to retry the operation if the error returned is not EIO.

You can configure the number of times CLDB retries and the interval between retries using the CLI.

Configuring the Number of Retries

- Set the value for the `cldb.gateway.retry.count` parameter, whose default value is 5, to configure the number of times that CLDB tries again. For example, to configure CLDB to retry to offload, recall, or abort at least 10 times, run the following command:

```
# maprcli config save -values {"cldb.gateway.retry.count":"10"}
```

Configuring the Interval Between Retries

- Set the value for the `cldb.gateway.retry.waittime.seconds` parameter, whose default value is 1800 seconds (30 minutes), to configure the amount of time CLDB waits between retries. For example, to configure CLDB to wait for up to 4 hours (14400 seconds), run the following command:

```
# maprcli config save -values
{"cldb.gateway.retry.waittime.seconds":"14400"}
```


last-updated: Apr 11, 2022

Running the Compactor Using the CLI and REST API

You can trigger the compactor using the CLI and REST API to purge recalled data on the MapR cluster or to purge stale data on the tier. See Data Compaction on page 485 for more information.

Running the Compactor to Purge Recalled Data on the MapR Cluster

CLI	Run the following command to trigger the compactor and purge data whether or not the expiry time for recalled data has been reached: `maprcli volume compact -name <volName> -forcerecallexpiry true`
REST	Send a request of type POST. For example: `curl -k -X POST 'https://<host>:8443/rest/volume/compact?name=<volName>&forcerecallexpiry=true' --user mapr:mapr`

For more information, see `volume compact` on page 1927.

Running the Compactor to Purge Stale Data on the Tier

CLI	Run the following command to trigger the compactor: `maprcli volume compact -name <volName>`
REST	Send a request of type POST. `curl -k -X POST 'https://<host>:8443/rest/volume/compact?name=<volName>' --user mapr:mapr`

When you trigger the compactor, the compactor purges stale data from the tier and also recalled data on the MapR cluster if the expiry time for recalled data has been reached. For more information, see `volume compact` on page 1927.

Retrieving the Status of a Volume-level Tiering Operation

You can check the status of an active volume offload, completed offload, aborted offload, and recall operations using the Control System, the CLI, and the REST API. For information on file level tiering job, see Retrieving Status of File-level Tiering Operation and File Data on page 1015.

Retrieving Status of Volume-level Operation Using the Control System

- Log in to the Control System and click **Data** > **Volumes**.
  The **Volumes** pane in the page displays the following for tiered volumes:
  - **Job** — The currently running or last completed tiering job for the volume.
  - **State** — The status of the tiering job.
  - **Progress** — The completion percentage of the tiering job.

Retrieving Status of Volume-level Operation Using the CLI and REST API

CLI

Run the following command to check the status of an active or completed offload, abort, and/or recall operation:

```
maprcli volume tierjobstatus -name
<volume_name> -json
```

Note: You must have full control (fc) permissions either at the cluster or at the volume level to run this command. To determine the status of a compaction operation, specify the `verbose` option with the `tierjobstatus` command. For example:

```
maprcli volume
tierjobstatus -name
<volume_name> -verbose true -json
```

REST

Send a request of type GET.

```
curl -k -X GET 'https://
<host>:8443/rest/volume/tierjobstatus?
name=<volume_name>' --user mapr:mapr
```

See Statuses on page 2040 for more information.

Retrieving Tiering Statistics Using guts

Explains how to use the `guts` utility to retrieve tiering statistics.

You can run the `/opt/mapr/bin/guts` utility to get granular information on ongoing offloads and recalls including:

- The number of objects that are offloaded to and recalled from the tier
- The number of reads and writes on MapR Database
- The number of reads and writes on MapR File System

Syntax

```
/opt/mapr/bin/guts <argument>:<options>
```

Arguments

Argument	Description
`mastgateway`	Refers to operations on the MAST Gateway node. See Usage on page 944 for information on the syntax for running the guts command with this argument.
`fstier`	Refers to operations on MapR File System node. See Usage on page 944 for information on the syntax for running the guts command with this argument.

last-updated: Apr 11, 2022

Options

Option	Description
`all`	Statistics for all operations.
`db`	Statistics for MAST Gateway operations currently running on MapR Database.
`mfsops`	Statistics for MAST Gateway operations on MapR File System.
`none`	Specifies the column(s) to exclude from the output.
`tier`	Statistics for MAST Gateway operations on the storage tier.

Usage

MAST Gateway Node

```
/opt/mapr/bin/guts mastgateway:all
/opt/mapr/bin/guts mastgateway:db
/opt/mapr/bin/guts mastgateway:tier
/opt/mapr/bin/guts mastgateway:mfsops
/opt/mapr/bin/guts mastgateway:none
```

MapR File System Node

```
/opt/mapr/bin/guts fstier:all
/opt/mapr/bin/guts fstier:none
```

Note: These commands might show statistics for several other fields. To skip, use `none` with the components whose fields you do not wish to retrieve. For example, to retrieve statistics for only the mastgateway tier, run the following command:

```
/opt/mapr/bin/guts allocator:none btree:none cache:none cleaner:none
client:none cpu:none db:none dbrepl:none disk:none fs:none fstier:none
gateway:none io:none kv:none log:none mastgateway:tier net:none
nfs:none rpc:none ssd:none streams:none time:none vcd:none
```

Output

`mastgateway:tier`

`objP`	Number of objects (whose maximum size is 8MB for cold-tier or whose size is computed based on the erasure coding scheme for warm-tier) that were offloaded to the storage tier.
`objG`	Number of objects (whose size is up to 1 MB for cold-tier and whose size is computed based on the erasure coding scheme for warm-tier) that were recalled from the storage tier.
`objD`	Number of deletions on the tier.

last-updated: Apr 11, 2022

`objPM`	Amount of data (in MB) offloaded per second to the storage tier.
`objGM`	Amount of data (in MB) recalled per second from the storage tier.

`mastgateway:db`

`tdbP`	Number of *puts* on MapR Database tables.
`tdbG`	Number of *gets* on MapR Database tables.
`tdbD`	Number of *deletes* on MapR Database tables.

`mastgateway:mfsops`

`moR`	Number of *read* requests from client to the MAST Gateway service to read from cache volumes.
`mp`	Number of MapR File System *purge* requests sent by the MAST Gateway service.
`mrw`	Number of MapR File System *reads* from the MAST Gateway service to perform modify/write operation.
`moRM`	Amount of MapR File System *reads* (in MB) sent by the MAST Gateway service to read offloaded data.
`mrwM`	Amount of *recall writes* (in MB) sent by the MAST Gateway service.

`fstier:all`

`tp`	Number of blocks (of 64KB) *purged* during an offload operation.
`trw`	Number of blocks (of 64KB) *written* during recall of offloaded data.
`trr`	Number of blocks (of 64KB) *read* during read of offloaded data.
`twr`	Number of blocks (of 64KB) *recalled* for partial overwrites.

Related reference

guts on page 2110

last-updated: Apr 11, 2022

`guts` is a tool to measure/analyse performance. In the default mode, it prints one line every second, and counts the number of operations or bytes-processed in one second intervals. `guts` is an internal utility, and is subject to change without notice.

cldbguts on page 2080
Monitors the activity of the Container Location Database (CLDB). This utility prints information about the CLDB service that is running on the node from which you run the utility.

Moving back end Volume from the Command Line

This section contains information on migrating the following volumes to a different topology:

- Metadata volume, which stores the DB tables for the metadata associated with the tier.
- Erasure-coded volume, which stores the erasure-coded data.

Moving Metadata Volume to Another Topology

By default, the volume, which stores the DB tables for the metadata associated with the tier, is created in `/var/mapr/tier/mapr.internal.tier.<volName>` and is in the `/data` topology. However, you can move the metadata volume to another topology using the `volume move` on page 2021 command. For example:

CLI

```
/opt/mapr/bin/maprcli
volume move -name
<metadataVolName> -topology <newTopo>
```

REST

```
curl -k -X POST 'https://<host>:8443/
volume/move?
name=<metadataVolName>&topology=<newTo
po>' --user mapr:mapr
```

Moving Erasure-Coded Volume to Another Topology

The erasure-coded volume is by default created in the same topology as the front-end volume. You can specify a different topology for the erasure-coded volume when creating a front-end volume. You can also move the erasure coded volume to a different topology using the `volume move` on page 2021 command. For example:

CLI

```
/opt/mapr/bin/maprcli volume
move -name <volName> -ectopology
<newTopo>
```

REST

```
curl -k -X POST 'https://<host>:8443/
volume/move?
name=<volName>&ectopology=<newTopo>'
 --user mapr:mapr
```

Here, the `name` parameter takes the name of the front-end volume and the `ectopology` parameter takes the topology to which to move the erasure-coded volume associated with the front-end volume. For more information, see `volume move` on page 2021.

last-updated: Apr 11, 2022

Using Volume Links for Read and Write Operations

When you mirror a volume, read requests to the source volume can be served by any of its mirrors on the same cluster via a volume link of type `mirror`. A volume link is similar to a normal volume mount point, except that you can specify whether it points to the source volume or its mirrors.

- To write to (and read from) the source volume, mount the source volume normally.

  As long as the source volume is mounted below a non-mirrored volume, you can read and write to the volume normally via its direct mount path. You can also use a volume link of type `writeable` to write directly to the source volume regardless of its mount point.

- To read from the mirrors, use the `volume link create` command to make a volume link (of type `mirror`) to the source volume.

  Any read requests from the volume link are distributed among the volume's mirrors. Since the volume link provides access to the mirror volumes, you do not need to mount the mirror volumes.

Managing Snapshots

This section provide information about managing snapshots.

A snapshot is a read-only image of a volume at a specific point in time. On clusters with an Enterprise Edition or higher license, you can create a snapshot manually or automate the process with a schedule. Snapshots are useful any time you need to be able to roll back to a known good data set at a specific point in time. For example, before performing a risky operation on a volume, you can create a snapshot to enable rollback capability for the entire volume.

A snapshot takes no time to create, and initially uses no disk space, because it stores only the incremental changes needed to roll the volume back to the state at the time the snapshot was created. The storage used by a volume's snapshots does not count against the volume's quota. When you view the list of volumes on your cluster in the MapR Control System, the value of the Snap Size column is the disk space used by all of the snapshots for that volume.

Note: Snapshot volumes inherit the auditing configurations of their original read-write volumes. For details about auditing, see Auditing.

You can perform the following tasks using the MapR Control System and the CLI.

Creating Volume Snapshots

Describes how to create snapshots of volumes using the Control System and the CLI.

You can create a snapshot manually using the Control System and the CLI, or use a schedule to automate snapshot creation.

Creating Snapshots of Multiple Volumes Using the Control System

To create snapshots of multiple (standard and/or mirror) volumes manually:

1. Log in to the Control System and go to the **Summary** tab in the **Data** > **Volumes** page.
2. Select the volume(s) for which you need to create the snapshots.
3. Click **Snapshot Volume** from the **Actions** drop-down menu.
   The **Snapshot Volume(s)** confirmation dialog displays.
4. Verify the list of volumes and enter a unique name for the snapshot in the **New Snapshot Name** field.
5. Click **Snapshot Volumes**.

last-updated: Apr 11, 2022

Creating a Snapshot of a Volume Using the Control System

You can create a snapshot manually or use a schedule to automate snapshot creation. To create a snapshot of a volume manually:

1. Log in to the Control System and go to the **Snapshots** tab in the volume information page.
2. Click **Create Snapshot** to display the **Create Snapshot** window.
3. Enter a unique name for the snapshot in the **New Snapshot Name** field.
4. Click **Create Snapshot** to create a snapshot of the volume.
   By default, manually created snapshots do not have an expiration date.

Creating Volume Snapshots Using the CLI or REST API

The basic command to create a snapshot is:

```
maprcli volume snapshot create -snapshotname <snapshot> -volume <volume>
```

For complete reference information, see `volume snapshot create` on page 2028.

Viewing the list of Snapshots

Describes how to view the list of snapshots that are present on a cluster, using the Control System or the CLI.

You can view the snapshots on the cluster using the Control System or the CLI.

Viewing All the Snapshots Using the Control System

- The **Snapshots** tab under the **Data** > **Volumes** page displays all the snapshots on the cluster.
  For each snapshot, you can view the following:

Column Name	Column Description
Snapshot Name	The name of the snapshot.
Volume	The volume with which the snapshot is associated.
Created On	The date when the snapshot was created.
Expires On	The date when the snapshot expires.
Reclaim Size	TBD

  You can select one or more snapshots to:

  - Preserve
  - Remove

Viewing the Snapshots Associated with a Volume Using the Control System

- Log in to the Control System and go to the **Snapshots** tab in the volume information page.
  The list of snapshots associated with the volume displays in this tab. For each snapshot, the pane displays the following:

Column Name	Column Description
Snapshot Name	The name of the snapshot.
Volume	The volume with which the snapshot is associated.
Created On	The date when the snapshot was created.

last-updated: Apr 11, 2022

Column Name	Column Description
Expires On	The date when the snapshot expires.
Reclaim Size	TBD

You can create a snapshot of the volume or select one or more snapshots to:

- Preserve
- Remove

Viewing Snapshots Using the CLI or REST API

The basic command to retrieve a list of snapshots is:

```
maprcli volume snapshot list
```

For complete reference information, see `volume snapshot list` on page 2030.

Viewing the Contents of a Snapshot from the Command Line
Describes how to view the contents of the `.snapshot` directory from the CLI.

At the top level of each volume is a directory named `.snapshot` containing all the snapshots for the volume. You can view the directory with `hadoop fs` commands or by mounting the cluster with NFS. To prevent recursion problems, `ls` and `hadoop fs -ls` do not show the `.snapshot` directory when you list the contents of the top-level volume directory. You must navigate explicitly to the `.snapshot` directory to view and list the snapshots for the volume.

Example:

```
hadoop fs -ls /myvol/.snapshot
   Found 1 items
   drwxrwxrwx   - root root          1 2011-06-01 09:57 /myvol/.snapshot/
2011-06-01.09-57-49
```

In the preceding example, `/myvol` is the mount point of the volume for which the snapshot named `2011-06-01.09-57-49` was created and stored in the `.snapshot` directory.

Preserving one or more Snapshots
Describes how to preserve a snapshot using the `volume snapshot preserve` command, or using the Control System.
Preserving Snapshots Using the Control System

You can preserve a snapshot to prevent it from expiring. To preserve one or more snapshots, in the **Snapshots** tab (under **Data** > **Volumes**):

Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

1. Select the snapshot(s) you want to preserve.
2. Click **Preserve Snapshot** to preserve the snapshot(s).
   The **Preserve Snapshot(s)** confirmation dialog displays.
3. Verify the list of snapshots to preserve and click **Preserve Snapshots**.
   The **Expires On** column for the selected snapshots will show **No Expiration**. You *cannot* set an expiration date for a preserved snapshot; instead, if necessary, remove the preserved snapshot.

last-updated: Apr 11, 2022

Preserving Snapshots Associated with a Volume Using the Control System

You can preserve a snapshot to prevent it from expiring. To preserve one or more snapshots (and prevent them from expiring) associated with a volume:

1. Log in to the Control System and go to the **Snapshots** tab in the volume information page.
   The list of snapshots associated with the volume displays.
2. Select the snapshots to preserve and click **Preserve Snapshot**.
   The **Preserve Snapshot(s)** confirmation dialog displays.
3. Verify the list of snapshots to preserve and click **Preserve Snapshots**.
   The **Expires On** column for the selected snapshots will show **No Expiration**. You *cannot* set an expiration date for a preserved snapshot; instead, if necessary, remove the preserved snapshot.

Preserving Snapshots Using the CLI or the REST API

The basic command to preserve the snapshots is:

```
maprcli volume snapshot preserve
```

For complete reference information, see `volume snapshot preserve` on page 2034.

Removing one or more Snapshots

Explains how to remove a snapshot using the `volume snapshot remove` command or using the Control System.

Each snapshot has a date and time at which it expires. You can remove a snapshot manually before its expiration, or you can preserve a snapshot to prevent it from expiring.

Removing Snapshots Using the Control System

To remove one or more snapshots, in the **Snapshots** tab under **Data** > **Volumes**:

Note: The **Volumes** page is under the **Volumes** in the Kubernetes version of the Control System.

1. Select the snapshots to remove.
2. Click **Remove Snapshot**.
   The **Remove Snapshots** confirmation dialog displays.
3. Verify the list of snapshots to remove and click **Remove Snapshots**.
   When you remove a snapshot, the snapshot is removed from the system and cannot be restored.

Removing Snapshots Associated with a Volume Using the Control System

To remove one or more snapshots associated with a volume:

1. Log in to the Control System and go to the **Snapshots** tab in the volume information page.
   The list of snapshots associated with the volume displays.
2. Select the snapshots to remove and click **Remove Snapshot**.
   The **Remove Snapshots** confirmation dialog displays.
3. Verify the list of snapshots to remove and click **Remove Snapshots**.
   When you remove a snapshot, the snapshot is removed from the system and cannot be restored.

last-updated: Apr 11, 2022

Removing Snapshots Using the CLI or REST API

The basic command to remove a snapshot is:

```
maprcli volume snapshot remove
```

For complete reference information, see `volume snapshot remove` on page 2036.

Copying From a Snapshot Using the CLI

Describes how to create a volume by copying data from a snapshot

Copying data from a snapshot involves a simple copy operation from the `.snapshot` directory to the destination, as in the following example. User input is marked in **bold**:

Note: This example assumes that the file system is mounted on /mapr using FUSE as explained in Mounting the Filesystem.

```
[user@host]$ maprcli volume snapshot create -snapshotname
uservolsnap -volume users
[user@host]$ maprcli volume snapshot list
snapshotid  sharedSize  volumename  ownername  cumulativeReclaimSizeMB
numSizeUpdates  snapshotname  enforcementMode      ownedsize
sizeUpdateRequestedAt         ownertype  numSizeUpdatesDesired  volumeid
creationtime                  volumepath  volumeSnapshotAces
256000049   0           users       mapr       0
0               uservolsnap   PolicyAceAndDataAce  0          Mon Jun 14
05:44:11 UTC 2021  1          1                      77144951  Mon Jun 14
05:44:11 UTC 2021  /user       ...

[user@host]$ ls -l /mapr/my.cluster.com/user/
total 1
drwxr-xr-x 2 mapr mapr 2 Jun 13 14:34 mapr


[user@host]$ ls -l /mapr/my.cluster.com/user/.snapshot
total 1
drwxr-xr-x 2 mapr mapr 2 Jun 13 14:37 uservolsnap

[user@host]$ cp /mapr/my.cluster.com/user/.snapshot/uservolsnap/mapr/* /
mapr/my.cluster.com/user/
```

Managing User Disk Usage

The **User Disk Usage** tab in the **Data** > **Volumes** page displays information about disk usage by cluster users. You can perform the following tasks to manage disk quotas using the MapR Control System and the CLI.

Viewing User Disk Usage Information

Explains how to view user disk usage information using either the Control System or the CLI.

Viewing User Disk Usage Information Using the Control System

To view information about disk usage by cluster users:

- Log in to the Control System and go to the **User Disk Usage** tab under **Data** > **Volumes**.

  Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

  The disk usage information for all the users displays. For each user, the **Accountable Entities** pane displays the following:

last-updated: Apr 11, 2022

Column Name	Column Description
Type	The type of *accountable entity (AE)*. Value can be User or Group.
Accountable Entity	The user or group responsible for the volume.
Disk Usage	The total disk space used by the user.
Volume Count	The number of volumes.
Hard Quota	The user's hard quota.
Advisory Quota	The user's advisory quota.
Email	The email address of the user.

Selecting the checkbox beside an accountable entity makes the **Edit Properties** button available. You can modify the quotas for the selected entities by clicking Edit Properties. Alternatively, you can click on an entity or associated to modify the email address and quotas for the entity.

Viewing User Disk Usage Information Using the CLI or REST API

The basic command to view user disk usage information is:

```
maprcli entity info -name <entity name> -type <type>
```

For complete reference information, see `entity info` on page 1646.

Set or Modify Quotas for Users and/or Groups
Explains how to set or modify quotas for one or more entities using either the Control System or the CLI.
Set or Modify Quotas for Multiple Users and/or Groups Using the Control System

To edit the quota, which limits the space used by all the volumes owned by a user or group, for one or more users, in the **User Disk Usage** tab under **Data** > **Volumes**:

Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

1. Select the users/groups from the list of users/groups in the **Accountable Entities** pane.
2. Click **Edit Properties**.
   The **Edit Properties** dialog displays.
3. Verify the list of users/groups and modify or set the following for the users/groups:
   a) Hard quota, which raises an alarm when the limit is reached and prevents further writes.
   b) Advisory quota, which raises an alarm when the threshold is reached, but does not prevent further writes.

   Note: Both, advisory and hard, quotas can be expressed in megabytes (MB), which is the default, gigabytes (GB), or terabytes (TB).
4. Click **Save Changes** for the changes to take effect.

Set or Modify Quotas for an Entity Using the Control System

To edit the quota, which limits the space used by all the volumes owned by a user or group, for a user, in the **User Disk Usage** tab under **Data** > **Volumes**:

last-updated: Apr 11, 2022

Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

1. Click the user/group from the list of users/groups or the associated with the user/group in the **Accountable Entities** pane to display the **Edit Properties** window.
2. Modify or set the following for the user or group:
   a) Enter the email address of the user/group.
   b) Hard quota, which raises an alarm when the limit is reached and prevents further writes.
   c) Advisory quota, which raises an alarm when the threshold is reached, but does not prevent further writes.
   The advisory quota must be less than the hard quota.

   Note: Both, advisory and hard, quotas can be expressed in megabytes (MB), which is the default, gigabytes (GB), terabytes (TB), petabytes (PB), exabytes (EB), and zettabytes (ZB).
3. Click **Save Changes** for the changes to take effect.

Set or Modify Quotas for Users and/or Groups Using the CLI or the REST API

The basic command to set or modify quotas for multiple entities is:

```
maprcli entity modify
        -entities <entities>
        -advisoryquota <advisory quota>
        -quota <quota>
```

The basic command to set or modify quotas for an entity is:

```
maprcli entity modify
        -name <entityname>
        -type <type>
        -advisoryquota <advisory quota>
        -quota <quota>
```

For complete reference information, see `entity modify` on page 1649.

Related information

accounting entity (AE) on page 6593
accountable entity (AE) on page 6593

Managing Schedules

A schedule is a group of rules that specify recurring points in time at which certain actions are determined to occur. You can use schedules to automate the creation of snapshots and mirrors and the offload of volume data to a storage tier; after you create a schedule, it appears as a choice in the scheduling menu when you are creating or editing a volume.

When you specify a *snapshot* schedule on a mirror volume, it specifies how often to take a snapshot of the mirror volume. This snapshot schedule is distinct from the snapshot schedule for the standard volume. A snapshot schedule for a promotable mirror volume has two purposes:

- It specifies how often to take a snapshot of the mirror volume for the purpose of preserving the state of the mirror before a subsequent mirror operation. This way, if corrupt data is copied from the source volume's snapshot into the mirror volume, the mirror contents can be rolled back to the snapshot.

last-updated: Apr 11, 2022

- If the promotable mirror volume is promoted to a read-write volume, the snapshot schedule specified for the mirror is used for the promoted read-write volume. Once a mirror volume is promoted to a read-write volume, the mirror schedule is disabled.

A *mirror* schedule specifies how frequently the mirror volume is synchronized with the source volume. In case of a disaster (or any type of data loss on a read-write source volume), the data can be recovered from the mirror volume, but any data written to the source volume since the last successful mirror operation will not be on the mirror volume. Therefore, you should set the mirror schedule such that it meets your RPO (Recovery Point Objective).

A *tier offload* schedule specifies how frequently data in the volume on the MapR cluster is offloaded to the tiered storage. The MAST Gateway uses this setting to automatically offload data to the storage tier.

Schedules require the Enterprise Edition license.

Viewing the List of Schedules
Explains how to view all the schedules using the Control System or the CLI.
Viewing the List of Schedules Using the Control System

To view all the schedules:

Log in to the Control System and go to the **Schedules** tab under **Data** > **Volumes**.

Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

The page displays all the schedules. For each schedule, the page displays the following:

Column Name	Column Description
Schedule Name	The name of the schedule.
ID	The schedule ID.
In Use	Checkmark (✓) indicates the schedule is currently being used.
Detail	The schedule details such as the recurring points in time when the associated actions occur and how long the data is preserved.

Selecting the checkbox associated with a schedule makes the **Remove Schedule** button available. You can:

- Create a new schedule by clicking **Create Schedule**
- Remove a schedule by selecting the checkbox beside the schedule (to remove) and then by clicking **Remove Schedule**
- Edit a schedule by clicking the schedule name

Retrieving the List of Schedules Using the CLI or REST API

The basic command to retrieve a list of schedules is:

```
maprcli schedule list
```

For complete reference information, see `schedule list` on page 1740.

Creating a Schedule
Explains how to create a schedule using the Control System or the CLI.

last-updated: Apr 11, 2022

Creating a Schedule Using the Control System

A schedule is a group of rules that specify recurring points in time at which certain actions are determined to occur. You can use schedules to automate the creation of snapshots and mirrors. To create a new schedule:

1. Log in to MCS and do one of the following:
   - Go to the **Schedules** tab in the **Data** > **Volumes** page to create a schedule.
   - Go to the **Schedules** tab in the **Create Volume**, **Edit Volume**, or **Edit Volumes** page to create a schedule and associate it with the volume.
2. Click one of the following based on the page you are on to display the **Create Schedule** window:
   - **Create Schedule** button in the **Schedules** tab of the **Data** > **Volumes** page.
   - **Create** link (associated with the type of schedule) in the **Schedules** tab of the **Create Volume**, **Edit Volume**, or **Edit Volumes** page.
3. Enter a name for the schedule in the **Schedule Name** text field.
4. Specify the schedule rules with the following components:

Frequency	Specify frequency (Once, Yearly, Monthly, Weekly, Daily, Hourly, Every X minutes).
Time	Specify the point of time within the specified frequency to perform the scheduled action. For example, if you selected Monthly from the first drop-down menu, select the day of the month from the second drop-down menu. Continue with each drop-down menu, proceeding to the right, to specify the time at which the scheduled action is to occur.  **Note:** This is available only if the selected frequency is Once, Yearly, Monthly, Weekly, or Daily.
Retain for	Specify how long the data should be preserved. For example, if the schedule is attached to a volume for creating snapshots, the **Retain for** specifies how far after creation the snapshot expiration date is set.

   If necessary, click **Add Another** to add another rule to the schedule or  to remove a rule.
5. Click **Create Schedule** to create the schedule.
   After the schedule is created, it appears as a choice in the scheduling menu when you are creating a new volume or editing a volume.

Creating a Schedule Using the CLI or REST API

The basic command to create a schedule is:

```
maprcli schedule create -schedule <JSON>
```

For complete reference information, see `schedule create` on page 1739.

Guidelines for Setting Mirror Schedules

Although MapR allows mirroring frequencies up to once per minute, setting a schedule at this frequency is not practical nor advisable. When you choose the mirror schedule, consider the amount of data on the volume and the load on the cluster. Remember that the mirroring frequency must allow enough time to complete one mirror operation before the next scheduled mirror operation starts. In addition, if you have a cascaded mirror setup (where A mirrors to B which mirrors to C), you cannot set a mirror schedule for C that starts before B finishes mirroring from A.

 **Warning:** In general, you should not set a mirror schedule for more often than every 30 minutes.

last-updated: Apr 11, 2022

If you set a mirror schedule to start mirroring before the previous mirror operation finishes, you will see an error message similar to this:

```
WARN Alarms [pool-2-thread-1]: Alarm raised: VOLUME_ALARM_MIRROR_FAILURE;
Cluster: Cluster1; Volume: ; Message: Cannot start new scheduled mirror
because existing mirror is in progress
```

Modifying a Schedule
Explains how to modify a schedule using either the Control System or the CLI.
Modifying a Schedule Using the Control System

When you modify a schedule, the new set of rules replaces any existing rules for the schedule. To edit a schedule, in the **Schedules** tab under **Data** > **Volumes**:

Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

1. Click the schedule name of the schedule to edit.
   The **Edit Schedule** dialog displays.
2. Modify the schedule as desired:
   a) Change the schedule name in the **Schedule Name** field.
   b) Add, modify, or remove rules in the **SCHEDULE RULE** section.
   Here:

Frequency	Specifies the frequency (Once, Yearly, Monthly, Weekly, Daily, Hourly, Every X minutes).
Time	Specifies the point of time within the specified frequency to perform the scheduled action. This is available only if the selected frequency is Once, Yearly, Monthly, Weekly, or Daily.
Retain for	Specifies how long the data should be preserved.

   To add another rule, click **Add Another** and to remove a rule, click  .
3. Click **Save Changes** for the changes to take effect.

Modifying a Schedule Using the CLI or REST API

The basic command to modify a schedule is:

```
maprcli schedule modify -id <schedule ID> -rules <JSON>
```

For complete reference information, see `schedule modify` on page 1744.

Removing one or more Schedules
Describes how to remove schedules not associated with any volumes using either the Control System or the CLI.

You can remove a schedule only if it is not associated with any volumes.

Removing one or more Schedules Using the Control System

To remove one or more schedules, in the **Schedules** tab under **Data** > **Volumes**:

Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

1. Select the schedule(s) to remove from the list of schedules.
2. Click **Remove Schedule** to display the **Remove Schedule** confirmation dialog.

3. Verify the list of schedules to remove click **Remove Schedule** to remove the schedules.
   When you remove a schedule, the schedule is removed from the system and cannot be restored.

Removing one or more Schedules Using the CLI or the REST API

The basic command to remove a schedule by ID is:

```
maprcli schedule remove -id <schedule ID>
```

For complete reference information, see `schedule remove` on page 1745.

Managing Tiers

You can create, modify, and remove tiers using the Control System and the CLI.

Enabling Tiering

Describes how to enable data tiering using both the Control System and the CLI.

For a primer on Data Tiering, see Data Tiering on page 469.

On all new installations, the data tiering functionality is enabled and available for all new volumes. If you are upgrading, you must enable data tiering; see Step 4: Enable New Features on page 330 for more information. If the data tiering functionality is enabled, you can then selectively enable tiering for a volume at the time of volume creation using the Control system and the CLI.

Data tiering is only available for new volumes; you cannot enable data tiering for existing volumes. Enable tiering for new volumes where read/write latency is not the dominant concern. You can decide later whether you want to do local (or warm) or remote (or cold) tiering. Data tiering cannot be disabled after it is enabled for a volume.

Enabling Tiering Using the Control System

1. Move the **Data Tier** slider to **Yes** (to enable tiering) in the **Create New Volume** page in the Control System.

   Proceed to the next step only if you wish to select a tier type for the volume. You can create a tiering-enabled volume without selecting a tier type and select a tier type later by editing the volume.

   Note: You cannot disable tiering for a volume after it is enabled.

2. (Optional) Select **Erasure Coding** (for warm tiering) or **Remote Archiving** (for cold tiering) from the **Tiering Type** drop-down menu.

   You:

   - Can enable a volume for either warm or cold tiering, but not for both.
   - Cannot modify the tier type after the volume is created.

3. Specify all other required and optional properties for creating the volume and click **Create Volume**.

   For information on required and optional properties, see Creating a Volume on page 864.

Enabling Tiering Using the CLI

- Run the following command to enable tiering:

  ```
  maprcli volume create -name <vol-name> -path <mount-path> -tieringenable
  true
  ```

  For more information, see `volume create` on page 1931.

last-updated: Apr 11, 2022

Advantages of Parallel Offload

Resiliency with Parallel Offload

Parallel Offload is resilient to the following scenarios:

Restart of the Primary Gateway	• Secondary gateways continue to run assigned tasks while the primary gateway is down or restarting. • CLDB reassigns the volume to another primary gateway. • CLDB restarts the tasks on the new primary gateway. • The primary gateway polls/reschedules the ongoing secondary gateway tasks.
Restart of the Secondary Gateway	• The primary gateway detects the failure of secondary gateway tasks when it polls the secondary gateway. • The primary gateway reschedules tasks that were terminated when the secondary gateway restarted.
Restart/Switchover of CLDB	• Reassign volume to the same primary gateway. • Reschedule pending volume task on the same primary gateway.

Load Balancing with Parallel Offload

Load Balancing involves:

Volume Level	• CLDB assigns each volume to a gateway with the least number of volumes. • Gateway Balancer reassigns volumes across gateways.
Task Level	CLDB balances tasks across MFS nodes.

Enabling Parallel Offload on an Upgraded Cluster

Creating a Storage Tier

Describes how to create a sotrage tier using the Control System and the CLI.

For a primer on Data Tiering, see Data Tiering.

You can create a tier using the Control System and the CLI.

When you create a tier, MapR File System creates a volume, whose mount point is `/var/mapr/tier/mapr.internal.tier.<tiername>`, for the tier. For warm tier volumes automatically created using the `ecenable` parameter or the Control System, by default, a corresponding tier volume named `mapr.internal.tier.autoec.<volName>.<creationTime>` is created in the `/var/mapr/autoectier` path. The tier volume is visible and stores all the metadata tables and information on all the jobs running on the tier. Do not modify, move, or remove this volume.

last-updated: Apr 11, 2022

Note: If the number of cluster nodes is more than five, by default, MapR (through the `enforceminreplicationforio` parameter) requires minimum number of copies of the tier volume for writes to succeed. If the number of cluster nodes is less than five, MapR does not enforce minimum number of copies for writes to succeed.

Creating a Warm Tier Using the Control System

When you create a volume enabled for erasure coding, the control system automatically creates a warm tier and associates the volume with that tier. See Creating a Volume on page 864 for more information. You cannot create a warm tier separately using the Control System.

Creating a Cold Tier Using the Control System

To create a cold tier:

1. Log in to the Control System, click **Data** > **Volumes**, and then do one of the following:
   - Go to the **Remote Targets** tab if you wish to create a remote target that is not (yet) associated with a volume.
   - Click **Create Volume** if you wish to create a remote target for a volume when you are creating the volume.

     Note: You must enable data tiering and select **Remote Archiving (Cold)** from the **Tiering Type** drop-down list to create the remote target.
   - Click **Edit Volume** in the volume information page if you want to create a remote target for the volume when you are editing the volume settings.

     Note: You can create a remote target only if a remote target is already not associated with the volume.
2. Click one of the following to display the **Create Remote Target** window.
   - **Create Target** if you are in the **Remote Targets** tab.
   - **Create** link associated with the **Remote Target** field if you are in the **Create New Volume** page.
   - **Create** link associated with the **Remote Target** field if you are in the **Edit Volume** page.
3. Specify a name for the tier in the **Remote Target Name** field.
4. Select a topology for the metadata volume associated with the tier from the list of topologies in the **Lookup Topology** drop-down menu.

   The volume stores all the metadata tables and information on all the jobs running on the tier. If many volumes share the same tier (and thus the same lookup topology), the lookups might have an adverse affect by inadvertently adding background load to nodes in that topology. This property allows you to setup the lookups on other nodes.
5. Specify the following properties.

last-updated: Apr 11, 2022

Vendor	The vendor from the **Vendor** drop-down list. **Note:** For cold tiering, MapR supports the following vendors: • AWS (Amazon AWS) • GCS (Google Cloud Platform) • HDS (Hitachi HCP) • IBM (IBM Cleversafe) • Azure Blob (Microsoft Azure) • Others (such as Minio) See Specifying the Vendor/Object Store for a Cold Tier on page 964 for more information on the supported vendors.
URL	The URL to use to connect to the tier in the following format: `<protocol>://<IP|hostname>.<domain>` If the protocol is `https`, the MAST Gateway uses HTTPS to upload data to the cold-tier. If the cold-tier storage does not support HTTPS, all tier related operations will fail. If the cold tier does not support HTTPS, set the protocol to `http`. See Specifying the Vendor/Object Store for a Cold Tier on page 964 for more information on the tier endpoint URLs and supported authentication protocols.
Bucket Name	The name of the bucket on the tier. This cannot be modified once associated with a tier. If the bucket does not already exist on the tier, MapR will attempt to create the bucket. **Note:** For Azure, the bucket/container is created in the region that is specified in the parent storage account.
Region	The S3 region to create the bucket on. This cannot be modified once configured (explicitly or using the default) and associated with a tier. See region on page 963 for more information.

6. Enter the credentials for accessing the tier in the **Access Key** and **Secret Key** fields.

   Tip: For Azure Blobs, the storage account name is the access key.

7. Click **Create Target** to create the cold tier that you can associate with a volume.

Creating a Tier Using the CLI and the REST API

CLI	Run the following command to create a tier:

last-updated: Apr 11, 2022

- Cold tier:

```
$maprcli tier
create -name <tier_name> -type
cold -url <tier_url> -credential
<credentials_file_path>
```

For using the `-credential` option, you must have the credential file already set up as described in Setting up a Credentials File for Connecting to a Cold Tier Using the CLI or REST API on page 962. On the other hand, if you do not have the file already set up, use the `-credential_str` option as follows:

```
$maprcli tier create -name
<tier_name> -type cold -url
<tier_url> -credential_str
'<credential_string>'
```

- Warm tier with default values:

```
$maprcli tier create -name
<tier_name> -type ectier
```

Note: You can associate the same tier with multiple volumes with different erasure coding scheme.

REST

Send a request of type POST. For example:

- Cold Tier:

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/create?
name=egColdTier&type=cold&url=s3.am
azonaws.com&credential=credentials.
txt' --user mapr:mapr
{"timestamp":1525724933919,"timeofd
ay":"2018-05-07 01:28:53.919
GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully
created tier: 'egColdTier'"]}
```


last-updated: Apr 11, 2022

- Warm tier:

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/create?
name=egWarmTier&type=ectier' --user
 mapr:mapr
{"timestamp":1525725105206,"timeofd
ay":"2018-05-07 01:31:45.206
GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully
created tier: 'egWarmTier'"]}
```

Note: You can associate the same tier with multiple volumes with different erasure coding scheme.

For more information, see `tier create` on page 1874.

Setting up a Credentials File for Connecting to a Cold Tier Using the CLI or REST API

Describes how to create a credential file, with examples for AWS and Microsoft Azure.

Set up a credentials file on the host you plan to use for creating the cold tier if you are planning on using the `credential` option (and not pass the credentials on the command-line using the `credential_str` option).

For example, your `credentials.txt` file for AWS should look similar to the following sample file:

```
{
      "bucketName" : "defaultbucket3",
      "region": "us-east-1",
      "credentials" : {
         "accessKey" : "AB956CDE8F2GO7H9I4J2",
         "secretKey" : "5K1LmN92e65oPQRsTUvOfSbURxyEtYl2MmAocGi"
      }
}
```

The sample for Microsoft Azure is as follows:

```
maprcli tier create -name tier_azure_1 -type cold -url https://
myobjectpools1.blob.core.windows.net -credential ~/creds_azure.txt
$ cat creds_azure.txt
      {
      "bucketName" : "bucket4",
      "credentials" : {
      "accessKey" : "myobjectpools1",
      "secretKey" :
"N6GKkDPttqNc6rfTzhb2JNKwvdr9EraN89Mg8WaodrvPBeINBTZwhQu+Q3vX4ENeW+RQN42f+P8
nXN0YasZWNA=="
                     }
      }
```

The credentials file (`.txt` file) contains the following properties in JSON format:

bucketName	The name of the bucket on the tier. If the bucket does not already exist on the tier, the command to create the tier attempts to create the bucket using the credentials in the credentials file. You can modify the name of the bucket only by using the `-force` option with the `tier modify` on page 1882 command

region

The S3 region to create the bucket on. Use the information in the following table to specify region information.

Vendor	Default Value	Notes
AWS	us-east-1	Specify region information as defined here. On AWS, each region can have a different URL. The URL must be provided with the `maprcli tier create` command. **Note:** Because bucket names are unique across regions, make sure you specify the correct region for a given bucket in the credentials file. For example, suppose a bucket called myBucket3 in us-east-1; you cannot offload data to myBucket3 by specifying us-west-1 as the region in the credentials file.
GCS	us-east-1	Specify region information as described here.
HDP	N/A	Not required. If specified, MapR ignores the value.

last-updated: Apr 11, 2022

Vendor	Default Value	Notes
IBM	us-east-region	Not required. If specified, MapR ignores the value.
Azure-Blobs	N/A	Not required. The region of the storage account is determined from the URL and data is offloaded into that region.
Minio	us-east-1	If you specify region, export the `MINIO_REGION` environment variable on the minio server as described in the Configuration Guide.
Scality	us-east-region	Not required. If specified, MapR ignores the value.

You can modify the region only by using the `-force` option with the `tier modify` on page 1882 command.

accessKey and secretKey

The credentials for accessing the tier.

Tip: For Azure Blobs, the storage account name is the access key.

Note: Once the tier is created, MapR does not require this file because CLDB stores the bucket, region, and credentials information.

Specifying the Vendor/Object Store for a Cold Tier

Specify the vendor or object store where you plan to offload the (cold) data. The following table lists the supported vendors, URL of the tier endpoint, and authentication protocol supported by MapR:

last-updated: Apr 11, 2022

Supported Vendor/Object Store	Tier URL/Endpoint	Supported Authentication Protocol
AWS (Amazon AWS)	The URL varies based on the region. For: • us-east-1: https://s3.amazonaws.com • us-east-2: https://s3.us-east-2.amazonaws.com • us-west-1: https://s3-us-west-1.amazonaws.com • us-west-2: https://s3-us-west-2.amazonaws.com • ap-south-1: https://s3.ap-south-1.amazonaws.com • ap-southeast-1: https://s3-ap-southeast-1.amazonaws.com • ap-southeast-2: https://s3-ap-southeast-2.amazonaws.com • ap-northeast-1: https://s3-ap-northeast-1.amazonaws.com • ap-northeast-2: https://s3.ap-northeast-2.amazonaws.com • ap-northeast-3: https://s3.ap-northeast-3.amazonaws.com • ca-central-1: https://s3.ca-central-1.amazonaws.com • cn-north-1: https://s3.cn-north-1.amazonaws.com • cn-northwest-1: https://s3.cn-northwest-1.amazonaws.com • eu-central-1: https://s3.eu-central-1.amazonaws.com • eu-west-1: https://s3-eu-west-1.amazonaws.com • eu-west-2: https://s3.eu-west-2.amazonaws.com • eu-west-3: https://s3.eu-west-3.amazonaws.com	• HTTP • HTTPS

Supported Vendor/Object Store	Tier URL/Endpoint	Supported Authentication Protocol
GCS (Google Cloud Platform)	`https://storage.googleapis.com`	• HTTP • HTTPS
HDS (Hitachi HCP)	`http://<hcptenant>.<hcphostname>` `https://<hcptenant>.<hcphostname>`	• HTTP • HTTPS
IBM (IBM Cleversafe)	`http://lb1.ait.cleversafelabs.com`	HTTP
Azure Blob (Microsoft Azure)	`https://<azureaccount>.blob.core.windows.net`	• HTTP • HTTPS
Minio	`http://10.10.88.198:9000`	HTTP
Scality	`https://<scality-instance-hostname>:8000`	HTTP

Viewing the List of Tiers

Describes how to view the list of tiers using the Control System, the CLI, and the REST API.

For a primer on Data Tiering, see Data Tiering.

In the Control System, you can only see the list of cold tiers (referred to as remote targets). Use the CLI or REST API to retrieve the list of both cold and warm tiers.

Viewing the List of Remote Targets Using the Control System

- Log in to the Control System and click **Data** > **Volumes** > **Remote Targets**.

  The page displays the following for each remote target:

Column Name	Column Description
Remote Target Name	The name of the remote target.
External Storage Vendor	The name of the third-party storage vendor.
Bucket	The name of the bucket.
Region	The region where the bucket resides.
URL	The URL of the remote target.

You can:

- Create a remote target
- Remove a remote target

Viewing the List of Tiers Using the CLI and REST API

CLI	Run the following command to retrieve the list of (warm and cold) tiers: `/opt/mapr/bin/maprcli tier list`

last-updated: Apr 11, 2022

REST

Send a request of type GET. For example:

```
# curl -k -X GET 'https://
10.10.82.24:8443/rest/tier/
list' --user mapr:mapr
{"timestamp":1525725765483,"timeofday"
:"2018-05-07 01:42:45.483 GMT-0700
PM","status":"OK","total":0,"data":
[{"tierid":"79082078","tiername":"egWa
rmTier","tiertype":"ectier","volume":"
mapr.internal.tier.egWarmTier","topolo
gy":"/data"},
{"tierid":"120198852","tiername":"egCo
ldTier","tiertype":"cold","url":"s3.am
azonaws.com","bucketname":"ksekhar-tes
t","region":"us-east-1","volume":"mapr
.internal.tier.egColdTier","topology":
"/data","objectstoretype":"S3_AWS"},
{"tierid":"158778422","tiername":"auto
ec.vol_tiered.1525463214","tiertype":"
ectier","volume":"mapr.internal.tier.a
utoec.vol_tiered.1525463214","topology
":"/data"}]}
```

For more information, see `tier list` on page 1880.

Editing a Tier

Describes how to modify a cold tier using the Control System, the CLI and the REST API.

For a primer on Data Tiering, see Data Tiering.

You cannot modify a warm tier. You can modify a cold tier (referred to as Remote Target in the Control System) using the Control System, the CLI, and REST API.

Modifying a Remote Target Using the Control System

1. Log in to the Control System and go to the **Remote Target** tab under **Data** > **Volumes**.
2. Click the name of the remote target (cold tier) to display the **Edit Remote Target** window.
3. Make necessary changes to the **CREDENTIALS**.
4. Click **Save Changes** to save the changes.

Modifying a Cold Tier Using the CLI and REST API

CLI

Run the following command to modify a cold tier:

```
$maprcli tier
modify -name <tier_name> -credential
<credentials_file_path>
```

REST

Send a request of type POST. For example:

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/modify?
name=<tier_name>&credential_str=<JSON>
' --user mapr:mapr
```

For more information, see `tier modify` on page 1882.

last-updated: Apr 11, 2022

Specifying a Tier

Describes how to associate a tier with a tiering-enabled volume using the Control System and the CLI.

Note:

For a primer on Data Tiering, see Data Tiering.

Using the Control System, you can only associate an exiting tier (referred to as **Remote Target** in the Control System) with a volume enabled for Remote Archiving (or cold-tier). You cannot associate an existing tier with a volume enabled for Erasure Coding (or warm-tier) because the Control System allows a new tier to be automatically created when you enable a volume for erasure coding. If you want to associate an existing tier with a volume enabled for erasure coding, use the CLI or REST API to create the volume.

Specifying a Remote Target Using the Control System

You can associate a remote target with a cold-tier enabled volume when you are:

- Creating the volume by clicking **Create Volume** button in the **Data** > **Volumes** page.
- Editing the volume by clicking **Edit Volume** button in the volume information page.

To associate a remote target with the volume, in the **Create Volume** or **Edit Volume** page:

1. Click the **Browse** link associated with the **Remote Target** field to display the **Browse Remote Target** window.
2. Review the name, vendor, bucket, region, and URL for each remote target and choose a remote target from the list.
3. Click **Select** to associate the remote target with the volume.
4. Complete the steps for creating or editing the volume.

Specifying a Tier Using the CLI and REST API

You can associate an existing tier with a volume when you are creating the tiering-enabled volume. You can associate an existing tier with a tiering-enabled volume when you are editing the volume only if the volume does not already have a tier associated with it. To associate an existing tier, you must specify the `tiername` parameter with the command.

CLI

Run a command similar to the following to associate a tier when:

- Creating a volume:

```
maprcli volume
create -name <volName> -path
<mountPath> -tieringenable
true -tiername <tierName> -json
```

For the list of all other required and optional parameters, see `volume create` on page 1931.

last-updated: Apr 11, 2022

- Editing the volume:

```
maprcli volume
modify -name <volName> -tiername
<tierName> -json
```

For the list of all other required and optional parameters, see volume modify on page 2005.

REST

Send a request of type POST. For example, to associate a tier when:

- Creating a volume:

```
curl -k -X POST 'https://
<host>:8443/rest/volume/create?
name=<volName>&path=<mountPath>&tie
ringenable=true&tiername=<tierName>
&tieringrule=<ruleName>' --user
mapr:mapr
```

For the list of all other required and optional parameters, see volume create on page 1931.

- Editing the volume:

```
curl -k -X POST 'https://
<host>:8443/rest/volume/modify?
name=<volName>&tieringrule=<ruleNam
e>' --user mapr:mapr
```

For the list of all other required and optional parameters, see volume modify on page 2005.

Removing a Tier

Describes how to remove a tier that is not associated with a tier, using the Control System, the CLI and the REST API.

For a primer on Data Tiering, see Data Tiering.

You can remove a tier that is not associated with a volume, using the Control System, the CLI, and REST API. In the Control System, a cold tier is referred to as remote target and you can only remove remote targets (or cold tiers) using the Control System. Use the CLI or REST API to remove cold and warm tiers.

Removing a Remote Target Using the Control System

1. Log in to the Control System and click **Data** > **Volumes** > **Remote Targets** to display the list of remote targets.
2. Select the checkbox associated with the tier to delete.
   Selecting the checkbox associated with a tier makes the **Remove Target** button available.
3. Click **Remote Target** to display the **Remove Remote Target** confirmation dialog.
4. Verify the list of remote targets to remove and click **Remove**.

last-updated: Apr 11, 2022

Removing a Tier Using the CLI and REST API

CLI

Run the following command to remove a tier:

```
/opt/mapr/bin/maprcli tier
remove -name <tier-name>
```

You cannot remove a tier associated with a volume. If you run the command to remove a tier that is associated with a volume, the command returns an error (shown in bold) similar to the following:

```
{
   "timestamp":1516771078126,
   "timeofday":"2018-01-23
09:17:58.126 GMT-0800",
   "status":"ERROR",
   "errors":[{
       "id":10003,
        "desc":"Cannot remove tier,
as some volumes are still using it."
   }]
}
```

REST API

Send a request of type POST to remove a tier. For example:

```
curl -k -X
POST 'https://abc.sj.us:8443/rest/
tier/remove?name=ksTestTier' --user
mapr:mapr
```

You cannot remove a tier associated with a volume. If you send a request to remove a tier that is associated with a volume, an error (shown in bold) similar to the following is returned:

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/remove?
name=ksTestTier' --user mapr:mapr
{"timestamp":1524675381333,"timeofday"
:"2018-04-25 09:56:21.333 GMT-0700
AM","status":"ERROR","errors":
[{"id":10003,"desc":"Can not remove
tier, as some volumes are still using
it."}]}
```

For more information, see tier remove on page 1884.

Managing Storage Policies

Data offload is driven by rules, which are configured per volume. Data offload rule can be based on size of file (s), owner (u, g, or p) of the file, and/or file modification timestamp (m). You can apply one rule per volume.

When a rule is associated with a volume, the rule is first applied on the files in the tiering-enabled volume. When applied on the files in the tiering-enabled volume, the offload is triggered for all files in the snapshot chain as well when the criteria in the rule is met. If the file does not exist in the tiering-enabled volume, rule is applied on the latest state of the file in the snapshot chain. If the file exists in the tiering-enabled volume but has no latest state or if the file was deleted in the tiering-enabled volume, offload does not happen.

last-updated: Apr 11, 2022

Rules can be defined using a combination of the following:

`u`	Username or user ID, as configured in the OS registry (such as `/etc/passwd` file, LDAP, etc.), of a specific user. **Usage:** `u:<username or user ID>`
`g`	Group name or group ID, as configured in the OS registry (such as `/etc/group` file, LDAP, etc.), of a specific group. **Usage:** `g:<groupname or group ID>`
`m`	(`mtime`) Time (in seconds or days) since the files were last modified. The number of seconds can be specified by appending `s` to value and the number of days can be specified by appending `d` to the value. **Usage:** • `"m:<value>s"` — specifies mtime in seconds • `"m:<value>d"` — specifies mtime in days All files that are not modified since the specified amount of time, are offloaded. **Note:** If the system time on CLDB and file server nodes are different, the mtime rule for offloading data may not work as intended.
`s`	The size of the file in bytes, kilobytes, megabytes, or gigabytes. The size of the file can be specified by appending one of the following to the value: `b` for bytes, `k` for kilobytes, `m` for megabytes, or `g` for gigabytes. **Usage** • `"s:<value>b"` — specifies file size in bytes • `"s:<value>k"` — specifies file size in KB • `"s:<value>m"` — specifies file size in MB • `"s:<value>g"` — specifies file size in GB All files whose size exceeds the specified size are offloaded.

Or, use the following:

`p`	(Default) Specifies all files. Specifies that this operation is applicable to all the files without restriction. This cannot be combined with any other operator.
`""`	Indicates none of the files. Specifies that this operation cannot be performed on any of the files.

Use the following to string multiple criteria for offload:

`&`	AND operation to combine multiple expressions as the criteria for the rule.
`|`	OR operation to indicate either of the expressions as the criteria for the rule.

last-updated: Apr 11, 2022

( )	Delimiters for subexpressions.

For volumes configured for erasure coding, a default storage policy, `default.ectier.rule` (ID `1` and expression p), is applied if one is not specified.

You can create, associate, and remove rules using the MapR Control System, the CLI, and REST API.

Creating a Storage Tier Policy
Explains how to create a tiering policy for storage using either the Control System, the CLI, or the REST API.
Creating a Storage Tier Policy Using the Control System

To create a storage tier policy (or rule) using the Control System:

1. Log in to the Control System, click **Data** > **Volumes**, and then do one of the following:

   Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

   - Go to the **Storage Policies** tab if you wish to create a storage policy that is not (yet) associated with a volume.
   - Click **Create Volume** if you wish to create a storage policy for a volume when you are creating the volume.

     Note: You must enable data tiering to create the storage policy.

   - Click **Edit Volume** in the volume information page if you wish to create a storage policy for a volume when you are editing the volume settings.

2. Click one of the following to display the **Create Storage Policy** dialog.
   - **Create Policy** if you are in the **Storage Policies** tab.
   - **Create** link associated with the **Storage Policy** field if you are in the **Create New Volume** page.
   - **Create** link associated with the **Storage Policy** field if you are in the **Edit Volume** page.
3. Enter a name for the storage policy in the **Storage Policy Name** text field.
4. Choose **Build rule** or **Rule expression** radio button to define the criteria for offloading data.

   Use the **Build rule** option to build simple rules. Click `Add Condition` to add one of the following entities:

   - Group
   - User
   - File size
   - Time since the file was modified
   - Time since the file was accessed

   Use a condition group, to add AND and OR conditions.

   Click `Add condition group` to add AND and OR conditions. You can toggle the AND and OR conditions as needed.

   Use the **Rule expression** option to create advanced rules that comprise a combination of the following expressions:

u	Username or user ID, as configured in the OS registry (such as `/etc/passwd` file, LDAP, etc.), of a specific user. **Usage:** `u:<username or user ID>`
g	Group name or group ID, as configured in the OS registry (such as `/etc/group` file, LDAP, etc.), of a specific group. **Usage:** `g:<groupname or group ID>`
m	(`mtime`) Time (in seconds or days) since the files were last modified. The number of seconds can be specified by appending `s` to value and the number of days can be specified by appending `d` to the value. **Usage:** • `"m:<value>s"` — specifies mtime in seconds • `"m:<value>d"` — specifies mtime in days All files that are not modified since the specified amount of time, are offloaded. **Note:** If the system time on CLDB and file server nodes are different, the mtime rule for offloading data may not work as intended.
s	The size of the file in bytes, kilobytes, megabytes, or gigabytes. The size of the file can be specified by appending one of the following to the value: `b` for bytes, `k` for kilobytes, `m` for megabytes, or `g` for gigabytes. **Usage** • `"s:<value>b"` — specifies file size in bytes • `"s:<value>k"` — specifies file size in KB • `"s:<value>m"` — specifies file size in MB • `"s:<value>g"` — specifies file size in GB All files whose size exceeds the specified size are offloaded.

Or, use the following:

p	(Default) Specifies all files. Specifies that this operation is applicable to all the files without restriction. This cannot be combined with any other operator.
`""`	Indicates none of the files. Specifies that this operation cannot be performed on any of the files.

Use the following to string multiple criteria for offload:

&	AND operation to combine multiple expressions as the criteria for the rule.
|	OR operation to indicate either of the expressions as the criteria for the rule.
( )	Delimiters for subexpressions.

last-updated: Apr 11, 2022

If a rule is not defined, the default rule, which is all files (p), is associated with the storage policy.

5. Click **Create Policy** to create the storage policy.

Creating a Rule Using the CLI and REST API

CLI

Run the following command to create a rule:

```
$ maprcli tier rule create -name
<rule_name> -expr <expressions>
```

REST

Send a request of type POST. For example:

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/rule/create?
name=rule1&expr=m:365d' --user
mapr:mapr
{"timestamp":1519681475025,"timeofday"
:"2018-02-26 01:44:35.025 GMT-0800
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
rule: 'rule1'"]}
```

For more information, see `tier rule create` on page 1885.

Viewing the List of Storage Tier Policies
Explains how to view tiering policies for storage using either the Control System, the CLI, or the REST API.
Viewing the List of Storage Tier Policies Using the Control System

- Log in to the Control System and click **Data** > **Volumes** > **Storage Policies**.

  Note: The **Storage Policies** tab is under the **Volumes** menu in the Kubernetes version of the Control System.

  The list of storage policies displays. For each storage policy, the page displays the following:

Column Name	Column Description
Policy Name	The name of the policy.
Detail	The policy details.

  You can:

  - Create a Policy
  - Edit a Policy
  - Remove a Policy

Viewing the List of Storage Tier Policies Using the CLI and REST API

CLI

Run the following command to retrieve the list of tiers:

```
# maprcli tier rule list
```

REST

Send a request of type GET. For example:

```
# curl -k -X GET 'https://
abc.sj.us:8443/rest/tier/rule/
```

```
list' --user mapr:mapr
{"timestamp":1525728727729,"timeofday"
:"2018-05-07 02:32:07.729 GMT-0700
PM","status":"OK","total":6,"data":
[{"ruleid":"1","rulename":"default.ect
ier.rule","expression":"m:1d"},
{"ruleid":"2","rulename":"rule2","expr
ession":"s:5g"},
{"ruleid":"3","rulename":"rule1","expr
ession":"m:365d"},
{"ruleid":"4","rulename":"rule3","expr
ession":"u:m7user1"},
{"ruleid":"5","rulename":"rule4","expr
ession":"p"},
{"ruleid":"6","rulename":"testRule","e
xpression":"u:m7user1 | (u:mapr &
(s:5g | m:365d))"}]}
```

For more information, see `tier rule list` on page 1892.

Modifying a Storage Tier Policy
Explains how to modify a storage tier policy using either the Control System, CLI, or the REST API.

If you modify a rule that is currently in use, the changes in the rule are only applied on future offloads; data offloaded using existing rule is not impacted by the change in the rule.

Modifying a Rule Using the Control System

1. Log in to the Control System and go to **Storage Policies** tab in the **Data** > **Volumes** page.

   Note: The **Storage Policies** tab is under the **Volumes** menu in the Kubernetes version of the Control System.

   The list of storage policies displays.

2. Click the storage policy name to display the **Edit Storage Policy** window.

3. Make changes to the rule:
   You can modify the basic rule to:

   - Add (⊕) or remove (⊖) users and/or groups.
   - Change the name of the users and/or groups.
   - Change the number of days since the file was last modified for users and/or groups.

   If you switch from a basic rule to an advanced rule, all expressions from the basic rule are carried over to the advanced rule. You can modify an advanced rule using a combination of the following expressions:

u	Username or user ID, as configured in the OS registry (such as `/etc/passwd` file, LDAP, etc.), of a specific user. **Usage:** `u:<username or user ID>`
g	Group name or group ID, as configured in the OS registry (such as `/etc/group` file, LDAP, etc.), of a specific group. **Usage:** `g:<groupname or group ID>`

`m`	(`mtime`) Time (in seconds or days) since the files were last modified. The number of seconds can be specified by appending `s` to value and the number of days can be specified by appending `d` to the value. **Usage:** • `"m:<value>s"` — specifies mtime in seconds • `"m:<value>d"` — specifies mtime in days All files that are not modified since the specified amount of time, are offloaded. **Note:** If the system time on CLDB and file server nodes are different, the mtime rule for offloading data may not work as intended.
`s`	The size of the file in bytes, kilobytes, megabytes, or gigabytes. The size of the file can be specified by appending one of the following to the value: `b` for bytes, `k` for kilobytes, `m` for megabytes, or `g` for gigabytes. **Usage** • `"s:<value>b"` — specifies file size in bytes • `"s:<value>k"` — specifies file size in KB • `"s:<value>m"` — specifies file size in MB • `"s:<value>g"` — specifies file size in GB All files whose size exceeds the specified size are offloaded.

Or, use the following:

`p`	(Default) Specifies all files. Specifies that this operation is applicable to all the files without restriction. This cannot be combined with any other operator.
`""`	Indicates none of the files. Specifies that this operation cannot be performed on any of the files.

Use the following to string multiple criteria for offload:

`&`	AND operation to combine multiple expressions as the criteria for the rule.
`|`	OR operation to indicate either of the expressions as the criteria for the rule.
( )	Delimiters for subexpressions.

You cannot switch from an advanced rule that includes the following to a basic rule because the following are not supported in a basic rule:

- `p` — All the files
- `s` — The size of the file
- `&` — The AND operation used for specifying multiple users (u), groups (g), or criteria
- `|` — The OR operation used with `s` or `m`

last-updated: Apr 11, 2022

- "" — None of the files.
- ( ) — Subexpressions

Note: The basic rule must contain mtime (`m`). It can also include one or more users or groups separated by the OR operation (`|`).

4. Click **Save Changes** to save the storage policy changes.

Modifying a Rule Using the CLI and the REST API

CLI

Run the following command to modify a storage policy:

```
$ maprcli tier rule modify -name
<rule_name> -json
```

REST API

Send a request of type POST. For example:

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/rule/modify?
name=sampleRule&expr=m:3d' --user
mapr:mapr
{"timestamp":1523587392465,"timeofday"
:"2018-04-12 07:43:12.465 GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully updated
rule: 'sampleRule'"]}
```

For more information, see `tier rule modify` on page 1893.

Specifying a Storage Tier Policy
Explains how to associate a storage tier policy with a tiering-enabled volume using either the Control System or the CLI.

Specifying a Storage Tier Policy Using the Control System

You can associate a storage policy with a tiering-enabled volume when you are:

- Creating a volume by clicking **Create Volume** in the **Data** > **Volumes** page.

  Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

- Editing the tiering-enabled volume by clicking **Edit Volume** button in the volume information page.

To associate a storage policy with the volume, in the **Create Volume** or **Edit Volume** page:

1. Click the **Browse** link associated with the **Storage Policy** field to display the **Browse Storage Policies** window.
2. Review the name and detail of each storage policy and choose a storage policy from the list.
3. Click **Select** to associate the storage policy with the volume.
4. Complete the steps for creating or editing the volume.

Specifying a Storage Tier Policy Using the CLI and REST API

You can associate a rule with a tiering-enabled volume by specifying the `tieringrule` parameter with the `volume create` on page 1931 or `volume modify` on page 2005 command.

last-updated: Apr 11, 2022

CLI

Run a command similar to the following to associate a rule when:

- Creating a volume:

```
maprcli volume
create -name <volName> -path
<mountPath> -tieringenable
true -tiername
<tierName> -tieringrule
<ruleName> -json
```

  For the list of all other required and optional parameters, see `volume create` on page 1931.

- Editing the volume:

```
maprcli volume modify -name
<volName> -tieringrule
<ruleName> -json
```

  For the list of all other required and optional parameters, see `volume modify` on page 2005.

REST

Send a request of type POST. For example, to associate a rule when:

- Creating a volume:

```
curl -k -X POST 'https://
<host>:8443/rest/volume/create?
name=<volName>&path=<mountPath>&tie
ringenable=true&tiername=<tierName>
&tieringrule=<ruleName>' --user
mapr:mapr
```

  For the list of all other required and optional parameters, see `volume create` on page 1931.

- Editing the volume:

```
curl -k -X POST 'https://
<host>:8443/rest/volume/modify?
name=<volName>&tieringrule=<ruleNam
e>' --user mapr:mapr
```

  For the list of all other required and optional parameters, see `volume modify` on page 2005.

Removing a Storage Policy

Explains how to remove tiering policies for storage using either the Control System, the CLI, or the REST API.

Warning: You cannot remove a storage policy that is associated with a volume.

Removing a Rule Using the Control System

1. Log in to the Control System and go to **Storage Policies** tab under **Data** > **Volumes**.

   Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

   The list of storage policies displays.

2. Select the checkbox associated with the storage policy to remove and click **Remove Policy**.
   The **Remove Policy** confirmation window displays.

3. Review the list of policies to remove and click **Remove**.

Removing a Rule Using the CLI and the REST API

CLI

Run the following command to remove a storage policy that is not associated with a volume:

```
# maprcli tier rule remove -name
<rule_name>
```

If you try to remove a rule associated with a volume, the command returns an error (shown in bold) similar to the following:

```
# maprcli tier rule remove -name
rule1 -json
{
   "timestamp":1516771655669,
   "timeofday":"2018-01-23
09:27:35.669 GMT-0800",
   "status":"ERROR",
   "errors":[{
      "id":10003,
      "desc":"Cannot remove rule, as
some volumes are still using it."
   }]
}
```

REST

Send a request of type POST. For example:

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/rule/remove?
name=sampleRule' --user mapr:mapr
{"timestamp":1523571783113,"timeofday"
:"2018-04-12 03:23:03.113 GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully deleted
rule: 'sampleRule'"]}
```

If you try to remove a storage policy associated with a volume, the response contains an error (shown in bold) similar to the following:

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/rule/remove?
name=sampleRule' --user mapr:mapr
{"timestamp":1523571741636,"timeofday"
:"2018-04-12 03:22:21.636 GMT-0700
PM","status":"ERROR","errors":
```

```
[{"id":10003,"desc":"Can not remove
rule, as some volumes are still using
it."}]}
```

For more information, see `tier rule remove` on page 1895.

Administering Files and Directories

The following sections provide configuration information that you can use to set chunk size and compression in MapR File System, as well as information on hard links, extended attributes, and core files:

Using Global File System Checking

Describes how to use the `gfsck` command to check and repair filesystem errors.

You can use the `gfsck` on page 2102 (global filesystem check) command to perform a consistency check and repair operation on a volume or volume snapshot, including the following entities:

- All cross-container links (for example, from file to *filelet*, or from table to tablets)
- The tabletmap key range
- The attributes of *filelet* (uid/gid/mode)

This command identifies the unreachable files, directories, and tables in the volume, and moves them to */lost+found* to be repaired. It also identifies and fixes any unreachable DB inodes or dangling pointers to lost inodes.

1. Take the affected storage pools offline by running the mrconfig sp offline on page 2175 command.
   For example:

   ```
   /opt/mapr/server/mrconfig sp offline /dev/sdc
   ```

2. Execute the `fsck` on page 2100 command on the storage pools or disks.
3. Bring the storage pools back online by running the `mrconfig sp online` on page 2176 command.
   For example:

   ```
   mrconfig sp online /dev/sdc
   ```

4. Run the `gfsck` command on the affected volumes, or snapshots, with the appropriate options.
   If there are alarms, such as `DataUnavailableAlarm` or `DataUnderReplicatedAlarm`, do not run the `gfsck` command with the `-r` (`--repair`) option. Running the `gfsck` command with the `-r` (`--repair`) option, might result in data loss. If necessary, first run `gfsck` without the `-r` (`--repair`) option, and attempt to repair only after analyzing the command output.

Related reference

gfsck on page 2102
Describes how you can use the `gfsck` command, under the supervision of Map R Support or Engineering, to perform consistency checks and appropriate repairs on a volume, or a volume snapshot.

Setting MapR File System Permissions

The MapR filesystem permissions are similar to the POSIX permissions model. Each file and directory is associated with a user (the *owner*) and a group. You can set read, write, and execute permissions separately for:

- the owner of the file or directory.
- members of the group associated with the file or directory.
- all other users.

The permissions for a file or directory are called its *mode*. The mode of a file or directory can be expressed in two ways:

- Text - a string that indicates the presence of the read (`r`), write (`w`), and execute (`x`) permission or their absence (`-`) for the owner, group, and other users respectively. Example: `rwxr-xr-x`
- Octal - three octal digits (for the owner, group, and other users), that use individual bits to represent the three permissions. Example: `755`

Both `rwxr-xr-x` and `755` represent the same mode; the owner has all permissions, and the group and other users have read and execute permissions only.

When you access the MapR filesystem layer over NFS, the file-level permissions are controlled through the Linux interface by using the `chmod` (change mode) command or the `chown` (change owner) command, as well as the `hadoop fs -chmod` and `hadoop fs -chown` equivalents. For example:

```
chown jsmith /mapr/my.cluster.com/jsmith/fileA
hadoop -fs chown jsmith /mapr/my.cluster.com/jsmith/fileA
chmod 744 /mapr/my.cluster.com/jsmith/fileA
hadoop -fs chmod 744 /mapr/my.cluster.com/jsmith/fileA
```

These commands grant a user whose username is `jsmith` the read, write, and execute privileges on `fileA`.

Once you set file permissions, authorization checks are performed when a file is opened, *and* on every file access.

Note: To further protect your data, the MapR filesystem data cache is never included in a file server core dump.

Text Modes

String modes are constructed from the characters in the following table:

Text	Description
u	The file's owner.
g	The group associated with the file or directory.
o	Other users (users that are not the owner, and not in the group).
a	All (owner, group and others).
=	Assigns the permissions Example: "a=rw" sets read and write permissions and disables execution for all.

last-updated: Apr 11, 2022

Text	Description
-	Removes a specific permission. Example: "a-x" revokes execution permission from all users without changing read and write permissions.
+	Adds a specific permission. Example: "a+x" grants execution permission to all users without changing read and write permissions.
r	Read permission
w	Write permission
x	Execute permission

Octal Modes

To construct each octal digit, add the value of each permission that you want to grant:

- Read: 4
- Write: 2
- Execute: 1

For example, 7 which provides read, write, and execute permissions because 4+2+1=7.

Syntax

You can change the modes of directories and files in the MapR storage using either the `hadoop fs` command with the `-chmod` option, or using the `chmod` command via NFS. The syntax for both commands is similar:

- ```
  hadoop fs -chmod [-R] <MODE>[,<MODE>]... | <OCTALMODE> <URI> [<URI> ...]
  ```
- ```
  chmod [-R] <MODE>[,<MODE>]... | <OCTALMODE> <URI> [<URI> ...]
  ```

Parameters and Options

The following table provides the command parameters and options with their descriptions:

Parameter/Option	Description
-R	If specified, this option applies the new mode recursively throughout the directory structure.
MODE	A string that specifies a mode.
OCTALMODE	A three-digit octal number that specifies the new mode for the file or directory.
URI	A relative or absolute path to the file or directory for which to change the mode.

Examples

The following examples are all equivalent:

last-updated: Apr 11, 2022

- `chmod 755 script.sh`
- `chmod u=rwx,g=rx,o=rx script.sh`
- `chmod u=rwx,go=rx script.sh`

Managing File and Directory ACEs

Describes the implications of setting access control expressions on files and directories.

File *ACE* allows you to define access (whitelist and blacklist) to files and directories for a combination of users, groups, and roles. If *ACE*s are not set, POSIX mode bits for the file or directory are used to grant or deny access to the file or directory.

When you set *ACE*s, MapR sets or resets the corresponding POSIX mode bits to match the permissions granted through *ACE*s. For more information, see Setting/Modifying File and Directory ACEs.

- If both *ACE*s and POSIX mode bits are set, access is granted if access is allowed through *ACE*s or POSIX mode bits.
- If *ACE*s are not set, POSIX mode bits are used to grant access.
- If neither *ACE*s nor POSIX mode bits are set, access is denied.

The owner of the file or directory (and mapr and root users) can set, modify, and remove *ACE*s for that file or directory using `hadoop mfs` commands.

File ACEs

You can set and modify permissions to read, write, and execute files using the `hadoop mfs` command or the FileACE Java APIs on page 1457 and FileACE C APIs on page 1457. Specifically, the following access types are supported.

Access Type		Description
Command-Line	**Java API (Enum)**	
`-readfile`	`READFILE`	Read a file.
`-writefile`	`WRITEFILE`	Write to a file.
`-executefile`	`EXECUTEFILE`	Execute a file.

For more information, see `hadoop mfs`, FileACE Java APIs on page 1457, and FileACE C APIs on page 1457.

Directory ACEs

You can set the same *ACE*s on directories as for files. In addition, directory *ACE*s support permissions to list, add child, delete child, and lookup directories using `hadoop mfs` command. Specifically, the following access types are supported.

Access Type		Description
Command-Line	**Java API (Enum)**	
`-readfile`	`READFILE`	Read a file.
`-writefile`	`WRITEFILE`	Write to a file.

last-updated: Apr 11, 2022

Access Type		Description
Command-Line	**Java API (Enum)**	
`-executefile`	`EXECUTEFILE`	Execute a file.
`-readdir`	`READDIR`	List the contents of a directory. This access is required to write and/or execute files in the directory.
`-lookupdir`	`LOOKUPDIR`	Lookup a file in a directory. This access is required to find, read, write, and/or execute files in the directory.
`-addchild`	`ADDCHILD`	Add a file or subdirectory.
`-deletechild`	`DELETECHILD`	Delete a file or subdirectory.

Although you can set both file and directory *ACE*s on directories, only the directory *ACE*s are used for determining access to the directory. The file *ACE* on the directory is used as the default *ACE* setting for new files under that directory.

By default, when you set *ACE*s on a parent directory:

- Permissions for existing files and subdirectories under that parent remain unchanged.
- New files under that parent inherit the file *ACE*s and corresponding POSIX mode bits of the parent directory, if available. Otherwise, new files get the default *ACE*, the empty string (""), which indicates that no one has permissions to read, write, or execute the file; POSIX mode bits are set on the file in the traditional way.
- New subdirectories under the parent inherit both the directory and file *ACE*s and corresponding POSIX mode bits from the parent directory.

Note: When accessing files and directories, the *ACE*s on files have no effect on accessing the parent directory.

Workaround for Execute Operation when ACES are set on an executable file

When *ACE*s are set on any file, mode bits are cleared. However, for a binary to execute, the kernel checks whether the execute bit is set or not, and restricts execution if it is not set. To run an executable file with *ACE*s set on it, use one of the following workarounds:

1. Set owner mode exec bit on binaries/shell scripts.
2. Set group mode exec bit on binaries/shell scripts.
3. Change owning group for the files to the group used in MapRAces, and set the executable group mode bit.

Setting File and Directory ACEs

Describes how to set access control expressions (ACEs) for files and directories.

For files and directories, run the `hadoop mfs` command to set *ACE*s. When *ACE*s are set, by default, the corresponding POSIX mode bits are also set. POSIX mode bits for owner and owning group are deduced by evaluating the corresponding *ACE*s. POSIX mode bits for others is set only if `"p"` is given as the value for an *ACE*.

The following table lists the POSIX mode bits that correspond to the access types.

last-updated: Apr 11, 2022

	ACE	POSIX Mode Bits
File	`readfile`	r
	`writefile`	w
	`executefile`	x
Directory	`readdir`	r
	`addchild`	w
	`deletechild`	w
	`lookupdir`	x

The POSIX mode bit granting write (`w`) access to directory is set only if user, role, or group is granted permission for both (`addchild` and `deletechild`) access types.

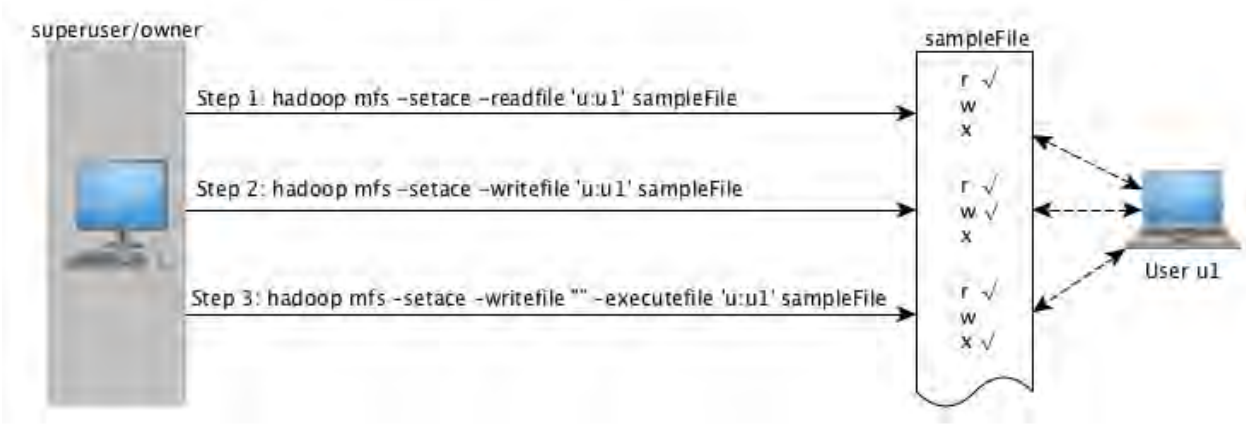
The `hadoop` command, by default, sets the POSIX mode bits corresponding to the given *ACE*s, and:

- Overwrites existing *ACE* values with new values, if specified, for access types that were previously set.
- Sets *ACE* values for access types that have not yet been set, if specified.
- Does not modify access types that are not specified with the command, whether or not they were previously set.

Warning: Changing the POSIX mode bits using `chmod` does not change the corresponding *ACE* setting and may result in different, conflicting permissions to files and directories.

File ACE Example
Illustrates setting access control expressions for files.

Suppose the following sequence of file *ACE* settings (and corresponding POSIX mode bits) are set for user u1.



As shown in the preceding illustration, in:

Step 1:

User u1 is granted permissions to read a file, `sampleFile`.

After the command runs, user u1 has permissions to (only) read the file. The POSIX mode bit for reading the file is set to u1 for owner/users.

There is no change in *ACE*s or POSIX mode bits for all other (write and execute) access types.

last-updated: Apr 11, 2022

Step 2:

User u1 is granted permissions to write to the same file.

After the command runs, user u1 has permissions to write to the file. The POSIX mode bit for writing to the file is set to u1 for owner/users.

There is no change in *ACE*s or POSIX mode bits for all other (read and execute) access types.

Step 3:

User u1's permissions are modified to remove write permission (using the empty string) and to grant access to execute file.

After the command runs, user u1 has permissions to execute the file, but user u1 can no longer write to the file. The POSIX mode bit for:

- Writing to the file is set to 0 for owner/users, groups, and others.
- Executing the file is set to u1 for owner/users.

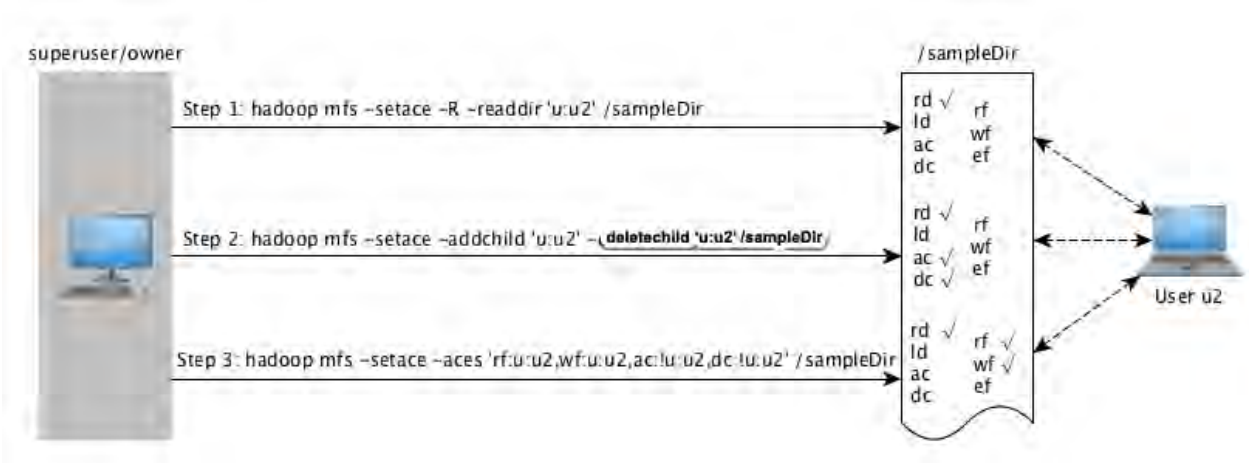
Note: When the empty string (`""`) is used to deny a specific type of file access, that type of file access is denied to all users, groups, and roles. To deny access to specific users only, use the negation operator (`!`).

There is no change in *ACE*s or POSIX mode bits for all other (read) access types.

Directory ACEs Example

Explains how to set access control expressions for directories.

For example, suppose the following diagram depicts the (command-line) sequence of directory *ACE* settings for user u2:



As shown in the preceding illustration, in:

Step 1:

User u2 is granted access to read directory, and sampleDir, while all other directory/file *ACE*s are not specified.

After the command runs, user u2 has permissions to list the contents of the directory. The POSIX mode bits for listing the contents of the directory (`r`) is set to u2 for owner/users.

last-updated: Apr 11, 2022

	There is no change in *ACE*s or POSIX mode bits for all other (file- and directory-level) access types.
Step 2:	User u2 is granted permission only to add and delete child directories, while all other directory/file *ACE*s are not specified. After the command runs, user u2 has permissions to create and delete child directories. The POSIX mode bit for writing (`w`) to the directory for owner/user is set to u2 because user u2 is granted access for both (`addchild` and `deletechild`) access types. If user u2 creates child directories, the child directories, by default, inherit the *ACE* settings of the parent directory. There is no change in *ACE*s or POSIX mode bits for all other (file- and directory-level) access types.
Step 3:	User u2's permissions are modified to grant access to read and write to files in the directory, user u2's permissions for adding and deleting child directories are removed (using the negation operator), and all other directory/file *ACE*s are not specified. After the command runs, user u2 can read and write to files in the directory, but user u2 can no longer add and delete child directories. The POSIX mode bits for directory write access (`w`) is set to 0 for owner/user. Although, at the directory level, user u2 has permissions to read and write to files in the directory, for existing files, the file level *ACE*s or the POSIX mode bits for the file are used to determine access. However, by default, user u2 gets read and write permissions to all new files created under the directory. If user u2 creates new files under the directory, the files inherit the file *ACE*s from the parent directory by default and the POSIX mode bits for read (`r`) and write (`w`) access are set to u2 for owner/user. There is no change in *ACE*s or POSIX mode bits for all other (`lookupdir` and `executefile`) access types.

Deleting File and Directory ACEs

Describes how to delete file and directory ACEs using the CLI.

You can remove all *ACE* associated with a file or directory using the `hadoop mfs -delace` command. When you delete all the *ACE*s, the system sets the *ACE* for the file or directory to the default value, which is the empty string (""). The POSIX mode bits are not reset; if necessary, run the `chmod` command to reset POSIX mode bits.

You cannot remove specific access types that have been set; instead, use the empty string to deny specific types of access. When the empty string ("") is used to deny a specific type of access, that type of access is denied to all users, groups, and roles. To deny access to specific users only, use the negation operator (`!`). If you use the empty string ("") or the negation operation (`!`) to deny a specific type of access, the corresponding POSIX mode bit are also reset to match the *ACE* setting.

Managing Chunk Size

Describes the considerations for managing the chunk size for map tasks.

Files in the MapR filesystem are split into *chunks* (similar to Hadoop *blocks*) that are normally 256 MB by default. Any multiple of 65,536 bytes is a valid chunk size, but tuning the size correctly is important:

last-updated: Apr 11, 2022

- Smaller chunk sizes result in larger numbers of map tasks, which can result in lower performance due to task scheduling overhead.
- Larger chunk sizes require more memory to sort the map task output, which can crash the JVM or add significant garbage collection overhead. MapR can deliver a single stream at upwards of 300 MB per second, making it possible to use larger chunks than in the stock Hadoop implementation. Generally, it is wise to set the chunk size between 64 MB and 256 MB.

Chunk size is set at the directory level. Files inherit the chunk size settings of the directory that contains them, as do subdirectories on which chunk size has not been explicitly set. Any files written by a Hadoop application, whether using the file APIs or over NFS, use chunk size specified by the settings for the directory where the file is written. If you change a directory's chunk size settings after writing a file, the file will keep the old chunk size settings. Further writes to the file will use the file's current chunk size.

Note: If chunk size is zero (`0`), when an application makes a request for block size, MapR will return 1073741824 (1GB); however, `hadoop mfs` on page 5373 commands will continue to display `0` for chunk size.

Configuring Chunk Size

Chunk size also affects parallel processing and random disk I/O during MapReduce applications. A higher chunk size means less parallel processing because there are fewer map inputs, and therefore fewer mappers. A lower chunk size improves parallelism, but results in higher random disk I/O during shuffle because there are more map outputs. Set the `io.sort.mb` parameter to a value between 120% and 150% of the chunk size.

Here are the general guidelines for chunk size:

- For most purposes, set the chunk size to the default 256 MB and set the value of the `io.sort.mb` parameter to the default 380 MB.
- On very small clusters or nodes with not much RAM, set the chunk size to 128 MB and set the value of the `io.sort.mb` parameter to 190 MB.
- If application-level compression is in use, the `io.sort.mb` parameter should be at least 380 MB.

Note: If you have Drill running in the cluster, change the `store.parquet.block-size` parameter in Drill so that the Parquet block size is the same as the chunk size in the MapR filesystem. See Configuring the Parquet Block Size for more information.

Setting Chunk Size

You can set the chunk size for a given directory in two ways:

- Change the `ChunkSize` attribute in the .dfs_attributes file at the top level of the directory
- Use the command `hadoop mfs -setchunksize <size> <directory>`

For example, if the volume `test` is NFS-mounted at `/mapr/my.cluster.com/projects/test` you can set the chunk size to 268,435,456 bytes by editing the file `/mapr/my.cluster.com/projects/test/.dfs_attributes` and setting `ChunkSize=268435456`. To accomplish the same thing from the `hadoop` shell, use the following command:

```
hadoop mfs -setchunksize 268435456 /mapr/my.cluster.com/projects/test
```

Managing Compression

Lists the advantages of using compression.

last-updated: Apr 11, 2022

MapR provides compression for files stored in the cluster. Compression is applied automatically to uncompressed files unless you turn compression off. The advantages of compression are:

- Compressed data uses less bandwidth on the network than uncompressed data.
- Compressed data uses less disk space.

Choosing a Compression Setting

Lists the compression algorithms supported by MapR.

MapR supports three different compression algorithms:

- lz4 (default)
- lzf
- zlib

Compression algorithms can be evaluated for compression ratio (higher compression means less disk space used), compression speed and decompression speed. The following table gives a comparison for the three supported algorithms. The data is based on a single-thread, Core 2 Duo at 3 GHz.

Compression Type	Compression Ratio	Compression Speed	Decompression Speed
lz4	2.084	330 MB/s	915 MB/s
lzf	2.076	197 MB/s	465 MB/s
zlib	3.095	14 MB/s	210 MB/s

Note that compression speed depends on various factors including:

- block size (the smaller the block size, the faster the compression speed)
- single-thread vs. multi-thread system
- single-core vs. multi-core system
- the type of codec used

Related Link

- LZO vs Snappy vs LZF vs ZLIB

Setting Compression on Files

Compression is set at the directory level. Any files written by a Hadoop application, whether via the file APIs or over NFS, are compressed according to the settings for the directory where the file is written. Sub-directories on which compression has not been explicitly set inherit the compression settings of the directory that contains them.

If you change a directory's compression settings after writing a file, the file will keep the old compression settings—that is, if you write a file in an uncompressed directory and then turn compression on, the file does not automatically end up compressed, and vice versa. Further writes to the file will use the file's existing compression setting.

Warning: Only the owner of a directory can change its compression settings or other attributes. Write permission is not sufficient.

File Extensions of Compressed Files

Lists extensions of compressed files.

last-updated: Apr 11, 2022

By default, MapR does not compress files whose filename extensions indicate they are already compressed. The default list of filename extensions is as follows:

- bz2
- gz
- lzo
- snappy
- tgz
- tbz2
- zip
- z
- Z
- mp3
- jpg
- jpeg
- mpg
- mpeg
- avi
- gif
- png

The list of filename extensions not to compress is stored as comma-separated values in the `mapr.fs.nocompression` configuration parameter and can be modified with the `config save` command. For example, you can add `parquet` to the default list:

```
maprcli config save -values
'{"mapr.fs.nocompression":"bz2,gz,lzo,snappy,tgz,tbz2,zip,z,Z,mp3, \
jpg,jpeg,mpg,mpeg,avi,gif,png,parquet"}'
```

The list can be viewed with the `config load` command. Example:

```
maprcli config load -keys mapr.fs.nocompression
```

Turning Compression On or Off on Directories Using the CLI
Explains how to turn off or turn on compression and optionally specify an algorithm, using the command line.

You can turn compression on or off for a given directory in two ways:

- Set the value of the `Compression` attribute in the .dfs_attributes file at the top level of the directory.
  - Set `Compression=lzf|lz4|zlib` to turn compression *on* for a directory.
  - Set `Compression=false` to turn compression *off* for a directory.

- Use the command hadoop mfs -setcompression on|off/lzf/lz4/zlib <dir|table>.

If you choose `-setcompression on` without specifying an algorithm, lz4 is used by default. This algorithm has improved compression speeds for MapR's block size of 64 KB.

The symbols for the various compression settings are explained here:

Symbol	Compression Setting
Z	lz4
z	zlib
L	lzf
U	Uncompressed, or previously compressed by another algorithm

Example

Suppose the volume `test` is NFS-mounted at `/mapr/my.cluster.com/projects/test`. You can turn off compression by editing the file `/mapr/my.cluster.com/projects/test/.dfs_attributes` and setting `Compression=false`. To accomplish the same thing from the `hadoop` shell, use the following command:

```
hadoop mfs -setcompression off /projects/test
```

You can view the compression settings for directories using the hadoop mfs -ls command. For example,

```
vrwxr-xr-x  Z U U   3 mapr mapr         11 2017-12-01 14:00  268435456 /.rw
           p mapr.cluster.root writeable 2049.36.131352 -> 2049.16.2
doc24.lab:5660
vrwxr-xr-x  Z U U   3 mapr mapr          0 2017-12-01 13:58  268435456 /abcd
           p abcd default 2049.1143.264886 -> 2181.16.2  doc24.lab:5660
vrwxrwxrwx  Z U U   3 root root          0 1969-12-31 16:00  268435456 /
abcdMirror
           p abcdMirror default 2049.1144.264888 -> 2182.16.2
doc24.lab:5660
vrwxr-xr-x  Z U U   3 mapr mapr          1 2017-11-28 08:13  268435456 /apps
           p mapr.apps default 2049.33.131346 -> 2051.16.2  doc24.lab:5660
vrwxr-xr-x  U U U   3 mapr mapr          0 2017-11-28 08:07   67108864 /
hbase
           p mapr.hbase default 2049.39.131358 -> 2064.16.2  doc24.lab:5660
drwxr-xr-x  Z U U   - mapr mapr          4 2017-11-28 08:13  268435456 /
installer
           p 2049.40.131360  doc24.lab:5660
drwxr-xr-x  Z U U   - mapr mapr          1 2017-11-28 08:15  268435456 /
oozie
           p 2049.203.131686  doc24.lab:5660
vrwxr-xr-x  Z U U   3 mapr mapr          0 2017-11-28 08:06  268435456 /opt
           p mapr.opt default 2049.38.131356 -> 2061.16.2  doc24.lab:5660
vrwxrwxrwx  Z U U   3 mapr mapr          0 2017-11-28 08:27  268435456 /tmp
           p mapr.tmp default 2049.32.131344 -> 2050.16.2  doc24.lab:5660
vrwxr-xr-x  Z U U   3 mapr mapr          2 2017-11-28 08:12  268435456 /user
           p users default 2049.37.131354 -> 2060.16.2  doc24.lab:5660
drwxr-xr-x  Z U U   - mapr mapr          1 2017-11-28 08:05  268435456 /var
           p 2049.34.131348  doc24.lab:5660
```


last-updated: Apr 11, 2022

Suppose three directories `abc`, `klm`, and `xyz`. You can turn on compression and set different compression algorithm for the directories by running the following commands:

```
hadoop mfs -setcompression on /ksTestVol1/abc
hadoop mfs -setcompression lzf /ksTestVol1/klm
hadoop mfs -setcompression zlib /ksTestVol1/xyz
```

You can then view the compression settings for the directories using the hadoop mfs -ls command. For example:

```
hadoop mfs -ls /ksTestVol1/
  Found 3 items
  drwxr-xr-x  Z U U   - root root    0 2017-12-11 08:41  268435456 /
ksTestVol1/abc
           p 2432.32.131194  doc24.lab:5660
  drwxr-xr-x  L U U   - root root    0 2017-12-11 08:42  268435456 /
ksTestVol1/klm
           p 2432.34.131198  doc24.lab:5660
  drwxr-xr-x  z U U   - root root    0 2017-12-11 08:42  268435456 /
ksTestVol1/xyz
           p 2432.33.131196  doc24.lab:5660
```

Setting Compression During Shuffle

By default, MapReduce uses compression during the Shuffle phase. You can use the `-Dmapreduce.maprfs.use.compression` switch to turn compression *off* during the Shuffle phase of a MapReduce job. For example:

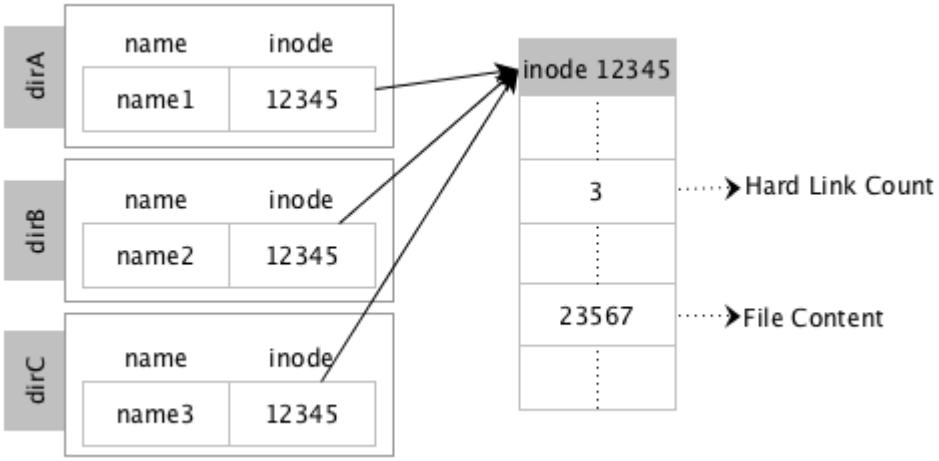
```
hadoop jar xxx.jar -Dmapreduce.maprfs.use.compression=false
```

Managing Hard Links

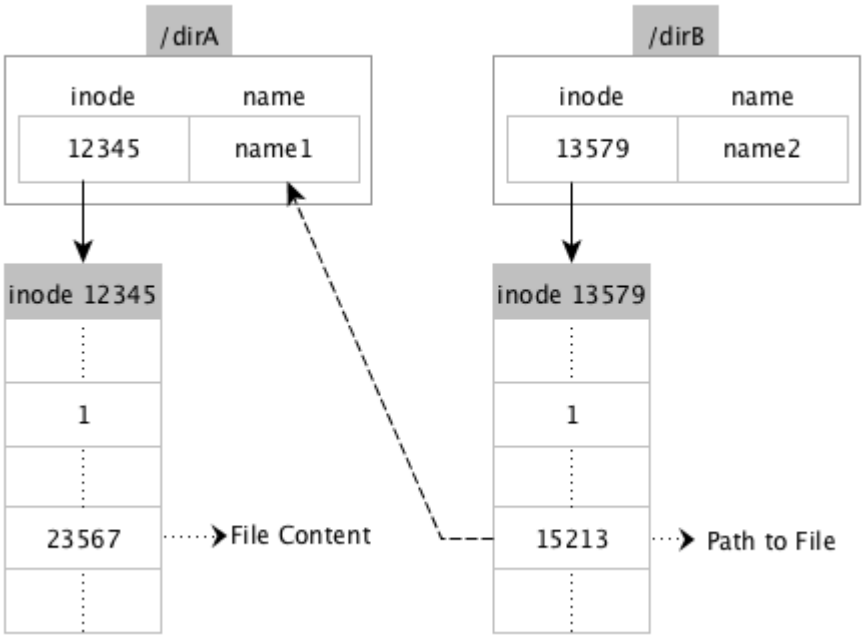
Explains what hard links are, and their limitations.

A hard link is a directory entry that associates a name with a file (or physical data) on the filesystem. Hard links allow multiple names to be associated with the same data (and associated inode) from within or outside of a directory. Every time a hard link is created, a directory entry is created and the inode (associated with the directory entry) remains the same across all hard links associated with that data. That is, all names associated with the data point to the same inode.

The following diagram illustrates the hard link semantics. Here, directory entries in dirA, dirB, and dirC for file names name1, name2, and name3 respectively all point to the inode 12345, which contains metadata including the text in the file and a count of the number of hard links to the file (or physical data).



In contrast, when a symbolic link is created, a new inode is created and the text part of the inode (associated with the symlink file) contains the path to the actual file. The following diagram illustrates the symbolic link semantics. Here, the directory entry in dirA for file name, name1, is associated with inode 12345, which contains the text in the file. The directory entry in dirB for symbolic link file, name2, is associated with inode 13579, which contains the path to file in dirA (*/dirA/name1*). Deleting the file, name1, in dirA will result in the symlink file in dirB, name2, pointing to stale content, which in turn will return errors when accessed.



Hard links can be created on regular files, symlink files, device files, and tables.

Limitations

- Hard links cannot be created on directories.
- Hard links cannot be created across volumes or clusters. Instead, use symbolic link to link to a file on a different volume.

last-updated: Apr 11, 2022

- Hard links within a volume are carried over to mirror volumes and volume snapshots. During cross-mirroring, there will be an error if support for hardlinks is not enabled on both the clusters.
- The `hadoop distcp` command cannot be used for creating and preserving copies of hard links.
- The maximum number of hard links is constrained by the integer width (32 bits), which means the maximum number possible for a file is 2^{32}.

Usage

Any user with access to the directory can create a hard link to any file under that directory. To create hard links, the user must have write permissions on the directory and execute permissions (to do the lookup for the path) on the target file. To read or modify the file, the user must have read or write permissions respectively on the file.

Errors

For information on the type of failures and errors, refer to the man page. In addition, please note that the EXDEV error is returned if command is run to create cross-volume or cross-cluster hard links.

Enabling Hard Links

By default, this feature is enabled on all new installation. If you upgrade, you must enable this feature. To enable this feature, run the following command:

```
maprcli cluster feature enable -name mfs.feature.hardlinks.support
```

Setting a Hard Link

Explains how to create a hardlink to a file.

To set a hard link using:

- POSIX loopbacknfs client or NFS client, run the following command:

  ```
  ln <sourcefile> <newfile>
  ```

  where <sourcefile> is the name of the file to link to and <newfile> is the name of the hard link, which must *not* already be present.

- Hadoop, run the following command:

  ```
  hadoop mfs -lnh <sourcefile> <newfile>
  ```

  where <sourcefile> is the name of the file (including full path) to link to and <newfile> is the name of the hard link (including the full path). When running this command, specify the full path to both files.

Retrieving the Number of Hard Links

Explains how to retrieve the number of hard links to a file.

To retrieve the number of hard links associated with a file, run the following command:

```
ls -l
```

The command, `ls -l`, will print the number of hardlinks in the second column. For example, your output will look similar to the following:

```
ls -l sample-link
  rw-r--r- 2 root root 0 Apr 21 11:09 sample-link
```

To retrieve the list of the hard links associated with a file, run the following command:

```
find <dirpath> -samefile <sourcefile>
```

where `<dirpath>` is the path to the source file and `<sourcefile>` is the source file for the hard link. For example, your output will look similar to the following:

```
find . -samefile file8
  ./file8-link2
  ./file8-link100
  ./file8-link101
  ./file8
  ./file8-link
```

For POSIX (loopback NFS) clients, the number of hardlinks to a file can be retrieved using the `stat64` system call. For example, your output will look similar to the following:

```
stat samplefile
  File: 'samplefile'
  Size: 0               Blocks: 0          IO Block: 131072 regular empty
file
  Device: 14h/20d Inode: 853785146   Links: 4
  Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (    0/    root)
  Access: 2016-05-12 13:06:21.000000000 -0700
  Modify: 2016-05-12 13:06:30.002560000 -0700
  Change: 2016-05-12 13:06:30.002560000 -0700
```

If you are not using NFS or the POSIX clients, to retrieve the number of hard links, you can run the `hadoop` command to retrieve the fid and then run the `maprcli` command to retrieve the number of hard links as follows. The `nlink` variable will print the number of links.

```
hadoop mfs -ls /p1
  Found 1 items
  -rw-r--r--  Z U U   3 root root       3054 2016-05-05 13:49  268435456 /p1
  p 2049.40.262550  node-31.lab:5660

maprcli fid stat -fid 2049.40.262550
  xattrInum  uid  atime       nblocks  deleteFlags     mtime
parent     nlink  type       version  size  mode  networkencryption
subtype   gid  compression
  0          0    1462481255  1        DeleteTypeNone  1462481376
2049.16.2  2      FTRegular  2097165  3054  644   false
FSTInval  0    lz4
```

Removing Hard Links

To remove a hard link using:

- Linux, run the following command:

  ```
  rm -f <hard link>
  ```

- Hadoop, run the following command:

  ```
  hadoop fs -rm <path to hard link>
  ```

Example

For example, suppose there are **4** hard links to file cite75_99.txt.

```
$ ls -l
total 1289433
-rwxr-xr-x 5 root root 264075431 Jul 28 13:46 cite75_99.txt
-rwxr-xr-x 5 root root 264075431 Jul 28 13:46 cite-link1
-rwxr-xr-x 5 root root 264075431 Jul 28 13:46 cite-link2
-rwxr-xr-x 5 root root 264075431 Jul 28 13:46 cite-link3
-rwxr-xr-x 5 root root 264075431 Jul 28 13:46 cite-link4

$ maprcli fid stat -fid 2142.34.131274
parent     deleteFlags     atime       gid  nlink  type       mtime
version  mode  uid  xattrInum  size       subtype   networkencryption
nblocks  compression
2142.16.2  DeleteTypeNone  1469738740  0    5      FTRegular  1469738771
1048600  755   0    0          264075431  FSTInval  false
8        lz4
```

To remove a hard link using:

- Linux, run the following command:

  ```
  rm -f cite-link1
  ```

  To verify that the command ran successfully, run the following command:

  ```
  $ ls -l
  total 1031546
  -rwxr-xr-x 4 root root 264075431 Jul 28 13:46 cite75_99.txt
  -rwxr-xr-x 4 root root 264075431 Jul 28 13:46 cite-link2
  -rwxr-xr-x 4 root root 264075431 Jul 28 13:46 cite-link3
  -rwxr-xr-x 4 root root 264075431 Jul 28 13:46 cite-link4
  ```

- Hadoop, run the following command:

  ```
  $ hadoop fs -rm /test-hl/cite-link2
  16/07/28 13:52:00 INFO Configuration.deprecation: io.bytes.per.checksum
  is deprecated. Instead, use dfs.bytes-per-checksum
  16/07/28 13:52:00 INFO fs.TrashPolicyDefault: Namenode trash
  configuration: Deletion interval = 0 minutes, Emptier interval = 0
  minutes.
  Deleted /test-hl/cite-link2
  ```

  To verify that the command ran successfully, run the following command:

  ```
  $ maprcli fid stat -fid 2142.34.131274
  parent  deleteFlags     atime       gid  nlink  type       mtime
  version  mode  uid  xattrInum  size       subtype   networkencryption
  nblocks  compression
  0.0.0   DeleteTypeNone  1469738740  0    3      FTRegular  1469738771
  1048603  755   0    0          264075431  FSTInval  false
  8        lz4
  ```

Managing Extended Attributes

Describes what extended attributes are, and the POSIX permissions that you need to manage them.

Extended attributes (referred to as `xattrs`) allow user applications to associate additional metadata with a regular file or directory. Unlike system-level inode metadata, such as file permissions or modification time, extended attributes are not interpreted by the system but are instead used by applications to store additional information about an inode. Multiple extended attributes can be associated with a single inode. The maximum size allowed for an extended attribute is 64 KB.

An extended attribute is a name-value pair, with a string name and binary value. The extended attribute names are prefixed with a namespace. For example, an `xattr` named `myXattr` in the user namespace would be specified as `user.myXattr`.

Limitations

- For the five valid namespaces supported by HDFS, MapR supports the following:

Namespace	MapR Functionality
user	Commonly used by client applications. Access to these extended attributes in the user namespace is controlled by corresponding file/directory permissions. For more information, see Permissions for Extended Attributes.
trusted	Available to superusers only. Access is denied for all other users. The extended attribute is not available through userspace methods.

Note: MapR does not support the raw namespace.

- Extended attributes cannot be associated with symbolic links. If extended attributes are used on symbolic links, they are instead applied to the symbolic link target file.
- The preserve options of commands like `cp -px` and `distcp -px` will work on extended attributes only in the following cases:
  - With Hadoop commands such as `hadoop fs`.
  - On FUSE mounted file paths.

Note: Extended attributes are not supported on NFS file paths.

Permissions for Extended Attributes

The following table lists the permissions (POSIX mode bits or *ACE*s) you will need to set, retrieve, or modify extended attributes.

To...	For directories, you need...	For files, you need...
Set extended attributes	Mode bits: write (OR) *ACE*: addchild	Mode bits: write (OR) *ACE*: writefile
Remove extended attributes	Mode bits: write (OR) *ACE*: deletechild	Mode bits: write (OR) *ACE*: writefile
Retrieve extended attributes	Mode bits: read (OR) *ACE*: readdir	Mode bits: read (OR) *ACE*: readfile

last-updated: Apr 11, 2022

Enabling Extended Attributes

Extended attributes are enabled by default on all new installation. If you are upgrading, this feature must be explicitly enabled. To enable, run the following command:

```
maprcli cluster feature enable -name mfs.feature.fileace.support
```

Setting, Retrieving, and Removing Extended Attributes

You can set and retrieve extended attributes on files, directories, and FUSE mounted file path using Hadoop commands, Linux commands, and Java APIs.

Hadoop Commands

Lists the Hadoop commands to set, retrieve, and remove extended attributes on files, directories and FUSE mounted paths.

You can set, retrieve, and remove extended attributes on files, directories, and FUSE mounted file path using the `hadoop fs` command. When setting an extended attribute:

- The name must be prefixed with a namespace.
- The extended attribute value must be encoded as one of the following:

text	The given string must be enclosed in double quotes to be treated as text.
hex	The given string must begin with 0x or 0X to be treated as hexadecimal number.
base64	The given string must begin with 0s or 0S to be treated as base64 encoding.

Note: You must have the right permissions to set, retrieve, and/or remove extended attributes.

Set Extended Attributes

To set an extended attribute name and value for a file or directory, run the following command:

```
hadoop fs -setfattr -n name [-v value] <path>
```

For example:

```
hadoop fs -setfattr -n system.name -v system-hadoopfs /volforsnap/
smallfile.txt
hadoop fs -setfattr -n user.name -v user-hadoopfs /volforsnap/smallfile.txt
hadoop fs -setfattr -n security.name -v security-hadoopfs /volforsnap/
smallfile.txt
hadoop fs -setfattr -n trusted.name -v trusted-hadoopfs /volforsnap/
smallfile.txt
```

Retrieve Extended Attributes

To retrieve the extended attributes associated with a file or directory, run the following command:

```
hadoop fs -getfattr [-R] -n name | -d [-e en] <path>
```

For example:

```
hadoop fs -getfattr  -d /volforsnap/smallfile.txt
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding
in [jar:file:/opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/common/lib/
slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/mapr/lib/
```

```
slf4j-log4j12-1.7.12.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an
explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
# file: /volforsnap/smallfile.txt
system.name="system-hadoopfs"
trusted.name="trusted-hadoopfs"
security.name="security-hadoopfs"
user.name="user-hadoopfs"
```

Remove Extended Attributes

To remove an extended attribute by name, run the following command:

```
hadoop fs -setfattr -x name <path>
```

For example:

```
hadoop fs -setfattr -x user.key1 /xattrs/m7user1/dir1
```

Linux Commands

You can set, retrieve, restore, and remove extended attributes on files, directories, and FUSE mounted file paths using Linux commands. For more information, refer to the respective Linux man page.

To use extended attributes on files on a MapR cluster with a FUSE client mounted path, see Configuring the MapR FUSE-Based POSIX Client on page 1240 to enable extended attributes through FUSE client.

Set Extended Attributes

To set an extended attribute name and value on a file/directory and/or a FUSE mounted file path, run the following command:

```
setfattr [-h] -n name [-v value] pathname...
```

For example:

```
setfattr -n system.name -v system /mapr_fuse/testcluster/volforsnap/
smallfile.txt
setfattr -n security.name -v test /mapr_fuse/testcluster/volforsnap/
smallfile.txt
setfattr -n trusted.name -v trusted /mapr_fuse/testcluster/volforsnap/
smallfile.txt
setfattr -n user.name -v user /mapr_fuse/testcluster/volforsnap/
smallfile.txt
```

For more information, refer to the Linux man page.

Retrieve Extended Attributes

To retrieve extended attributes, run one of the following commands:

```
getfattr [-hRLP] -n name [-e en] pathname...
getfattr [-hRLP] -d [-e en] [-m pattern] pathname...
```

For example:

```
getfattr  -d -m - /mapr_fuse/testcluster/volforsnap/smallfile.txt
getfattr: Removing leading '/' from absolute path names
  file: mapr_fuse/testcluster/volforsnap/smallfile.txt
```


last-updated: Apr 11, 2022

```
  security.name="test"
  system.name="system"
  trusted.name="trusted"
  user.name="user"
```

For more information, refer to the Linux man page.

Remove Extended Attributes

To remove an extended attribute by name, run the following command:

```
setfattr [-h] -x name pathname...
```

For example:

```
setfattr -x user.test test2
```

For more information, refer to the Linux man page.

Restore Extended Attributes

To restore extended attributes from a file, which must be in the format generated by the `getfattr` command with the `--dump` option, run the following command:

```
setfattr [-h] --restore=file...
```

For example:

```
setfattr --restore=testout
getfattr -d test2
 file: test2
 user.test="test"
```

For more information, refer to the Linux man page.

Java APIs

Java APIs to manage extended attributes

You can set, retrieve, and remove extended attributes on files, directories, and FUSE mounted file path using Extended Attribute Java APIs.

Set Extended Attributes

To set extended attributes, use the following APIs:

`public void setXAttr(Path path, String name, byte[] value) throws IOException`	Set an extended attribute on a file or directory. The name must be prefixed with the namespace followed by `"."`. For example, `"user.attr"`. By default, if a given extended attribute exists, then it will be replaced with the specified attribute.
`public void setXAttr(Path path, String name, byte[] value, Enum<SetXAttrSetFlag> flag) throws IOException`	Set an extended attribute on a file or directory. The name must be prefixed with the namespace followed by `"."`. For example, `"user.attr"`. The `XAttrSetFlag` value can be: • `CREATE` to create a new extended attribute. An error is returned if an extended attribute with the given name already exists.

last-updated: Apr 11, 2022

- REPLACE to replace an existing extended attribute. An error is returned if the specified extended attribute does not already exist.

Retrieve Extended Attributes

To retrieve extended attributes, use the following APIs:

`public byte[] getXAttr(Path path, String name) throws IOException`	Get an extended attribute name and value for a file or directory. The name must be prefixed with the namespace followed by `"."`. For example, `"user.attr"`.
`public Map<String,byte[]> getXAttrs(Path path) throws IOException`	Get all the extended attribute name/value pairs for a file or directory. Only those extended attributes that the logged-in user has permissions to view, are returned.
`public Map<String,byte[]> getXAttrs(Path path, List<String> names) throws IOException`	Get the extended attributes specified by the given list of names. Only those extended attributes that the logged-in user has permissions to view, are returned.
`public List<String> listXAttrs(Path path) throws IOException`	Get all the extended attribute names for a file or directory. Only those extended attribute names that the logged-in user has permissions to view, are returned.

Remove Extended Attributes

To remove an extended attribute associated with a file or directory, use the following API:

`public void removeXAttr(Path path, String name) throws IOException`	Remove an extended attribute of a file or directory. The name must be prefixed with the namespace followed by `"."`. For example, `"user.attr"`.

Managing Core Files

Describes how to set the location for core files.

The Linux `core_pattern` file (in `/proc/sys/kernel/core_pattern`) can be used to specify the location for storing core files. If any process launched by MapR crashes, the core files are created in the directory specified by the `core_pattern` file. A valid location in the `core_pattern` file is a full path to the directory you want to use. For example:

```
/tmp/dir1/cores/%e.core.%p.%h
```

> **Tip:** For details about the standard Linux `%` specifiers that you can use to name core files, see the core man page.

If the `core_pattern` file is empty, if the file does not contain a full path, or if it uses the `"|"` redirection feature, by default, MapR sets the location for core files (in the event of a core dump on a node) to `/opt/cores` directory when:

- The `configure.sh` utility is run.
- The Warden init script is run.
- The MapR File System init script is run by Warden.

The default directory (`/opt/cores`) is also used if the `core_pattern` file contains the default MapR value for core files.

For MapR software, the directory being used to store core files should not be used for other purposes and should be empty. The cores directory cannot be the home directory, as it can cause problems during SSH

access. The hoststats service monitors the directory specified in the `core_pattern` file and raises the node-level alarm if the directory contains any entry other than "`.`" and "`..`". When Warden is started, sticky bit is set on the cores directory.

Migrating Files From MapR Edge Cluster to AWS S3

The File Migration service for MapR Edge on page 6579 clusters monitors a set of configured directory trees (referred to as *policies* in the service UI) on the MapR cluster for new and changed files. When the service detects new or changed files, it automatically uploads them to AWS S3 (Amazon's storage environment) using the AWS SDK TransferManager object. The File Migration service is a long-running process that is controlled by Warden.

Note: For information on installing File Migration Service on your Edge cluster, see Installing the File Migration Service on the Edge Cluster on page 222.

How the File Upload Process Works

The File Migration service maintains in memory information regarding directories, subdirectories, and files within each directory tree to monitor. When started, the service first looks for changed and new files in the configured directories. To detect changes to existing files, the service looks at the extended attribute `user.s3uploadtime`, which is set on each file and contains the modification time (in milliseconds) when the file was last uploaded to the AWS S3 bucket. It then places the new and changed files in a queue for upload, sorted by the oldest modification timestamp. The service then uploads multiple files in parallel and provides logs as files upload, fail to upload, or complete uploading.

Note: For each directory tree, you can set up any file or directory that matches a specified regex pattern to be ignored. For example, all files ending in `.tmp` or all files in `/tmp` directory can be set up to be ignored. Additionally, only files are considered for upload; empty directories, symbolic links, and tables or streams are ignored.

Scanning for New and Changed Files

The service scans for new and changed files in the monitored directories using a full scan or a lite scan.

Full Scan	The service performs a full scan when it is first started and periodically or randomly for a subset of the directories in the directory tree after. This allows you to spread the cost of the full scan over time rather than performing a full scan of all directories at once. During a full scan, the service examines each file in a directory to determine if it has changed since the last time the directory was examined. If the file contains changes, then the file is examined more carefully. If the file was uploaded, the service retrieves the modification time of the file, which was set on the file at the time the file was uploaded. If the file was not uploaded or if the file was uploaded but the modification time changed since the last upload, the service places the file in the queue for upload. A full scan also discovers new directories and adds them to the list of directories to be monitored.
Lite Scan	The service performs an incremental scan to detect the last modified timestamp on each directory recursively using the in-memory directory tree from the last scan. If there is no change in the timestamp since the last scan, the service does not examine the directory contents and proceeds to the next directory; this makes the incremental scan very fast. If there is

last-updated: Apr 11, 2022

a change in the timestamp since the last scan, the service performs a full scan on the directory.

Note: Before starting an incremental scan, the service checks to see if a directory tree is due for a full scan. Only a full scan allows the service to detect if a file was modified after upload. With a lite scan, the service checks only the modification time of the directory, which is not changed by the modification (such as due to late writes) of a file.

Queueing Files for Upload

Files are placed in the queue with a configurable delay from the last known modification time of the file to avoid uploading a file that is still changing.

For example, with the default of 15 seconds, a file that was modified 5 seconds ago would remain on the queue for another 10 seconds, whereas a file modified 5 minutes ago would be immediately considered for upload. This allows files to be uploaded roughly in order by last modification time.

The service places (by default, 10) files in the queue for parallel upload. The scanning service is paused if the pending queue exceeds 10 times the number of files allowed for parallel update to avoid too many pending files.

Uploading Files to S3

Uploads are in one of three states: waiting to be uploaded, uploading, finished with uploading. By default, the service performs a check on the status of the files being uploaded every 3 seconds and detects uploads that are approaching completion. The service then sets the extended attribute (`user.s3uploadtime`) to the file on MapR File System to indicate that the file was uploaded.

If an error occurs during upload, the file is requeued for another try after a brief wait, which is 5 minutes by default. If the file could not be uploaded after a specified number of attempts, the file is dequeued to allow other files to upload. Note that the file is rediscovered, if it is still there, during the next full scan and queued for upload.

Note: When files are uploaded, although the file name remains the same, but the leading slash is removed and the full path is converted to a key on the AWS S3 bucket. For example, `/tmp/foo` after upload becomes the key `tmp/foo` on AWS S3 bucket.

Setting Extended Attributes

When the upload completes, the service uses an extended attribute, `user.s3uploadtime`, to set the timestamp on the file to when the file was uploaded. This extended attribute, which contains the last upload timestamp, is used by the service to determine if a file has changed since the file was last uploaded.

Purging the Files and Directories

The service also optionally deletes files after uploading and deletes empty directories. You can configure file deletion after upload in hours or fractions of an hour and configure empty directory deletion in minutes. Both can be set to zero, which disables the deletion function. Both functions are disabled by default.

Deletion of files and directories happen only during a full scan:

For files	For files, the modification timestamp on the file is examined to see how far back in time it was last modified.

last-updated: Apr 11, 2022

	If the file has been uploaded in its present form and if it matches or exceeds the interval configured for file deletion, the file is immediately deleted. If it does not match the configured interval for file deletion, a message is printed (to the log file, `filemigrate.log`) as to when the file will be deleted if it will be deleted fairly soon. Files that were never uploaded (such as the files that were ignored) will not be deleted by the service.
For directories	For directories, the contents of the directory and the modification timestamp on the directory are examined. If the directory is empty and the modification timestamp matches or exceeds the configured interval of time for directory deletion, the directory is deleted. The act of deleting a file or directory from a parent directory causes the modification timestamp of the parent directory to change. Therefore, an idle or empty (parent) directory is deleted only after the configured interval of time since the deletion of its children (file or subdirectory). The recursive deletion of directories has a slow bottom up behavior: first empty leaf directories are deleted and then the parents are deleted during a future full scan after the interval of time has elapsed. If a directory contains files that were never uploaded (ignored), the directory will not be deleted.

Configuring MapR for HTTPS Upload to S3

Describes how to validate and trust security certificates to permit files to be uploaded to Amazon S3.

By default, MapR only trusts its own self-signed certificates. To configure MapR to trust the certificates used by AWS S3 for HTTPS upload, you must configure additional trusted certificates. Add one of the following to the `/opt/mapr/conf/ssl_truststore` file on every node in the cluster:

- The actual certificate used by the S3 endpoints you are using
- A signer of the actual certificate used by the S3 endpoints you are using
- A certificate higher in the trust chain that ultimately does sign the certificate for the S3 endpoint

Note: Currently, the root certificate used by AWS S3 is the Baltimore CyberTrust root certificate provided by Digicert.

The following instructions are based on the assumption that you are adding the root certificate (known as the Baltimore CyberTrust root) provided by Digicert with a fingerprint of D4DE20D05E66FC53FE1A50882C78DB2852CAE474 and an expiration date of May 12, 2025. You can also add other certificates to the truststore.

Warning: Most Baltimore CyberTrust root certificates will expire in 2025 and expired certificates cannot be used for connecting to S3. When Amazon replaces its certificates with those issued by new certificate authorities, you must update the truststore to hold both the old and new root certificates for a smooth transition.

1. Download the Baltimore CyberTrust root certificate from the URL specified by Digicert, as in the following example:

```
cd /tmp
wget https://www.digicert.com/CACerts/BaltimoreCyberTrustRoot.crt
```

2. Add the certificate to the MapR truststore.
   a) Go to the directory where the `ssl_truststore` file is located.
      For example:

```
cd /opt/mapr/conf
```

   b) Run the command to add the certificate.
      For example:

```
keytool -importcert -file /tmp/BaltimoreCybrTrustRoot.crt -keystore
ssl_truststore
```

   c) Enter the keystore password when prompted.
      The default password is `mapr123`.

   d) Specify whether to trust this certificate by entering y when prompted.

      Note: The default value is **[no]**.

      For example:

```
cd /opt/mapr/conf
keytool  -importcert -file /tmp/BaltimoreCyberTrustRoot.crt -keystore
ssl_truststore
 Enter keystore password:
 Owner: CN=Baltimore CyberTrust Root, OU=CyberTrust, O=Baltimore, C=IE
 Issuer: CN=Baltimore CyberTrust Root, OU=CyberTrust, O=Baltimore,
C=IE
 Serial number: 20000b9
 Valid from: Fri May 12 14:46:00 EDT 2000 until: Mon May 12 19:59:00
EDT 2025
 Certificate fingerprints:
     MD5:  AC:B6:94:A5:9C:17:E0:D7:91:52:9B:B1:97:06:A6:E4
     SHA1: D4:DE:20:D0:5E:66:FC:53:FE:1A:50:88:2C:78:DB:28:52:CA:E4:74
     SHA256:
16:AF:57:A9:F6:76:B0:AB:12:60:95:AA:5E:BA:DE:F2:2A:B3:11:19:D6:44:AC:9
5:CD:4B:93:DB:F3:F2:6A:EB
     Signature algorithm name: SHA1withRSA
     Version: 3
 ...

 Trust this certificate? [no]:  y
 Certificate was added to keystore
```

      Wait for the message "Certificate was added to keystore" before proceeding.

last-updated: Apr 11, 2022

e) Copy the `ssl_truststore` file to all other MapR nodes in the same location (for example, `/opt/mapr/conf/`).

Ensure that the correct truststore is available on all nodes because the service can be run on any node.

Accessing the Service UI
Describes how to access the MapR Service interface.

You can access the service UI directly and using the Control System.
Accessing the Service UI Using the Control System (from MapR v6.0.1)

- Log in to the Control System and click **Services**.
  The list of services installed on the cluster displays.
- Locate File Migrate service in the list and click **File Migrate**.
  The login page for the service UI displays.
- Enter the credentials (username and password) to log in to the page and click **login**.

Accessing the Service UI Directly

- Enter the following URL in your browser (for both secure and unsecure clusters):

```
https://<hostname>:9444/
```

To change the port:

a) Modify the environment variable property that specifies the `JETTY_PORT` (whose default value is 9444) in the `/opt/mapr/conf/conf.d/warden.filemigrate.conf` file, as shown in the following example:

```
service.env=JAVA_HEAP=200m,JETTY_PORT=9444
```

b) Wait for a few minutes and then restart the service by running the following command:

```
maprcli node services -name filemigrate -nodes <node to
restart> -action restart
```

Setting up Authentication for Users

The File Migration service uses PAM Shiro to authenticate the user in a dialog presented when opening the UI. The service authentication consults the PAM configurations in `/etc/pam.d` in the following order (first file in list that exists in that directory is the one used):

1. mapr-admin
2. sudo
3. sshd
4. passwd

The UI authorizes the following users to login and use the filemigrate service:

- The owner of the filemigrate process (on the cluster, this will be the MAPR_USER).
- Any users belonging to comma-separated list of authorized users in the properties file.
- Any users belonging to any group in comma-separated list of authorized groups in the properties file.

Setting up Authentication for Users in the Properties File

Specify a comma-separated list of users and/or groups who are authorized to log in to the UI in the `FileMigrate.properties` file in `maprfs:///var/mapr/filemigrate` directory. If the properties file is not in the directory, follow the steps in Configuring the File Migration Service Using the Properties File on page 1009.

To set up the list of users and/or groups manually, set the following properties in the `FileMigrate.properties` file:

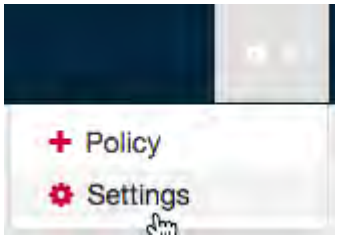
- authorized.users
- authorized.groups

To edit the file, see Managing Policies on page 1010. For more information on the properties, see FileMigrate.properties on page 2191.

Setting up Authentication for Users Using the UI

Only authorized users can log in to the UI. To set up authentication for other users:

1. Log in to the UI and select Settings from the drop-down menu.



2. Specify a comma-separated list of:
   - Users in the **Authorized Users** field under **Upload Settings**.
   - Groups in the **Authorized Groups** filed under **Upload Settings**.
3. Click **Save Changes** for the changes to take effect.

Configuring File Migration Service Using the UI

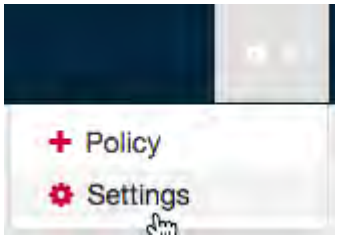
After installing and starting the service, you can access the service UI by directly entering the URL in the browser. You can then proceed to configure the properties for scanning, uploading, and monitoring the files.

Configuring Settings

Describes the interface settings for scanning and uploading files.

You can configure the settings in the interface for scanning and uploading the files.

1. Log in to the interface and select Settings from the drop-down menu.



   The **File Migration Settings** page displays.
2. Optionally, set the following for file upload:

- Full Scan Frequency
- Upload Scan Frequency
- Completion Scan Frequency
- Minimum Wait Before Upload
- Maximum Active Uploads
- Minimum Wait After Failure
- Maximum Retries Per File
- Maximum System Retries
- Minimum Idle Time Before Error

See FileMigrate.properties on page 2191 for more information.

3. Set the following to enable the service to connect and upload to AWS S3:
   - Proxy Host
   - Proxy Port
   - Proxy Username
   - Proxy Password

   See FileMigrate.properties on page 2191 for more information.
4. Click **Save Changes** for the changes to take effect.

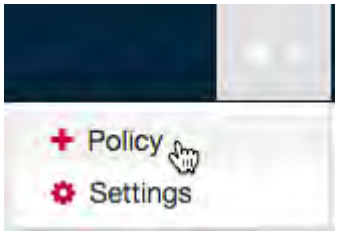
Setting up Policy

To add a new policy for file upload:

1. Log in to the UI and click **Add New File Migration Policy**.

   Alternatively, select Policy from the drop-down menu.

2. Create policies in the **Add Data Migration Policy** page.
   To create a new policy, set the following:
   - Directory Path
   - Target Bucket
   - Purge Interval

- Delete Empty Directories
- Ignore Files Regex
- X-Attributes

See FileMigrate.properties on page 2191 for more information.

3. Click **Create New Policy**.
4. Click **Restart File Migration Service**.

Configuring the File Migration Service Using the Properties File
Lists the procedure to configure the File Migration Serivce using a properties file.

To configure the File Migration Service:

1. Stop the service (if it is currently running).
   See Starting, Stopping, and Restarting the Service Using the CLI on page 1010 for more information.
2. Make a copy of the `/opt/mapr/filemigrate/filemigrate-1.0.0/conf/FileMigrate.properties.default` file and rename it to `FileMigrate.properties` if you have not yet set any properties and wish to start from the defaults, as in the following example.

   ```
   cp /opt/mapr/filemigrate/filemigrate-1.0.0/conf/
   FileMigrate.properties.default FileMigrate.properties
   ```

   Alternatively, you can use the `maprfs:///var/mapr/filemigrate/FileMigrate.properties` file if you have already used the interface to set some properties and want to perform further edits manually. See Managing Policies on page 1010 for more information.
3. Edit the `FileMigrate.properties` file to change the settings for the service, if/where necessary, and save the file.
   For more information, see FileMigrate.properties on page 2191.
4. Copy the `FileMigrate.properties` file to the `/var/mapr/filemigrate/` directory, as in the following example.

   ```
   hadoop fs -copyFromLocal /tmp/FileMigrate.properties /var/mapr/
   filemigrate
   ```

   Note: If the directory does not already exist, create it by running the `hadoop fs -mkdir` command. The `FileMigrate.properties` file contains sensitive information and the recommendation is to limit read access to the directory. If the service creates the directory, by default, only the MAPR_USER and MAPR_GROUP have read access to the directory.
5. Restart the service using the `maprcli` command, as in the following example.

   ```
   maprcli node services -name filemigrate -nodes <node to restart> -action
   restart
   ```

Managing the Service
Describes how to start, stop, and restart services using either the Control System or the CLI.

You can start, stop, and restart the service and view the service dashboard.

Starting, Stopping, and Restarting the Service Using the Control System

1. Log in to the Control System and click **Services** to display the list of services installed on the cluster.
2. See Managing Services on page 827 for information on starting, stopping, and restarting the services using the Control System.

Starting, Stopping, and Restarting the Service Using the CLI

- To:
  - Start the service, run the following command:

    ```
    maprcli node services -name filemigrate -nodes <node to start> -action start
    ```

  - Stop the service, run the following command:

    ```
    maprcli node services -name filemigrate -nodes <node to start> -action stop
    ```

  - Restart the service, run the following command:

    ```
    maprcli node services -name filemigrate -nodes <node to start> -action restart
    ```

Restarting the Service Using the Service UI

- Log in to the UI and click **Restart File Migration Service**.

Viewing the Service Dashboard

- Log in to the service UI to view the following:
  - **Uploaded Files** - Displays the number of files uploaded in the last hour.
  - **Waiting Uploads** - Displays the number of files waiting to be uploaded.
  - **Uploaded Data** - Displays the number of bytes uploaded in the last hour.

Managing Policies
Explains how to edit the Purge Interval policy and enable or disable Server Side encryption.

You can modify the Purge Interval policy and Server Side Encryption settings using the UI. You can also modify the properties directly in the properties file.

Note: Editing the properties both through the service UI and directly in the properties file may lead to inconsistencies.

Modifying Purge Interval

1. Log in to the UI and click associated with the purge interval value to edit.
2. Enter the value you want in the field.
3. Click to save the changes.

   If you want, you can click to cancel the edit operation.
4. Restart the service for the changes to take effect.

   See Restarting the Service Using the Service UI on page 1010.

Enabling or Disabling Server Side Encryption

1. Log in to the UI and select (to enable) or clear (to disable) the checkbox ( ) associated with the policy.
2. Restart the service for the changes to take effect.
   See Restarting the Service Using the Service UI on page 1010.

Editing Properties in the Properties File

You can directly edit the `maprfs:///var/mapr/filemigrate/FileMigrate.properties` file. See `FileMigrate.properties on page 2191` for complete list of properties that can be set or modified in the file.

1. Stop the File Migration service.
2. Make the necessary changes.
3. Restart the service using the `maprcli` command.
   See Starting, Stopping, and Restarting the Service Using the CLI on page 1010.

   Note: If you modified the file directly, you must restart the service using the `maprcli` command.

Removing a Policy

- Log in to the UI and click associated with the policy to remove.

Configuring Logging

Provides an overview of standard Log4J configuration.

The service uses standard Log4J configuration to control its logging. The `log4j.properties` file can be found in `<MAPR_HOME>/filemigrate/filemigrate-1.0.0/conf/` directory. By default, INFO logging is enabled. You can edit the file and restart the service to change the logging behavior.

The service logs the key actions related to scanning, uploading files, and deleting files at the INFO level in the `filemigrate.log` file. The log file also includes WARN and ERROR messages if there were problems preventing the upload of files (for example, AWS S3 was unreachable). If additional detail is needed, edit the `log4j.properties` file to increase logging to the DEBUG or TRACE levels in the properties file.

Troubleshooting

Determining if a File was Uploaded

The service externalizes the status of file uploads as an extended attribute on the files uploaded. Therefore, the service recovers easily from arbitrary crashes and restarts. Files not uploaded will be rediscovered and then uploaded.

To determine if a particular file has been uploaded by the service to AWS S3, you can examine the extended attribute on the file as shown in the following example using the unsupported script `printupload.sh`, which is included in `<MAPR_HOME>/filemigrate/filemigrate-1.0.0/bin/` directory:

```
/opt/mapr/filemigrate/filemigrate-1.0.0/bin/printupload.sh /tmp/afile
    Thu Jan  5 00:32:26 EST 2017
```

Ensuring that Login is Successful

If the username and password do not work correctly, make sure that the following properties are configured correctly in the properties file:

- `authorized.users`

last-updated: Apr 11, 2022

- `authorized.groups`

See `FileMigrate.properties` on page 2191 for information on these properties.

Managing Tiered Files from the Command-line

After creating a tiered volume and associating a tier with the volume, you can manually trigger offload and recall of individual files in the volume using the CLI. This section describes how to offload file to and recall file from the tier, retrieve status of a file-level tiering operation, and how to perform certain tiering operations when hadoop and maprcli are not available on the host you wish to use for triggering the tiering operation.

Offloading a File to a Tier Using the CLI and REST API

Describes how to offload files to a tier using the CLI and the REST API.

Files, in tiering-enabled volumes, can be offloaded individually using the CLI and REST API. See Data Offload and Purge on page 475 for more information. For information on offloading files using (loopbacknfs or FUSE-based) POSIX or NFS clients when `maprcli` or `hadoop` commands are not available, see Running Tiering Commands when maprcli and hadoop Commands are not Available on page 1014.

Note: Offloading a single file to a warm-tier might result in wasted space. See Data Offload and Purge on page 475 for more information

The user offloading the file data must have write permission (mode bit or *ACE*) on the file to offload data.

You can also offload all data in a tiering-enabled volume to the associated tier. see Offloading a Volume to a Tier on page 933 for more information.

CLI	Run one of the following commands to offload file data to a tier: • `hadoop mfs -offload <file-path>` For more information, see `hadoop mfs` on page 5373. • `/opt/mapr/bin/maprcli file offload -name <file>` For more information, see `file offload` on page 1659.
REST API	Send a request of type POST. For example: `curl -k -X POST 'https://abc.sj.us:8443/rest/file/offload?name=fileName' --user mapr:mapr`

If the manual offload succeeds, the command returns nothing. If the offload fails, the command returns an error code. For more information on the codes, see Retrying Failed Operation on page 939.

Recalling a File to MapR File System Using the CLI and REST API

You can recall individual files from a storage tier. When you recall a file, the MAST Gateway fetches a copy of the data to the cluster. Based on the expiration time setting on the volume, recalled data is automatically:

- Offloaded again based on the storage policy (rules) if there are changes to the recalled data.
- Purged when the compactor runs if there are no changes to the recalled data.

See Data Reads, Writes, and Recalls on page 481 for more information.

Note: You can recall all data from the tier to the volume. See Recalling a Volume to MapR File System on page 936 for more information.

Recalling a File Manually Using the CLI

- Run one of the following commands to recall a file:
  - ```
    hadoop mfs -recall <pathToFile>
    ```

    For more information, see `hadoop mfs` on page 5373.
  - ```
    maprcli file recall -name <pathToFile>
    ```

    For more information, see `file recall` on page 1660.

Note: For information on recalling files using (loopbacknfs or FUSE-based) POSIX or NFS clients when maprcli or hadoop commands are not available, see Running Tiering Commands when maprcli and hadoop Commands are not Available on page 1014.

Recalling a File Using the REST API

- Send a request of type POST to the URL. For example, send a request similar to the following:

```
curl -k -X POST 'https://abc.sj.us/rest/file/recall?name=fileName' --user
mapr:mapr
```

Terminating a Running File-Level Tiering Job

Explains how to terminate file tiering jobs using the Control System and the CLI.

You can terminate an ongoing offload or recall of a file using the CLI. Terminating a running:

- Offload operation does not prevent future offloads; if a schedule is associated with the volume, data that is still on the cluster is automatically offloaded based on the rules as per schedule. You can also manually offload data again at any time by running the `file offload` on page 1659 or `hadoop mfs` on page 5373 command.
- Recall operation does not prevent future recalls; you can run the recall command again to recall the remaining data on the tier. Based on the expiry time set on the volume (associated with the recalled data), recalled data is offloaded if there are changes or purged if there are no changes. See Recalling a Volume to MapR File System on page 936 for more information.

For information on terminating a running offload or recall operation using (loopbacknfs or FUSE-based) POSIX or NFS clients when maprcli or hadoop commands are not available, see Running Tiering Commands when maprcli and hadoop Commands are not Available on page 1014. For information on terminating a running volume-level job, see Terminating a Running Volume-Level Tiering Job on page 938.

Terminating a Running File-Level Offload or Recall Operation Using the CLI and REST API

CLI	Run the following command to terminate a currently running file-level offload or recall operation: ``` maprcli file tierjobabort -name <filePath> ``` For more information, see `file tierjobabort` on page 1662 .

last-updated: Apr 11, 2022

REST API	Send a request of type POST. For example: `curl -k -X POST 'https://<host>:8443/rest/file/tierjobabort?name=<filePath>' --user mapr:mapr`

Running Tiering Commands when maprcli and hadoop Commands are not Available

You can offload and/or recall a file with NFS, loopbacknfs, and FUSE-based POSIX clients even if `maprcli` or `hadoop` commands are not available. To perform file-level data tiering operations like offload and/or recall using NFS, loopbacknfs, and FUSE-based POSIX clients when `maprcli` or `hadoop` commands are not available, after mounting, provide the tiering command such as `offload`, `recall`, `tierjobstatus`, and/or `tierjobabort` as described below. When you run the command, the command creates a hidden `.tier_attributes` file (similar to `.dfs_attributes` file) that is purged immediately after the operation is submitted to the server.

Note: When you run the command, the tiering command is triggered immediately and storage policy (or rule), if any, at the volume-level is ignored.

Usage

```
/bin/echo "<command> <file-name>" > .tier_attributes
```

Note: Do not use `echo` in the terminal; instead, use `/bin/echo`.

Options

Option	Description
command	The tiering related command to run. The following commands are supported: • `offload` • `recall` • `tierjobstatus` • `tierjobabort`
file-name	The name of the file.

Return Values

On success, the command returns nothing. Otherwise, the command returns one of the following `/bin/echo` return codes, which are displayed as write errors:

Code	Message	Description
EEXIST	File exists	Indicates tier job is queued or is already in progress.
ENOTEMPTY	Directory not empty	Indicates that tier job queue is full.
ENOENT	No such file or directory	Indicates that the specified file or job ID does not exist.

Code	Message	Description
EIO	I/O error	Indicates that the job could not be submitted. Run the `tierjobstatus` command to determine the reason for this error.
EINVAL	Invalid argument	Indicates that the given command is invalid or not available. See Options on page 1014 for the list of supported commands.

Examples

Offload file named test

```
# /bin/echo "offload test"
> .tier_attributes
/bin/echo: write error: File exists
```

Recall a file named test

```
# /bin/echo "recall test"
> .tier_attributes
/bin/echo: write error: File exists
```

Check the status of a running job for a file

```
# /bin/echo "tierjobstatus test"
> .tier_attributes
```

Abort a running job

```
# /bin/echo "tierjobabort test"
> .tier_attributes
/bin/echo: write error: No such file
or directory
```

Retrieving Status of File-level Tiering Operation and File Data

You can retrieve the status of a file-level tiering operation using the CLI and REST API. For information on volume-level tiering operation, see Retrieving the Status of a Volume-level Tiering Operation on page 942.

Retrieving the Status of a Running Tiering Operation

- Run the maprcli command or send a request of type GET to check the status of an active or completed offload, abort, and/or recall operation.

  See `file tierjobstatus` on page 1663 for more information. For example:

  CLI

  ```
  maprcli file tierjobstatus -name
  <file_name> -json
  ```

  REST

  ```
  curl -k -X
  GET 'https://<host>:8443/rest/file/
  tierstatus?name<file_name>' --user
  mapr:mapr
  ```

  See Output on page 1663 for more information.

Retrieving Status of File Data

- Run the maprcli command or send a request of type GET to determine the status of file data.

  See `file tierstatus` on page 1676 for more information. For example:

CLI	`maprcli file tierstatus -name <file_name>`
REST	`curl -k -X GET 'https://<host>:8443/rest/file/tierstatus?name=<file_name>' --user mapr:mapr`

See Output on page 1677 for more information.

Administering Tables

Administration of the MapR Database is done primarily via the command line (maprcli) or with the Managed Control System (MCS). Regardless of whether the MapR Database table is used for binary files or JSON documents, the same types of commands are used with slightly different parameter options. MapR Database administration is associated with tables, columns and column families, and table regions.

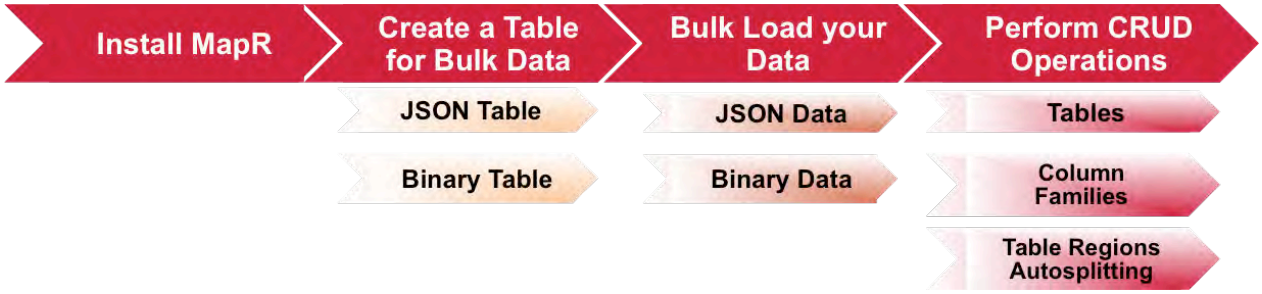
Why use MapR Database?

From an administrator's point-of-view, MapR Database provides the following capabilities:

- **Minimal administration**: Single namespace for files, tables and streams, flexible schema that allows built-in data management and protection, automatic splits and merges as data grows and shrinks, and easy bulk data loading.
- **Self-healing from HW and SW failures**: Replicated state and data for instant recovery and automated re-replication of data.
- **Global low-latency replication**: Multi-master (that is, active to active) replication which is important for disaster recovery. Includes reduced risk of data loss, application failover, and faster data access.
- **High performance and low latency**: Integrated system with fewer software layers, single hop to data, and no compactions with low I/O amplification.
- **Fine-grained security**: Access permissions can be granted to tables (as well as files and streams) at a granular level using MapR Access Control Expressions (ACEs), which are designed for flexibility and ease-of-use.

How Do I Get Started?

The following graphic shows the basics steps (with hotspot links) for getting started.



1. Install MapR
2. Creating a table for bulkloading data involves specifying the table type (JSON or binary) and setting the bulkload flag.

3. Bulkloading can be done either as a full or incremental bulkload. Different utilities are used for the bulk load depending on what you are trying to accomplish.
4. Both full and incremental bulkloads can be performed for MapR Database JSON tables. This topic describes the three command-line utilties available for loading documents into JSON tables.
5. Administration of tables describes how to create, read, update, and delete tables as well as other tasks such as managing permissions and auditing.
6. This section cover the administration of column families including how to create column families, alter them, delete them, set permissions on them, and set and display parameter values.
7. This topic describes administrative tasks associated with table regions including how to set autosplitting.

Useful Administrator Resources

Links to Resources	Descriptions
maprcli and REST API Syntax on page 1522	Command line reference for MapR operations. For MapR Database, the commands particularly applicable are assocociate with the `maprcli` table on page 1788 command. These commands include not only table CRUD operations but also table column family, table region, and table replication operations.
Utilities for MapR Database JSON Tables on page 5312	Utilities for MapR Database JSON tables. These utilities are used form managing JSON tables icluding importing and exporting data to and from JSON tables. Particularly useful are: • MapR Database JSON ImportJSON on page 5322 utility with imports JSON documents into a MapR Database JSON table. • MapR Database Shell (JSON Tables) on page 5286utility which performs CRUD operations on JSON documents and tables.
Utilities for MapR Database Binary Tables on page 5329	Utilities for MapR Database binary tables. These utitlies are used for managing binary tables. Particulary useful is MapR Database Binary CopyTable on page 5329 which is used to copy data from one MapR Database binary table to another. **Note:** To import HFile or Result files in a MapR Database binary table, the hbase command can be used. See Loading Data into Binary Tables on page 1040.
Configuring Security on page 1402	Information on security tasks for configuring MapR security, managing secure clusters, and administering auditing.
Hadoop and Big Data Security	MapR information on Security and Big Data Governance that identified key unique advantages including authentication, authorization, auditing, and encryption.
Provisioning Secure Access Controls in MapR Database	MapR blog discussing MapR's boolean Access Control Expressions (ACEs) which provide granular-level permissions including topics and examples of best practices.

Managing Tables

MapR Database supports two types of table: binary tables and JSON tables. This section covers how to create, edit, and delete tables, as well as how to set parameter values, display parameter values, grant permissions and access, replicate tables, and more using the Control System and the CLI.

When you log in to the Control System and click **Data** > **Table**, the **Tables** page displays the following in the various panes:

- Active Alarms — The active table (replication) alarms
- **Recently Viewed Tables** — The tables that were most recently accessed from the Control System
- List of volumes that you have access to and a search field to retrieve a table by table path

Click **Create Table** button to create a new binary or JSON table. For a conceptual overview of:

- Binary tables, see MapR Database Binary Tables on page 540.
- JSON tables, which contain JSON documents, see MapR Database JSON Tables on page 524.

Creating a Table

Explains how to create binary and JSON tables using either the Control System, or the CLI.

There are several methods that you can use to create MapR Database tables including the maprcli, and hbase shell, mapr dbshell commands, and through the Control System. There are two methods that you can use to create both binary tables and JSON tables:

- Control System
- CLI

Creating a Table Using the Control System

To create a table from the Control System, under **Data** > **Tables**:

Note: This option is not available on the Kubernetes version of the Control System.

1. Click **Create Table**.
   The **Create New Table** page displays.
2. Choose the table type:
   - JSON — see MapR Database as a Document Database on page 507
   - Binary — see MapR Database as a Column-Oriented Database on page 539
3. Specify the following properties under **PROPERTIES**.
   a) Enter path to the table in the **Table Path** field.
   Tables are stored in the MapR filesystem. When providing the path to a table, use these conventions.
   - For a path on the local cluster, start the path at the volume mount point. For example, for a table named test under a volume with a mount point at /volume1, specify the following path: `/volume1/test`
   - For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named customer in volume1 in the sanfrancisco cluster, specify the following path: `/mapr/sanfrancisco/volume1/customer`

b) Select the interval of time to apply when logging table metrics.

You can choose to log metrics every 10 seconds, 1 minute (default), or 10 minutes. For visualizing the metrics, see Visualizing Table Metrics in the Control System on page 1303.

c) Enable (**Yes**) or disable (**No**) auto-splitting of table.

If enabled, the table will be split automatically into regions as the table grows. If disabled, the table can be split manually into regions. By default, auto-splitting is enabled.

d) Enable (**Yes**) or disable (**No**) full bulk load of the table.

For more information, see Loading Documents into JSON Tables on page 1036. By default, full bulk load is disabled.

e) Enable (**Yes**) or disable (**No**) auditing of table operations.

If auditing is enabled at the cluster and volume levels, enabling auditing will cause auditing to start for the table operations. For more information, see Auditing of Operations on MapR Database Binary Tables and JSON Tables on page 700.

4. Specify users, groups, and/or roles that have and/or do not have the following types of access to the table under **USER ACCESS CONTROLS**.

JSON Table

Administration	Can view and edit the permissions for the table.
Force Pack	Can pack table regions.
Split Merge	Can take the following actions: • Split the table into regions or merge regions of the table together. • Change the size of the region.
Index	Can create index for this table.
Bulkload	Can load this table with bulk loads if the table was created with bulk load support.
Replication Access	Can set up replication either to or from a table.
Create/Rename Column Family	Can create column families for this table or rename existing column families.
Delete Column Family	Can delete column families associated with the table.

Binary Table

Administration	Can view and edit the permissions for the table.
Force Pack	Can pack table regions.

last-updated: Apr 11, 2022

Split Merge	Can take the following actions: • Split the table into regions or merge regions of the table together. • Change the size of the region.
Bulkload	Can load this table with bulk loads if the table was created with bulk load support.
Replication Access	Can set up replication either to or from a table.
Create/Rename Column Family	Can create column families for this table or rename existing column families.
Delete Column Family	Can delete column families associated with the table.

By default, all permissions are given to the user creating the table.

To grant or block access to users, groups, and/or roles, from the:

- Basic settings, select the type — public, (OR) user, group, or role — from the drop-down menu, specify the name of the user, group, or role, and select one or more checkbox to grant permissions.

  Tip: Click to create a copy of the associated access control setting. Click to remove the associated access control expression.

  To add *ACE*s for another user, group, or role, click **Add Another** and repeat this step.

- Advanced settings, specify public (p) or user (u), group (g), and/or role (r) who have or do not have the type of access using the following boolean expressions and subexpressions:
  - ! — Negation operator.
  - & — AND operation.
  - | — OR operation.

  Use ( ), parentheses, for subexpressions.

  Note: You *cannot* specify user, group, or role individually if access is granted to all users (public).

  Alternatively, click associated with the type of access to use the **Access Control Expression** window to define access for public or users, group, and/or role. See Defining ACEs on page 1473 for more information.

Note: If you switch from **Basic** to **Advanced**, the basic settings, if any, are carried over to the advanced settings. If you switch from **Advanced** to **Basic**, all the settings are lost because the subexpressions and AND (`&`) and negation (`!`) operations that are supported by advanced settings are not supported in the basic settings.

You can proceed to:

- Step 5 to set default column family permissions.
- Step 6 to create the table.

5. Click **Default Column Family Authorization**.
   You can set up default permissions for column families on this page.

JSON Table

Specify public, (OR) users, groups, and/or roles that have and/or do not have the following types of access to the column families under **USER ACCESS CONTROLS**.

Read Data	Can do column reads. Reads require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Write Data	Can do column writes. Writes require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.

last-updated: Apr 11, 2022

Traverse Data	Can pass over fields in JSON documents. For example, suppose that a JSON table contains documents of this general structure: `{ "_id" : "ID", "a" : { "b" : "value", "c" : "value" } }` Suppose further that the user sjohnson has read permission on a.b, but not on a. For sjohnson to read a.b, the user needs the traverse permission on a. The user can then pass over field a to a.b. This permission is inherited by fields within the column family.
Set Compression	Can set or change the compression setting for the column family.

Binary Table

Specify public, (OR) users, groups, and/or roles that have and/or do not have the following types of access to the column families under **USER ACCESS CONTROLS**.

Read Data	Can do column reads. Reads require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Write Data	Can do column writes. Writes require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Append Data	Can do column appends. Column appends require permission both at the column-family level and at the column level.

Set Version	Can set or change the maximum and minimum number of versions of column values to keep.
Set Compression	Can set or change the compression setting for the column family.

By default, all permissions are given to the user creating the table.

To grant or block access to users, groups, and/or roles, from the:

- Basic settings, select the type — public, (OR) user, group, or role — from the drop-down menu, specify the name of the user, group, or role, and select one or more checkbox to grant permissions.

  Tip: Click to create a copy of the associated access control setting. Click to remove the associated access control expression.

  To add *ACE*s for another user, group, or role, click **Add Another** and repeat this step.

- Advanced settings, specify public (p) or user (u), group (g), and/or role (r) who have or do not have the type of access using the following boolean expressions and subexpressions:
  - ! — Negation operator.
  - & — AND operation.
  - | — OR operation.

  Use ( ), parentheses, for subexpressions.

  Note: You *cannot* specify user, group, or role individually if access is granted to all users (public).

  Alternatively, click associated with the type of access to use the **Access Control Expression** window to define access for public or users, group, and/or role. See Defining ACEs on page 1473 for more information.

Note: If you switch from **Basic** to **Advanced**, the basic settings, if any, are carried over to the advanced settings. If you switch from **Advanced** to **Basic**, all the settings are lost because the subexpressions and AND (&) and negation (!) operations that are supported by advanced settings are not supported in the basic settings.

6. Click **Create Table** to create the table.
   You can proceed to:
   - Add column families to the table.
   - View the table information for the newly created table.

Creating a Table Using the CLI or the REST API

The basic command to create a binary table is:

```
maprcli table create -path <path>
```


last-updated: Apr 11, 2022

To create a JSON table, include the `-tabletype` parameter and set it to `json`:

```
maprcli table create -path <path> -tabletype json
```

The `-tabletype` parameter is set to `binary` by default.

The format of the value of the `-path` parameter depends on whether you are creating a table on a local cluster or a remote cluster:

- For a path on the local cluster, start the path at the volume mount point. For example, for a table named `test` under a volume with a mount point at /volume1, specify the following path: `/volume1/test`
- For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customer` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`

  Note: You cannot use the following characters in the table name:

  ```
  < > ? % \
  ```

  To use the following characters in the table name, enclose them either in single or double quotes:

  ```
  ; | ( ) /
  ```

  For example:

  ```
  maprcli table create -path "/♦=#;{}&()/"  (or)
  maprcli table create -path '/♦=#;{}&()/'
  ```

  To use either the **'** or the **"** character in the table name, enclose:

  - the **'** character within double quotes (")
  - the **"** character within single quote (')

  For example:

  ```
  maprcli table create -path "/'♦=#;{}&()/"  (or)
  maprcli table create -path '/"♦=#;{}&()/'
  ```

When you create a table with this command, you can set a number of properties. See `table create` on `page 1788`.

Creating Tables Using shell Command

JSON Tables	The MapR Database shell command is used only on JSON tables. To run this command, execute the following: `mapr dbshell` After starting the shell, run the `create` command. See `MapR Database Shell (JSON Tables)` on `page 5286`.

Binary Tables	The HBase shell command is used on binary tables only. To run this command, execute the following: `hbase shell` After starting the HBase shell, run the `create` command. Type `help` to see a list of commands and their syntax. See MapR Database HBase Shell (Binary Tables) on page 5325.

Configuring Maximum Row Sizes Using the CLI

The default maximum row size at installation is 32MB. You can configure this maximum by changing the value of the `mfs.db.max.rowsize.kb` parameter with the `maprcli config save` command.

Tables support rows up to 2 GB in size. Rows in excess of 100MB might show decreased performance.

Here is an example of changing the maximum row size:

```
maprcli config save -values {"mfs.db.max.rowsize.kb":<value in KB>}
```

The value of this parameter affects both JSON tables and binary tables.

To view the current setting of this parameter, use the `maprcli config load` command, as in this example:

```
maprcli config load -json | grep mfs.db.max.rowsize.kb
```

Editing Tables

Use either the `maprcli` command `table edit` or the Control System to edit the attributes of a MapR Database binary or JSON table. You can also use the HBase shell to edit a binary table.

Editing Tables Using the Control System

To edit a (JSON or Binary) table using the Control System:

1. Go to the table information page.
   See Viewing Table Information on page 1032.
2. Click **Edit Table** to display the **Edit Table** page.
3. Make changes (optional) to the following properties:

Metrics Interval	The interval of time for logging metrics.
Auto Split	Enable (**Yes**) or disable (**No**) auto-splitting of table. If enabled, the table will be split automatically into regions as the table grows. If disabled, the table can be split manually into regions. By default, this is enabled.
Bulkload	Enable (**Yes**) or disable (**No**) full bulk load of the table.
Enable Auditing	Enable (**Yes**) or disable (**No**) auditing of table operations. If auditing is enabled at the cluster and volume levels, enabling auditing will cause auditing to start for the table.

4. Make changes (optional) to table administration control settings.
   a) Make changes to **USER ACCESS CONTROLS** under **Table Setup and Authorization**.
      Modify the list of users, groups, and/or roles that have and/or do not have the following types of access to the table:

Admin	Can view and edit the permissions for the table.
Force Pack	Can pack table regions.

last-updated: Apr 11, 2022

Split Merge	Can take the following actions: • Split the table into regions or merge regions of the table together. • Change the size of the region.
Bulkload	Can load this table with bulk loads if the table was created with bulk load support.
Replication Access	Can set up replication either to or from a table.
Create/Rename Column Family	Can create column families for this table or rename existing column families.
Delete Column Family	Can delete column families associated with the table.

To grant or block access to users, groups, and/or roles, from the:

- Basic settings, select the type — public, (OR) user, group, or role — from the drop-down menu, specify the name of the user, group, or role, and select one or more checkbox to grant permissions.

  Tip: Click to create a copy of the associated access control setting. Click to remove the associated access control expression.

  To add *ACE*s for another user, group, or role, click **Add Another** and repeat this step.

- Advanced settings, specify public (p) or user (u), group (g), and/or role (r) who have or do not have the type of access using the following boolean expressions and subexpressions:
  - ! — Negation operator.
  - & — AND operation.
  - | — OR operation.

  Use ( ), parentheses, for subexpressions.

  Note: You *cannot* specify user, group, or role individually if access is granted to all users (public).

  Alternatively, click associated with the type of access to use the **Access Control Expression** window to define access for public or users, group, and/or role. See Defining ACEs on page 1473 for more information.

Note: If you switch from **Basic** to **Advanced**, the basic settings, if any, are carried over to the advanced settings. If you switch from **Advanced** to **Basic**, all the settings are lost because the subexpressions and AND (&) and negation (!) operations that are supported by advanced settings are not supported in the basic settings.

b) Make changes to **Default Column Family Authorization**.

JSON Table	Modify the list of users, groups, and/or roles that have and/or do not have the following types of access to the default column family:

last-updated: Apr 11, 2022

Read Data	Can do column reads. Reads require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Write Data	Can do column writes. Writes require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Traverse Data	Can pass over fields in JSON documents. For example, suppose that a JSON table contains documents of this general structure: `{ "_id" : "ID", "a" : { "b" : "value", "c" : "value" } }` Suppose further that the user sjohnson has read permission on a.b, but not on a. For sjohnson to read a.b, the user needs the traverse permission on a. The user can then pass over field a to a.b. This permission is inherited by fields within the column family.
Set Version	Can set or change the maximum and minimum number of versions of column values to keep.
Set Compression	Can set or change the compression setting for the column family.

Binary Table

Modify the list of users, groups, and/or roles that have and/or do not have the following types of access to the default column family:

last-updated: Apr 11, 2022

Read Data	Can do column reads. Reads require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Write Data	Can do column writes. Writes require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Append Data	Can do column appends. Column appends require permission both at the column-family level and at the column level.
Set Version	Can set or change the maximum and minimum number of versions of column values to keep.
Set Compression	Can set or change the compression setting for the column family.

To grant or block access to users, groups, and/or roles, from the:

- Basic settings, select the type — public, (OR) user, group, or role — from the drop-down menu, specify the name of the user, group, or role, and select one or more checkbox to grant permissions.

  Tip: Click to create a copy of the associated access control setting. Click to remove the associated access control expression.

  To add *ACE*s for another user, group, or role, click **Add Another** and repeat this step.

- Advanced settings, specify public (p) or user (u), group (g), and/or role (r) who have or do not have the type of access using the following boolean expressions and subexpressions:
  - ! — Negation operator.
  - & — AND operation.
  - | — OR operation.

  Use ( ), parentheses, for subexpressions.

  Note: You *cannot* specify user, group, or role individually if access is granted to all users (public).

  Alternatively, click associated with the type of access to use the **Access Control Expression** window to define access for public or users, group, and/or role. See Defining ACEs on page 1473 for more information.

Note: If you switch from **Basic** to **Advanced**, the basic settings, if any, are carried over to the advanced settings. If you switch from **Advanced** to **Basic**, all the settings are lost because the subexpressions and AND (`&`) and negation (`!`) operations that are supported by advanced settings are not supported in the basic settings.

5. Click **Save Changes** for the changes to take effect.

Editing Tables Using the CLI or the REST API

Run the command `maprcli table edit -path <path>`.

- For a path on the local cluster, start the path at the volume mount point. For example, for a table named `test` under a volume with a mount point at /volume1, specify the following path: `/volume1/test`
- For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customer` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`

  Note: You cannot use the following characters in the table name:

  ```
  < > ? % \
  ```

  To use the following characters in the table name, enclose them either in single or double quotes:

  ```
  ; | ( ) /
  ```

  For example:

  ```
  maprcli table create -path "/♦=#;{}&()/"  (or)
  maprcli table create -path '/♦=#;{}&()/'
  ```

  To use either the **'** or the **"** character in the table name, enclose:

  - the **'** character within double quotes (")
  - the **"** character within single quote (')

  For example:

  ```
  maprcli table create -path "/'♦=#;{}&()/"  (or)
  maprcli table create -path '/"♦=#;{}&()/'
  ```

When you edit a table with this command, you can change a number of properties.

- Enable or disable auditing, autosplitting, and bulkloading
- Set permissions on table
- Set permissions for default column families

For full reference for this command, see `table edit` on page 1822.

Editing Binary Tables Using HBase Shell

After starting the HBase shell, run the `alter` command. Type `help` to see a list of commands and their syntax.

Removing a Table
Use either the Control System or the `maprcli table delete` command to drop a MapR Database table.

Removing a Table Using the Control System

To remove a table:

1. Go to the table information page.
   See Viewing Table Information on page 1032.
2. Click **Remove Table**.
   The **Remove Table** confirmation window displays.
3. Click **Remove Table** to remove the table.
   After the table is removed, data in the table cannot be recovered.

Removing a Table Using the CLI or REST API

Run the command `maprcli table delete -path <path>`.

- For a path on the local cluster, start the path at the volume mount point. For example, for a table named `test` under a volume with a mount point at /volume1, specify the following path: `/volume1/test`
- For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customer` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`

  Note: You cannot use the following characters in the table name:

  ```
  < > ? % \
  ```

  To use the following characters in the table name, enclose them either in single or double quotes:

  ```
  ; | ( ) /
  ```

  For example:

  ```
  maprcli table create -path "/♦=#;{}&()/"  (or)
  maprcli table create -path '/♦=#;{}&()/'
  ```

  To use either the **'** or the **"** character in the table name, enclose:

  - the **'** character within double quotes (")
  - the **"** character within single quote (')

  For example:

  ```
  maprcli table create -path "/'♦=#;{}&()/"  (or)
  maprcli table create -path '/"♦=#;{}&()/'
  ```

For more information, see `table delete` on page 1820.

Defining ACEs
Indicates how to build access control expressions (ACEs) using the Expression Builder.

To define access control expressions using the **Access Control Expression** builder in the MapR Control System:

1. Choose **All** or **Any** (from the drop-down menu) of the settings to match for access.
   Here:

All	AND (&) operation	Indicates that all of the conditions must be met for public or user, group, and/or role to access the volume.
Any	OR (|) operation	Indicates that any one of the conditions must be met for public or user, group, and/or role to access the volume.

2. Click:

+	To add an expression.
( )	To add a subexpression.
x	To remove an expression or subexpression.

3. Select Public or User, Group, or Role from the drop-down menu and:
   a) Choose **Is** to grant or **Is not** to block access to the user, group, or role.
   b) Enter name of the user, group, or role.
4. Click **Save Changes** to create an Access Control Expression.

Setting Whole Volume ACEs Using the CLI

See Setting Whole Volume ACEs on page 1459.

Setting Table ACEs Using the CLI

See Enabling Table and Stream Authorizations with ACEs on page 1461.

Setting Stream ACEs Using the CLI

See Enabling Table and Stream Authorizations with ACEs on page 1461.

Viewing the List of Tables

Describes how to view the list of tables using either the Control System or the CLI.

Viewing the Tables in a Volume Using the Control System

1. Log in to the Control System and click **Data** > **Tables** to view all the volumes to which you have access.

   Note: This option is not applicable on the Kubernetes version of the Control System.

   For each volume, the pane displays the following:

Column Name	Column Description
Name	The name of the volume.
Type	The type. Value can be: • — Volume • — Directory • — Table

last-updated: Apr 11, 2022

Column Name	Column Description
Owner	The name of the owner.
Last Modified	The last modification date and timestamp.

2. Click on the name of the volume (to browse to the path to the table) or enter the name of the volume in the text field.
   The tables in the selected volume display. If necessary, click the name of the directory to browse further or  to return to **All** volumes view.

Viewing a Table by Table Path Using the Control System

1. Log in to the Control System and click **Data** > **Tables**.

   Note: This option is not applicable on the Kubernetes version of the Control System.

2. Enter the path to the table and click **GO**.
   The tables information page for the specified table displays.

Listing the Tables in a Directory From the Command-line
Method for Binary Tables Only (HBase Shell)

After starting the HBase shell, run the `list` command. Include the directory path in the command if you want to list tables that are not in your home directory. Type `help` to see a list of commands and their syntax.

Method for JSON Tables Only (`mapr dbshell`)

After starting the shell, run the `list` command. Include the directory path in the command if you want to list tables that are not in your home directory. See MapR Database Shell (JSON Tables) on page 5286.

Retrieving a Table by Table Path Using the CLI or the REST API

To retrieve table details for a table by specifying the table path from the CLI, run the following command:

```
maprcli table info -path <table-path>
```

For information on this command, see `table info` on page 1838.

Viewing Table Information
Explains how to view table information including table properties, column families, regions, replicas, upstream source, and indexes using either the Control System or the CLI.

Use either the `maprcli` command or the Control System to display all of the information that MapR Database stores about a particular table including:

- The path to the table
- Whether a table is a binary or JSON table
- The number of rows in the table
- The logical size (in bytes) of the table
- The physical size (in bytes) of the table
- The maximum size (in bytes) of regions of the table
- Whether auditing is enabled or not

- Whether autosplitting is on or off
- Whether the bulkload flag is set or not
- The interval for logging metrics
- The permissions on the table

Note: The logical and physical sizes of the table are estimated values.

Displaying Table Information Using the Control System

To view table information using the Control System:

1. Search and retrieve the table either by volume or by table path.
   For information on retrieving, see:
   - Viewing the Tables in a Volume Using the Control System on page 1031
   - Viewing a Table by Table Path Using the Control System on page 1032
2. Click the name of the table to see the table details.
   The page displays the following tabs:
   - Summary
   - Column Families
   - Regions
   - Replication
   - Change Data Capture
   - Indexes
   - Metrics

   You can:
   - Edit table
   - Remove table

   The **Summary** tab displays:
   - The active and recent alarms in the **Active Alarms** pane.
   - The table settings and permissions in the **Detail** pane.

Displaying Table Information Using CLI

Run the command `maprcli table info -path <path> -json` .

- For a path on the local cluster, start the path at the volume mount point. For example, for a table named `test` under a volume with a mount point at /volume1, specify the following path: `/volume1/test`
- For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customer` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`

Note: You cannot use the following characters in the table name:

```
< > ? % \
```

To use the following characters in the table name, enclose them either in single or double quotes:

```
; | ( ) /
```

For example:

```
maprcli table create -path "/♦=#;{}&()/"  (or)
maprcli table create -path '/♦=#;{}&()/'
```

To use either the **'** or the **"** character in the table name, enclose:

- the **'** character within double quotes (")
- the **"** character within single quote (')

For example:

```
maprcli table create -path "/'♦=#;{}&()/"  (or)
maprcli table create -path '/"♦=#;{}&()/'
```

The `json` parameter displays the output as a JSON document.

Viewing Table Regions
Use either the Control System or the CLI to list the regions in which a table's data is located.

MapR Database tables are split into regions on an ongoing basis. Administrators and developers do not need to manage these regions or restructure data on disk when data is added and deleted. These operations happen automatically. You can view region information for tables to get a sense of the size and location of table data on the MapR cluster.

Displaying the Regions Using the Control System

To display the regions of a table:

1. Go to the table information page.
   See Viewing Table Information on page 1032.
2. Do one of the following:
   - Click **Regions** to view the list of regions for the table.
   - Click the name of the index in the **Indexes** tab to view the regions for the index.

   For each region, the **Regions** pane displays the following:

Column Name	Column Description
Start Key	Value of the start key for this region. For the first region in a table, this value is exclusive. For all other regions, it is inclusive.
End Key	Value of the end key for this region. This value is always exclusive.
Physical Size	The physical size of the region with data compression (excluding replication).

Column Name	Column Description
Logical Size	The logical size of the region without data compression (excluding replication).
No of Rows	Number of rows in the region.
Primary Node	The host name and port of the primary node for this region.
Secondary Node	The host names and ports of the secondary nodes where this region is replicated.
Last HB	The time since last heartbeat from the region's primary node.
Region Identifier	The region's FID.

Displaying Region Information Using the CLI or the REST API

The basic command to retrieve the list of regions that make up the table is:

```
maprcli table region list -path <path>
```

- For a path on the local cluster, start the path at the volume mount point. For example, for a table named `test` under a volume with a mount point at /volume1, specify the following path: `/volume1/test`
- For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customer` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`

  Note: You cannot use the following characters in the table name:

  ```
  < > ? % \
  ```

  To use the following characters in the table name, enclose them either in single or double quotes:

  ```
  ; | ( ) /
  ```

  For example:

  ```
  maprcli table create -path "/♦=#;{}&()/"  (or)
  maprcli table create -path '/♦=#;{}&()/'
  ```

  To use either the **'** or the **"** character in the table name, enclose:

  - the **'** character within double quotes (")
  - the **"** character within single quote (')

  For example:

  ```
  maprcli table create -path "/'♦=#;{}&()/"  (or)
  maprcli table create -path '/"♦=#;{}&()/'
  ```

The `json` parameter displays the output as a JSON document.

To run this command, your user ID must have the following permissions:

- `readAce` on the volume

last-updated: Apr 11, 2022

- `lookupdir` on directories in the path

See `table region list` on page 1844.

Loading Documents into JSON Tables
There are three command-line utilities for loading documents into JSON tables: `mapr copytable`, `mapr importtable` (which works in conjunction with the `mapr exporttable` utility), and `mapr importJSON`

You can choose whether to have these utilities perform bulk loads or incremental loads.

For bulk loads, the `-bulkload` parameter of the JSON table must be set to `true`. During a bulk load, client applications are unable to access the table. After the utility is finished, you must set the table's `-bulkload` parameter to `false`, so that client applications can access the table again.

When you set the `-bulkload` parameter to true, you cannot enable replication on the table. Since this effectively disables logging on the table, MapR Database also does not capture log data that Elasticsearch can use to index the table.

Note: Incremental loads allow client applications to access the table as the documents are loaded. However, incremental loads are slower than bulk loads.

`mapr copytable`

The `mapr copytable` utility copies documents -- all documents or a subset determined by a range of row keys, and all fields or a subset of fields -- directly from oneJSON table to another.

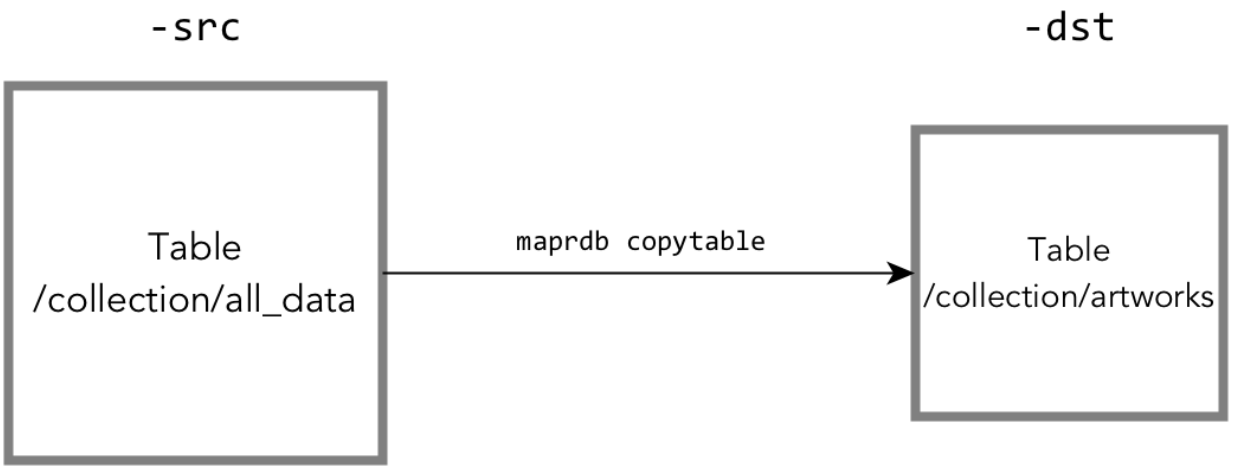


Figure 14: Copying a subset of data from one table to another

For reference information about this utility, see `mapr copytable`.

`mapr exporttable` and `mapr importtable`

The `mapr exporttable` utility exports data from a JSON table to binary sequence files that you can import into another JSON table by using the `mapr importtable` utility.

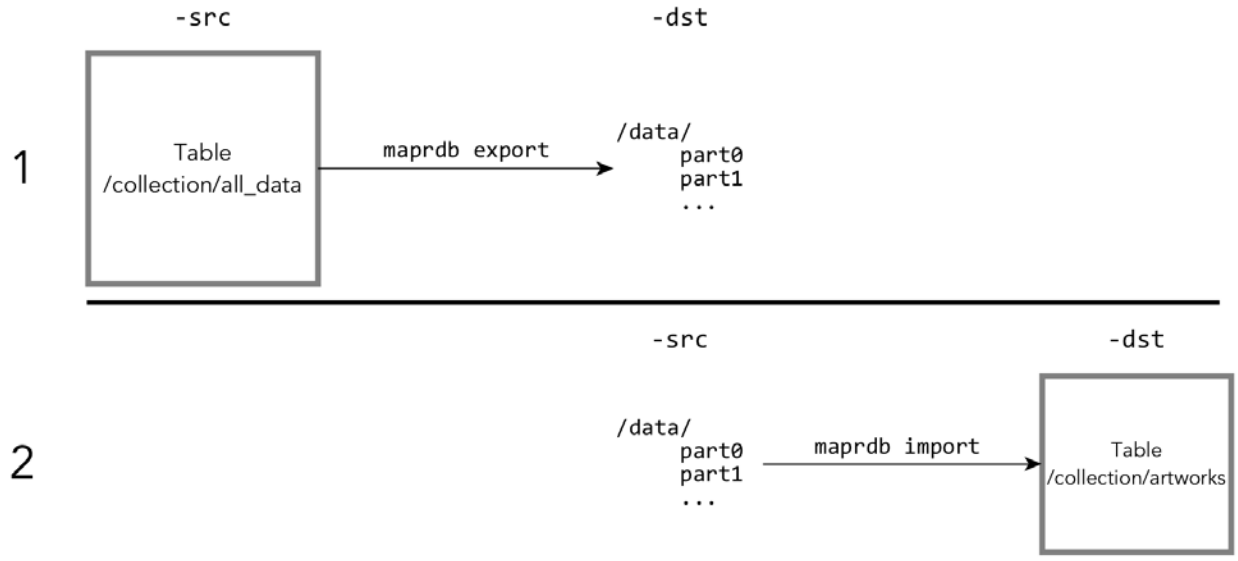


Figure 15: JSON documents exported from a JSON table as binary sequence files and then imported into another JSON table

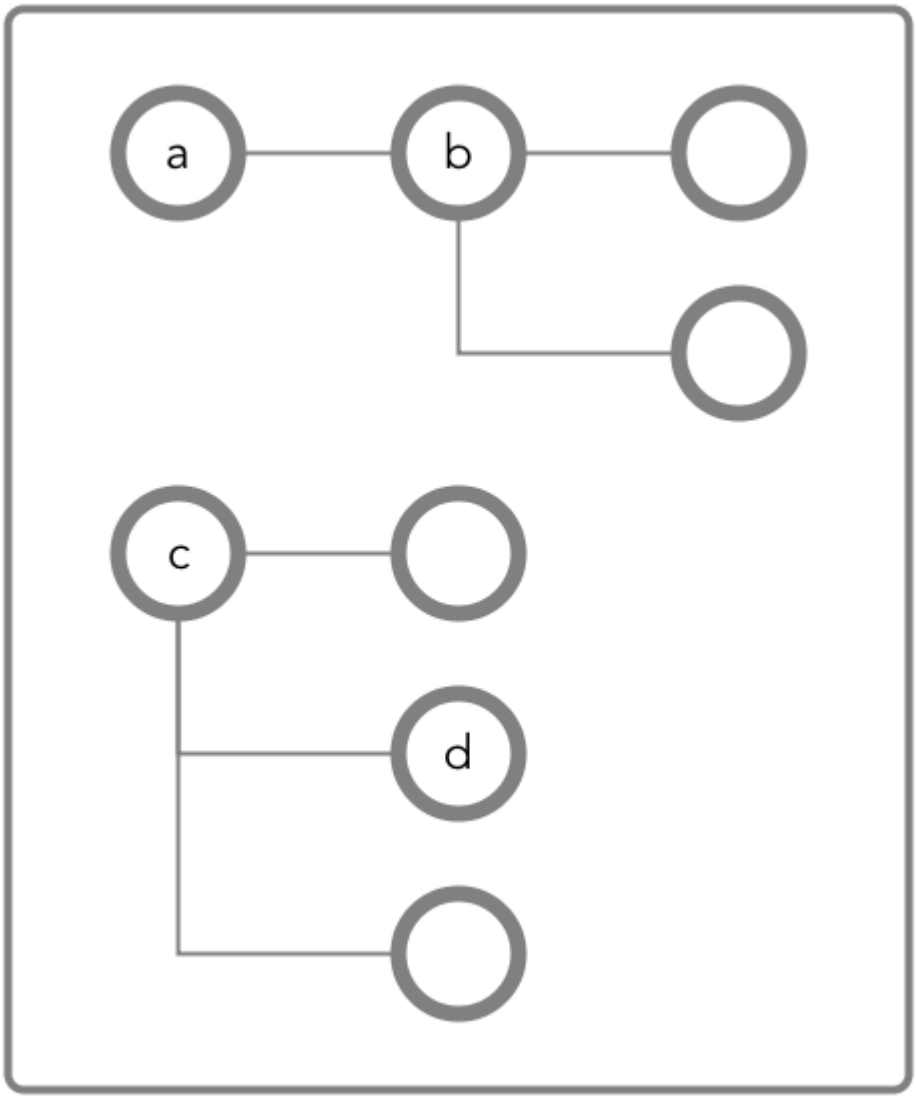
The destination directory is created by the `mapr exporttable` utility. To prevent accidental overwriting of data, the `mapr exporttable` utility fails if the destination directory already exists.

The command for running the `mapr importtable` utility in step 2 of the diagram above would look like this:

```
mapr importtable -src /data/* -dst /collection/artworks
```

The `-columns` parameter of the `mapr exporttable` utility lets you export subsets of the fields in the documents that are in a table. For example, to export field b, the fields under it, and field d from documents with the following structure, the command to run the `mapr exporttable` utility would look like this:

```
mapr exporttable -columns a.b,c.d -src /collection/all_data -dst /data
```



For reference information about these commands, see MapR Database JSON ExportTable and ImportTable on page 5320.

`mapr importJSON`

This utility imports one or more JSON documents that are text files into a JSON table.

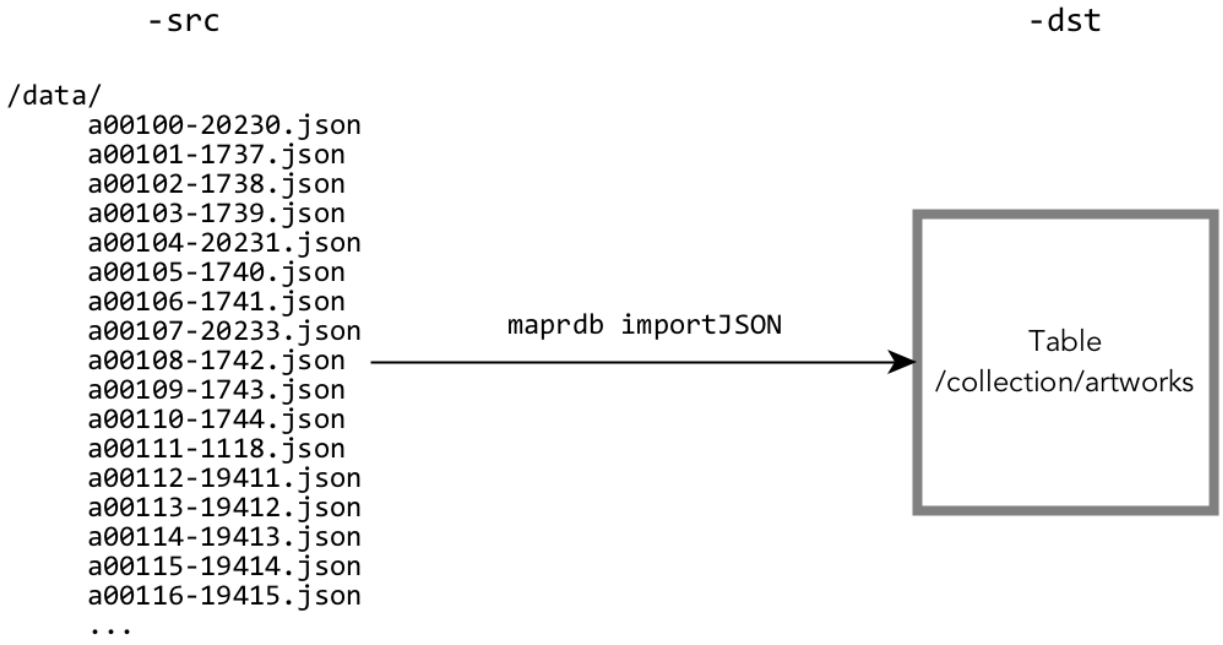


Figure 16: JSON documents in a folder named /data being imported into a JSON table

If the each document does not already contain an `_id` field to use as a document ID, the `mapr importJSON` utility adds an `_id` field during the import. Use the `-idfield` parameter to specify the name of the field that contains the value to copy into the value of the `_id` field.

For example, each document might have a `product_ID` field that contains a universally unique identifier. You could run the utility with this command:

```
mapr importJSON -idfield "product_ID" -src /data/* -dst /collection/artworks
```

For reference information about this command, see `mapr importJSON`.

Accessing MapR Database Binary Tables Using HBase APIs
This section describes how to access MapR Database binary tables via HBase APIs, HBase Shell, and MapReduce applications.

MapR Core extends the HBase component to handle access to MapR Database binary tables. MapR Database binary tables do not support low-level HBase API calls that are used to manipulate the state of an Apache HBase cluster.

For a full list of supported HBase Java APIs, see Creating Java Apps - Binary Tables on page 2478.

For a full list of supported commands in the HBase shell, see HBase Shell for MapR Database.

To enable the HBase API and `hbase shell` access, install the `mapr-hbase` package on every node in the cluster. The HBase component of the MapR distribution for Hadoop is typically installed under `/opt/mapr/hbase`. Install the `mapr-hbase` package that provides the version that corresponds with the current EEP and MapR version you are running.

For MapR installation procedures, see Installing MapR and MapR Ecosystem Components on page 128.

Note: The version of HBase provided by MapR has been modified to work with MapR Database binary tables. Do not download and install stock Apache HBase on a MapR cluster that uses MapR tables.

Note: If you use fat JARs to deploy your application as a single JAR including all dependencies, be aware that the fat JAR may contain versions of HBase that override the installed MapR versions, leading to problems. Check your fat JARs for the presence of stock HBase to prevent this problem.

Loading Data into Binary Tables
Bulkload operations can be performed as a full bulkload or as an incremental bulkload.

The most common way of loading data into a MapR Database Binary Tables is with a put operation. However, at large scales, bulk loads offer a performance advantage over put operations.

Bulk loading is supported by the following tools, which can be used for both full and incremental bulkload operations:

- Hbase MapR Database Binary CopyTable on page 5329 utility which copies MapR Database binary table data, table metadata, access control expressions, and more to another MapR Database binary table.

  ```
  hbase com.mapr.fs.hbase.tools.mapreduce.CopyTable
  ```

- Hbase `ImportFiles` utility which imports HFile or Result files into MapR Database binary tables. For example:

  ```
  hbase com.mapr.fs.hbase.tools.mapreduce.ImportFiles
     -Dmapred.reduce.tasks=2
     -inputDir < input directory, for example: /test/tabler.kv >
     -table < table name, for example: /table2 >
     [ -format < Result|HFile > ]
     [ -sample < true|false > ]
     [ -mapOnly < true|false > ]
  ```

Full Bulk Loads

Full bulkload operations offer the best performance advantage because it skips the write-ahead log (WAL) typical of MapR Database binary table operations. Full bulkload operations can only be performed on empty tables that have the `bulkload` attribute set to **true**. This value is set only when creating a table.

When you set the `bulkload` attribute, you cannot enable replication on the table. Since this effectively disables logging on the table, MapR Database also does not capture log data that Elasticsearch can use to index the table.

Important: Tables are unavailable for normal client operations, including put, get, and scan operations, while a full bulkload operation is in progress.

To create a MapR Database binary table for bulkloading, use one of the following:

- maprcli table create command with tthe `-bulkload` parameter set to true.
- Apache HBase shell `create` command with the `BULKLOAD` parameter set to `true`. For example:

  ```
  hbase> create '/a0','f1', BULKLOAD => 'true'
  ```

  If you want to pre-split a table, separate the `BULKLOAD` parameter from the `SPLITS` parameter. For example:

  ```
  hbase> create '/t1', 'f1', {SPLITS => ['10', '20', '30']}, {BULKLOAD =>
  'true'}
  ```

- Control System with **Will table be bulkload?** option set to **Yes** under table **PROPERTIES**.

Note: Attempting a full bulkload on a table that does not have the bulkload attribute set to true results in an incremental bulkload being performed instead.

After you perform a full bulkload on a table, you cannot perform a full bulkload on it again. For example:

- You cannot use the `maprcli table edit` command to set the `bulkload` parameter to TRUE again.
- You cannot use the Apache HBase shell `alter` command to set the `BULKLOAD` parameter to TRUE again.
- In the Control System, the **Will table be bulkload?** option cannot be modified after table creation.

Incremental Bulk Loads

Incremental bulk loads can add data to existing tables concurrently with other table operations, with better performance than put operations. This type of bulk load makes use of write-ahead log files.

Note: Tables are available for client operations, such as put, get, and scan operations, during incremental bulk loads.

You can use incremental bulk loads to ingest large amounts of data to an existing table. Tables remain available for standard client operations such as put, get, and scan while the bulk load is in process. A table can perform multiple incremental bulk load operations simultaneously.

Note: Whether you create a table with the `maprcli table create` command, with the hbase shell's `create` command, or in the Control System, incremental loads are supported by default.

Performing File System Operations on MapR Database Tables

The data-fabric file system stores tables in the same namespace as files. You can move and delete tables in much the same way as you can with files. All file system operations remain accessible with the `hadoop fs` command.

Volume properties, such as replication factor or rack topology, that apply to the specified location also apply to tables stored at that location. You can move a table with the Linux `mv` command or the `hadoop fs -mv` command.

When you use Direct Access NFS or the `hadoop fs -ls` command to access a MapR cluster, tables and files are listed together. Because the client's Linux commands are not table-aware, other Linux file manipulation commands, notably file read and write commands, are not available for MapR Database tables.

This section describes the operations that you can perform on MapR Database tables through a Linux command line when you access the cluster through NFS or with the `hadoop fs` commands.

Setting Permissions

MapR Database tables do not support setting user permissions through the UNIX `chmod` command or the `hadoop fs -chmod` analogue. Instead, MapR Database table access is controlled with Access Control Expressions (ACEs). See Enabling Table and Stream Authorizations with ACEs on page 1461.

Read and Write

You cannot perform read or write operations on a MapR Database table from a Linux file system context. For example, you cannot use the `cat` command to view the content of a table or search through a table with the `grep` command. MapR File System returns an error when an application attempts to read or write to a MapR Database table.

last-updated: Apr 11, 2022

Move

You can move a MapR Database table within a volume with the `mv` command over NFS or with the `hadoop fs -mv` command. These moves are subject to the standard permissions restrictions. Moves across volumes are not currently supported.

Remove

You can remove a table with the `rm` command over NFS or with the `hadoop fs -rm` command. These commands remove the table from the namespace and asynchronously reclaims the disk space. You can remove a directory that includes both files and tables with the `rm -r` or `hadoop fs -rmr` commands.

Note: To prevent users from deleting a particular table, you must ensure that users do not have WRITE permission on the folder in which the table is located. Permissions on the table itself are not used in evaluating whether a user can delete the table. This convention follows standard UNIX rules for file and directory permissions.

Copy and Recursive/Directory Copy

Table copying at the file-system level is not supported. See Migrating Between Apache HBase Tables and MapR Database Tables for information on copying tables using the HBase shell.

Managing Column Families and Columns

This section covers overviews of column families in binary tables and JSON tables, how to create column families, alter them, delete them, set permissions on them, and set and display parameter values.

For a conceptual overview of column families in binary tables, see Column Families in Binary Tables.

For a conceptual overview of column families in JSON tables, see Managing Column Families.

Creating Column Families

Explains how to create column families using either the Control System, the CLI, or the HBase shell.

There are several methods that you can use to create column families in MapR Database tables. To create column families, you must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path to the table
- `createrenamefamilyperm` on the table

Creating Column Families Using the Control System

To create a column family from the Control System, under **Data** > **Tables**:

Note: This option is not available on the Kubernetes version of the Control System.

1. Click:
   - **Take me to Add Column Family** after creating a new table.
   - **Add Column Family** in the **Column Families** tab in the table information page.

     See Viewing Table Information on page 1032.
2. Specify the following properties to set up column families.

JSON Table

Column Family Name	The name of the column family.

JSON Path	The path to the column family in dotted notation. For example, suppose the table contained JSON documents that were of this general structure: `{ "_id" : "ID", "a" : { "b" : { "c" : "value", }, "e" : "value" } }` If you want to create a column family at the field `d` nested within `b`, your new path would be `a.b.d`. **Note:** Ensure that the field at which you want to create the column family does not yet exist. If the field exists, it could become inaccessible after the column family is created.
Compression	The compression setting to use for the column family. Valid options are off, lzf, lz4, and zlib. The default setting is the same as the compression setting for the directory where the table is located. To find out whether a directory is compressed and the type of compression, see Turning Compression On or Off on Directories Using the CLI on page 990.

last-updated: Apr 11, 2022

Time-to-Live	Specifies whether to purge data when the age of the data in this column family exceeds the value specified here. Data can remain forever or can be purged after specified amount of time (in seconds). Setting the value to 0 is equivalent to allowing data to remain indefinitely or forever.  **Note:** If the value for an existing column family in a JSON table is not 0, you cannot add another column family.

In Memory	Determines whether preference is given to values of this column family for storage with row keys. Because row keys are cached in memory in preference to row data, column-family data that is stored inline with the row keys is also cached in memory. For all column families in a table together, up to 200 bytes of row data will be stored inline with each row key. Storing data inline with a row key might speed retrieval of the data from a column family because disk access can often be avoided. For each column family, up to 32 bytes can be stored inline with each row key even if this is disabled (No), but preference will be given to column families where this is enabled (Yes). A column family can have more than 32 bytes stored inline if this is enabled. If the total number of bytes for all column families together exceeds 200 for a row, then preference for inclusion within the inline storage for that row is given to column families that have this enabled.  **Note:** All of the data for a column family will be stored in-line with the row key, or none will be. If the contents in a column family for a particular row are larger than the maximum number of bytes that are allowed to be stored for that column family, no data will be stored in-line for that column family. By default, this is enabled.

last-updated: Apr 11, 2022

Binary Table

Column Family Name	The name of the column family.
Version	• Minimum — The minimum number of versions of column values to keep. The default is zero. • Maximum — Maximum number of versions of column values to keep. The default is one.
Compression	The compression setting to use for the column family. Valid options are off, lzf, lz4, and zlib. The default setting is the same as the compression setting for the directory where the table is located. To find out whether a directory is compressed and the type of compression, see Turning Compression On or Off on Directories Using the CLI on page 990.
Time-to-Live	Specifies whether to purge data when the age of the data in this column family exceeds the value specified here. Data can remain forever or can be purged after specified amount of time (in seconds). Setting the value to 0 is equivalent to allowing data to remain indefinitely or forever.

In Memory	Determines whether preference is given to values of this column family for storage with row keys. Because row keys are cached in memory in preference to row data, column-family data that is stored inline with the row keys is also cached in memory. For all column families in a table together, up to 200 bytes of row data will be stored inline with each row key. Storing data inline with a row key might speed retrieval of the data from a column family because disk access can often be avoided. For each column family, up to 32 bytes can be stored inline with each row key even if this is disabled (No), but preference will be given to column families where this is enabled (Yes). A column family can have more than 32 bytes stored inline if this is enabled. If the total number of bytes for all column families together exceeds 200 for a row, then preference for inclusion within the inline storage for that row is given to column families that have this enabled.  **Note:** All of the data for a column family will be stored in-line with the row key, or none will be. If the contents in a column family for a particular row are larger than the maximum number of bytes that are allowed to be stored for that column family, no data will be stored in-line for that column family. By default, this is enabled.

last-updated: Apr 11, 2022

3. Set up access to column families for users, groups, and/or roles.
   You can use either the default permissions or proceed to define new permissions for this column family.

JSON Table

Read Data	Can do column reads. Reads require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Write Data	Can do column writes. Writes require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Traverse Data	Can pass over fields in JSON documents. For example, suppose that a JSON table contains documents of this general structure: `{ "_id" : "ID", "a" : { "b" : "value", "c" : "value" } }` Suppose further that the user sjohnson has read permission on `a.b`, but not on `a`. For sjohnson to read `a.b`, the user needs the traverse permission on `a`. The user can then pass over field `a` to `a.b`. This permission is inherited by fields within the column family.
Set Version	Can set or change the maximum and minimum number of versions of column values to keep.
Set Compression	Can set or change the compression setting for the column family.

Binary Table

Read Data	Can do column reads. Reads require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Write Data	Can do column writes. Writes require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Append Data	Can do column appends. Column appends require permission both at the column-family level and at the column level.
Set Version	Can set or change the maximum and minimum number of versions of column values to keep.
Set Compression	Can set or change the compression setting for the column family.

By default, all permissions are given to the user creating the table.

To grant or block access to users, groups, and/or roles, from the:

- Basic settings, select the type — public, (OR) user, group, or role — from the drop-down menu, specify the name of the user, group, or role, and select one or more checkbox to grant permissions.

  Tip: Click to create a copy of the associated access control setting. Click to remove the associated access control expression.

  To add *ACE*s for another user, group, or role, click **Add Another** and repeat this step.

- Advanced settings, specify public (p) or user (u), group (g), and/or role (`r`) who have or do not have the type of access using the following boolean expressions and subexpressions:
  - `!` — Negation operator.
  - `&` — AND operation.
  - `|` — OR operation.

  Use `( )`, parentheses, for subexpressions.

  Note: You *cannot* specify user, group, or role individually if access is granted to all users (public).

  Alternatively, click associated with the type of access to use the **Access Control Expression** window to define access for public or users, group, and/or role. See Defining ACEs on page 1473 for more information.

last-updated: Apr 11, 2022

Note: If you switch from **Basic** to **Advanced**, the basic settings, if any, are carried over to the advanced settings. If you switch from **Advanced** to **Basic**, all the settings are lost because the subexpressions and AND (`&`) and negation (`!`) operations that are supported by advanced settings are not supported in the basic settings.

4. Specify:

Field Permissions (for JSON Tables)

Specify a name for the field and the permissions to access the field. By default, a field inherits permissions from the column in which the field is located. Permissions set at this level override permissions inherited from the column. You can set the following permissions by selecting the associated checkbox:

Read Data	Can read from the field. This permission extends to fields that are nested below as well unless explicitly denied on any of the nested fields.
Write Data	Can delete the field, insert a value into the field, or overwrite the field's value. **Note:** Deleting a field also deletes all fields that are nested within that field, even those fields on which the write permission is explicitly denied.
JSON Traverse	Can descend a hierarchy of fields to access the fields to read or write.

By default, all permissions are given to the user creating the table. See Permission Types for Fields and Column Families in JSON Tables on page 1462 for more information.

Column Permission (for Binary Tables)

Create (by clicking **Add Column** and specifying a name in the **Column Name** field) and set permissions for columns in the column family. You can set the following permissions by selecting the associated checkbox:

Read Data	Can do column reads. Reads require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.

last-updated: Apr 11, 2022

Write Data	Can do column writes. Writes require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Append Data	Can do column appends. Column appends require permission both at the column-family level and at the column level.

Note: When a user, group, or role requests to read data from, write data to, or append data to a column, MapR Database checks whether that user, group, or role has read or write permission for the column family AND read or write permission for the column. By default, columns allow read and write access to all users; in such cases, only the read or write permission for the column family matters.

However, suppose that a table contains columns `col1` and `col2` in column family `cf1`, and these columns grant read and write permission only to the table creator. A different user tries to write data to these columns. MapR Database checks whether this user has write permission on `cf1` AND `col1` AND `col2`. If the user does not have all three permissions, MapR Database returns an error that says access for the write is denied.

If this user were to try to read from the same two columns, MapR Database would simply not return the data. If the user tried to read from those two columns and additional columns on which the user had read permissions, the results would contain the data for those additional columns, but exclude the data for `col1` and `col2`.

You can add columns to a table at any time. Null columns for a given row don't take up any storage space.

Note: Extremely wide tables with very large numbers of columns can sometimes reach the recommended size for a table split at a comparatively small number of rows because MapR Database tables split at the row level, not the column level.

To grant or block access to users, groups, and/or roles, from the:

- Basic settings, select the type — public, (OR) user, group, or role — from the drop-down menu, specify the name of the user, group, or role, and select one or more checkbox to grant permissions.

last-updated: Apr 11, 2022

Tip: Click to create a copy of the associated access control setting. Click to remove the associated access control expression.

To add *ACE*s for another user, group, or role, click **Add Another** and repeat this step.

- Advanced settings, specify public (p) or user (u), group (g), and/or role (r) who have or do not have the type of access using the following boolean expressions and subexpressions:
  - ! — Negation operator.
  - & — AND operation.
  - | — OR operation.

  Use ( ), parentheses, for subexpressions.

  Note: You *cannot* specify user, group, or role individually if access is granted to all users (public).

  Alternatively, click associated with the type of access to use the **Access Control Expression** window to define access for public or users, group, and/or role. See Defining ACEs on page 1473 for more information.

Note: If you switch from **Basic** to **Advanced**, the basic settings, if any, are carried over to the advanced settings. If you switch from **Advanced** to **Basic**, all the settings are lost because the subexpressions and AND (&) and negation (!) operations that are supported by advanced settings are not supported in the basic settings.

5. Click **Add Column Family** to add the column family to the table.

Creating Column Families Using CLI or the REST API

JSON Table

To create a column family in a JSON table, include the parameters `-jsonpath` and `-force` :

```
maprcli table cf
create -path <path> -cfname
<name_of_column_family> -jsonpath

<path> -force true
```

For the full list of options for this command, see `table cf create` on page 1799.

The `-jsonpath` parameter specifies the path to the column family. The path is in dotted notation. For example, suppose the table contained JSON documents that were of this general structure:

```
{
     "_id" : "ID",
     "a" :
          {
               "b" :
                    {
                         "c" :
"value",
                    },
               "e" : "value"
```

```
            }
}
```

You want to create a column family at the field `d` in the new path `a.b.d` because you plan to store image files in fields in that column family.

Important: Ensure that the field at which you want to create the column family does not yet exist. Also ensure that there are no secondary indexes defined on the field. If the field does exist or is a field in an index, the data in the field could become inaccessible after you create the column family.

By default, every time you try to create a non-default column family in a JSON table, this command fails and returns a warning message that you should ensure there is no existing data at the specified path. Set the `-force` parameter to `true` if you want to override this warning mechanism and create a column family.

Binary Table

The command to create a column family for a binary table is:

```
maprcli table cf create -path
<path> -cfname <name_of_column_family>
```

For the full list of options for this command, see `table cf create` on page 1799.

The format of the value of the `-path` parameter depends on whether you are creating a table on a local cluster or a remote cluster.

Creating a Column Family for a Binary Table Using HBase Shell

After starting the HBase shell, run the `alter` command. Type `help` to see a list of commands and their syntax.

Listing Column Families

Explains how to view the column families for a table using either the Control System or the CLI.

Viewing Table Column Families Using the Control System

To view the column families for a table:

1. Go to the table information page.
   See Viewing Table Information on page 1032.
2. Click **Column Families**.
   The page displays the default permissions for the column families in the **Default Column Family Authorization** pane and for each column family, the **All** pane displays:

Column Name	Column Description
Column Family Name	The name of the column family.
JSON Path	The JSON path for the column in the JSON file in dotted notation.
Compression	The compression scheme used for the column family.
Time-to-Live	The amount of time to keep the data in the column family.
In Memory	Whether or not this column value resides in memory.

last-updated: Apr 11, 2022

Selecting the checkbox associated with the column family makes the **Remove Column Family** button available. You can:

- Add a column family to the table
- Remove a column family associated with the table

Viewing Table Column Families Using the CLI or REST API

The command to list the column families that are in a table is:

```
maprcli table cf list -path <path> -cfname <name_of_column_family>
```

The format of the value of the `-path` parameter depends on whether you are viewing a table on a local cluster or a remote cluster:

- For a path on the local cluster, start the path at the volume mount point. For example, for a table named `test` under a volume with a mount point at /volume1, specify the following path: `/volume1/test`
- For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customer` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`

Note: You cannot use the following characters in the table name:

```
< > ? % \
```

To use the following characters in the table name, enclose them either in single or double quotes:

```
; | ( ) /
```

For example:

```
maprcli table create -path "/♠=#;{}&()/"  (or)
maprcli table create -path '/♠=#;{}&()/'
```

To use either the **'** or the **"** character in the table name, enclose:

- the **'** character within double quotes (")
- the **"** character within single quote (')

For example:

```
maprcli table create -path "/'♠=#;{}&()/"  (or)
maprcli table create -path '/"♠=#;{}&()/'
```

To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the paths
- `adminaccessperm` on the table

For complete reference, see `table cf list` on page 1810.

Removing Column Families

Explains how to delete column families using either the Control System or the CLI.

Important: Starting in the 6.0 release, you cannot delete a column family from a JSON table.

Removing Column Families Using the Control System

To remove one or more column families:

1. Go to the table information page.
   See Viewing Table Information on page 1032.
2. Click **Column Families** tab.
   The page displays:
   - Default column family permissions
   - All the column families for the table
3. Select the checkbox associated with the column families to delete in the **All** pane.
4. Click **Remove Column Family**.
   The **Remove Column Families** confirmation dialog displays.
5. Verify the list of column families and click **Remove Column Family**.
   If necessary, click ✕ to remove a column family from the list of column families to delete.

Removing a Column Family Using the CLI or the REST API

To remove a column family by name using the CLI or the REST API, run the following command:

```
maprcli table cf delete -path <path> -cfname <name>
```

See `table cf delete` on page 1805 for more information.

Altering Column Families

Explains how to modify the permissions and properties of column families using either the Control System, the CLI, or the HBase shell.

There are several methods that you can use to edit column families in MapR Database tables. These methods also let you change permissions on column families.

Modifying a Column Family Using the Control System

1. Go to the table information page.
   See Viewing Table Information on page 1032.
2. Click **Column Families** tab.
   The page displays:
   - Default column family permissions
   - All the column families for the table
3. Click the name of the column family to modify.
   The **Edit Column Family** page displays.
4. Make changes to the following properties (under **PROPERTIES**) as desired.

JSON Table

Column Family Name	The name of the column family.

last-updated: Apr 11, 2022

Compression	The compression setting to use for the column family. Valid options are off, lzf, lz4, and zlib. The default setting is the same as the compression setting for the directory where the table is located. To find out whether a directory is compressed and the type of compression, see Turning Compression On or Off on Directories Using the CLI on page 990.

In Memory	Determines whether preference is given to values of this column family for storage with row keys. Because row keys are cached in memory in preference to row data, column-family data that is stored inline with the row keys is also cached in memory. For all column families in a table together, up to 200 bytes of row data will be stored inline with each row key. Storing data inline with a row key might speed retrieval of the data from a column family because disk access can often be avoided. For each column family, up to 32 bytes can be stored inline with each row key even if this is disabled (No), but preference will be given to column families where this is enabled (Yes). A column family can have more than 32 bytes stored inline if this is enabled. If the total number of bytes for all column families together exceeds 200 for a row, then preference for inclusion within the inline storage for that row is given to column families that have this enabled.  **Note:** All of the data for a column family will be stored in-line with the row key, or none will be. If the contents in a column family for a particular row are larger than the maximum number of bytes that are allowed to be stored for that column family, no data will be stored in-line for that column family. By default, this is enabled.

last-updated: Apr 11, 2022

Binary Table

Column Family Name	The name of the column family.
Version	• Minimum — The minimum number of versions of column values to keep. The default is zero. • Maximum — Maximum number of versions of column values to keep. The default is one.
Compression	The compression setting to use for the column family. Valid options are off, lzf, lz4, and zlib. The default setting is the same as the compression setting for the directory where the table is located. To find out whether a directory is compressed and the type of compression, see Turning Compression On or Off on Directories Using the CLI on page 990.
Time-to-Live	Specifies whether to purge data when the age of the data in this column family exceeds the value specified here. Data can remain forever or can be purged after specified amount of time (in seconds). Setting the value to 0 is equivalent to allowing data to remain indefinitely or forever.

In Memory	Determines whether preference is given to values of this column family for storage with row keys. Because row keys are cached in memory in preference to row data, column-family data that is stored inline with the row keys is also cached in memory. For all column families in a table together, up to 200 bytes of row data will be stored inline with each row key. Storing data inline with a row key might speed retrieval of the data from a column family because disk access can often be avoided. For each column family, up to 32 bytes can be stored inline with each row key even if this is disabled (No), but preference will be given to column families where this is enabled (Yes). A column family can have more than 32 bytes stored inline if this is enabled. If the total number of bytes for all column families together exceeds 200 for a row, then preference for inclusion within the inline storage for that row is given to column families that have this enabled.  **Note:** All of the data for a column family will be stored in-line with the row key, or none will be. If the contents in a column family for a particular row are larger than the maximum number of bytes that are allowed to be stored for that column family, no data will be stored in-line for that column family. By default, this is enabled.

last-updated: Apr 11, 2022

5. Make changes to permissions as desired.
   a) Modify USER ACCESS CONTROLs for the column family as desired.

JSON Table

Modify or set the following permissions for the column family:

Read Data	Can do column reads. Reads require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Write Data	Can do column writes. Writes require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Traverse Data	Can pass over fields in JSON documents. For example, suppose that a JSON table contains documents of this general structure: `{ "_id" : "ID", "a" : { "b" : "value", "c" : "value" } }` Suppose further that the user sjohnson has read permission on a.b, but not on a. For sjohnson to read a.b, the user needs the traverse permission on a. The user can then pass over field a to a.b. This permission is inherited by fields within the column family.
Set Compression	Can set or change the compression setting for the column family.

Binary Table

Modify or set the following permissions for the column family:

last-updated: Apr 11, 2022

Read Data	Can do column reads. Reads require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Write Data	Can do column writes. Writes require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Append Data	Can do column appends. Column appends require permission both at the column-family level and at the column level.
Set Version	Can set or change the maximum and minimum number of versions of column values to keep.
Set Compression	Can set or change the compression setting for the column family.

b) Modify or set:

Field Permission (for JSON Tables)

Modify permissions for the following on existing fields and/or create new fields and set permissions for the following on the new fields:

Read Data	Can read from the field. This permission extends to fields that are nested below as well unless explicitly denied on any of the nested fields.
Write Data	Can delete the field, insert a value into the field, or overwrite the field's value.  **Note:** Deleting a field also deletes all fields that are nested within that field, even those fields on which the write permission is explicitly denied.

last-updated: Apr 11, 2022

JSON Traverse	Can descend a hierarchy of fields to access the fields to read or write.

Column Permission (for Binary Tables)

Modify following permissions for existing columns and/or create new columns and set permissions for the following:

Read Data	Can do column reads. Reads require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Write Data	Can do column writes. Writes require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Append Data	Can do column appends. Column appends require permission both at the column-family level and at the column level.

To grant or block access to users, groups, and/or roles, from the:

- Basic settings, select the type — public, (OR) user, group, or role — from the drop-down menu, specify the name of the user, group, or role, and select one or more checkbox to grant permissions.

  Tip: Click to create a copy of the associated access control setting. Click to remove the associated access control expression.

  To add *ACE*s for another user, group, or role, click **Add Another** and repeat this step.

- Advanced settings, specify public (p) or user (u), group (g), and/or role (r) who have or do not have the type of access using the following boolean expressions and subexpressions:
  - ! — Negation operator.
  - & — AND operation.
  - | — OR operation.

  Use ( ), parentheses, for subexpressions.

  Note: You *cannot* specify user, group, or role individually if access is granted to all users (public).

  Alternatively, click associated with the type of access to use the **Access Control Expression** window to define access for public or users, group, and/or role. See Defining ACEs on page 1473 for more information.

Note: If you switch from **Basic** to **Advanced**, the basic settings, if any, are carried over to the advanced settings. If you switch from **Advanced** to **Basic**, all the settings are lost because the subexpressions and AND (&) and negation (!) operations that are supported by advanced settings are not supported in the basic settings.

6. Click **Save Changes** for the changes to take effect.

Modifying a Column Family Using the CLI or REST API

The basic command to edit a column family is:

```
maprcli table cf edit -path <path> -cfname <name_of_column_family> options
```

For the full list of options for this command, see `table cf edit` on page 1806.

The format of the value of the `-path` parameter depends on whether you are creating a table on a local cluster or a remote cluster:

- For a path on the local cluster, start the path at the volume mount point. For example, for a table named `test` under a volume with a mount point at /volume1, specify the following path: `/volume1/test`
- For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customer` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`

  Note: You cannot use the following characters in the table name:

  ```
  < > ? % \
  ```

  To use the following characters in the table name, enclose them either in single or double quotes:

  ```
  ; | ( ) /
  ```

  For example:

  ```
  maprcli table create -path "/♦=#;{}&()/"  (or)
  maprcli table create -path '/♦=#;{}&()/'
  ```

  To use either the **'** or the **"** character in the table name, enclose:

  - the **'** character within double quotes (")
  - the **"** character within single quote (')

  For example:

  ```
  maprcli table create -path "/'♦=#;{}&()/"  (or)
  maprcli table create -path '/"♦=#;{}&()/'
  ```

Modifying a Column Family in a Binary Table Using HBase shell

After starting the HBase shell, run the `alter` command. Type `help` to see a list of commands and their syntax.

Displaying Default Column Family Permissions

Use either the Control System or the `maprcli` command to find out the users, groups, or roles that have permissions on the default column family.

last-updated: Apr 11, 2022

Viewing Default Column Family Permissions in the Control System

- Log in to the Control System and go to the **Column Families** tab in the table information page.

  The **Default Column Family Authorization** pane displays the following permissions for users, groups, and roles.

Binary Table

Read Data	Can do column reads. Reads require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Write Data	Can do column writes. Writes require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Append Data	Can do column appends. Column appends require permission both at the column-family level and at the column level.
Set Version	Can set or change the maximum and minimum number of versions of column values to keep.
Set Compression	Can set or change the compression setting for the column family.

JSON Table

Read Data	Can do column reads. Reads require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.
Write Data	Can do column writes. Writes require permission both at the column-family level and at the field level. This permission is inherited by fields within the column family.

last-updated: Apr 11, 2022

Traverse Data	Can pass over fields in JSON documents. For example, suppose that a JSON table contains documents of this general structure: `{ "_id" : "ID", "a" : { "b" : "value", "c" : "value" } }` Suppose further that the user sjohnson has read permission on `a.b`, but not on `a`. For sjohnson to read `a.b`, the user needs the traverse permission on `a`. The user can then pass over field `a` to `a.b`. This permission is inherited by fields within the column family.
Set Version	Can set or change the maximum and minimum number of versions of column values to keep.
Set Compression	Can set or change the compression setting for the column family.

Retrieving the Default Column Family Permissions Using the CLI

To display the permissions on a column family, run this command:

```
maprcli table cf colperm get -path <path> -cfname <name of column
family> -json
```

To display the permissions on a column, add the `-name` parameter:

```
maprcli table cf colperm get -path <path> -cfname <name of column
family> -name
            <name of column> -json
```

The format of the value of the `-path` parameter depends on whether you are viewing a table on a local cluster or a remote cluster.

The `json` parameter displays the output as a JSON document.

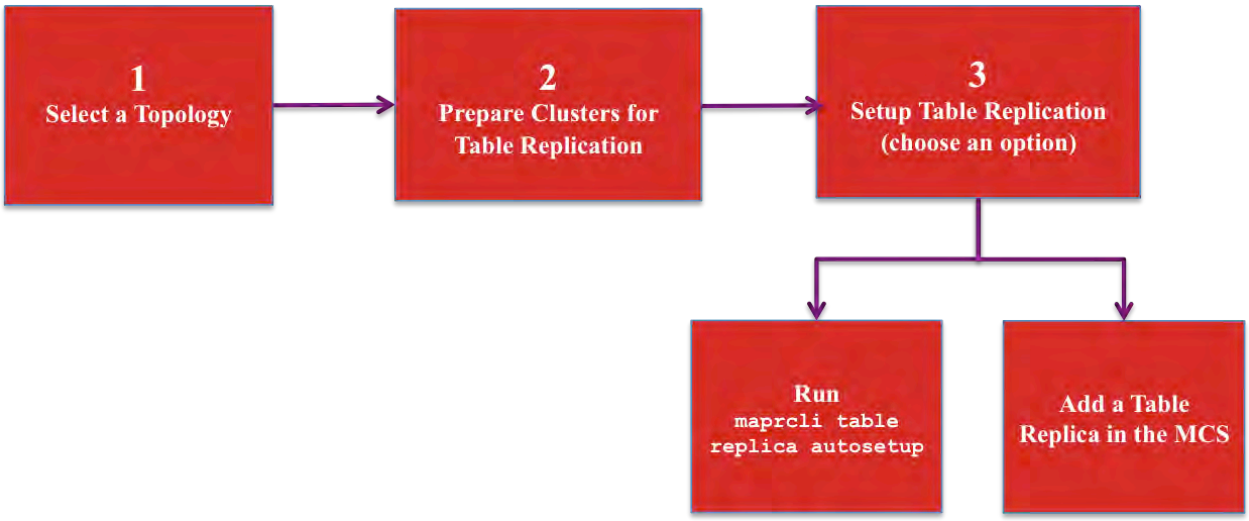
Managing Table Replication

This section contains topics about setting up table replication and administering existing replicas.

last-updated: Apr 11, 2022

The process to set up table replication consists of the following steps:

1. Select a Topology
2. Prepare Clusters for Table Replication.
3. Set up Table Replication using one of the following options:
   - Run maprcli table replica autosetup.
   - Add a Table Replica in the Control System.



1. Select a Table Topology
2. Prepare Clusters for Table Replication
3. Run maprcli table replica autosetup
4. Add a Table Replica in the MCS

Note: After setting up replication, replicas can be administered using either the Control System or the CLI.

Preparing Clusters for Table Replication
Preparing clusters for table replication includes configuring gateways on destination clusters, configuring the `mapr-clusters.conf` file on the source cluster, and, if the clusters are secure, setting up secure communications between the clusters. After you prepare the clusters for table replication, you can setup replication between tables.

Before You Begin

The following topics identify concepts and tasks that you need to do before setting up your environment for table replication.

- Plan which replication topology you want to use. For information about the various topologies, see Supported replication topologies on page 611.

- In general, if you are replicating tables, you should store them in their own volumes to avoid overlap with volume mirroring. Otherwise, if a source volume fails, you may have a scenario where a table in a promoted mirror lags behind the table's replica.

  For example, suppose `/vol` mirrors to `/vol.mirror` and contains a table `srcTab` that replicates to `/replVol/replTab`. If `/vol` fails, `/vol.mirror/srcTab` may lag `/replVol/replTab` when `/vol.mirror` is promoted.

  To avoid this problem, starting with the 6.1 release, after MapR Database promotes a mirror volume, replication terminates with `REPLICA_STATE_UNEXPECTED` for any tables in that volume.

  The following sample output shows this behavior:

```
[mapr]# /opt/mapr/bin/maprcli table replica list -path /vol.mirror/
srcTab -refreshnow true -json
{
  "timestamp":1534805233244,
  "timeofday":"2018-08-20 03:47:13.244 GMT-0700 PM",
  "status":"OK",
  "total":1,
  "data":[
    {
      "cluster":"mirrorSrc",
      "table":"/replVol/replTab",
      "type":"MapRDB",
      "replicaPath":"/replVol/replTab",
      "replicaState":"REPLICA_STATE_UNEXPECTED",
      "paused":false,
      "throttle":false,
      "idx":1,
      "networkencryption":false,
      "synchronous":false,
      "networkcompression":"lz4",
      "propagateExistingData":false,
      "isUptodate":true,
      "minPendingTS":0,
      "maxPendingTS":0,
      "bytesPending":0,
      "putsPending":0,
      "bucketsPending":0,
      "uuid":"8b4563e1-884d-7852-f257-078c397b5b00",
      "copyTableCompletionPercentage":0,
      "errors":{
        "Code":"ErrReplicaTableUpstreamMismatch",
        "Host":"10.10.104.35",
        "Msg":"OpenStream: Upstream table does not match original
Upstream cluster mirrorSrc table /replVol/replTab"
      }
    }
    ]
}
```

  This change in behavior applies to only tables that have replication enabled starting in 6.1. See Table Replication States on page 625 for more details.

- Ensure that your user ID has the `readAce` permission on the volume where the source tables are located and the `writeAce` permission on the volumes where the replicas are located. For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

- Ensure that you have administrative authority on the clusters that you plan to use.

- If you upgraded your source cluster from a previous version of MapR, enable table replication by running this `maprcli` command: `maprcli cluster feature enable -name mfs.feature.db.repl.support`
- Depending on your use case, replication requires the installation of gateways and may also require the HBase client. For more information about installation requirements, see Service Layout Guidelines for Replication on page 116

Setting Up Table Replication

The following steps show how to set up your environment for table replication including setup for secure clusters.

1. In the `mapr-clusters.conf` file on every node in your source cluster, add an entry that lists the CLDB nodes that are in the destination cluster. This step is required so that the source cluster can communicate directly with the destination cluster's CLDB nodes. See mapr-clusters.conf for the format to use for the entries.
2. On the destination cluster, configure gateways through which the source cluster sends updates to the destination cluster. See Configuring Gateways for Table and Stream Replication on page 1152.
3. If your clusters are secure, configure each cluster so that you can access them all from one cluster. This way, you need not log into each secure cluster separately and run `maprcli` commands locally on them. See Configuring Secure Clusters for Running Commands Remotely on page 1484 for more information.
4. If your clusters are secure, add a cross-cluster ticket to the source cluster, so that it can replicate data to the destination cluster. See Configuring Secure Clusters for Cross-Cluster Mirroring and Replication on page 1486.
5. **Optional:** If your clusters are secure, configure your source cluster so that you can use the Control System to set up and administer table replication from the source to the destination cluster.

   These steps make it convenient to use the Control System for setting up and managing replication involving two secure clusters. However, before following them, perform these prerequisite tasks.

   Note:

   - Ensure that both clusters are managed by the same team or group. The UIDs and GIDs of the users that are able to log in to the Control System on the source cluster must exactly match their UIDs and GIDs on the destination cluster. This restriction applies only to access to both clusters through the Control System, and does not apply to access to both clusters through the maprcli. If the clusters are managed by different teams or groups, use the maprcli instead of the Control System to set up and manage table replication involving two secure clusters.
   - Ensure that the proper file-system and table permissions are in place on both clusters. Otherwise, any user who can log into the Control System and has the same UID or GID on the destination cluster will be able to set up replication either from the source cluster to the destination cluster or vice versa. A user could create one or more tables on the destination cluster, enable replication to them from the source cluster, load the new tables with data from the source cluster, and start replication. A user could also create tables on the source cluster, enable replication to them from tables in the destination cluster, load the new tables with data from the destination cluster, and start replication.

a. On the source cluster, generate a service ticket by using the `maprlogin` command:

```
maprlogin generateticket -type service -cluster <destination cluster>
-user mapr -duration <duration> -out <output folder>
```

Where `-duration` is the length of time before the ticket expires. You can specify the value in either of these formats:

- `[Days:]Hours:Minutes`
- `Seconds`

b. To every node of the source cluster, add the service ticket to the file `/opt/mapr/conf/maprusertickets` file that was created when you secured the source cluster:

```
at <path and filename of the service ticket> >> /opt/mapr/conf/
maprusertickets
```

c. Restart the web server by running the `maprcli node services` command. For the syntax of this command, see node services on page 1730.

d. Add the following two properties to the `core-site.xml` file. For Hadoop 2.7.0, edit the file `/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/core-site.xml`.

```
<property>
      <name>hadoop.proxyuser.mapr.hosts</name>
      <value>*</value>
 </property>
 <property>
      <name>hadoop.proxyuser.mapr.groups</name>
      <value>*</value>
 </property>
```

Setting Up Table Replication Using the CLI

You can run the `maprcli table replica autosetup` command to set up primary-secondary or multi-master replication from an existing source table.

Note: This procedure describes how to use the `maprcli` to automatically set up table replication. As an alternative, you can use the Control System to automatically setup table replication or use the `maprcli` command to manually setup primary-secondary replication or manually setup multi-master replication.

Before you begin, complete the following steps:

- Verify that you have completed the steps to configure the clusters for table replication. For more information, see Preparing Clusters for Table Replication on page 1066.
- On the source table, run the `maprcli table info` command to verify that you have the following permissions:
  - `readperm`, which is required for reading from the table.
  - `replperm`, which is required for replicating from the table.

  On the destination table (if it already exists), run the `maprcli table info` command to verify that you have the following permissions:
  - `bulkload`, which is required for the initial copy of source data into the destination table.

- `replperm`, which is required for receiving replicated updates from the source table.

All updates from a source table arrive at a replica after having been authenticated at a gateway. Therefore, access control expressions on the replica that control permissions for updates to column families and columns are irrelevant; gateways have implicit authority to update replicas.

Note: If you are setting up a primary-secondary replication loop for *n* clusters, repeat these steps for *n-1* primary-secondary relationships to set up basic primary-secondary topologies.

1. Log into both the source and destination clusters.
2. Run the `maprcli table replica autosetup` command.
   - For primary-secondary replication:

     ```
     maprcli table replica autosetup -path /mapr/<source cluster>/<path to table> -replica /mapr/<destination cluster>/<path to table>
     ```

   - For multi-master replication:

     ```
     maprcli table replica autosetup -path /mapr/<source cluster>/<path to table> -replica /mapr/<destination cluster>/<path to table> -multimaster true
     ```

     Note:

     The parameter `-multimaster` is an optional parameter that you use to set up multi-master replication.

   For example, to set up multi-master replication between the `customers` table in the `sanfrancisco` cluster and a new `customers` table in the `newyork` cluster, you could use this command:

   ```
   maprcli table replica autosetup -path /mapr/sanfrancisco/
   customers -replica /mapr/newyork/customers -multimaster true
   ```

   To set up primary-secondary replication between the `customersA` table in the `sanfrancisco` cluster and a new `customersB` table in the same cluster, you could use this command:

   ```
   maprcli table replica autosetup -path /mapr/sanfrancisco/
   customersA -replica /mapr/sanfrancisco/customersB
   ```

   Note: For information about additional parameters that you can configure, see table replica autosetup on page 1854.
3. To check the replication status, run table replica list on page 1863.

Additional Information:

- With multi-master replication, if one of the tables goes offline, you can direct client applications to the other table. For more information, see Multi-Master Replication on page 616.
- Be aware that changes to the structure of a source table are not replicated automatically to replicas. For more information, see Adding a Column Family to a Replica on page 1086
- Check the Control System for alarms related to replication and whether synchronous replication is switched temporarily to asynchronous replication. See Table-Replication Alarms.

last-updated: Apr 11, 2022

Setting Up Primary-Secondary Replication Manually
You can run `maprcli` commands to set up primary-secondary replication manually.

1. Ensure that you have followed these prerequisite steps:
   - Verify that you have complete the steps to configure the clusters for table replication. For more information, see Preparing Clusters for Table Replication on page 1066.
   - Run the `maprcli table info` command on the source table to verify that you have the following permissions:
     - `readperm`, which is required for reading from the table.
     - `replperm`, which is required for replicating from the table.
   - Run the `maprcli table info` command on the destination table (if it already exists) to verify that you have the following permissions:
     - `bulkload`, which is required for the initial copy of source data into the destination table.
     - `replperm`, which is required for receiving replicated updates from the source table.
2. Create the replica manually with the `maprcli` command `table create`. Use the `-copymetafrom` option to ensure that the metadata for the replica is identical to the metadata for the source table. Metadata includes column families, access control expressions (ACEs), and other attributes.

   ```
   maprcli table create -path <path to the replica> -copymetafrom <path to the source table>
   ```

   For example, to create the replica `customers` in the `newyork` cluster and use the metadata from the source table in the `sanfrancisco` cluster, you could use this command:

   ```
   maprcli table create -path /mapr/newyork/customers -copymetafrom /mapr/sanfrancisco/customers
   ```

3. Register the replica as a replica of the source table by running the `maprcli table replica add` command.

   ```
   maprcli table replica add -path <path to the source table> -replica <path to the replica> -paused true
   ```

   For example, to register the `customers` table in the `newyork` cluster as a replica of the `customers` table in the `sanfrancisco` cluster, you could use this command:

   ```
   maprcli table replica add -path /mapr/sanfrancisco/customers -replica /mapr/newyork/customers -paused true
   ```

   The `-paused` parameter ensures that replication does not start immediately after you register the source table as a source for this replica. You do this registration in step 4.

   Note: For more information about additional parameters that you can configure, see table replica add on page 1851.

last-updated: Apr 11, 2022

4. Verify that you specified the correct replica by running the `maprcli table replica list` command.

   ```
   maprcli table replica list -path <path to the source table>
   ```

   To verify that the customers table in the `newyork` cluster is a replica of the `customers` table in the `sanfrancisco` cluster, you could look at the output of this command:

   ```
   maprcli table replica list -path /mapr/sanfrancisco/customers
   ```

5. Authorize replication between the tables by defining the source table as the upstream table for the replica by running the `maprcli table upstream add` command. Definition of the upstream table ensures that a table cannot replicate updates to any replica. Replication depends on a two-way agreement between the owners of the two tables.

   ```
   maprcli table upstream add -path <path to the replica> -upstream <path
   to the source table>
   ```

   To add the `customers` table in the `sanfrancisco` cluster as an upstream source for the `customers` table in the `newyork` cluster:

   ```
   maprcli table upstream add -path /mapr/newyork/customers -upstream /mapr/
   sanfrancisco/customers
   ```

6. Verify that you specified the correct source table by running the `maprcli table upstream list` command.

   ```
   maprcli table upstream list -path <path to the replica>
   ```

   To verify this in our example scenario, you could use this command:

   ```
   maprcli table upstream list -path /mapr/newyork/customers
   ```

7. If you set `-paused` to `true` when adding the replica, follow these steps:
   a) Load the replica with data from the source table by using the MapR Database CopyTable utility for binary tables or the MapR Database JSON CopyTable on page 5312 utility for JSON tables.

   b) Start replication with the command `maprcli table replica resume`. Here is the `maprcli` command:

      ```
      maprcli table replica resume -path <path to the source
      table> -replica <path to the replica>
      ```

      For our example scenario, you could use this command:

      ```
      maprcli table replica resume -path mapr/sanfrancisco/
      customers -replica /mapr/newyork/customers
      ```

- Be aware that changes to the structure of a source table are not replicated automatically to replicas. For more information, see Adding a Column Family to a Replica on page 1086
- You can check for alarms related to replication and whether synchronous replication is switched temporarily to asynchronous replication by looking in the Control System. See Table-Replication Alarms.

Setting Up Multi-Master Replication Manually

You can run `maprcli` commands to set up multi-master replication, first establishing replication in one direction, and then establishing it in the other direction.

- Verify that you have complete the steps to configure the clusters for table replication. For more information, see Preparing Clusters for Table Replication on page 1066.
- Run the `maprcli table info` command on the source table to verify that you have the following permissions:
  - `readperm`, which is required for reading from the table.
  - `replperm`, which is required for replicating from the table.
- Run the `maprcli table info` command on the destination table (if it already exists) to verify that you have the following permissions:
  - `bulkload`, which is required for the initial copy of source data into the destination table.
  - `replperm`, which is required for receiving replicated updates from the source table.

1. For manual setup in one direction, follow these steps:

a) Create the replica manually with the `maprcli table create command`. Use the `-copymetafrom` option to ensure that the metadata for the replica is identical to the metadata for the source table. Metadata includes column families, access control expressions (ACEs), and other attributes.

```
maprcli table create -path <path to the replica> -copymetafrom <path
to the source table>
```

For example, to create the replica `customers` in the `newyork` cluster and use the metadata from the source table in the `sanfrancisco` cluster, you could use this command:

```
maprcli table create -path /mapr/newyork/customers -copymetafrom /
mapr/sanfrancisco/customers
```

b) Register the replica as a replica of the source table by running the `maprcli table replica add` command.

```
maprcli table replica add -path <path to the source table> -replica
<path to the replica> -paused true
```

For example, to register the `customers` table in the `newyork` cluster as a replica of the `customers` table in the `sanfrancisco` cluster, you could use this command:

```
maprcli table replica add -path /mapr/sanfrancisco/
customers -replica /mapr/newyork/customers -paused true
```

The `-paused` parameter ensures that replication does not start immediately after you register the source table as a source for this replica. You do this registration in step d.

last-updated: Apr 11, 2022

c) Verify that you specified the correct replica by running the `maprcli table replica list` command.

```
maprcli table replica list -path <path to the source table>
```

To verify that the customers table in the `newyork` cluster is a replica of the `customers` table in the `sanfrancisco` cluster, you could look at the output of this command:

```
maprcli table replica list -path /mapr/sanfrancisco/customers
```

d) Authorize replication between the tables by registering the source table as the upstream table for the replica by running the `maprcli table upstream add` command. Definition of the upstream table ensures that a table cannot replicate updates to any replica. Replication depends on a two-way agreement between the owners of the two tables.

```
maprcli table upstream add -path <path to the replica> -upstream
<path to the source table>
```

To add the `customers` table in the `sanfrancisco` cluster as an upstream source for the `customers` table in the `newyork` cluster:

```
maprcli table upstream add -path /mapr/newyork/customers -upstream /
mapr/sanfrancisco/customers
```

e) Verify that you specified the correct source table by running the `maprcli table upstream list` command.

```
maprcli table upstream list -path <path to the replica>
```

To verify this in our example scenario, you could use this command:

```
maprcli table upstream list -path /mapr/newyork/customers
```

f) If you set `-paused` to `true` when adding the replica, follow these steps:

1. Load the replica with data from the source table by using the MapR Database CopyTable utility for binary tables or the MapR Database JSON CopyTable on page 5312 utility for JSON tables.
2. Start replication with the command `maprcli table replica resume`. Here is the `maprcli` command:

```
maprcli table replica resume -path <path to the source
table> -replica <path to the replica>
```

For our example scenario, you could use this command:

```
maprcli table replica resume -path mapr/sanfrancisco/
customers -replica /mapr/newyork/customers
```

2. For manual setup in the other direction, follow these steps:

a) Log into both the source and destination clusters.

b) Register the replica as a replica of the source table by running the `maprcli table replica add` command.

```
maprcli table replica add -path <path to the source table> -replica
<path to the replica>
```

c) Verify that you specified the correct replica by running the `maprcli table replica list` command.

```
maprcli table replica list -path <path to the source table>
```

d) Authorize replication between the tables by defining the source table as the upstream table for the replica by running the `maprcli table upstream add` command. Definition of the upstream table ensures that a table cannot replicate updates to any replica. Replication depends on a two-way agreement between the owners of the two tables.

```
maprcli table upstream add -path <path to the replica> -upstream
<path to the source table>
```

- With multi-master replication, if one of the tables goes offline, direct client applications to the other table. For more information, see Multi-Master Replication on page 616.
- Be aware that changes to the structure of a source table are not replicated automatically to replicas. For more information, see Adding a Column Family to a Replica on page 1086
- You can check for alarms related to replication and whether synchronous replication is switched temporarily to asynchronous replication by looking in the Control System. See Table-Replication Alarms.

Adding Table Replicas
Explains how to add table replicas using either the Control System, the CLI or the REST API.

You can register a MapR Database binary or JSON table as a replica of another MapR Database binary or JSON table using the Control System and the CLI. When you add a replica using the Control System, you can also setup and start replication between a source MapR Database Binary or JSON table to a replica MapR Database Binary or JSON table. Before you begin, complete the steps to prepare MapR clusters for table replication.

Adding Table Replica Using the Control System

To replicate a table:

1. Log in to the Control System and go to the **Replication** tab in the table information page.
2. Click **Add Replica** and follow the steps for:
   - Adding JSON Table Replicas on page 1076
   - Adding Binary Table Replicas on page 1078
3. Click **Add Replica**.

Adding Table Replica Using the CLI or the REST API

To add a replica, run the following command:

```
/opt/mapr/bin/maprcli table replica add -path <table path> -replica
<replica table path>
```

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on both the source volume and the target volume
- `lookupdir` on directories in the paths of both tables
- `readperm` and `replperm` permissions on the source table

For complete reference information, see `table replica add` on page 1851.

Note: You also have the option to setup table replication with `maprcli table replica autosetup` which will setup and start replication. For more information, see Setting Up Table Replication Using the CLI on page 1069.

Adding JSON Table Replicas
Explains how to add replicas of JSON tables using either the Control System, the CLI or the REST API.

You can register a MapR Database JSON table as a replica of another MapR Database JSON table using either the Control System or the CLI. When you add a replica using the Control System, you can also setup and start replication between a source MapR Database JSON table to a replica MapR Database JSON table. Before you begin, complete the steps to prepare MapR clusters for table replication.

Adding JSON Table Replica Using the Control System

To create a replica:

1. Go to the table information page.
   See Viewing Table Information on page 1032.
2. Click **Replicas** tab.
   The list of replicas associated with the table displays.
3. Click **Add Replica**.
   The **Add Replica** page displays.
4. Specify the following settings:

Destination Cluster	The destination cluster for the replica, where gateways are configured to allow source cluster to send updates.
Path to Replica	The path to the replica. • For a table on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under a volume with a mount point at `/volume1`, specify the following path: `/volume1/testdst` • For a table on another cluster, you must also specify the cluster name in the path. For example, for a table named `customerdst` under `volume1` in the `sanfrancisco` cluster, specify the following path: `/mapr/sanfrancisco/volume1/customerdst` **Note:** For replication to a table, the command will fail if the replica path you specify points to table that already exists.
Replication State	Specify whether or not to start replication by choosing one of the following: • Automatic Setup — Creates the table on the destination cluster, registers the table on the destination cluster as a replica, adds the current table as an upstream source, copies the content of the current table into the replica, and starts replication.

	• Pause Replication — Registers the table on the destination cluster as a replica and adds the current table as an upstream source, but prevents replication from immediately starting after. Pausing replica like this allows you to load the data into the replica from the current table, after which you can restart replication. **Note:** Although visible, this option is not supported if the source or replica is on a remote secure cluster.
Multi-Master Setup	(Available only with **Automatic Setup**) Multi-master topology, in which there are two primary-secondary relationships, with each table playing both the primary and secondary roles. Client applications update both tables and each table replicates updates to the other. See Multi-Master Replication on page 616. If this is not selected, table replication will be basic primary-secondary topology. In this topology, you replicate one way from source tables to replicas. The replicas can be in a remote cluster or in the cluster where the source tables are located. See Primary-Secondary Replication on page 611. **Note:** Access control expressions on the replica that control permissions for updates to column families and columns are irrelevant because all updates from a source table arrive at a replica after having been authenticated at a gateway, which has the implicit authority to update replicas.

5. Set the following optional properties:

Throttle	Specify whether (**Yes**) or not (**No**) to throttle replication operations. Throttle the replication stream to minimize the impact of the replication process on incoming operations during periods of heavy load.
Replicate Synchronously	Specify whether replication is synchronous (**Yes**) or asynchronous (**No**). See Modes of replication on page 610 for more information.
Encrypt On Wire	Specify whether (**Yes**) or not (**No**) to enable on-wire encryption. If you enable this, the local cluster and any other cluster that is part of the replication process must be enabled for security.
Compress On Wire	The type of on-wire compression. Choose one of the following: • Inherited • OFF • LZF • LZ4 • ZLib

6. Choose whether to:
   - **Replicate Entire Document**
   - **Replicate Selected Field Paths** — Specify the full path to the field in dotted notation. For example, suppose the table contained JSON documents that were of this general structure:

```
{
     "_id" : "ID",
     "a" :
```

```
            {
                 "b" :
                      {
                           "c" : "value",
                      },
                 "e" : "value"
            }
}
```

To replicate field `c`, you must specify `a.b.c` as the field path. Do not use quotation marks and do not include spaces after each dot. Click:

- **Add Field** to add another field to replicate.
- to remove a field.

By default, the entire document in the source table is replicated.

Note: If a field is added at a later date, replication for that field will start at that time.

7. Click **Add Replica**.
   A table with the specified fields is created in the destination cluster, the new table is declared to be a replica of the source table, and the source table is registered as an upstream source for the replica.

Adding JSON Table Replica Using the CLI or the REST API

To add a replica, run the following command:

```
maprcli table replica add -path <table path> -replica <replica table path>
```

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on both the source volume and the target volume
- `lookupdir` on directories in the paths of both tables
- `readperm` and `replperm` permissions on the source table

For complete reference information, see `table replica add` on page 1851.

Note: You also have the option to use `maprcli table replica autosetup` which will setup and start replication. For more information, see `table replica autosetup` on page 1854.

Adding Binary Table Replicas
Explains how to add replicas of binary tables using either the Control System or the CLI.

You can register a MapR Database Binary table as a replica of another MapR Database Binary table using the Control System and CLI. When you add a replica using the Control System, you can also setup and start replication between a source MapR Database Binary table to a replica MapR Database Binary table. Before you begin, complete the steps to prepare MapR clusters for table replication.

Adding Binary Table Replica Using the Control System

To create a replica:

1. Go to the table information page.
   See Viewing Table Information on page 1032.
2. Click **Replicas** tab.
   The list of replicas associated with the table displays.

3. Click **Add Replica**.
   The **Add Replica** page displays.

4. Specify the following settings:

Destination Cluster	The destination cluster for the replica, where gateways are configured to allow source cluster to send updates.
Path to Replica	The path to the replica. • For a table on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under a volume with a mount point at `/volume1`, specify the following path: `/volume1/testdst` • For a table on another cluster, you must also specify the cluster name in the path. For example, for a table named `customerdst` under `volume1` in the `sanfracisco` cluster, specify the following path: `/mapr/sanfrancisco/volume1/customerdst` **Note:** For replication to a table, the command will fail if the replica path you specify points to table that already exists.
Replication State	Specify whether or not to start replication by choosing one of the following: • Automatic Setup — Creates the table on the destination cluster, registers the table on the destination cluster as a replica, adds the current table as an upstream source, copies the content of the current table into the replica, and starts replication. • Pause Replication — Creates the table on the destination cluster, registers the table on the destination cluster as a replica, adds the current table as an upstream source, but prevents replication from immediately starting after. Pausing replica like this allows you to load the data into the replica from the current table, after which you can restart replication. **Note:** Although visible, this option is not supported if the source or replica is on a remote secure cluster.
Multi-Master Setup	(Available only with **Automatic Setup**) Multi-master topology, in which there are two primary-secondary relationships, with each table playing both primary and secondary roles. Client applications update both tables and each table replicates updates to the other. See Multi-Master Replication on page 616. If this is not selected, table replication will be basic primary-secondary topology. In this topology, you replicate one way from source tables to replicas. The replicas can be in a remote cluster or in the cluster where the source tables are located. See Primary-Secondary Replication on page 611. **Note:** Access control expressions on the replica that control permissions for updates to column families and columns are irrelevant because all updates from a source table arrive at a replica after having been authenticated at a gateway, which has the implicit authority to update replicas.

5. Set the following optional properties:

Throttle	Specify whether (**Yes**) or not (**No**) to throttle replication operations. Throttle the replication stream to minimize the impact of the replication process on incoming operations during periods of heavy load.
Replicate Synchronously	Specify whether replication is synchronous (**Yes**) or asynchronous (**No**). See Modes of replication on page 610 for more information.

last-updated: Apr 11, 2022

Encrypt On Wire	Specify whether (**Yes**) or not (**No**) to enable on-wire encryption. If you enable this, the local cluster and any other cluster that is part of the replication process must be enabled for security.
Compress On Wire	The type of on-wire compression. Choose one of the following: • Inherited • OFF • LZF • LZ4 • ZLib

6. Choose whether to:
   - **Replicate all column families**
   - **Replicate Selected Column Families** — Specify the column family name and select:
     - **Include All Columns** — to replicate all the columns associated with the column family.
     - **Assign Columns** — to specify specific columns associated with the column family. To add more columns, click +.

   By default, all columns in the source table are replicated.

   Note: While the column families that you specify must already exist in the source table, the columns that you specify do not have to exist in the destination table for replication to succeed. If a column is added at a later date, replication for that column will start at that time.

7. Click **Add Replica**.
   A table with the specified column families is created in the destination cluster, the new table is declared to be a replica of the source table, and the source table is registered as an upstream source for the replica.

Adding Binary Table Replica Using the CLI or the REST API

To add a replica, run the following command:

```
maprcli table replica add -path <table path> -replica <replica table path>
```

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on both the source volume and the target volume
- `lookupdir` on directories in the paths of both tables
- `readperm` and `replperm` permissions on the source table

For complete reference information, see `table replica add` on page 1851.

Note: You also have the option to use `maprcli table replica autosetup` which will setup and start replication. For more information, see `table replica autosetup` on page 1854.

Displaying the List of Table Replicas
Describes how to view information on the table replicas using the Control System or the CLI.

last-updated: Apr 11, 2022

Displaying the List of Table Replicas Using the Control System

To view table replicas:

1. Log in to the Control System and go to the table information page.
2. Click **Replication**.
   The page displays all the replicas and for each replica, the pane displays the following statistics:

Column Name	Column Description
Paused	Whether replication is paused.
Destination Cluster	The cluster on which the replica resides.
Destination Path	The path to the destination.
Up to Date	Whether replica is up-to-date. Data may not be up-to-date for the following reasons: • Data Pending — Indicates the amount of data that is yet to be replicated to the replica. • Puts Pending — Indicates the number of puts that are yet to be replicated to the replica. • Bytes Pending — Indicates the number of bytes that are yet to be replicated to the replica.
Earliest	The epoch time in milliseconds of the oldest operation that is yet to be replicated to the replica.
Latest	The epoch time in milliseconds of the newest operation that is yet to be replicated to the replica.
Errors	Error () information, if any.
Compression Type	The type of on-wire compression.
Synchronous	Whether replication is synchronous or asynchronous.
Throttled	Whether replication is throttled.
Encrypted	Whether replication is encrypted.

Retrieving List of Table Replicas Using the CLI or the REST API

To view table replicas and associated replica statistics for a table, run the following command:

```
maprcli table replica list -path <table-path>
```

For more information, see `table replica list` on page 1863

Modifying Table Replica

Explains how to edit the properties of a replica using the Control System and the CLI.

Modifying a Table Replica Using the Control System

To modify the properties of a table replica:

1. Go to the table information page.
   See Viewing Table Information on page 1032.
2. Click **Replicas** to go to that tab.
3. Click  associated with the replica to modify.

last-updated: Apr 11, 2022

The **Edit Replica** page displays.

4. Make changes to the following as desired:

Path to Replica	The path to the replica. • For a table on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under a volume with a mount point at `/volume1`, specify the following path: `/volume1/testdst` • For a table on another cluster, you must also specify the cluster name in the path. For example, for a table named `customerdst` under `volume1` in the `sanfrancisco` cluster, specify the following path: `/mapr/sanfrancisco/volume1/customerdst` **Note:** For replication to a table, the command will fail if the replica path you specify points to a table that already exists.
Replication State	Specify whether or not to start replication by choosing one of the following: • Automatic Setup — Creates the table on the destination cluster, registers the table on the destination cluster as a replica, adds the current table as an upstream source, copies the content of the current table into the replica, and start replication. • Pause Replication — Creates the table on the destination cluster, registers the table on the destination cluster as a replica, adds the current table as an upstream source, but prevents replication from immediately starting after. Pausing replica like this allows you to load the data into the replica from the current table, after which you can restart replication.
Multi-Master Setup	(Available only with **Automatic Setup**) Multi-master topology, in which there are two primary-secondary relationships, with each table playing both primary and secondary roles. Client applications update both tables and each table replicates updates to the other. See Primary-Secondary Replication on page 611. If this is not selected, table replication will be basic primary-secondary topology, which is the default. In this topology, you replicate one way from source tables to replicas. The replicas can be in a remote cluster or in the cluster where the source tables are located. See Primary-Secondary Replication on page 611. **Note:** Access control expressions on the replica that control permissions for updates to column families and columns are irrelevant because all updates from a source table arrive at a replica after having been authenticated at a gateway, which has the implicit authority to update replicas.

5. Set the following optional properties:

Throttle	Specify whether (**Yes**) or not (**No**) to throttle replication operations. Throttle the replication stream to minimize the impact of the replication process on incoming operations during periods of heavy load.
Replicate Synchronously	Specify whether replication is synchronous (**Yes**) or asynchronous (**No**).
Encrypt On Wire	Specify whether (**Yes**) or not (**No**) to enable on-wire encryption. If you enable this, the local cluster and any other cluster that is part of the replication process must be enabled for security.

last-updated: Apr 11, 2022

Compress On Wire	The type of on-wire compression. Choose one of the following: • Inherited • OFF • LZF • LZ4 • ZLib

6. Choose whether to:
   - For JSON table replica:
     - **Replicate Entire Document**
     - **Replicate Selected Field Paths** — Specify the full path to the field in dotted notation. For example, suppose the table contained JSON documents that were of this general structure:

```
{
     "_id" : "ID",
     "a" :
          {
               "b" :
                    {
                         "c" : "value",
                    },
               "e" : "value"
          }
}
```

       To replicate field `c`, you must specify `a.b.c` as the field path. Do not use quotation marks and do not include spaces after each dot. Click:
       - **Add Field** to add another field to replicate.
       - ⛔ to remove a field.
   - For Binary table replica:
     - **Replicate All Column Families**
     - **Replicate Selected Column Families** — Specify the column family name and select:
       - **Include All Columns** — to replicate all columns associated with the column family.
       - **Assign Columns** — to specify specific columns associated with the column family. To add more columns, click +.
7. Click **Edit Replica** for the changes to take effect.

Modifying a Table Replica Using the CLI or REST API

The basic command to modify a table replica is:

```
maprcli table replica edit -path <table path> -replica <replica table path>
```

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on both the source volume and the target volume

last-updated: Apr 11, 2022

- `lookupdir` on directories in the paths of both tables
- `readperm` and `replperm` permissions on the source table

For complete reference, see `table replica edit` on page 1859.

Removing Table Replicas
Explains how to un-register one or more replicas using either the Control System or the CLI.
Removing Replication Using the Control System

1. Go to the table information page.
   See Viewing Table Information on page 1032.
2. Click **Replication** to go to that tab.
3. Select the replicas to remove by clicking the associated checkbox.
   Selecting the checkbox next to a replica makes the **Actions** drop-down menu available.
4. Select **Remove Replica(s)** from the **Actions** drop-down menu.
   The **Remove Replica(s)** confirmation window displays.
5. Verify the list of replicas to remove and click **Remove Replica**.
   If necessary, click ✕ to remove a replica.
   The selected replicas are no longer replicas of the source table and will no longer receive updates from the source table. You must also remove upstream source to remove the association between the source table and the replica table. For more information, see Removing Upstream Sources.

Removing Replication Using the CLI or REST API

The un-register a replica, run the following command:

```
maprcli table replica remove -path <table path> -replica <replica table
path>
```

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on both the source volume and the target volume
- `lookupdir` on directories in the paths of both tables
- `readperm` and `replperm` permissions on the source table

For complete reference, see `table replica remove` on page 1867.

Pausing Table Replication
Explains how to pause table replication of data from a source MapR Database binary or JSON table to a replica MapR Database binary or JSON table respectively using either the Control System or the CLI.
Pausing Table Replication Using the Control System

1. Go to the table information page.
   See Viewing Table Information on page 1032.
2. Click **Replicas** to go to that tab.
3. Select the replicas to pause by clicking the associated checkbox.
   Selecting the checkbox next to a replica makes the **Actions** drop-down menu available.

4. Select **Pause Replication** from the **Actions** drop-down menu.
   The **Pause Replication** confirmation window displays.
5. Verify the list of replicas to pause and click **Pause Replication** to pause replication on the selected replicas.
   If necessary, click ✕ to remove a replica from being paused.

Pausing Table Replication Using the CLI or REST API

The pause replication, run the following command:

```
maprcli table replica pause -path <table path> -replica <replica table path>
```

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on both the source volume and the target volume
- `lookupdir` on directories in the paths of both tables
- `readperm` and `replperm` permissions on the source table

For complete reference, see `table replica pause` on page 1866.

Resuming Table Replication
Explains how to resume replication between a source MapR Database binary or JSON table and a replica of that table when the replication state is set to paused from the Control System or by the `maprcli table replica add` or the `maprcli table replica pause` command.
Resuming Replication Using the Control System

1. Go to the table information page.
   See Viewing Table Information on page 1032.
2. Click **Replicas** to go to that tab.
3. Select the replicas in paused state by clicking the associated checkbox.
   Selecting the checkbox next to a replica makes the **Actions** drop-down menu available.
4. Select **Resume Replication** from the **Actions** drop-down menu.
   The **Resume Replication** confirmation window displays.
5. Verify the list of replicas for which to resume replication and click **Resume Replication**.
   If necessary, click ✕ to leave the replica in paused state.

Resuming Replication Using the CLI or REST API

The resume replication, run the following command:

```
maprcli table replica resume -path <table path> -replica <replica table
path>
```

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on both the source volume and the target volume
- `lookupdir` on directories in the paths of both tables
- `readperm` and `replperm` permissions on the source table

For complete reference, see `table replica resume` on page 1869.

Adding a Column Family to a Replica
Changes to the structure of a source table are not automatically replicated to replica table. You must complete the following steps to add a column family to a replica.

When an entire table is being replicated, new column families are not automatically created at the replica. However, once you add the new column family to the replica then updates to the new column family will immediately start being replicated.

When you are replicating a subset of column families and columns, you must add the new column family to the replica and also run the copytable utility to initially populate the new column family in the replica table.

1. Pause replication by running the `maprcli table replica pause` command.
2. Add the new column family to the replica by running the `maprcli table replica edit` command.
3. If the replica includes a subset of column families and columns from the source table, copy the data from the new column family from the source table into the replica by using the CopyTable utility. Use the `-columns` parameter to specify the name of the column family.
4. Resume replication by running the `maprcli table replica resume` command.

Viewing Active Table Replication Alarms
Describes how to view active table replication alarms using the Control System and the CLI.

You can view table replication alarms using the Control System, the log files, and the CLI.

Viewing Active Table Alarms in the Control System

- Log in to the Control System and click **Data** > **Tables** to view table replication alarms in the **Active Alarms** pane.

  You can:

  - View information related to the alarm.
  - Dismiss an alarm.
  - Mute an alarm.

  See Table-Replication Alarms on page 2237 for more information on the table alarms.

Retrieving Active Table Replication Alarms Using the CLI or REST API

Alarms for replication are issued per volume rather than per source table. To retrieve table replication alarms, run the following command:

```
maprcli alarm list -cluster <cluster name> -type volume
```

For complete reference information, see `alarm list` on page 1545.

Viewing Table Replication Alarms in the Log Files

The log files `mfs.log-5` and `cldb.log` display these alarms. These files are located in the `/opt/mapr/logs` directory.

Managing Upstream Source for Table Replicas

You can set up a table to be the upstream source for replicas. This is especially useful if you did not set up replication automatically when setting up replicas.

Setting Table as Upstream Source for a Replica
Explains how to set up the current table as the upstream source for a replica if the replica was not configured to automatically re-sync with the current table.
Setting Up Current Table as Upstream Source for a Replica Using the Control System

To set up a table as the upstream source for replicas:

1. Go to the **Replication** tab in the table information page.
2. Select the checkbox beside the replicas that do not have the current table configured as upstream source for automatic re-sync.
   Selecting a checkbox next to a replica makes the **Actions** drop-down menu available.
3. Select **Set Current Table as Upstream Source** from the **Actions** drop-down menu.
   The **Set Current Table as Upstream Source** dialog displays.
4. Review the list of selected replicas and click **Set Upstream Source**.
   The current table will automatically send updates to the replica(s).

Setting Up Table as Upstream Source for a Replica Using the CLI or REST API

The basic command to set a table as the upstream source for a replica is:

```
maprcli table upstream add -path <replica table path> -upstream <source
table path>
```

See `table upstream add` on page 1870 for complete reference information.

Adding Upstream Source for Table
Describes how to add an upstream source for a table using either the Control System or the CLI.
Adding Upstream Source Using the Control System

To add an upstream source for the current table:

1. Log in to the Control System and go to the **Replication** tab in the table information page.
2. Click **Add Upstream Source Table** in the **Upstream Sources** pane.
   The **Add Upstream Source Table** window displays.
3. Enter the path to the upstream source table.
   - For a path on the local cluster, start the path at the volume mount point. For example, for a table named `testsrc` under `volume1` which has a mount point at `/volume1`, specify the following path: `/volume1/testsrc`
   - For a path on another cluster, you must also specify the cluster name in the path. For example, for a table named `customersrc` under `volume1` in the `sanfrancisco` cluster, specify the following path: `/mapr/sanfrancisco/volume1/customersrc`
4. Click **Add Upstream Source Table**.

Adding Upstream Source Using the CLI or the REST API

The basic command to add upstream source is:

```
maprcli table upstream add -path <table path> -upstream <upstream table
path>
```

See `table upstream add` on page 1870 for complete reference information.

Listing all Upstream Sources for a Table
Explains how to retrieve and view the list of upstream sources for a table using either the Control System, or the CLI.
Viewing Upstream Sources Using the Control System

To view the list of upstream sources:

- Log in to the Control System and go to the **Replication** tab in the table information page.
  The list of upstream sources for the table displays in the **Upstream Source** pane. For each upstream source, the pane displays the following:

Column Name	Column Description
IDX	The index number of the replica table.
Upstream Source Cluster	The cluster on which the upstream table resides.
Upstream Source Path	The path to the upstream source table.
UUID	The replica table's universally unique identifier.

  Selecting the checkbox beside an upstream source makes the **Remove Upstream Source(s)** button available. You can:

  - Add an upstream source table
  - Remove an upstream source table

Retrieving Upstream Sources Using the CLI or the REST API

The basic command to retrieving the list of upstream sources for a table is:

```
maprcli table upstream list -path <table path>
```

See `table upstream list` on page 1871 for complete reference information.

Removing Upstream Source
Explains how to remove a table as an upstream source using either the Control System or the CLI.
Removing Upstream Source Using the Control System

To remove upstream source:

1. Log in to the Control System and go to the **Replication** tab in the table information page.
   See Viewing Table Information on page 1032.
2. Select the upstream sources to remove in the **Upstream Source** pane by clicking the associated checkbox.
3. Click **Remove Upstream Source Table** in the **Upstream Source** pane.
   The **Remove Upstream Source Table** confirmation dialog displays.
4. Verify the list of upstream sources to remove and click **Remove Upstream Source Table**.

Removing Upstream Source Using the CLI or the REST API

The basic command to remove upstream source is:

```
maprcli table upstream remove -path <table path> -upstream <upstream table
path>
```

See `table upstream remove` on page 1873 for complete reference information.

Managing Secondary Indexes

Describes how to manage secondary indexes, including adding, deleting, and listing indexes, setting up your cluster for querying, and troubleshooting index usage.

You can add and manage secondary indexes using either the `maprcli table index` commands or the Control System. The `maprcli table index` commands are also available using the REST API. The following diagram provides links to information about managing indexes.

1. Describes the tasks needed to prepare your environment so you can query MapR Database JSON tables using secondary indexes.
2. Describes how to add secondary indexes on MapR Database JSON tables.
3. Describes how to view information about secondary indexes. This includes information about whether an index is lagging its parent JSON table.
4. Describes how to use the MapR Database shell to examine the contents of a secondary index. This includes displaying information about errors encountered inserting into the index.
5. Describes how to verify that the data in a secondary index is consistent with its JSON table.
6. This topic describes how to remove secondary indexes that are no longer needed.
7. Describes how to debug and troubleshoot usage of secondary indexes.

The following permissions are required to add, remove, and list indexes. Indexes share the same volume and topology as its JSON table, so the applicable permissions are on the volume and JSON table path.

- `readAce` on the volume
- `lookupdir` on directories in the table path

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to perform this operation unless that user is given the relevant permission or permissions with access-control expressions.

Also, to add or remove an index, the user must be the creator of the table or a user with `indexperm` permission.

Preparing Clusters for Querying using Secondary Indexes on JSON Tables

Describes the tasks needed to prepare your environment so you can query MapR Database JSON tables using secondary indexes.

Installing with the MapR Installer

To install MapR using the MapR installer, follow the steps outlined at Installing with the MapR Installer on page 141.

last-updated: Apr 11, 2022

Starting with MapR 6.0.1, you do not have to enable a separate query service to use secondary indexes. The **Operational Applications with MapR Database** template installs and configures the replication gateways needed to update secondary indexes in MapR Database JSON and includes the components needed to run OJAI queries.

You must enable the OJAI Distributed Query Service on page 505 to use certain features. The following table summarizes the differences in the functionality of OJAI queries when you do and do not have the service enabled:

Service Not Enabled	Service Enabled
• Can run queries that use a single secondary index • Can sort data in your queries up to a configurable limit	• Can run queries that use multiple secondary indexes • Can sort data in your queries without any limit • Can run queries in parallel

Selecting any of the following templates enables the OJAI Distributed Query Service:

- **Operational Applications with MapR Database and Distributed Query Service**
- **MapR Converged Cluster: Batch, interactive and real-time analytics**
- **Analytics with MapR Database**

You can also explicitly enable the OJAI Distributed Query Service by selecting the service in the **Custom Services** template.

For more information about installer templates, see Auto-Provisioning Templates on page 5448.

For more information about how secondary index selection and execution works in MapR Database JSON, see Selection and Execution of Secondary Indexes on page 582.

Note: The **OJAI Query Service** has been renamed to the **OJAI Distributed Query Service** in MapR 6.0.1. All information about the OJAI Distributed Query Service applies to the OJAI Query Service, except where noted.

Installing without the MapR Installer

Other sections of the documentation describe the detailed steps for installing and configuring without the MapR installer. Generally, you need to perform the following steps:

1. Install software.

   To install MapR without using the MapR installer, follow the steps outlined at Installing without the MapR Installer on page 141. In addition to installing MapR core packages, you also need to install MapR Drill if you want advanced secondary index selection, sorts on large data sets, and parallel query execution. When installing Drill, make sure to Configure the OJAI Distributed Query Service on page 182.

2. Install and configure replication gateways.

   Updates are propagated from the JSON tables using the Gateways for Replicating MapR Database Tables on page 621. You need to install the replication gateways. Since the source JSON table and the secondary index are on the same volume within a cluster, configure an intracluster gateway. In this type of gateway, the source and destination clusters are the same.

   If your gateways are running on the same nodes as CLDB, then no additional configuration steps are required. See Configuring Gateways for Table and Stream Replication on page 1152 for details about this scenario and other options for configuring your gateways.

Upgrades

Other sections of the documentation describe the detailed steps for upgrades. Generally, you need to perform the following steps:

1. Upgrade your MapR software by following the instructions at Upgrading MapR Core or EEP Components on page 284.
2. Install MapR Drill, if you have not already done so and want to sort large data sets and run queries in parallel.
3. When installing Drill, make sure to Configure the OJAI Distributed Query Service on page 182.
4. If you are upgrading without the MapR installer, follow step 2 in the previous section to install and configure replication gateways.
5. Enable the replication support needed to propagate index updates by running the following command:

   ```
   maprcli cluster feature enable -name mfs.feature.db.streams.v6.support
   ```

   See Step 4: Enable New Features on page 330 for further details.

Important: If you are using a Manual Rolling Upgrade Description on page 319, you must upgrade all nodes running replication gateways before performing updates on tables with indexes. Otherwise, the index updates will hang.

Adding Secondary Indexes on JSON Tables

Describes how to add secondary indexes on MapR Database JSON tables.

You can add secondary indexes using the Control System, or the `maprcli table index` commands.

If you are adding an index on a large table, particularly if it contains complex data, you should consider modifying the `mfs.db.parallel.copyregions` parameter using the maprcli config save command. The parameter controls the degree of parallel processing in your MapR cluster. Increasing parallelism can improve the index build time by optimizing the build's intermediate copy operation.

Permissions

You need the following permissions to add an index:

- `readAce` on the volume
- `lookupdir` on directories in the table path
- `indexperm` permission on the table

If you created the table in version 6.0 or later, you automatically have `indexperm` permission. For tables created before 6.0, even if you are the owner of the table, you must explicitly add `indexperm` permission.

Note: The mapr user is not treated as a superuser. MapR Database does not allow the mapr user to perform this operation unless that user is given the relevant permission or permissions with access-control expressions.

See Restrictions on Secondary Indexes on page 565 for information about other restrictions.

Adding Indexes Using the Control System

1. Log in to the Control System and go to the **Indexes** tab in the table information page.
2. Click **Add Index** to display the **Add Index** page.
3. Specify the following settings:

last-updated: Apr 11, 2022

a) Specify the name of the index in the **Index Name** field.

b) Specify whether (**Yes**) or not (**No**) the index is hashed in the **Hashed** field. If **Yes**, specify the number of hash index partitions for distributing the keys.

See Hashed Indexes on page 555 for information about whether you should create a hashed index.

4. Specify the list of indexed fields under **FIELDS INDEXED**.
   a) Specify the name of the indexed field in the **Field Name** text field.

   b) Select the ordering (**Ascending** or **Descending**) for the field from the **Order** drop-down menu.

   c) Select the function of the field from the **Function** drop-down menu.

   Before defining an index that specifies index keys with CAST functions, see Using Casts in Secondary Indexes on page 557 for more information on creating indexes using CAST functions.

   Tip: To add more indexed fields, click **Add Another** and repeat step 4.

5. Specify the names of the included fields under **INCLUDED FIELDS**.
   For more information, see Covering Indexes on page 560.

   Tip: To add additional included fields, click **Add Another** and repeat step 5.

6. Click **Add Index** to create the index.

Adding Indexes Using the CLI

The following is the basic command for adding a secondary index on a JSON table.

```
maprcli table index add
  -path <path>
  -index <index name>
  -indexedfields < indexed field names >
```

See `table index add` on page 1827 for a description of the complete syntax.

Troubleshooting Secondary Indexes
Describes how to debug and troubleshoot usage of secondary indexes.

The following table lists problems you may encounter when using secondary indexes. Based on the symptoms listed in the first column, refer to the section in the third column to further troubleshoot the issue.

Symptom	Possible Cause	Troubleshooting Steps
Query performance is slow	Query is not using secondary indexes	1. Determining the Query Execution Path for OJAI Queries on page 1097 2. Determining Secondary Index Usage on page 1098
	Non-optimal OJAI query plan chosen	Examining the OJAI Query Plan on page 1098
	Non-optimal query plan chosen by OJAI Distributed Query Service	Determining Index Use on page 3364

Symptom	Possible Cause	Troubleshooting Steps
Inconsistent query results	Secondary index update lag	Identifying Secondary Index Lag on page 1099
	Unresolved encoding errors	Troubleshooting Secondary Index Encoding Errors on page 1101
Query runs out of memory	Memory configuration in the OJAI Distributed Query Service set too low	Adjusting Memory Settings in the OJAI Distributed Query Service on page 1103

Secondary Index Restrictions

When troubleshooting secondary indexes, you should also keep in mind the following restrictions:

Name Restrictions

You cannot use the following characters in the index name and in the indexed fields:

```
< > ? % \
```

To use the following characters in the index name and in the indexed fields, enclose them either in single or double quotes:

```
; | ( ) /
```

For example:

```
maprcli table index
add -path /volume1/MYTABLE -index
"MYTABLE1_ANALYSIS_1♦=#;{}&()/" \
  -indexedfields "●timestamp":desc,"●
♥,"LOTNo" -includedfields \
  "♥,"♦=#;{}&()/" (or)

maprcli table index
add -path /volume1/MYTABLE -index
'MYTABLE1_ANALYSIS_1♦=#;{}&()/' \
  -indexedfields "●timestamp":desc,"●
♥,"LOTNo" -includedfields \
  '♦,'♦=#;{}&()/'
```

To use either the ' or the " character in the index name and in the indexed fields, enclose:

- the ' character within double quotes (")
- the " character within single quote (')

For example:

```
maprcli table index
add -path /volume1/MYTABLE -index
"'MYTABLE1_ANALYSIS_1♦=#;{}&()/" \
  -indexedfields "●timestamp":desc,"♦●
♥,"LOTNo" -includedfields \
  "♦","♦=#;{}&()/" (or)

maprcli table index
add -path /volume1/MYTABLE -index
'"MYTABLE1_ANALYSIS_1♦=#;{}&()/' \
  -indexedfields "'●timestamp":desc,"●
```

```
🐰,"LOTNo" -includedfields \
  '🐰',"🍁=#;{}&()/"
```

Type Restrictions

- If a composite index includes the same subfield in multiple indexed fields, the implied types of the subfields must be consistent.

  For example, you cannot create an index with the following indexed fields:

  ```
  a.b[].c, a.b.d
  ```

  Although subfield b appears in both indexed fields, in the first, it is an array and in the second, it is a nested document.

  See Composite Indexes and Container Field Paths on page 554 for more details.

Size Restrictions

- The maximum size of all indexed fields in an index is 32 KB.

  If the collective size exceeds 32 KB, then an insert of the corresponding document results in an encoding error (`INDEX_ROW_KEY_ENCODER_ERROR_ENCODING_IS_TOO_LONG`).

- The maximum number of indexes that you can create on a JSON table is 32.

Field Definition Restrictions

- You cannot specify individual array elements as indexed fields.
- You cannot specify a table's `_id` field as an indexed field.
- If a field contains an array of nested documents and you want to index on subfields in the nested documents, then you must define the indexed field using a container field path.
- You can include a specific field only once as either an indexed or included field, with the following two exceptions:
  - The indexed field is a container field path:

    ```
    maprcli table index add -path /
    people \
       -index phoneNumberIdx \
       -indexedfields
    Phones[].Number \
       -includedfields
    Phones[].Number
    ```

- The field specifies a cast to another type.

  You can create an index in which the `score` field is an indexed field cast as a `double` type, and `score` is also an included field. The included field retains the original data type of the `score` field:

  ```
  maprcli table index add -path /
  castTable \
     -index castIdx1 \
     -indexedfields
  '$CAST(score@DOUBLE)' \
     -includedFields score
  ```

  You can create an index in which the `score` field is an indexed field, cast as a `double` type, and the `score` field is also another indexed field, cast as a `long` type:

  ```
  maprcli table index add -path /
  castTable \
     -index castIdx2 \
     -indexedfields
  '$CAST(score@DOUBLE)','$CAST(scor
  e@LONG)'
  ```

- You cannot use casts with included fields.
- You cannot specify a field as either an indexed or included field if the field is also specified as a column family JSON path name.

  For example, suppose you have the following JSON table:

  ```
  {
       "_id" : "ID",
       "a" :
            {
                 "b" :
                      {
                           "c" :
  "value",
                           "d" :
  "value"
                      },
                 "e" : "value"
            }
  }
  ```

  If you create a column family at field c in the JSON path `a.b.c`, you cannot define field `a.b.c` as either an indexed or included field. You can define the fields a, `a.b`, and `a.b.d` as either indexed or included fields.

last-updated: Apr 11, 2022

- You cannot specify an included field in which the data in the field spans more than one column family.

  In the following example, the included field `sl1.sl2` spans column families, `cf2` and `cf3`:

```
maprcli table cf list -path /cftab
compressionperm  readperm
traverseperm  jsonfamilypath
writeperm  minversions
maxversions  compression
ttl         inmemory  cfname
memoryperm
u:root           u:root
u:root
u:root     0
1            lz4
2147483647  false     default
u:root
u:root           u:root
u:root         sl1
u:root     0
1            lz4
2147483647  false     cf1
u:root
u:root           u:root
u:root         sl1.sl2.sl3
u:root     0
1            lz4
2147483647  false     cf2
u:root
u:root           u:root
u:root         sl1.sl2.sl3.sl4
u:root     0
1            lz4
2147483647  false     cf3
u:root

maprcli table index add -path /
cftab -index i1 -indexedfields
sl1.sl2.sl3.sl4.l4a,
sl1.l1a -includedfields
sl1.sl2,sl1.sl2.sl3.sl4.sl5.l5b -js
on
{
        "timestamp":1507419777919,
        "timeofday":"2017-10-07
04:42:57.919 GMT-0700 PM",
        "status":"ERROR",
        "errors":[
                {
                        "id":22,

"desc":"Data for included field
sl1.sl2 may not span more than one
column family."
                }
         ]
}
```

- You cannot specify a composite index with more than one container field path as your indexed fields, unless the prefixes of the container field paths are the same.

  See Composite Indexes and Container Field Paths on page 554 for more details.

- You cannot specify a composite index with an indexed field that is a subfield of another indexed field.

  For example, you cannot create an index with the following indexed fields:

  ```
  a, a.b
  ```

  The indexed field `a.b` is a subfield of the indexed field `a`.

Option Restrictions

- As indexes are automatically split, you cannot disable splits when you create your index.

Index Use Restrictions

- Indexes do not optimize non-existence filter conditions.

Related reference

table index list on page 1834
This topic describes how to list information about the secondary indexes created on MapR Database JSON tables.

MapR Database JSON verifyindex on page 5324
Describes how to use the MapR Database JSON `verifyindex` command to verify that the data in a secondary index is consistent with its JSON table.

dbshell indexscan on page 5298

Determining the Query Execution Path for OJAI Queries

You can determine whether an OJAI query directly accesses MapR Database JSON or leverages the OJAI Distributed Query Service by enabling Java OJAI tracing. Java OJAI tracing logs information that enables you to determine which execution path your queries use.

Follow the instructions at Enable OJAI Tracing on page 2667 to output tracing messages that include query plans.

If the query does not use the OJAI Distributed Query Service, you will see tracing like the following:

```
2017-07-17 17:35:59 TRACE OjaiDocumentStore:132 -
Query Plan: '[{"streamName":"DBDocumentStream","parameters":
{"queryConditionPath":false,"indexName":"abc_Idx","projectionPath":
["c","b","a"],"primaryTable":"/tmp/test-728918932/ei_suffix_sort"}}]'
```

If a query uses the OJAI Distributed Query Service, you will see tracing like the following instead:

```
2017-07-17 18:51:13 TRACE OjaiDocumentStore:132 - Query
Plan:'[{"streamName":"DrillDocumentStream","parameters":{"sql":"select t.`$
$ENC00NQYF6YJUL5UW45AAL5UWI`,t.`$$document` from dfs.`/tmp/testTable` t
where (t.`l0_a4_int` = -92) order by t.`l0_a4_int` ASC,t.`_id` DESC"}}]'
```

```
2017-07-17 18:51:14 DEBUG DrillDocumentStream:120 -
DocumentResultsListener[1].queryIdArrived(queryId =
```

```
2692966d-0888-96e2-fa09-0d9befcd3173 ,sql string = select t.`$
$ENC00NQYF6YJUL5UW45AAL5UWI`,t.`$$document` from dfs.`/tmp/testTable` t
where (t.`l0_a4_int` = -92) order by t.`l0_a4_int` ASC,t.`_id` DESC)
```

Note that the **Query Plan** in the second trace fragment above contains a **DrillDocumentStream** instead of a **DBDocumentStream**. The **sql** parameter in that stream shows a SQL query. This is also missing in the first trace fragment above. The presence of the SQL query indicates that OJAI passes the query to the OJAI Distributed Query Service, as noted in the third trace fragment.

For further information about OJAI query plans, see Examining the OJAI Query Plan on page 1098. For background information about different query execution paths, see OJAI Distributed Query Service on page 505.

Determining Secondary Index Usage
This section describes how to determine whether a query is using secondary indexes, depending on the query execution path used.

Determine whether your query uses the OJAI Distributed Query Service by following the steps outlined at Determining the Query Execution Path for OJAI Queries on page 1097. The following sections describe next steps depending on the execution path.

Simple OJAI Queries

Using the tracing described at Determining the Query Execution Path for OJAI Queries on page 1097, you will see the following in the log output:

```
2017-07-17 17:35:59 TRACE MapRDBTableImplHelper:703 - Scan on Index:
'testIndex', Primary Table is: '/tmp/testTable', Index Scan QueryCondition:
'((field < {"$numberLong":101}))',
Index Scan startRow: '\x0FZ', Index Scan stopRow: '\x0F\x89\xCA\x80\x00'
```

The "**Scan on Index**", highlighted in bold, indicates OJAI used a secondary index to process the query.

Queries Requiring the OJAI Distributed Query Service

The tracing output described at Determining the Query Execution Path for OJAI Queries on page 1097 contains a **queryId**, highlighted in bold:

```
2017-07-17 18:51:14 DEBUG DrillDocumentStream:120 -
DocumentResultsListener[1].queryIdArrived(queryId =
2692966d-0888-96e2-fa09-0d9befcd3173 ,
sql string = select t.`$$ENC00NQYF6YJUL5UW45AAL5UWI`,t.`$$document` from
dfs.`/tmp/testTable` t where (t.`l0_a4_int` = -92) order by t.`l0_a4_int`
ASC,t.`_id` DESC)
```

Use this **queryId** to retrieve more information through the Drill Web Console, including the query plan selected by the OJAI Distributed Query Service. See Query Profile on page 3365 for details.

Examining the OJAI Query Plan
This section describes two ways to access a Java OJAI query plan and provides general information about how to interpret the query plan. You can examine the query plan to determine if the Java OJAI client chooses an appropriate execution path.

Using OJAI Tracing

After following the steps at Determining the Query Execution Path for OJAI Queries on page 1097, if you determine that your query directly accesses MapR Database JSON and does not use the OJAI Distributed Query Service, you can further examine the query plan in the trace output.

As noted in the referenced topic, to enable tracing, set the following property in your `log4j.properties` file, located in the `/opt/mapr/conf` directory:

```
log4j.logger.com.mapr.ojai.store.impl=TRACE, stdout
```

In the following logged output, the query plan uses an index named `i1_idx` and projects field `id1`. It also limits the result to two documents:

```
2017-10-18 11:29:32,876 TRACE
[main] com.mapr.ojai.store.impl.OjaiDocumentStore -
Query Plan: '[{"streamName":"DBDocumentStream","parameters":
{"queryConditionPath":false,"indexName":"i1_idx","projectionPath":
["id1"],"primaryTable":"/tmp/test-728918932/tab"}},
{"streamName":"LimitStream","parameters":{"limit":2}}]'
```

Calling `QueryResult.getQueryPlan`

Instead of using OJAI tracing, you can programmatically retrieve query plans by calling QueryResult.getQueryPlan. The method returns a JSON document that is a list of `Maps`. Each `Map` in the list represents a DocumentStream in the query plan, which corresponds to an operation. The order of the list represents the order the MapR Database client processes each operation. Each `Map` entry contains the name of the DocumentStream (`streamName`) and its parameters. You may see `Map` entries corresponding to the following `DocumentStreams` in a query plan:

- `DBDocumentStream` - Accesses MapR Database without the OJAI Distributed Query Service
- `DrillDocumentStream` - Uses the OJAI Distributed Query Service to process the query
- `LimitStream` - Limits the number of documents to return
- `OffsetStream` - Skips past specified number of documents before reading

Important: The `DocumentStream` names and their parameters are subject to change from one release to the next. Take that into consideration if you plan to write tools that interpret the contents of an OJAI query plan

Identifying Secondary Index Lag

This section describes the commands you use to determine if updates to a secondary index are lagging.

Secondary index lag can occur due to aysnchronous secondary index updates. Use maprcli table index list and mapr verifyindex to assess if there is update lag in an index and to see details on the updates that are lagging.

Run maprcli table index list

The `maprcli table index list` command lists information about indexes created on a table. To retrieve the most current status, specify the optional `refreshnow` parameter. Examine the values of the following fields to determine if an index is lagging:

- isUptodate - True if the index is up-to-date
- bytesPending - The number of bytes pending propagation to the index
- putsPending - The number of puts pending propagation to the index
- minPendingTS - The timestamp of the oldest put pending propagation to the index
- maxPendingTS - The timestamp of the most recent put pending propagation to the index

last-updated: Apr 11, 2022

In the sample output below, the index is up-to-date.

```
maprcli table index list -path /demo/business -json
{
        "timestamp":1506617667735,
        "timeofday":"2017-09-28 04:54:27.735 GMT+0000 PM",
        "status":"OK",
        "total":1,
        "data":[
                {
                        "cluster":"my.cluster.com",
                        "type":"maprdb.si",
                        "indexFid":"2049.93.10257820",
                        "indexName":"i1",
                        "hashed":false,
                        "indexState":"REPLICA_STATE_REPLICATING",
                        "idx":1,
                        "indexedFields":"a.b:ASC",
                        "isUptodate":true,
                        "minPendingTS":0,
                        "maxPendingTS":0,
                        "bytesPending":0,
                        "putsPending":0,
                        "bucketsPending":0,
                        "copyTableCompletionPercentage":100,
                        "numTablets":1,
                        "numRows":4,
                        "totalSize":24576
                }
        ]
}
```

Run mapr verifyindex

If `maprcli table index list` shows that an index is lagging, use the mapr verifyindex command to gather more information. The following example illustrates the output in the case where an index is lagging. The following updates have not yet propagated to the index:

- Document with _id=997 not yet inserted into the index.
- Update to the `city` field in document with _id=998 not yet propagated to the index.

```
// Display contents of parent JSON table
# mapr dbshell

maprdb root:> find /t1
{"_id":"1000","city":"city3","misc":{"a":"misc.a","b":2}}
{"_id":"997","city":"city3","misc":{"a":"misc.a","b":2}}
{"_id":"998","city":"city4","misc":{"a":"misc.a","b":2}}
{"_id":"999","city":"city3","misc":{"a":"misc.a","b":2}}
4 document(s) found.


// Display contents of an index that is lagging.
// Document with _id=997 is missing from the index and
// there is a mismatch in the index data for document with _id=998
maprdb root:> indexscan /t1 --indexname i2
{"_id":"1000","city":"city3"}
{"_id":"998","city":"city3"}
{"_id":"999","city":"city3"}
3 document(s) found.
maprdb root:> quit
```


last-updated: Apr 11, 2022

```
// Output of verifyindex
# mapr verifyindex -path /t1 -index i2

Missing Document in Index:
{"_id":"997","city":"city3"}

Mismatch Document Found!!
Table side-->{"_id":"998","city":"city4"}
Index side-->{"_id":"998","city":"city3"}

Number of rows in table but not in index: 1
Number of rows in index but not in table: 0
Mismatch row count: 1
```

Troubleshooting Secondary Index Encoding Errors

This section describes how to locate secondary index encoding errors in log files, and then resolve them.

Unresolved secondary index encoding errors can result in queries returning incomplete results. See Secondary Index Encoding Error on page 2240 for details.

To troubleshoot secondary index encoding errors, follow these steps:

1. Determine whether any table has index encoding alarms by using one of the following two options:
   - Run the following `grep` command, searching for the strings `index` and `encoding` in the `mfs.log-5` file:

     ```
     # grep -i "index.*encoding" /opt/mapr/logs/mfs.log-5
     2018-07-10 11:06:07,7042 INFO DB db/repl/araggregator.cc:524 Table
     2050.43.262440 hit index row-key encoding error
     2018-07-10 11:06:07,7042 INFO DB db/repl/araggregator.cc:914 Raising
     alarm VOLUME_ALARM_TABLE_INDEX_ENCODING_ERROR for volume 195503497
     ```

     The fid, `2050.43.262440`, in the sample output indicates the table corresponding to the alarm. If you are not sure which table this corresponds to, you can convert the fid to a table path by following the instructions at Converting fid and volid on page 2438.

   - Check for any table index encoding alarms in the MapR Control System, as described at Viewing Active Table Replication Alarms on page 1318.

     The alarm details indicate the table corresponding to the alarm.

2. Find the index on the table from step 1 that is causing the error.

   Run dbshell indexscan on page 5298 with `--mode` set to `err` on each index to see the index's error output. You need to run the command multiple times if a table has multiple indexes with errors.

   For example, if `table1` has three secondary indexes and all three secondary indexes have errors, you must run `indexscan` three times:

   ```
   indexscan /table1 --indexname index1 --mode err
   indexscan /table1 --indexname index2 --mode err
   indexscan /table1 --indexname index3 --mode err
   ```

   The following example shows error output from running the `dbshell indexscan` command:

   ```
   maprdb root:> indexscan /IndexEncodingErrorAlarmsTest1/tab1 --indexname
   idx1 --mode err
   {"_id":"100","$ERROR":"Index field 1: INVALID_CAST"}
   ```


last-updated: Apr 11, 2022

3. Address the identified errors by attempting the following suggested resolutions:

Error	Suggested Resolutions
KEY_TOO_LONG: The collective size of the index key is limited to less than 32 KB.	• Reinsert the row in the JSON table so the collective size of all the indexed fields is less than 32 KB. • Redesign the secondary index so fields with large values are included fields instead of indexed fields. • Reduce the number of indexed fields in the secondary index.
INVALID_CAST: An error was encountered applying the CAST function on an indexed field.	• Verify that you have specified the correct types when using the CAST function with indexed fields. • If the types are correct, reinsert the row in the JSON table so the values of indexed fields can be cast to the specified type. See Using Casts in Secondary Indexes on page 557.

Adjusting Memory Settings in the OJAI Distributed Query Service

This section describes how to verify, through log output, that your OJAI query is running out of memory due to memory limits in the OJAI Distributed Query Service. It then describes how to adjust the memory settings in the service.

1. Before adjusting the OJAI Distributed Query Service memory settings, first confirm that your query has run out of memory due to limits in the service.

   You should see output like the following in your client application log:

```
15:32:46.465 [Thread-21] - Error caused in scan Drill submissionFailed
for "select t.`$$ENC00FIAF62LE`,t.`$$document` from dfs.`/tables/
business` t where ((t.`city` = 'Currie') and (t.`state` = 'PA') and
(t.`review_count` > 5100)) limit 1
org.ojai.exceptions.OjaiException: Drill submissionFailed for "select
t.`$$ENC00FIAF62LE`,t.`$$document` from dfs.`/tables/business` t where
((t.`city` = 'Currie') and (t.`state` = 'PA') and (t.`review_count` >
5100)) limit 128" please ch
        at
com.mapr.ojai.store.impl.DrillDocumentStream$DocumentResultsListener.subm
issionFailed(DrillDocumentStream.java:220)
        at
com.mapr.ojai.store.impl.DelegatingResultsListener$2.run(DelegatingResult
sListener.java:84)
        at
com.mapr.ojai.store.impl.RunnableQueue$QueueRunner.run(RunnableQueue.java
:59)
        at
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java
:1142)
        at
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.jav
a:617)
        at java.lang.Thread.run(Thread.java:745)
Caused by: org.apache.drill.common.exceptions.UserRemoteException:
RESOURCE ERROR: One or more nodes ran out of memory while executing the
query.

Failure trying to allocate initial reservation for Allocator. Attempted
to allocate 5000000 bytes and received an outcome of FAILED_LOCAL.
Fragment 0:0
```

   In the OJAI Distributed Query Service, log files are in the `/opt/mapr/drill/<drill-version>/logs/drillbit.log` on each node where the Query Service is running.

   You should see output like the following:

```
2017-10-07 15:32:41,693 [BitServer-3] INFO
o.a.drill.exec.ops.FragmentContext - User Error Occurred: One or more
nodes ran out of memory while executing the query. (Failure trying to
allocate initial reservation for Allocator. Attempted
org.apache.drill.common.exceptions.UserException: RESOURCE ERROR: One or
more nodes ran out of memory while executing the query.

Failure trying to allocate initial reservation for Allocator. Attempted
to allocate 7000000 bytes and received an outcome of FAILED_LOCAL.
Fragment 1:1
```

2. After confirming, increase the Query Service memory settings by editing the `/opt/mapr/conf/conf.d/warden.drill-bits.conf` file on each Drillbit node. The file contains the following entries:

```
service.env=DRILL_HEAP=3072m,DRILL_MAX_DIRECT_MEMORY=1024m,DRILLBIT_CODE_
CACHE_SIZE=512m
service.heapsize.min=4608
service.heapsize.max=4608
```

   Perform the following steps on **each** Drillbit node:

   a) Modify `DRILL_HEAP` and `DRILL_MAX_DIRECT_MEMORY` in the `service.env` entry based on your requirements.

   b) Update `service.heapsize.min` and `service.heapsize.max` to reflect the updates you made. The numbers sum to the 3 memory settings in the `service.env` entry.

   c) Restart warden on the node by running the following command:

```
service mapr-warden restart
```

Listing Secondary Indexes

Describes how to list information about the secondary indexes created on MapR Database JSON tables.

You can view secondary indexes using the Control System or the `maprcli table index` commands. You need the following permissions.

- `readAce` on the volume
- `lookupdir` on directories in the table path

Note: The mapr user is not treated as a superuser. MapR Database does not allow the `mapr` user to perform this operation unless that user is given the relevant permission or permissions with access-control expressions.

Listing Indexes in the Control System

- Log in to the Control System and go to the **Indexes** tab in the table information page.
  The list of indexes displays in the **All indexes** pane and for each index, the page displays the following:

Column Name	Column Description
Index Name	The name of the index
Fields Indexed	The number of fields on the JSON table that are indexed and used for ordering
Fields Covered	The number of fields on the JSON table that are indexed, but not used for ordering
State	The replication state of the index
Up to Date	Whether the index is up to date
Hashed	Whether the index is hashed
Size	The size of the index

To view more details on individual indexes, see Viewing Secondary Index Details on page 1105.

Listing Indexes Using the CLI

The following is basic command for listing secondary indexes.

```
maprcli table index list
            -path <path>
            -refreshnow < true | false >
```

See `table index list` on page 1834 for more information.

Viewing Secondary Index Details

Describes how to use the Control System to view more specific details on secondary indexes.

You can view secondary index details using the Control System.

1. Go to the **Indexes** tab in the table information page for the JSON table.
2. Click the name of the index to display the details.
   The page displays **Summary** and **Metrics** tabs.

Summary	The **Summary** tab displays the following:	
	Throughput - By Op Type	See Viewing Throughput by Operation Type Using the Control System on page 1303
	Region Distribution	See Viewing Region Distribution on page 1306.
	SETTINGS AND AUDITING	Displays the index name, whether or not the index is hashed and if hashed, the number of hashed partitions, the status of the index and if the index is current, and the number of bytes, puts, and buckets that are yet to be replicated to the index.
	INCLUDED FIELDS	The fields in the JSON table that are indexed, but not used for ordering.
	FIELDS INDEXED	The fields in the JSON table that are indexed and used for ordering.
Metrics	The **Summary** tab displays charts for the secondary index metrics. For more information, see Viewing Secondary Index Metrics on page 1305.	

Removing Secondary Indexes on JSON Tables

Describes how to remove secondary indexes that are no longer needed.

You can remove secondary indexes using the Control System or the `maprcli table index` commands. You need the following permissions.

- `readAce` on the volume
- `lookupdir` on directories in the table path

last-updated: Apr 11, 2022

- `indexperm` permission on the table, if you did not create the table

Note: The mapr user is not treated as a superuser. MapR Database does not allow the `mapr` user to perform this operation unless that user is given the relevant permission or permissions with access-control expressions.

Removing Indexes Using the Control System

1. Log in to the Control System and go to the **Indexes** tab in the table information page.
2. Select the indexes to remove and click **Remove Index**.
   The **Remove Index** confirmation window displays.
3. Review the index(es) to remove and click **Remove Index**.

Removing Indexes Using the CLI

The following is the basic command for removing a secondary index on a JSON table.

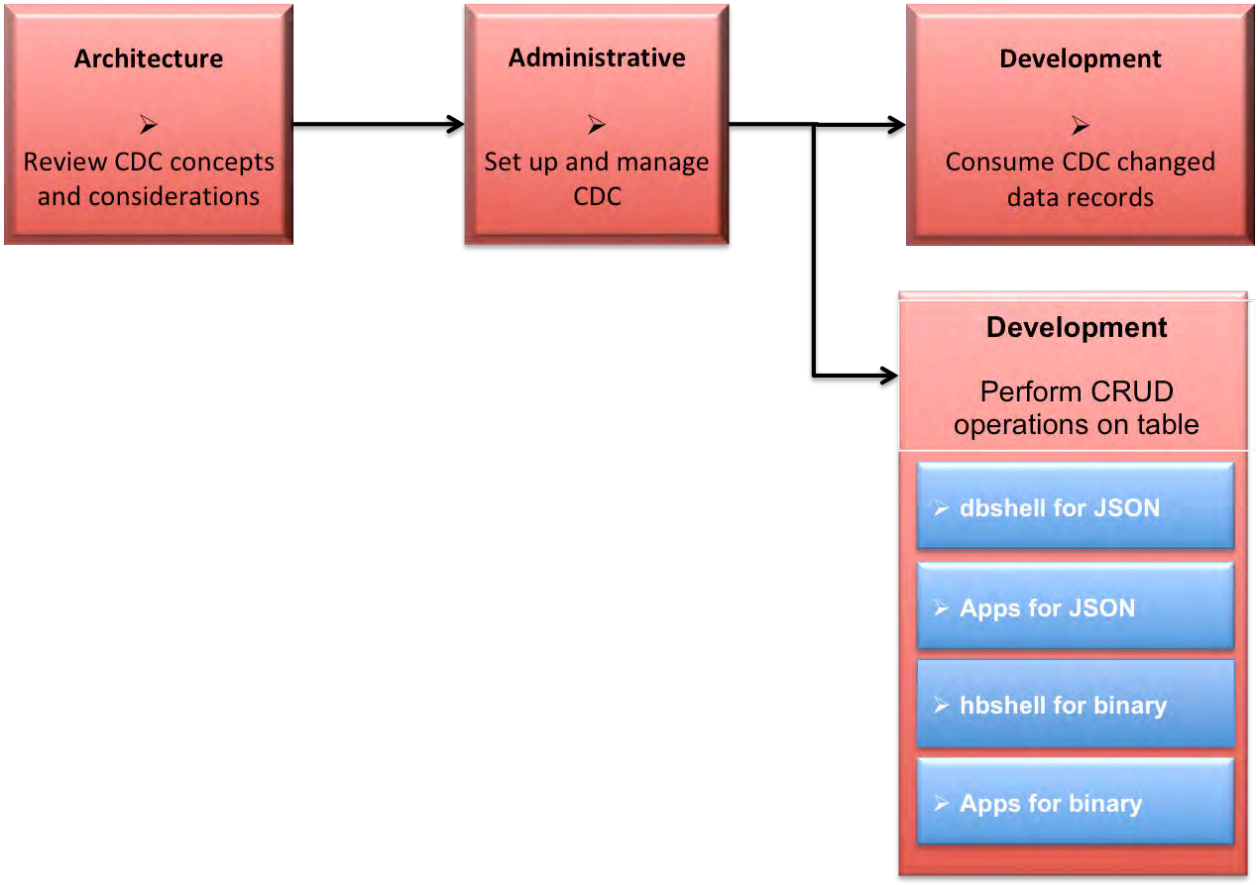
```
maprcli table index remove
  -path <path>
  -index <index name>
```

See `table index remove` on page 1837 for more information.

Administering Change Data Capture

This topic covers the Control System and `maprcli` tools for managing the Change Data Capture (CDC) feature.

The following topics provide information you need to understand the CDC feature, to setup and use CDC and the `maprcli` commands used to perform tasks.



1. Learning about CDC
2. Setting up the CDC environment
3. Consuming CDC changed data records
4. Using dbshell to perform CRUD operations on MapR Database JSON tables
5. Developing client applications for MapR Database JSON tables.
6. Using hbshell to perform CRUD operations on MapR Database binary tables.
7. Developing client applications for MapR Database binary tables.

Additional Information

- table changelog on page 1813: The `maprcli table changelog` commands for managing the changelog relationship between the source table and the destination stream topic.

Setting Up CDC

To set up the Change Data Capture (CDC) feature, the following must exist or be created: a MapR Database source table (JSON or binary), a MapR Event Store For Apache Kafka changelog stream, a MapR Event Store For Apache Kafka stream topic, and a MapR Database table changelog relationship between the source table and the destination stream topic.

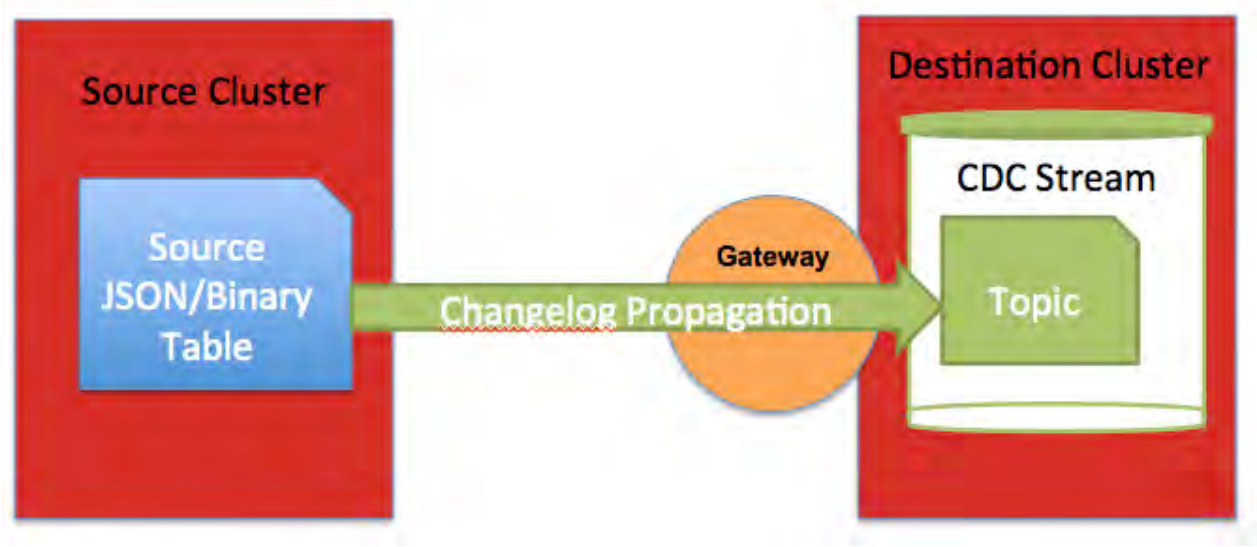
Before Setting Up CDC

The destination MapR Event Store For Apache Kafka stream can be in same cluster as the MapR Database source table or it can be on a remote MapR cluster. If you are propagating changed data from

a source table on a source cluster to a destination stream topic on a remote destination cluster, a gateway must be setup. Gateways are setup by installing the gateway on the destination cluster and specifying the gateway node(s) on the source cluster. See Administering MapR Gateways on page 1150 and Configuring Gateways for Table and Stream Replication on page 1152.

The following diagram shows a simple CDC data model, with one source table to one destination topic on one stream. Because this scenario has the destination stream topic on a remote destination cluster, a gateway must be setup and configured.

Note: More complex CDC scenarios can be implemented and multiple gateways can be setup. See Data Modeling and CDC on page 604 for more information about CDC data models.



Important: If you have a secure cluster, secure configuration must be setup. See Configuring Secure Clusters for Cross-Cluster Mirroring and Replication on page 1486.

Create Table

A MapR Database table (JSON or binary) must be established for the CDC data. You can create a new table and add data or use an existing table with data. See maprcli table create for creating a new table or use the Control System. Example code is provided for completing this task using either the maprcli or REST. Alternatively, depending on whether you are establishing JSON documents or binary files, you can use the following:

- mapr dbshell for MapR Database JSON documents
- hbshell for MapR Database binary data

Attention: Ensure that a volume exists and mounted for both tables and streams. Even though MapR Database tables and MapR Event Store For Apache Kafka stream can exist in the same volume, for organizational purposes, you could create separate volumes for both tables and streams.

The following code examples show how to:

- Create and mount a volume for a source table.
- Create a new binary table. The -tabletype parameter's default setting is binary so you don't need to specify this parameter.
- Create a new JSON table.

last-updated: Apr 11, 2022

CLI

```
// Create Volume for table
maprcli volume create -name
tableVolume -path /tableVolume

// Create Binary table
maprcli table create -path /
tableVolume/cdcTable

// Create JSON table
maprcli table create -path /
tableVolume/cdcTable -tabletype json
```

REST

```
// Create Volume for table
https://10.10.100.17:8443/rest/
volume/create?name=tableVolume&path=/
tableVolume

// Create Binary table
https://10.10.100.17:8443/rest/table/
create?path=/tableVolume/cdcTable

// Create JSON table
https://10.10.100.17:8443/rest/
table/create?path=/tableVolume/
cdcTable&tabletype=json
```

Create Stream

A MapR Event Store For Apache Kafka changelog stream must be created for the propagated changed data records using the `maprcli stream create -ischangelog` parameter. See maprcli stream create or use the Control System.

Note: Ensure that a volume exists and mounted for both tables and streams. Even though MapR Database tables and MapR Event Store For Apache Kafka stream can exist in the same volume, for organizational purposes, you could create separate volumes for tables and streams.

Important: The changelog stream's default partitions can impact how many partitions a stream topic can have. This is because once you create a stream topic for a changelog stream, the number of topic partitions is *locked*. The number of topic partitions cannot change.

- If the `stream topic create` command is used to create a stream topic, then the number of topic partitions can be set at creation time and then is *locked*.
- If the `table changelog add` command is used to add a stream topic (as well as establish a relationship between the source table and the changelog stream), then the number of topic partitions is inherited from the changelog stream and is *locked*.

The following code examples show how to:

- Create and mount a volume for a changelog stream.
- Create a changelog stream using the default partitions value of one (1).
- Create a changelog stream changing the default partitions to three (3).

last-updated: Apr 11, 2022

CLI

```
// Create Volume for stream
maprcli volume create -name
streamVolume -path /streamVolume

// Create stream (default partitions:
1)
maprcli
stream create -path /streamVolume/
changelogStream -ischangelog true

// Create stream  (default
partitions: 3)
maprcli
stream create -path /streamVolume/
changelogStream -ischangelog
true -defaultpartitions 3
```

REST

```
// Create Volume for stream
https://10.10.100.17:8443/rest/volume/
create?name=streamVolume&path=/
streamVolume

// Create stream (default partitions:
1)
https://10.10.100.17:8443/rest/stream/
create?path=/streamVolume/
changelogStream&ischangelog=true

// Create stream (default partitions:
3)
https://10.10.100.17:8443/rest/stream/
create?path=/streamVolume/
changelogStream&ischangelog=true&defau
ltpartitions=3
```

Create Topic

A MapR Event Store For Apache Kafka stream topic must be created for the changed data records. This can be accomplished in a variety of ways:

- Use the maprcli table changelog add command. This command establishes a changelog relationsip between the source table and the destination stream topic.
- Use the maprcli stream topic create command. I
- Use the REST equivalent of the above maprcli commands.
- Use the Control System.

Important: Once a changelog relationship is established between the source table and the destination stream topic, the number of topic partitions is *locked*. (The `maprcli table changelog add` command is used to establish the changelog relationship.) The `stream topic edit` command can not be used to modify the topic's number of partitions.

The following describes when to create a stream topic a specific way.

- If the changelog stream's default partitions are acceptable for the stream topic (because the topic inherits the stream's default partitions), you can either:

last-updated: Apr 11, 2022

  - Go directly to adding the changelog relationship with the `maprcli table changelog add` command and create the topic there.
  - Create the topic with the `stream topic create` command and *not* specify the `-partitions` parameter.
- If you want to change the topic's partitions, create the topic with the `stream topic create` command and set the `-partitions` parameter.
- If you use the Control System, either of the above methods are available.

The following code examples show how to create a stream topic and change the default partition to five (5).

CLI

```
// Create topic (default partitions: 5
maprcli stream topic create -path /
streamVolume/changelogStream -topic
cdcTopic1 -partitions 5
```

REST

```
// Create topic (default partitions: 5
https://10.10.100.17:8443/rest/stream/
topic/create?path=/streamVolume/
changelogStream&topic=cdcTopic1&partit
ions=5
```

Add Changelog

A table changelog relationship must be added between the source table and the destination stream topic by using the maprcli table changelog add command or the Control System. By adding a table changelog relationship, you are creating an environment that propagates changed data records from a source table to a MapR Event Store For Apache Kafka stream topic.

- If you are creating a changelog relationship and the stream topic does not exist, you specify the stream path and topic.
- If you are creating a changelog relationship and the stream topic *does* exist, specify the stream path and topic plus the `-useexistingtopic` parameter. The `-useexistingtopic` parameter can only be used with a changelog stream's newly created topic or a previous changelog stream topic *for the same source table*.

Note: Propagation of existing table data is enabled by default. If you do *not* want to propagate existing source table data, set the `-propagateexistingdata` parameter to **false**. The default is true.

Note: Propagation is enabled as soon as the table changelog relationship is added. If you do *not* want propagation to begin, set the `-pause` parameter to **true**. The change data records are stored in a bucket until you resume the changelog relationship; at this point, the stored change data records are propagated to the stream topic. See table changelog resume on page 1820 for more information.

The following code examples show how to:

- Create a changelog relationship between the source table and the destination stream topic, where the stream topic *does not* exist.
- Create a changelog relationship between the source table and the destination stream topic, where the stream topic *does* exist.

last-updated: Apr 11, 2022

CLI	`maprcli table changelog add -path /tableVolume/cdcTable -changelog /streamVolume/changelogStream:cdcTopic1`
	
	`maprcli table changelog add -path /tableVolume/cdcTable -changelog /streamVolume/changelogStream:cdcTopic1 -useexistingtopic true`

REST	`https://10.10.100.17:8443/rest/table/changelog/add?path=/tableVolume/cdcTable&changelog=/streamVolume/changelogStream:cdcTopic1`
	
	`https://10.10.100.17:8443/rest/table/changelog/add?path=/tableVolume/cdcTable&changelog=/streamVolume/changelogStream:cdcTopic1&useexistingtopic=true`

The following example verifies that the table changelog relationship exists:

```
maprcli table changelog list -path /tableVolume/cdcTable
```

What's Next: Modifying and Consuming Data

To have CDC changed data records to consume, you need to perform inserts, updates, and deletes on the MapR Database table data. See CRUD operations on documents using mapr dbshell for JSON documents, mapr hbshell for binary data, Java applications for MapR Database JSON, C or Java applications for MapR Database Binary.

A MapR Event Store For Apache Kafka Kafka/OJAI consumer application subscribes to the topic and consumes the change data records. See Consuming CDC Records on page 2723 for more information.

Example: Setting Up CDC with Default Topic Partitions

This example creates the following: a volume for a MapR Database table, a MapR Database JSON table, a MapR Event Store For Apache Kafka changelog stream without changing the default partitions, creates a topic while adding a table changelog relationship from the source table to the destination stream topic and and views the changelog information.

CLI Example

```
// Creating and mounting a volume for the source table
   maprcli volume create -name tableVolume -path /tableVolume

// Creating and mounting a volume for the destination stream
   maprcli volume create -name streamVolume -path /streamVolume

// Creating a new JSON table
   maprcli table create -path /tableVolume/cdcTable -tabletype json

// Creating a stream for CDC data
   maprcli stream create -path /streamVolume/changelogStream -ischangelog
true

// Creating a changelog relationship between the source table and the stream
   maprcli table changelog add -path /tableVolume/cdcTable -changelog /
```

```
streamVolume/changelogStream:cdcTopic1

// Viewing the changelog information
   maprcli table changelog info -changelog  /streamVolume/
changelogStream:cdcTopic1 -json
```

REST Example

```
// Creating and mounting a volume for the source table
   https://10.10.100.17:8443/rest/volume/create?name=tableVolume&path=/
tableVolume

// Creating and mounting a volume for the destination stream
   https://10.10.100.17:8443/rest/volume/create?name=streamVolume&path=/
streamVolume

// Creating a stream for CDC data
   https://10.10.100.17:8443/rest/stream/create?path=/streamVolume/
changelogStream&ischangelog=true

// Creating a changelog relationship between the source table and the stream
   https://10.10.100.17:8443/rest/table/changelog/add?path=/tableVolume/
cdcTable&changelog=/streamVolume/changelogStream:cdcTopic1

// Viewing the changelog information
   https://10.10.100.17:8443/rest/table/changelog/info?changelog=/
streamVolume/changelogStream:cdcTopic1
```

Example: Setting Up CDC with Non-default Topic Partitions
This example creates the following: a volume for a MapR Database table, a MapR Database JSON table, a MapR Event Store For Apache Kafka changelog stream with default partitions, a stream topic with custom partitions, a table changelog relationship from the source table to the destination stream topic, and views the changelog information.

CLI Example

```
// Creating and mounting a volume for the source table
   maprcli volume create -name tableVolume -path /tableVolume

// Creating and mounting a volume for the destination stream
   maprcli volume create -name streamVolume -path /streamVolume

// Creating a new JSON table
   maprcli table create -path /tableVolume/cdcTable -tabletype json

// Creating a stream for CDC data
   maprcli stream create -path /streamVolume/changelogStream -ischangelog
true -defaultpartitions 3

// Creating a stream topic that overrides the stream's default partitions
   maprcli stream topic create -path /streamVolume/changelogStream -topic
cdcTopic1 -partitions 5

// Creating a changelog relationship between the source table and the
stream plus using an existing topic that has custom partitions
   maprcli table changelog add -path /tableVolume/cdcTable -changelog /
streamVolume/changelogStream:cdcTopic1  -useexistingtopic true

// Viewing the changelog information
```

```
   maprcli table changelog info -changelog  /streamVolume/
changelogStream:cdcTopic1 -json
```

REST Example

```
// Creating and mounting a volume for the source table
   https://10.10.100.17:8443/rest/volume/create?name=tableVolume&path=/
tableVolume

// Creating and mounting a volume for the destination stream
   https://10.10.100.17:8443/rest/volume/create?name=streamVolume&path=/
streamVolume

// Creating a stream for CDC data
   https://10.10.100.17:8443/rest/stream/create?path=/streamVolume/
changelogStream&ischangelog=true&defaultpartitions=3

// Creating a stream topic that overrids the stream's default partitions
    https://10.10.100.17:8443/rest/stream/topic/create?path=/streamVolume/
changelogStream&topic=cdcTopic1amp;partitions=5

// Creating a changelog relationship between the source table and the
stream plus using an existing topic that has custom partitions
   https://10.10.100.17:8443/
rest/table/changelog/add?path=/tableVolume/cdcTable&changelog=/streamVolume/
changelogStream:cdcTopic1&useexistingtopic=true

// Viewing the changelog information
   https://10.10.100.17:8443/rest/table/changelog/info?changelog=/
streamVolume/changelogStream:cdcTopic1
```

Managing Table Changelogs
Describes how to manage CDC table changelogs through the Control System and maprcli.

Adding a Change Data Log
Explains how to add a change data log using the Control System and the CLI.

To add a change data log, you must have the following permissions:

- **Replication Access** (UI) or `replperm` (CLI) on the source table on the source cluster
- **Topic** (UI) or `topicperm` (CLI) on the destination stream in the destination cluster

If you are a normal user and want to create a changelog between your own MapR Database table and someone else's stream topic, you must have **Topic** (UI) permission or `topicperm` (CLI) on the destination stream.

Adding a Change Data Log Using the Control System

1. Log in to t he Control System and go to the **Change Data Capture** tab in the table information page.
2. Click **Add Change Log** to display the **Add Change Log** page.
3. Set values for the following.

Destination Cluster	(Required) The path to the cluster on which the changelog stream exists. If the destination stream is on a remote secure cluster, then a gateway and secure configuration must first be setup. For more information, see Table Replication on page 610, Administering MapR Gateways on page 1150, and Configuring Secure Clusters for Cross-Cluster Mirroring and Replication on page 1486.
Destination Stream Topic Name	(Required) The target of the changelog stream, specified as `<stream_path>:<topic_name>`, to which all change data records will be published. The stream must exist as a changelog stream or the operation fails.
Publish to Existing Topic	Whether (**Yes**) or not (**No**) to publish to existing topic. If value is **No** and the topic does not already exist, it will be created.
Publish Existing Data	Whether (**Yes**) or not (**No**) to initiate publishing of the existing data to the stream. If value is **No**, only new changes will be propagated.
Defer Publishing	Whether (**Yes**) or not (**No**) to pause propagation after creating the change log.
Throttle	Whether (**Yes**) or not (**No**) the data transfer to the stream for this change log must be throttled.
Synchronously	Whether (**Yes**) or not (**No**) to acknowledge the client writes to the table before the CDC gateway receives the data.
Encrypt on Wire	Whether (**Yes**) or not (**No**) the data transfer between MapR File System and gateway for this change log is encrypted.
Compress on Wire	The compression scheme of the data transfer between MapR File System and gateway for this change log instance.

4. Choose one of the following:
   - For JSON table:
     - **Publish Entire Document** — to publish the entire document to the stream topic.
     - **Publish Selected Field Path** — to specify the paths to the fields to publish to the stream topic.
   - For Binary table:
     - **Publish Entire Document** — to publish the entire document to the stream topic.
     - **Publish Selected Column Families** — to specify the column families to publish to the stream topic.
5. Click **Add Change Log**.

Adding a Change Data Log Using the CLI and REST API

The basic command to add a change data log to a table is:

```
maprcli table changelog add
```

For complete reference, see table changelog add on page 1813.

last-updated: Apr 11, 2022

Viewing the List of Change Logs
Explains how to view the list of change logs using the Control System or the CLI.
Viewing the List of Change Logs Using the Control System

- Log in to the Control System and go to the **Change Data Capture** tab in the table information page.
  For each change log, the page displays the following:

Column Name	Column Description
Edit Change Log	Shortcut to the **Edit Change Log** window for editing a change log.
Paused	Specifies whether the change propagation is paused for the associated change log.
Destination Cluster	Specifies the destination cluster on which the stream exists.
Destination Stream Topic Name	Specifies the name of the stream associated with the change log.
Up to Date	Specifies whether (**Yes**) or not (**No**) the change log is up to date. If value is **No**, hover the cursor over the value to see the number of pending bytes, puts, and buckets.
Errors	Indicates whether there were any errors during change propagation.
Compression Type	The type of compression for data transfer between MapR File System and gateway for the associated change data log instance
Synchronous	Specifies whether client writes to the table should be acknowledged before the CDC gateway receives the data.
Throttle	Specifies whether data transfer to the stream for the associated change data log is throttled.
Encrypted	Specifies whether the data transfer between MapR File System and gateway for the associated change data log is encrypted.
Field Path	Specifies whether only specific field paths are being published to the stream topic.

Retrieving the List of Change Logs Using the CLI or REST API

The basic command to retrieve the list of change data logs is:

```
maprcli table changelog list
```

For complete reference, see table changelog list on page 1816.

Viewing Change Log Information
Explains how to view information about a specific change log using the CLI.
Retrieving Change Log Information Using the CLI or REST API

The basic command to retrieve the list of change data logs is:

```
maprcli table changelog info
```

For complete reference, see table changelog info on page 1815.

Editing a Change Log
Explains how to modify a change log associated with a table using either the Control System or the CLI.

Editing a Change Log Using the Control System

1. Log in to the Control System and go to the **Change Data Capture** tab in the table information page.
2. Click  associated with the change log to modify.
   The **Edit Change Log** window displays.
3. Make the necessary changes to any of the following:

Throttle	Whether (**Yes**) or not (**No**) the data transfer must be throttled.
Synchronously	Whether (**Yes**) or not (**No**) to acknowledge the client writes to the table before the CDC gateway receives the data.
Encrypt on Wire	Whether (**Yes**) or not (**No**) the data transfer is encrypted.
Compress on Wire	The compression scheme of the data propagation.

4. Click **Save Changes** for the changes to take effect.

Editing a Change Log Using the CLI or REST API

The basic command to modify the change log is:

```
maprcli table changelog edit
```

For complete reference, see table changelog edit.

Pausing Data Propagation
Explains how to pause data propagation using either the Control System or the CLI.
Pausing Data Propagation Using the Control System

1. Log in to the Control System and go to the **Change Data Capture** tab in the table information page.
2. Select the change log(s) to pause.
   Selecting the checkbox next to a change log makes the **Actions** drop-down menu available.
3. Select **Pause Data Propagation** from the **Actions** drop-down menu to display the **Pause Data Capture** confirmation dialog.
4. Review the change log(s) to pause and click **Pause Data Capture**.

Pausing Data Propagation Using the CLI or REST API

The basic command to pause change data propagation is:

```
maprcli table changelog pause
```

For complete reference, see table changelog pause.

Resuming Data Propagation
Explains how to resume a paused change log using either the Control System or the CLI.
Resuming Data Propagation Using the Control System

1. Log in to the Control System and go to the **Change Data Capture** tab in the table information page.
2. Select the change log(s) to pause.
   Selecting the checkbox next to a change log makes the **Actions** drop-down menu available.

last-updated: Apr 11, 2022

3. Select **Resume Data Propagation** from the **Actions** drop-down menu to display the **Resume Data Capture** confirmation dialog.
4. Review the change log(s) to resume and click **Resume Data Capture**.

Resuming Data Propagation Using the CLI or REST API

The basic command to resume change data propagation is:

```
maprcli table changelog resume
```

For complete reference, see table changelog resume.

Removing Change Logs
Describes how to remove change logs using either the Control System or the CLI.
Removing Change Logs Using the Control System

1. Log in to the Control System and go to the **Change Data Capture** tab in the table information page.
2. Select the change log(s) to remove.
   Selecting the checkbox next to a change log makes the **Actions** drop-down menu available.
3. Select **Remove Change Logs** from the **Actions** drop-down menu to display the **Remove Change Log(s)** confirmation dialog.
4. Review the change log(s) to remove and click **Remove Change Logs**.

Removing Change Logs Using the CLI or REST API

The basic command to remove a change log is:

```
maprcli table changelog remove
```

For complete reference, see table changelog remove.

Troubleshooting Changelogs

checkandcreate topic failed with error 17

I'm getting a **checkandcreate topic** error while trying to edit a changelog topic.

Because the `maprcli table changelog add` command is an asynchronous command, the CDC relationship (same as replication relationship) involves storing information at both the source and destination sides. This results in the following behavior:

- When the `maprcli table changelog add` operation succeeds, it means that the add request is received. To check whether there is an error during the add operation, run the `maprcli table changelog list` operation.
- The `maprcli table changelog edit` operation only modifies the information on the source side, even if an error is display in the `maprcli table changelog list` output, the changelog can still be modified.

Troubleshooting methods:

- If the stream topic already exists in the destination and you are getting an error, delete the topic. The `maprcli table changelog add` operation automatically retries and finishes.
- If the error can not be fixed, delete the partial relationship from the source side with the `maprcli table changelog remove` operation and retry.

Enabling/Disabling Propagation

Propagation of existing table data is enabled by default. If you do *not* want to propagate existing source table data, set the `-propagateexistingdata` parameter to **false**. The default is true.

Propagation is enabled as soon as the table changelog relationship is added. If you do *not* want propagation to begin, set the `-pause` parameter to **true**. The change data records are stored in a bucket until you resume the changelog relationship; at this point, the stored change data records are propagated to the stream topic. See table changelog resume on page 1820 for more information.

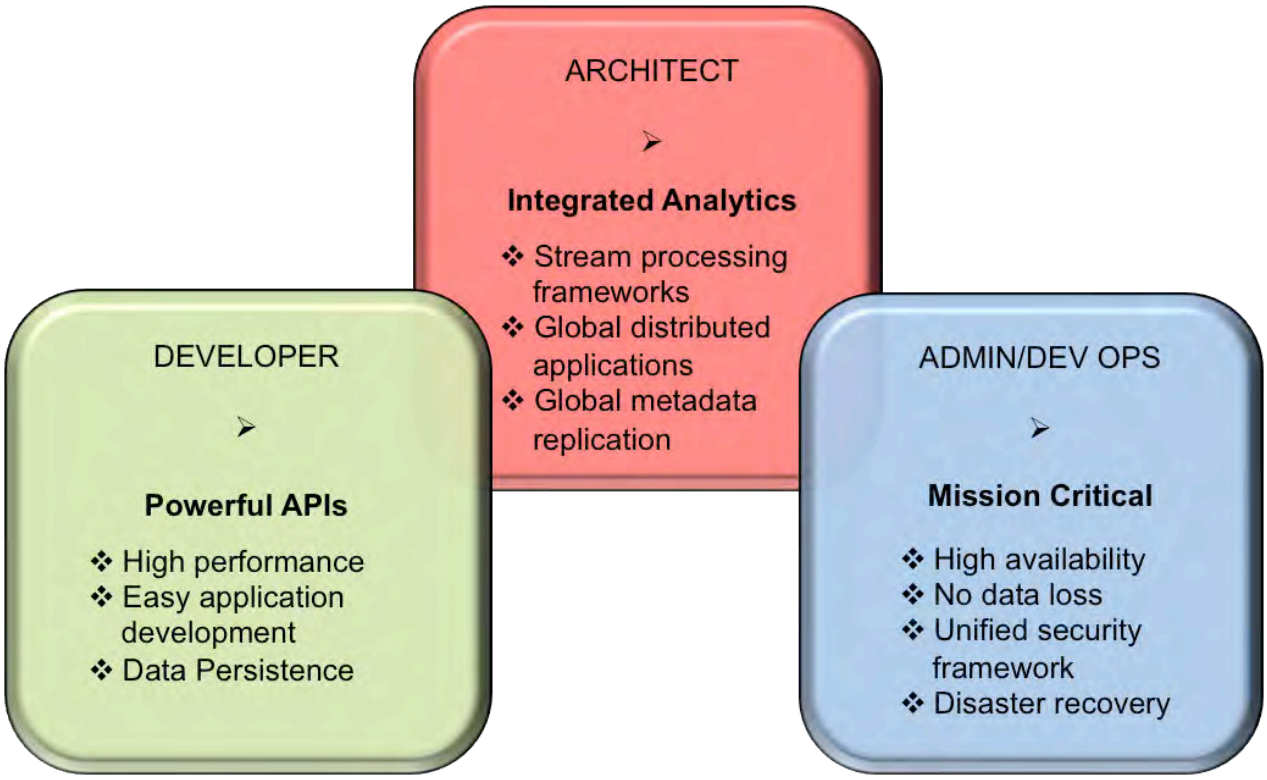
Indexing MapR Database Binary Tables with Elasticsearch

As of with MapR 6.0, MapR Database Elastic Search integration capability is deprecated and no longer available in the MapR Database product.

Attention: MapR Database Change Data Capture (CDC) framework can be used to integrate with latest versions of Elasticsearch. See Change Data Capture on page 597 for more information.

Administering Streams

MapR Event Store For Apache Kafka brings integrated publish and subscribe messaging to the MapR Data Platform. Producer applications can publish messages to topics, which are logical collections of messages, and Consumer applications can read those messages at their own pace. Topics are grouped into streams, for which administrators can apply security, retention, and replication policies.



1. The MapR Event Store For Apache Kafka and Apps section information and examples for developing Producer and Consumer applications.
2. The MapR Event Store For Apache Kafka section provides conceptual information.

last-updated: Apr 11, 2022

3. The Administering Streams section provides information about creating and managing streams, topics, and stream replication.

Managing Streams

This topic provides information about managing streams in MapR Event Store For Apache Kafka.

Creating a Stream

Explains how to create a stream using the Control System and the CLI.

Your decision about what streams to create should take into account whatever topics you want to replicate. Replication is between streams, not individual topics.

For example, suppose that you plan to create the stream `pollution_monitors` to collect various measurements about pollution levels in cities in Europe. However, during a planning session, the representative from Amsterdam says that her country wants to perform analyses of the data for its cities, and would like your company to replicate the data to its own MapR cluster, where its own consumers can read the replicated messages.

You would create a separate stream of topics that contain data from only the pollution sensors in the cities in that country. You might even decide to do the same for each center, in case other centers eventually want to perform their own analyses, too. The streams you might decide to create could be `pollution_monitors_netherlands`, `pollution_monitors_sweden`, and so on.

Creating a Stream Using the Control System

To create a stream:

1. Log in to the Control System and click **Create Stream** under **Data** > **Streams**.

   Note: This option is not available in the Kubernetes version of the Control System.

   The **Create New Stream** page displays.

2. Specify the following properties.

Property	Description
Stream Path	The path and name of the stream to create. The path to the stream can include any character allowed by MapR. For example, `/my/path/with:/to/mystream:topic1` is valid, but `/my/path/with:/to/mystream:withcolon:topic1` is invalid. The name of the stream cannot include a colon (`:`) or a forward slash (`/`).
Time To Live	The amount of time to elapse between the publication of a message in a topic in this stream and the expiration of that message. Choose: • Forever to retain messages indefinitely • Seconds to specify the number of seconds. A value of 0 causes messages to be retained indefinitely. Messages that have expired are deleted during the next purge process. See Time-to-Live for Messages on page 637 for details.

last-updated: Apr 11, 2022

Property	Description
Compression	The compression setting to use for the stream. Producer client libraries can bundle messages that are to be published on the same partition and compress them. The messages are sent to the server compressed, are stored compressed, are replicated to other containers compressed, and (if stream replication is configured) replicated to replica streams compressed. Consumer client libraries receive compressed data, decompresses it, and passes it to client applications. Choose one of the following: • Inherited (to inherit from the directory where the stream is stored), which is the default setting • OFF (to disable compression) • LZF • LZF4 • ZLIB
Auto Create Topics	Whether (**Yes**) or not (**No**) to create a topic automatically when a producer tries to write the first message to it.
Default Partitions	The default number of partitions to allocate to new topics in the stream. When deciding how many partitions to create by default for new topics in a stream, consider the expected volume of messages that will be published to the topics in the stream. High message volumes can be handled more efficiently by multiple consumers in consumer groups reading from multiple partitions than by individual consumers reading from a single partition.  **Note:** You can override the default and specify a different number of partitions for each topic in the stream at the time of creating the topic or after creating the topic.
Use for Change Log	Whether (**Yes**) or not (**No**) to create the stream for changed data records (as a result of inserts, updates, and deletes) in a MapR Database table.

3. Set up access to streams for users, groups, and roles.

   For each user, group, and/or role, you can grant (by selecting the associated check box) or block (by deselecting the associated checkbox) the following types of access:

Administer	Can modify the access-control expressions for the stream, set up replication from the stream, and modify attributes of the stream. This permission includes the topic permission.
Copy Stream	Can copy data from one MapR stream to another MapR stream (using the `mapr copystream` utility) and compare the message IDs, metadata, and data in two MapR streams (using the `mapr diffstreams` utility).
Topic	Can create, edit, or remove topics in the stream.
Producer	Can publish messages to topics in the stream.
Consumer	Can listen to topics in the stream.

last-updated: Apr 11, 2022

By default, all permissions are given to the user creating the stream. To grant or block access to other users, groups, and/or roles, choose one of the following:

- Basic Settings: Select the type — public, (OR) user, group, or role — from the drop-down list and grant read and/or write permissions.

  Tip: Click to create a copy of the associated access control setting. Click to remove the associated access control expression.

  Click **Add Another** to add permissions for another user, group, or role.

- Advanced Settings: Within empty strings (""), specify user (u), group (g), role (r), or public (p) who have and do not have read and/or write access using the following boolean expressions and subexpressions:
  - ! — Negation operator.
  - & — AND operation.
  - | — OR operation.

  Use ( ), parentheses, for subexpressions.

  Alternatively, click associated with the type of access to use the **Create Access Control Expression** window to define access for public or users, group, and/or role.

Note: If you switch from Basic to Advanced, the basic settings, if any, will be carried over to the Advanced settings. If you switch from Advanced to Basic, all the settings will be lost because the subexpressions and AND (&) and negation (!) operations that are supported by Advanced settings are not supported in the Basic settings.

To add access control expressions for another user, group, or role, click **Add Another** and repeat this step.

4. Click **Create Stream** to create the stream.

Creating a Stream Using the CLI or REST API

The basic command to create a stream is:

```
maprcli stream create -path <Stream Path>
```

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path

For complete reference information, see `stream create` on page 1758.

Editing a Stream

Describes how to edit streams using the Control System and the CLI.

Editing a Stream Using the Control System

1. Log in to the Control System and go to the stream information page.
2. Click **Edit Stream**.

   The **Edit Stream** page displays.
3. Make necessary changes to one or more of the following:

last-updated: Apr 11, 2022

Property	Description
Time To Live	The amount of time to elapse between the publication of a message in a topic in this stream and the expiration of that message. Choose: • Forever to retain messages indefinitely • Seconds to specify the number of seconds. A value of 0 causes messages to be retained indefinitely. Messages that have expired are deleted during the next purge process. See Time-to-Live for Messages on page 637 for details.
Compression	The compression setting to use for the stream. Producer client libraries can bundle messages that are to be published on the same partition and compress them. The messages are sent to the server compressed, are stored compressed, are replicated to other containers compressed, and (if stream replication is configured) replicated to replica streams compressed. Consumer client libraries receive compressed data, decompress it, and pass it to client applications. Choose from one of the following compression settings: • Inherited (to inherit from the directory where the stream is stored), which is the default setting • OFF (to disable compression) • LZF • LZF4 • ZLIB
Auto Create Topics	Whether (**Yes**) or not (**No**) to create a topic automatically when a producer tries to write the first message to it.
Default Partitions	The default number of partitions to allocate to new topics in the stream.
Compact	Enable (**Yes**) or disable (**No**) log compaction. If enabled, obsolete records from topics are detected and deleted. By default, this is disabled (**No**).

4. Add, modify, or remove access to streams for users, groups, and roles.

   For each user, group, and/or role, you can grant (by selecting the associated check box) or deny (by deselecting the associated checkbox) the following types of access:

Administer	Can modify the *ACE* for the stream, set up replication from the stream, and modify attributes of the stream. This permission includes the topic permission.
Copy Stream	Can copy data from one MapR stream to another MapR stream (using the `mapr copystream` utility) and compare the message IDs, metadata, and data in two MapR streams (using the `mapr diffstreams` utility).
Topic	Can create, edit, or remove topics in the stream.
Producer	Can publish messages to topics in the stream.
Consumer	Can listen to topics in the stream.

last-updated: Apr 11, 2022

To grant or deny access to users, groups, and/or roles, choose one of the following:

- Basic Settings: Select the type — public, (OR) user, group, or role — from the drop-down list and grant read and/or write permissions.

  Tip: Click to create a copy of the associated *ACE* setting. Click to remove the associated *ACE*.

  Click **Add Another** to add permissions for another user, group, or role.

- Advanced Settings: Within empty strings (""), specify user (u), group (g), role (r), or public (p) who have and do not have read and/or write access using the following boolean expressions and subexpressions:

  - ! — Negation operator.
  - & — AND operation.
  - | — OR operation.

  Use ( ), parentheses, for subexpressions.

  Alternatively, click associated with the type of access to use the **Access Control Expression** window to define access for public or users, group, and/or role. See Defining ACEs on page 1473 for more information.

Note: If you switch from Basic to Advanced, the basic settings, if any, are carried over to the Advanced settings. If you switch from Advanced to Basic, all the settings are lost because the subexpressions, and AND (&) and negation (!) operations that are supported by Advanced settings are not supported in the Basic settings.

To add *ACE*s for another user, group, or role, click **Add Another** and repeat this step.

5. Click **Save Changes** for the changes to take effect.

Editing a Stream Using the CLI or REST API

The basic command to edit a stream is

```
/opt/mapr/bin/maprcli stream edit -path <Stream Path>
```

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminperm` permission on the stream

For complete reference information, see `stream edit` on page 1765.

Encrypting a Stream

Apply an additional layer of security to streams by encrypting them.

To set encryption on a stream:

1. Before encrypting a stream, ensure that wire-level security is enabled for the cluster. See Enabling Wire-level Security on page 1411.

2. Determine whether a directory or stream is encrypted by running the following command:

```
hadoop mfs -ls <path>
```

   Note: Streams inherit the value of the `-setnetworkencryption` setting from the directory in which they are created.

3. If the directory is not encrypted, set the encryption on the streams with the following command:

```
hadoop mfs -setnetworkencryption on <path of stream>
```

Example

Suppose that the streams that you want to encrypt are all in the `/test` directory. You run this command to discover whether the directory is encrypted:

```
# hadoop mfs -lsd /test
Found 1 items
drwxr-xr-x Z U U   - root root          0 2015-09-07 02:37  268435456 /test
           p 2049.43.131260  localhost:5660
```

The second flag U after the permissions indicates that the directory `test` is unencrypted. Because you want to encrypt your stream to enhance data security, you run this command, which encrypts the entire directory:

```
hadoop mfs -setnetworkencryption on /test
```

If you run the `-lsd` command again, you will see that the U is replaced by an E, indicating that the directory is now encrypted:

```
# hadoop mfs -lsd /test
Found 1 items
drwxr-xr-x Z E U   - root root          0 2015-09-07 02:40  268435456 /test
           p 2049.43.131260  localhost:5660
```

Defining ACEs

Indicates how to build access control expressions (ACEs) using the Expression Builder.

To define access control expressions using the **Access Control Expression** builder in the MapR Control System:

1. Choose **All** or **Any** (from the drop-down menu) of the settings to match for access.

   Here:

All	AND (&) operation	Indicates that all of the conditions must be met for public or user, group, and/or role to access the volume.
Any	OR (|) operation	Indicates that any one of the conditions must be met for public or user, group, and/or role to access the volume.

last-updated: Apr 11, 2022

2. Click:

+	To add an expression.
( )	To add a subexpression.
x	To remove an expression or subexpression.

3. Select Public or User, Group, or Role from the drop-down menu and:
   a) Choose **Is** to grant or **Is not** to block access to the user, group, or role.
   b) Enter name of the user, group, or role.
4. Click **Save Changes** to create an Access Control Expression.

Setting Whole Volume ACEs Using the CLI

See Setting Whole Volume ACEs on page 1459.

Setting Table ACEs Using the CLI

See Enabling Table and Stream Authorizations with ACEs on page 1461.

Setting Stream ACEs Using the CLI

See Enabling Table and Stream Authorizations with ACEs on page 1461.

Removing Streams

Explains how to delete a stream using either the Control System or the CLI.

Deleted streams cannot be recovered unless they were replicated before deletion. After a stream is deleted, producers will not be able to publish messages to topics in the stream, and consumers will not be able to read messages from topics in the stream.

Removing a Stream Using the Control System

1. Log in to the Control System and go to the stream information page.
2. Click **Remove Stream** to display the **Remove Stream** confirmation dialog.
3. Confirm the action by clicking **Remove Stream**.

Removing a Stream Using the CLI or REST API

The command to delete a stream is:

```
stream delete -path <Stream Path>
```

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path

For complete reference, see stream delete on page 1764.

Viewing a List of Streams

Describes how to view the list of streams using the Control System.

Listing Streams in a Volume

1. Log in to the Control System and click **Data** > **Streams** to view all the volumes that you have access to.

   Note: This option is not available in the Kubernetes version of the Control System.

For each volume, the pane displays the following:

Column Name	Column Description
Name	The name of the volume.
Type	The type. Value can be: • — Volume • — Directory • — Stream
Owner	The name of the owner.
Last Modified	The last modification date and timestamp.

2. Click the name of the volume (to browse to the path to the stream) or enter the name of the volume in the text field.

   The streams in the selected volume display. If necessary, click to return to **All** volumes view.

Listing Streams by Stream Path

1. Log in to the Control System and click **Data** > **Streams**.

   Note: This option is not available in the Kubernetes version of the Control System.

2. Enter the path to the stream in the search field and click **GO**.

Viewing Stream Information

Explains how to view stream information including stream properties, topics, and replication settings using the Control System and the CLI.

Viewing Stream Information Using the Control System

To view stream information:

1. Log in to the Control System and click **Data** > **Streams**.

   Note: This option is not available in the Kubernetes version of the Control System.

2. Locate the stream (by searching or browsing the volumes or by entering the full path to the stream) and click on the stream name.

   The stream information page displays the following tabs:

   - Summary
   - Topics
   - Replication

   You can:

   - Modify the stream
   - Remove the stream

   The **Summary** tab displays:

   - The active and recent alarms in the **Alarms** pane.

last-updated: Apr 11, 2022

- The stream settings and permissions in the **Detail** pane.

Retrieving Stream Information Using the CLI or REST API

The basic command to retrieve stream information is:

```
maprcli stream info -path <Stream Path>
```

To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the path
`adminperm`	When a user with this permission runs the command, the output includes the access-control expressions for the `adminperm` and `topicperm` permissions.
`produceperm`, `consumeperm`, or `topicperm`	When a user with one of these permissions runs the command, the output does not include any access-control expressions.

For complete reference information, see `stream info` on page 1768.

Managing Topics

Topics are logical collections of messages. The following sections describe how to create and manage topics.

Adding a Topic to a Stream
Explains how to add a topic to a stream using either the Control System or the CLI.
Adding a Topic to a Stream Using the Control System

1. Log in to the Control System and go to the **Topics** tab in the stream information page.
2. Click **Add Topic** in the **All** topics pane.
3. Specify the name of the topic in the **Topic Name** field.
   A name can include alphanumeric characters and the following characters: `.` (dot), _ (underscore), and `-` (hyphen).
4. Specify the number of partitions to use for the topic.
   After you create the topic, you can increase the number of partitions, but you cannot reduce the number. If the topic is associated with a stream for change log, you cannot modify the partitions after you create the topic.
5. Click **Add Topic** to create the topic.

Adding a Topic to a Stream Using the CLI or the REST API

The basic command to create a topic is:

```
maprcli stream topic create -path <Stream Path> -topic <Topic Name>
```

For complete reference information, see `stream topic create` on page 1781.

Removing Topics in a Stream
Describes how to delete one or more topics from the stream using either the Control System or the CLI.

Consumers do not have to stop consuming from a topic before the topic is deleted.

Removing Topics in a Stream Using the Control System

1. Log in to the Control System and the go to the **Topics** tab in the stream information page.
2. Select the topics to remove in the **All** topics pane and click **Remove Topic(s)**.
   The **Remove Topic(s)** confirmation window displays.
3. Verify the list of topics and click **Remove Topic**.
   The topic and the messages are immediately deleted.

Removing Topics in a Stream Using the CLI or REST API

The command to delete a topic is:

```
maprcli stream topic delete -path <Stream Path> -topic <Topic Name>
```

For complete reference information, see `stream topic delete` on page 1783.

Viewing the List of Topics in a Stream

Explains how to view the list of topics in a stream using the Control System and the CLI.

Viewing the List of Topics in a Stream Using the Control System

- Log in to the Control System and go to the **Topics** tab in the stream information page.
  The All topics pane displays the list of topics in the stream and for each topic, the pane displays the following:

Column Name	Column Description
Topic Name	The name of the topic.
Maximum Lag	The consumer lag time (in milliseconds).
Partitions	The number of partitions in the topic.
Consumers	The number of consumers for the topic.
Physical Size	The physical size (in MB) of the topic.

  You can view a topic from the list of topics by entering the topic name in the search field. You can also:

  - Add a topic to the stream.
  - Remove one or more topics.
  - Modify the number of partitions for a topic.

Viewing the List of Topics in a Stream Using the CLI or REST API

The command to view the list of topics in a stream is:

```
stream topic list -path <Stream Path>
```

For complete reference information, see `stream topic list` on page 1787.

Modifying Topic Partitions

Explains how to modify the number of partitions using the Control System and the CLI.

Note: In general, you can increase the number of partitions, but you cannot reduce the number. However, if the topic is associated with a stream for change logs, you *cannot* modify the number of partitions.

last-updated: Apr 11, 2022

Modifying Topic Partitions Using the Control System

1. Log in to the Control System and go to the **Topics** tab in the stream information page.
2. Click in the **Partitions** column for the topic.
   The **Edit Partition** window displays.
3. Modify the number of partitions and click **Save Changes** for the changes to take effect.

Modifying Topic Partitions Using the CLI or REST API

The command to modify topic partitions is:

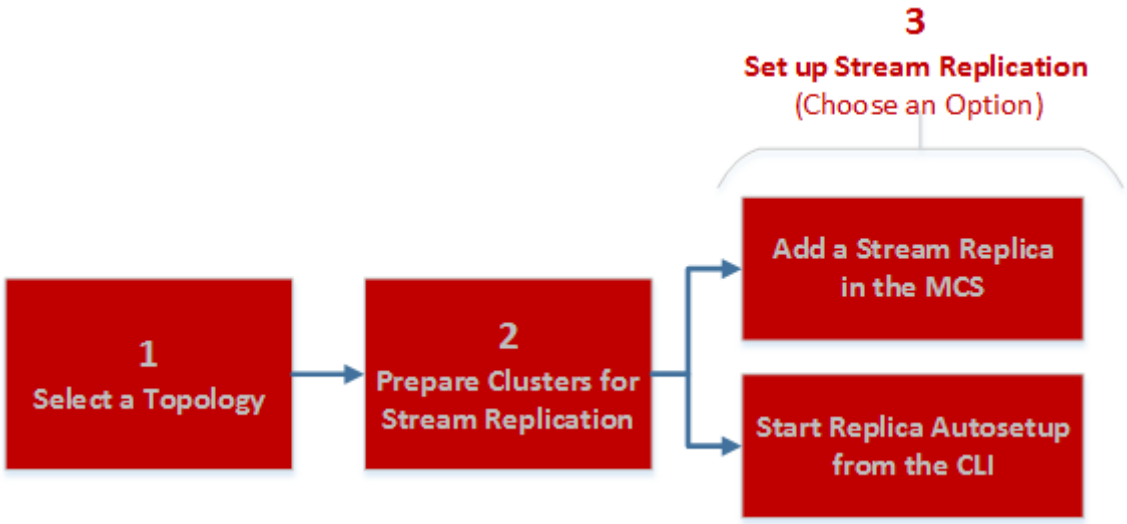
```
stream topic edit -path <stream path> -topic <topic name> -partitions
<number of partitions>
```

For complete reference information, see `stream topic edit` on page 1784.

Managing Stream Replication

This section contains topics about setting up stream replication and administering existing replicas.

The process to set up stream replication consists of 3



steps:

1. Stream Replication on page 656
2. Preparing Clusters for Stream Replication on page 1130
3. Adding Stream Replicas on page 1131
4. Setting Up Stream Replication Using the CLI on page 1133

After you set up replication, you can administer replicas using the Control System or the CLI. To view the replication status, run stream replica list on page 1775.

Preparing Clusters for Stream Replication

Configuring clusters for participation in the replication of MapR Event Store For Apache Kafka streams involves configuring two or more gateways on destination clusters and, if the clusters are secure, setting up secure communications between the clusters.

- Plan which replication topology you want to use: basic primary-secondary, multi-master, or a combination of these. For more information about replication topologies, see Stream Replication on page 656.

last-updated: Apr 11, 2022

- Ensure that you have administrative authority on the clusters that you plan to use.
- Replicating streams requires the installation of gateways. For more information about installation requirements, see Service Layout Guidelines for Replication on page 116

To configure clusters for replication between streams:

1. In the mapr-clusters.conf file on every node in your source cluster, add an entry that lists the CLDB nodes that are in the destination cluster.

   Note:

   This step is required so that the source cluster can communicate directly with the destination cluster's CLDB nodes. See mapr-clusters.conf on page 2200 for the format to use for the entries.

2. Configure gateways on the destination clusters.

   See Configuring Gateways for Table and Stream Replication.

3. **For secure clusters:** Optionally, configure source clusters so that you can locally run maprcli commands that are executed on the destination cluster.

   See Configuring Secure Clusters for Running Commands Remotely.

4. **For secure clusters:** Add one cross-cluster ticket to each source cluster for each cluster that it replicates to.

   See Configuring Secure Clusters for Cross-Cluster Mirroring and Replication.

5. Ensure that the user ID of the person who starts the replication process has the `readAce` permission on the volume where the source streams are located and the `writeAce` permission on the volumes where the replica streams are located. For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

Adding Stream Replicas

You can create stream replicas using the Control System and the CLI. When you add a replica using the Control System, you can also set up and start replication between a source and replica stream.

Before creating a replica:

1. Review the following:
   - Modes of Stream Replication on page 659
   - Security for Stream Replication on page 662
   - Preparing Clusters for Stream Replication on page 1130

Note:

- You must replicate all of the topics that are in a stream. You cannot select only a subset of topics to replicate.
- The maximum number of replicas that a stream can replicate to is 64.
- The maximum number of upstream sources that a replica can accept data from is 64.
- In
  - Multi-master replication, names of topics must be unique on all streams. Messages are assigned sequential offsets. The offsets for messages in a topic in one copy could conflict with the offsets for messages in the other copy. As a result, messages could be lost.

last-updated: Apr 11, 2022

- Many-to-one replication, topics with the same name should not be replicated to an aggregate replica.

Adding a Replica Using the Control System

To add a replica using the Control System:

1. Log in to the Control System and go to the **Replication** tab in the Viewing Stream Information on page 1127.
2. Click **Add Replica**.
   The **Add Replica** page displays.
3. Specify the following settings.

Path to Source Stream	The path and name of the stream that you want to create a replica for.
Destination Cluster	The destination cluster for the replica, where gateways are configured to allow source cluster to send updates.
Path to Replica	The path and name of the replica stream.
Replication State	Specify whether or not to start replication by choosing one of the following: • **Automatic Setup** — Creates the stream on the destination cluster, registers the stream on the destination cluster as a replica, adds the current stream as an upstream source, copies the content of the current stream into the replica, and starts replication. In this case, the replica stream starts empty and accumulates messages over time. • **Pause Replication** — Creates the stream on the destination cluster, registers the stream on the destination cluster as a replica, adds the current stream as an upstream source, but prevents replication from immediately starting after. Pausing replica like this allows you to load the data into the replica from the current stream, after which you can restart replication.  **Note:** If you are interested only in the messages that are published to the source stream after replication starts, then you do not need to pause replication initially. However, if you want the full set of messages from the source stream that have not yet been purged or marked for deletion, then pause replication initially.
Multi-Master Setup	(Available only with **Automatic Setup**) Multi-master topology, in which there are two primary-secondary relationships, with each stream playing both the primary and secondary roles. Client applications update both streams and each stream replicates updates to the other. If this is not selected, stream replication will be basic primary-secondary topology. In this topology, you replicate in one direction. See Stream Replication on page 656 for more information.

4. Select values for the following optional settings.

last-updated: Apr 11, 2022

Throttle	Specifies whether (**Yes**) or not (**No**) to throttle replication operations. Throttle the replication stream to minimize the impact of the replication process on incoming operations during periods of heavy load. By default, throttling is disabled. Throttling has two effects, both of which allow MapR Event Store For Apache Kafka to use more system resources to process new messages: • Throttling slows down the rate at which changes to a stream are replicated. • Throttling slows down the rate at which messages to be replicated are read from disk.
Replicate Synchronously	Specifies whether replication is synchronous (**Yes**) or asynchronous (**No**). The default value is asynchronous replication.
Encrypt On Wire	Specifies whether (**Yes**) or not (**No**) to enable on-wire encryption. By default, this is disabled (**No**). If you enable on-wire encryption, the local cluster and any other cluster that is part of the replication process must be enabled for security.
Compress On Wire	Specifies the type of compression to use when replicating messages.

5. Click **Add Replica** to create the replica.

Adding a Replica Using the CLI or REST API

The basic command to create a replica is:

```
maprcli stream replica add -path <stream path> -replica <remote stream path>
```

To run this command, your user ID must have the following permissions on the:

- Source cluster:
  - `readAce` and `writeAce` on the volume
  - `lookupdir` on directories in the path
  - `adminperm` and `copyperm` permissions on the source stream
- Target cluster:
  - `readAce` and `writeAce` on the volume
  - `lookupdir` on directories in the path

For complete reference, see `stream replica add` on page 1770.

Note: You also have the option to set up replication with `maprcli table replica autosetup` which will set up and start replication. For more information, see Setting Up Stream Replication Using the CLI on page 1133.

Setting Up Stream Replication Using the CLI

Describes how to run the `maprcli stream replica autosetup` command to set up primary-secondary or multi-master replication from an existing source stream.

Note: This procedure describes how to use the maprcli to automatically set up stream replication. As an alternative, you can use the Control System to automatically set up table replication or use the `maprcli` command to manually set up primary-secondary replication.

Before you begin, review the following requirements:

- You must replicate all of the topics that are in a stream. You cannot select only a subset of topics to replicate.

last-updated: Apr 11, 2022

- The maximum number of replicas that a stream can replicate to is 64.
- The maximum number of upstream sources that a replica can accept data from is 64.
- In multi-master replication, names of topics must be unique on all streams. Messages are assigned sequential offsets. The offsets for messages in a topic in one copy could conflict with the offsets for messages in the other copy. As a result, messages could be lost.
- In many-to-one replication, topics with the same name should not be replicated to an aggregate replica.

In general, you should store your streams on their own volumes to avoid overlap with volume mirroring. Otherwise, if a source volume fails, you may have a scenario where a stream in the promoted mirror lags behind the stream's replica. See Preparing Clusters for Table Replication on page 1066 for more details.

Set up replication automatically by following these steps:

1. Log into both the source and destination clusters.
2. Run the command `maprcli stream replica autosetup`:
   - For primary-secondary replication:

     ```
     maprcli stream replica autosetup -path <path to source
     stream> -replica <path to replica stream>
     ```

     For example, to set up primary-secondary replication from the `activity` stream in the `sanfrancisco` cluster to a new `activity` stream in the `newyork` cluster, you could use this command:

     ```
     maprcli stream replica autosetup -path
     /mapr/sanfrancisco/activity -replica /mapr/newyork/activity
     ```

   - For multi-master replication:

     ```
     maprcli stream replica autosetup -path <path to source
     stream> -replica <path to replica stream> -multimaster yes
     ```

     For example, to set up multi-master replication between the `activity` stream in the `sanfrancisco` cluster and a new `activity` stream in the `newyork` cluster, you could use this command:

     ```
     maprcli stream replica autosetup -path /mapr/sanfrancisco/activity
     -replica /mapr/newyork/activity -multimaster yes
     ```

     Note: The parameter `-multimaster` is an optional parameter that you use to set up multi-master replication.

   Note: By default, `maprcli stream replica autosetup` sets up asynchronous replication. If you want to set up synchronous replication or use any of the other optional parameters, see `stream replica autosetup` on page 1771.

3. To check the replication status, run stream replica list on page 1775.

Setting Up Primary-Secondary Stream Replication Manually
Describes how to setup a primary-secondary stream replica that replicates in one direction.

Replica streams can be in a remote MapR cluster or in the MapR cluster where their source streams are located. All updates from a source stream arrive at a replica stream after having been authenticated at a gateway. Therefore, the `produceperm` access control expressions on the replica stream is irrelevant; gateways have the implicit authority to publish messages to topics in replica streams.

To set up primary-secondary replication of streams:

1. Create the replica manually with the `maprcli stream create` command. Use the `-copymetafrom` option to ensure that the metadata for the replica is identical to the metadata for the source stream.

   ```
   maprcli stream create -path <path to replica>
   -copymetafrom <path to source stream>
   ```

   For example, to create the replica `activity` in the `newyork` cluster and use the metadata from the source stream in the `sanfrancisco` cluster, you could use this command:

   ```
   maprcli stream create -path /mapr/newyork/activity
   -copymetafrom /mapr/sanfrancisco/activity
   ```

2. Register the replica as a replica of the source stream by running the `maprcli stream replica add` command.

   ```
   maprcli stream replica add -path <path to source stream>
   -replica <path to replica> -paused true
   ```

   For example, to register the `activity` stream in the `newyork` cluster as a replica of the `activity` stream in the `sanfrancisco` cluster, you could use this command:

   ```
   maprcli stream replica add -path /mapr/sanfrancisco/activity
   -replica /mapr/newyork/activity -paused true
   ```

   The `-paused` parameter ensures that replication does not start immediately after you register the source stream as a source for this replica. You do this registration in step 4.

3. Verify that you specified the correct replica by running the `maprcli stream replica list` command.

   ```
   maprcli stream replica list -path <path to source stream>
   ```

   To verify that the `activity` stream in the `newyork` cluster is a replica of the `activity` stream in the `sanfrancisco` cluster, you could look at the output of this command:

   ```
   maprcli stream replica list -path /mapr/sanfrancisco/activity
   ```

4. Authorize replication between the streams by defining the source stream as the upstream stream for the replica by running the `maprcli stream upstream add` command.

   Definition of the upstream stream ensures that a stream cannot replicate updates to any replica. Replication depends on a two-way agreement between the owners of the two streams.

   ```
   maprcli stream upstream add -path <path to replica> -upstream
   <path to source stream>
   ```

   To add the `activity` stream in the `sanfrancisco` cluster as an upstream source for the `activity` stream in the `newyork` cluster:

   ```
   maprcli stream upstream add -path /mapr/newyork/activity -upstream
   /mapr/sanfrancisco/activity
   ```

5. Verify that you specified the correct source stream by running the `maprcli stream upstream list` command.

   ```
   maprcli stream upstream list -path <path to the replica>
   ```

   To verify this in our example scenario, you could use this command:

   ```
   maprcli stream upstream list -path /mapr/newyork/activity
   ```

6. Load the replica with data from the source stream by using the `mapr copystream` on page 5339 utility.

7. Start replication with the command `maprcli stream replica resume`.

   ```
   maprcli stream replica resume -path <path to the source stream>
   -replica <path to the replica>
   ```

   For our example scenario, you could use this command:

   ```
   maprcli stream replica resume -path mapr/sanfrancisco/activity
   -replica /mapr/newyork/activity
   ```

Viewing the List of Stream Replicas

Explains how to view the list of replicas for a stream using the Control System and the CLI.

Viewing the List of Stream Replicas Using the Control System

- Log in to the Control System and go to the **Replication** tab in the stream information page.

  The **Replicas** pane displays the the list of replicas for the selected stream and for each replica, the pane displays the following:

Column Name	Column Description
Paused	Specifies whether replication is paused.
Destination Cluster & Type	The cluster on which the replica stream resides.
Destination Path	Specifies the name and path of the replica stream.
Up to Date	Specifies whether or not the replica is up-to-date.
Earliest	The date of the oldest message that has yet to be replicated.

last-updated: Apr 11, 2022

Column Name	Column Description
Latest	The date of the newest message that has yet to be replicated.
Errors	Indicates if there were errors during replication.
Compression Type	The type of on-wire compression.
Synchronous	Specifies whether replication is synchronous.
Throttled	Specifies whether replication operations are throttled.
Encrypted	Specifies whether on-wire encryption is enabled.

Selecting the checkbox beside a replica makes the Actions drop-down menu available. You can:

- Add a replica
- Edit a replica
- Pause a replica
- Resume replication
- Un-register replica(s)
- Set current stream as an upstream source

Viewing the List of Stream Replicas Using the CLI or REST API

The command to pause a replication is:

```
maprcli stream replica pause -path <stream path> -replica <remote stream
path>
```

For complete reference information, see `stream replica pause` on page 1777.

Editing a Stream Replica
Explains how to edit a stream replica using the Control System and the CLI to modify the way in which messages are replicated from the source stream to the replica.
Editing a Stream Replica Using the Control System

To un-register one or more stream replicas from the Control System:

1. Log in to the Control System and go to the **Replication** tab in the stream information page.
2. Click  associated with the replica to edit in the **Replicas** pane.
   The **Edit Replica** page displays.
3. Make necessary changes to any of the following properties.

Path to Replica	Specify path and name of the replica stream.
Throttle	Enable (**Yes**) or disable (**No**) throttling of replication operations. Throttle the replication stream to minimize the impact of the replication process on incoming operations during periods of heavy load. By default, throttling is disabled. Throttling has two effects, both of which allow MapR Event Store For Apache Kafka to use more system resources to process new messages: • Throttling slows down the rate at which changes to a stream are replicated.

last-updated: Apr 11, 2022

	• Throttling slows down the rate at which messages to be replicated are read from disk.
Replicate Synchronously	Set up synchronous (**Yes**) or asynchronous (**No**) replication.
Encrypt On Wire	Enable (**Yes**) or disable (**No**) on-wire encryption. By default, this is disabled (**No**). If you enable on-wire encryption, the local cluster and any other cluster that is part of the replication process must be enabled for security.
Compress On Wire	Specify type of compression to use when replicating messages.

4. Click **Save Changes** for the changes to take effect.

Editing a Stream Replica Using the CLI or REST API

The basic command to modify a replica is:

```
stream replica edit -path <stream path> -replica <remote stream path>
```

For complete reference information, see `stream replica edit` on page 1773.

Removing Stream Replicas

Explains how to unregister one or more replicas using the Control System and the CLI.

Removing Stream Replicas Using the Control System

To un-register one or more stream replicas from the Control System:

1. Log in to the Control System and go to the **Replication** tab in the stream information page.
2. Select the replicas to remove in the **Replicas** pane.
   Selecting the checkbox next to a replica makes the **Actions** drop-down menu available.
3. Select **Remove Replica(s)** from the **Actions** drop-down menu.
   The **Remove Replica(s)** confirmation window displays.
4. Verify the list of replicas to remove and click **Remove Replica**.
   This action un-registers the stream as the replica of the source stream.

Removing a Stream Replica Using the CLI or REST API

The command to remove a replica is:

```
stream replica remove -path <stream path> -replica <remote stream path>
```

For complete reference information, see `stream replica remove` on page 1778.

Pausing Stream Replication

Explains how to pause replication from a source stream to a replica stream using the Control System and the CLI.

Pausing Stream Replication Using the Control System

To pause one or more replications:

1. Log in to the Control System and go to the **Replication** tab in the stream information page.
2. Select the replicas to pause in the **Replicas** pane.
   Selecting the checkbox next to a replica makes the **Actions** drop-down menu available.

3. Select **Pause Replication** from the **Actions** drop-down menu.
   The **Pause Replication** confirmation window displays.
4. Verify the list of replicas to pause and click **Pause Replication**.
   This action pauses replication from the source stream to the selected replica stream(s).

Pausing Stream Replication Using the CLI or REST API

The command to pause a replication is:

```
maprcli stream replica pause -path <stream path> -replica <remote stream
path>
```

For complete reference information, see stream replica pause on page 1777.

Resuming Stream Replication

Explains how to resume replication from one stream to another stream using either the Control System or the CLI.

Resuming Stream Replication Using the Control System

To resume one or more replications:

1. Log in to the Control System and go to the **Replication** tab in the stream information page.
2. Select the replicas (in **Paused** state) in the **Replicas** pane.
   Selecting the checkbox next to a replica makes the **Actions** drop-down menu available.
3. Select **Resume Replication** from the **Actions** drop-down menu.
   The **Resume Replication** confirmation window displays.
4. Verify the list of replicas to resume and click **Resume Replication**.
   This action resumes replication from the source stream to the selected replica stream(s).

Resuming Stream Replication Using the CLI or REST API

The command to resume a replication is:

```
maprcli stream replica resume -path <stream path> -replica <remote stream
path>
```

For complete reference information, see stream replica resume on page 1778.

Managing Upstream Sources for Stream Replicas

You can set up a stream to be the upstream source for replicas. This is especially useful if you did not set up replication automatically when setting up replicas.

Set up Stream as Upstream Source

Describes how to set up the current stream as the upstream source for a replica if the replica was not set up to automatically resync with the current stream.

Setting Up Current Stream as Upstream Source for a Replica Using the Control System

To set up a stream as the upstream source for a replica:

1. Go to the **Replication** tab in the stream information page
2. Select the checkbox beside the replica(s) that do not have the current stream configured as upstream source for automatic resync.
   Selecting a checkbox next to a replica makes the **Actions** drop-down menu available.

3. Select **Set Current Stream as Upstream Source** from the **Actions** drop-down menu.
   The **Set Current Stream as Upstream Source** dialog displays.
4. Review the list of selected replicas and click **Set Upstream Source**.
   The current stream will automatically send updates to the replica(s).

Setting Up Stream as Upstream Source for a Replica Using the CLI or REST API

The basic command to set a table as the upstream souce for a replica is:

```
maprcli stream upstream add -path <replica stream path> -upstream <source
stream path>
```

See `stream upstream add` on page 1779 for complete reference information.

Adding Upstream Source for a Stream
Explains how to add an upstream source for a stream using either the Control Panel or the CLI.

You can register a stream as an upstream source for a stream using the Control System and the CLI. When you register a stream as an upstream source, the registered upstream source stream will send updates to the stream.

Adding Upstream Source for a Stream Using the Control System

To register a stream as an upstream source:

1. Log in to the Control System and go the stream information page.
2. Click **Add Upstream Source** in the **Upstream Sources** pane.
   The **Add Upstream Sources** window displays.
3. Specify the path and name of the source stream in the **Upstream Source** field.
4. Click **Add Upstream Source** to register the source stream as an upstream source for this stream.

Adding Upstream Source for a Stream Using the CLI or REST API

The command to add an upstream source is:

```
maprcli stream upstream add -path <stream path> -upstream <upstream stream
path>
```

For complete reference information, see `stream upstream add` on page 1779.

Listing all Upstream Sources for a Stream
Describes how to list all the upstream sources for a stream using the Control System and the CLI.
Listing all Upstream Sources for a Stream Using the Control System

- Log in to the Control System and go to the **Replication** tab in the stream information page.
  The **Upstream Sources** pane displays the list of upstream sources for the selected stream and for each upstream source, the pane displays the following:

Column Name	Column Description
IDX	The index number of the upstream stream.
Upstream Source Cluster	The name of the MapR cluster in which the upstream stream is located.
Upstream Source Path	The path and name of the upstream stream.
UUID	The upstream stream's universally unique identifier.

last-updated: Apr 11, 2022

You can add an upstream source and by selecting the checkbox beside a stream, you can decouple the selected upstream stream.

Listing all Upstream Sources for a Stream Using the CLI or REST API

The command to list all upstream sources for a stream is:

```
maprcli stream upstream list -path <stream path>
```

For complete reference information, see stream upstream list on page 1780.

Removing Upstream Sources for a Stream
Explains how to un-register a stream as an upstream source for a stream using either the Control System or the CLI.

When you remove a stream as an upstream source for a stream, the upstream source stream will stop sending updates to the stream.

Removing Upstream Sources for a Stream Using the Control System

1. Log in to the Control System and go the **Replication** tab in the stream information page.
2. Select the upstream sources to remove in the **Upstream Sources** pane and click **Remove Upstream Source(s)**.
   The **Remove Upstream Source(s)** confirmation window displays.
3. Verify the list and click **Remove Upstream Source**.
   This action un-registers the selected stream(s) as upstream source(s) for this stream.

Removing Upstream Sources for a Stream Using the CLI or REST API

The command to decouple upstream sources is:

```
maprcli stream upstream remove -path <stream path> -upstream <upstream
stream path>
```

For complete reference information, see stream upstream remove on page 1781.

Preparing Clusters for Log Compaction

Describes how to prepare your environment so you can use log compaction.

Installing with the MapR Installer

When you use the MapR Installer to install MapR Event Store For Apache Kafka, a local gateway is also locally installed so that log compaction can be implemented. To configure for log compaction, see maprcli stream create on page 1758 and stream edit on page 1765.

Installing without the MapR Installer

Other sections of the documentation describe the detailed steps for installing and configuring without the MapR installer. Generally, you need to perform the following steps:

1. Install the MapR software.To install MapR without using the MapR installer, follow the steps outlined at Installing without the MapR Installer on page 141.
2. Install the MapR gateway on your local cluster. Since gateways for log compaction are installed on the local cluster, no configuration is needed. See Gateways and Stream Replication on page 662 for general information about gateways.

last-updated: Apr 11, 2022

Adding Gateways for Load Balancing

Since the number of gateways impacts the compaction process, you might want to increase the number of gateways on the cluster to improve the load distribution. To add gateways for log compaction, you install additional MapR gateways on your local cluster. See Gateways and Stream Replication on page 662 for general information about gateways.

Note: No configuration is required because the additional gateways are installed on the local clusster.

For More Information

See the following topics for more information:

- maprcli stream create on page 1758 and stream edit on page 1765
- MapR Event Store For Apache Kafka Java Applications on page 2754, MapR Event Store For Apache Kafka Java API Library on page 2756, and Enabling Log Compaction on page 2765

Mirroring Topics with Apache Kafka MirrorMaker

Use the Apache Kafka MirrorMaker utility either to mirror topics that are in Apache Kafka clusters to streams that are in MapR Data Platform clusters or to Mirror topics that are in MapR Data Platform clusters to Apache Kafka clusters.

Mirroring is a type of replication that takes place in this sequence of steps:

1. Messages that are published to topics in a source cluster are read by consumers that MirrorMaker manages.
2. These consumers send the messages to producers that MirrorMaker also manages.
3. The producers publish the messages in topics that are in the destination cluster.

Mirroring can continue indefinitely. Alternatively, you can mirror your data as a way of migrating it from Apache Kafka to MapR Event Store For Apache Kafka. If you use it for this purpose, you can stop mirroring after migrating your producers and consumers to use MapR Event Store For Apache Kafka, as described in Migrating Apache Kafka 0.9.0 Applications to MapR Event Store For Apache Kafka.

Attention: MirrorMaker does not provide the same reliability guarantees as the replication features in MapR Event Store For Apache Kafka. In particular, MirrorMaker does not replicate cursors or message positions, which makes disaster recovery much more difficult than with replication of MapR Event Store For Apache Kafka. Therefore, MapR Data Platform recommends MirrorMaker for use only for mirroring between MapR Event Store For Apache Kafka and Apache Kafka, not for replication of MapR Event Store For Apache Kafka.

Prerequisites

- Ensure that the destination stream in the MapR Data Platform cluster exists. To create a stream, run the command `maprcli stream create`.
- Ensure that the ID of the user that runs MirrorMaker has the `produceperm` and `topicperm` permissions on the stream.

Command Syntax and Descriptions of Parameters

```
bin/kafka-mirror-maker.sh
--consumer.config <File that lists consumer properties and values>
--num.streams <Number of consumer threads>
--producer.config <File that lists producer properties and values>
[--whitelist=<Java-style regular expression for specifying the topics to
```

```
mirror>]
[--blacklist=<Java-style regular expression for specifying the topics not
to mirror>]
```

Parameter	Description
`consumer.config`	The path and name of the file that lists the consumer properties. See the Consumer Properties and Descriptions on page 1143 section for detailed information.
`num.streams`	Use the --num.streams option to specify the number of mirror consumer threads to create. Note that if you start multiple mirror maker processes then you may want to look at the distribution of partitions on the source cluster. If the number of consumption streams is too high per mirror maker process, then some of the mirroring threads will be idle by virtue of the consumer rebalancing algorithm (if they do not end up owning any partitions for consumption).
`producer.config`	The path and name of the file that lists the producer properties. See the Producer Properties and Descriptions on page 1144 section for detailed information.
`whitelist`	A Java-style regular expression for specifying the topics to copy. Commas (',') are interpreted as the regex-choice symbol ('|'). If you use this parameter, do not use the `blacklist` parameter.
`blacklist`	A Java-style regular expression for specifying the topics not to copy. Commas (',') are interpreted as the regex-choice symbol ('|'). If you use this parameter, do not use the `whitelist` parameter.

Consumer Properties and Descriptions

```
zookeeper.connect=<IP address>:<ZooKeeper port>
zookeeper.connection.timeout.ms=<Timeout value in milliseconds>
group.id=<ID>
bootstrap.servers=<IP address>:<port>
shallow.iterator.enable=false
```

Property	Description
`zookeeper.connect`	The IP address and port number of the ZooKeeper instance for the Apache Kafka cluster.
`zookeeper.connection.timeout.ms`	The max time that the client waits to establish a connection to zookeeper. If not set, the value in `zookeeper.session.timeout.ms` is used.
`group.id`	A unique string that identifies the consumer group this consumer belongs to. This property is required if the consumer uses either the group management functionality by using `subscribe(topic)` or the Kafka-based offset management strategy. If `group.id` is not set and the value of the `num.streams` option is > 1 , messages might go multiple times to a stream.

last-updated: Apr 11, 2022

Property	Description
`bootstrap.servers`	A list of host/port pairs to use for establishing the initial connection to the Kafka cluster. The client will make use of all servers irrespective of which servers are specified here for bootstrapping—this list only impacts the initial hosts used to discover the full set of servers. This list should be in the form `host1:port1,host2:port2,....` Since these servers are just used for the initial connection to discover the full cluster membership (which may change dynamically), this list need not contain the full set of servers (you may want more than one, though, in case a server is down).
`shallow.iterator.enable`	Set this value to `false`.

Producer Properties and Descriptions

```
key.serializer=<serializer class>
value.serializer=<serializer class>
streams.producer.default.stream=<Path and name of the stream to copy the
topics to>
auto.create.topics.enable=true
```

Property	Description
`key.serializer`	The name of the appropriate serialization class in the `org.apache.kafka.common.serialization` package or a class that implements the `Serializer` interface for serializing keys.
`value.serializer`	The class that implements the `Serializer` interface for serializing values.
`streams.producer.default.stream`	Specifies the path and name of stream that the topics will be copied to.
`auto.create.topics.enable`	Enables auto-creation of topics within the stream specifed with the `streams.producer.default.stream` parameter.

Mirroring Topics from an Apache Kafka Cluster to the MapR Cluster
You can use MirrorMaker to mirror data continuously from Apache Kafka clusters to streams in MapR Event Store For Apache Kafka clusters.

- Because this procedure requires that MirrorMaker be run from the MapR Data Platform cluster, ensure that the mapr-kafka package is installed on the node that you choose to run MirrorMaker from.
- Configure the node as a mapr client.
- Ensure that the ID of the user that runs MirrorMaker has the `produceperm` and `topicperm` permissions on the destination stream.

Alternatively, you can stop mirroring after you migrate the consumers and producers for your applications from your Apache Kafka cluster to your data-fabric cluster where the stream is located. See in Migrating Apache Kafka 0.9.0 Applications to MapR Event Store For Apache Kafka for details. After you start MirrorMaker, it launches a configurable number of consumer threads to read topics that are in a Kafka cluster and a number of producers to write the messages from those topics into topics in MapR Event Store For Apache Kafka.

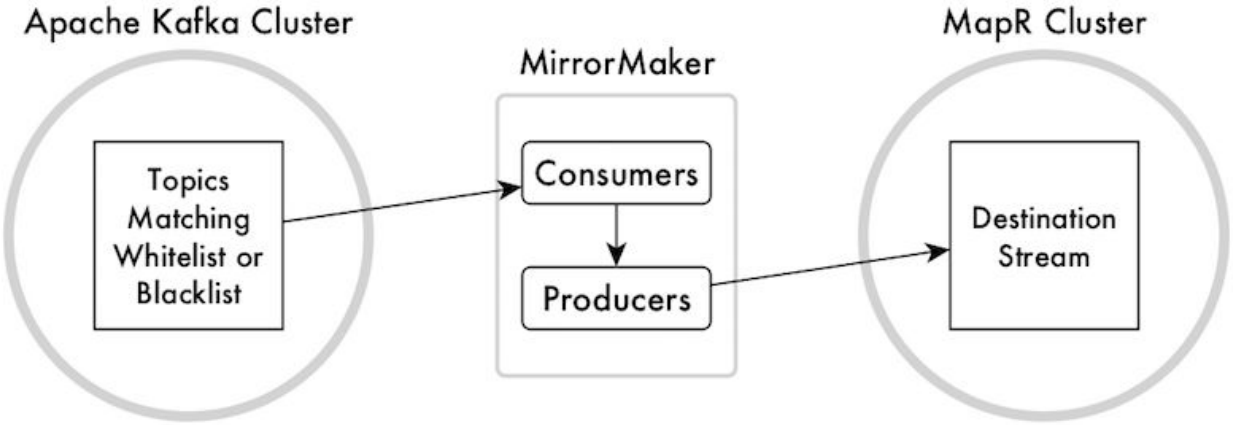


Figure 17: Mirroring from Apache Kafka to MapR Event Store For Apache Kafka

Before running MirrorMaker, you create a file that contains the required configuration parameters for the consumers that read from the Apache Kafka cluster. You also create a file that contains the required configuration parameters for the producers that publish to the stream in the MapR Data Platform cluster. You point to these files in the MirrorMaker command.

You can specify the topics you want to mirror or the topics you do not want to mirror. To mirror topics, use the `whitelist` parameter to provide a Java-style regular expression that matches the names of the topics that you want to mirror. To specify topics you do not want mirrored, use the `blacklist` parameter to provide a Java-style regular expression that matches the names of the topics that you do not want mirrored.

1. Create a file that contains the required properties and values for consumers to use. When you run MirrorMaker, you point to this file by using the `consumer.config` parameter.

   The descriptions of these properties, except for the last, are taken from the documentation for Apache Kafka. The last property is not documented.

Property	Description
`zookeeper.connect`	The IP address and port number of the ZooKeeper instance for the Apache Kafka cluster.
`zookeeper.connection.timeout.ms`	The max time that the MirrorMaker waits to establish a connection to Zookeeper.
`group.id`	A unique string that identifies the consumer group the consumers started by MirrorMaker belong to.
`bootstrap.servers`	A list of host/port pairs to use for establishing the initial connection to the Kafka cluster. The client will make use of all servers irrespective of which servers are specified here for bootstrapping—this list only impacts the initial hosts used to discover the full set of servers. This list should be in the form `host1:port1,host2:port2,...`. Since these servers are just used for the initial connection to discover the full cluster membership (which may change dynamically), this list need not contain the full set of servers (you may want more than one, though, in case a server is down).

2. Create a file that contains the required properties and values for producers to use. When you run MirrorMaker, you point to this file by using the `producer.config` parameter.

Property	Description
`streams.producer.default.stream`	Specifies the path and name of the stream in the MapR Data Platform cluster that the topics will be mirrored to.
`auto.create.topics.enable`	Set the value to `true`. The producers will therefore be able to create topics in the destination stream automatically.

3. Run MirrorMaker with this command to start mirroring topics from Apache Kafka to MapR Event Store For Apache Kafka:

   Syntax

```
/opt/mapr/kafka/kafka-0.9.0/bin/kafka-mirror-maker.sh
--consumer.config <File that lists consumer properties and values>
--num.streams <Number of consumer threads>
--producer.config <File that lists producer properties and values>
[--whitelist=<Java-style regular expression for specifying the topics to
mirror>]
[--blacklist=<Java-style regular expression for specifying the topics
not to mirror>]
```

Parameter	Description
`consumer.config`	The path and name of the file that lists the consumer properties and their values.
`num.streams`	Use this option to specify the number of mirror consumer threads to create. Note that if you start multiple mirror maker processes then you may want to look at the distribution of partitions on the source cluster. If the number of consumption streams is too high per mirror maker process, then some of the mirroring threads will be idle by virtue of the consumer rebalancing algorithm (if they do not end up owning any partitions for consumption).
`producer.config`	The path and name of the file that lists the producer properties and their values.
`whitelist`	A Java-style regular expression for specifying the topics to copy. Commas (',') are interpreted as the regex-choice symbol ('|'). If you use this parameter, do not use the `blacklist` parameter.
`blacklist`	A Java-style regular expression for specifying the topics not to copy. Commas (',') are interpreted as the regex-choice symbol ('|'). If you use this parameter, do not use the `whitelist` parameter.

   Example

   In this example, the file that lists the properties and values for the consumers that will read messages from the topics in Apache Kafka is named `consumers.props`. It contains this list:

```
zookeeper.connection.timeout.ms=6000
group.id=cg1
```


last-updated: Apr 11, 2022

```
bootstrap.servers=10.10.100.87:9093
shallow.iterator.enable=false
```

The file that lists the properties and values for the producers that will publish messages to topics in MapR Event Store For Apache Kafka is named `producers.props`. It contains this list:

```
streams.producer.default.stream=/newStream
auto.create.topics.enable=true
```

The topics to mirror all have names that begin with `na_west`. When running the command, we can use `"na_west.*"` as the regular expression to use for the `whitelist` parameter.

Here is the command:

```
bin/kafka-mirror-maker.sh --consumer.config consumers.props
--num.streams 2 --producer.config producers.props --whitelist="na_west.*"
```

Mirroring Topics from the MapR Cluster to an Apache Kafka Cluster
You can use MirrorMaker to mirror data continuously from streams in MapR Data Platform clusters to Apache Kafka clusters.

- This procedure requires MirrorMaker to run from the MapR Data Platform cluster. Verify that the mapr-kafka package is installed on the node that you choose to run MirrorMaker on.
- Configure the node as a `mapr` client.
- Ensure that the ID of the user who runs MirrorMaker has the `consumeperm` permission on the stream.

After you start MirrorMaker, it launches a configurable number of consumer threads to read topics that are in a stream in a MapR Data Platform cluster and a number of producers to write the messages from those topics into topics in an Apache Kafka cluster.

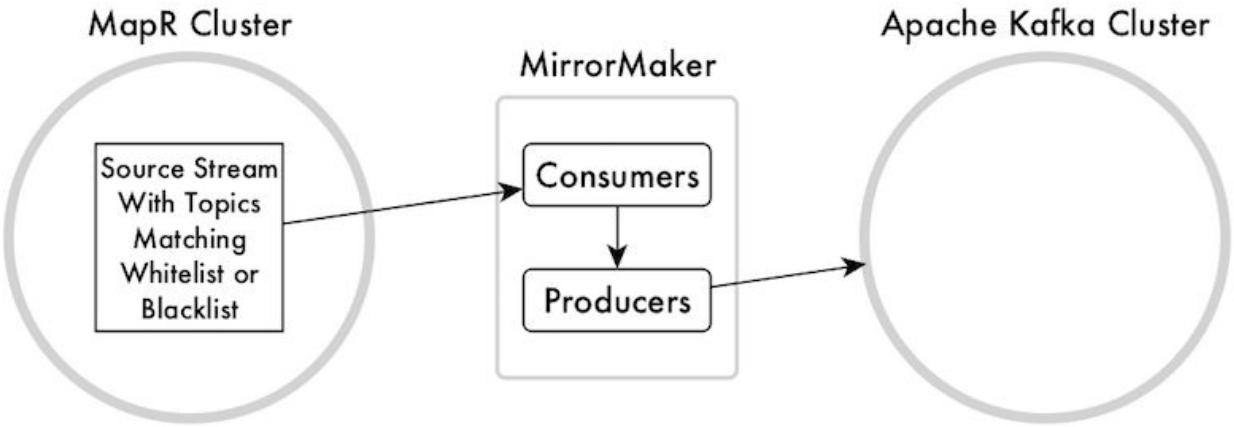


Figure 18: Mirroring from MapR Event Store For Apache Kafka to Apache Kafka

Before running MirrorMaker, you create a file that contains the required configuration parameters for the consumers that read from the stream in the MapR Data Platform cluster. You also create a file that contains the required configuration parameters for the producers that publish to the Apache Kafka cluster. You point to these files in the MirrorMaker command.

You can specify the topics you want to mirror or the topics you do not want to mirror. To mirror topics, use the `whitelist` parameter to provide a Java-style regular expression that matches the names of the topics that you want to mirror. To specify topics you do not want mirrored, use the `blacklist` parameter

last-updated: Apr 11, 2022

to provide a Java-style regular expression that matches the names of the topics that you do not want mirrored.

1. Create a file that contains the required properties and values for consumers to use. When you run MirrorMaker, you point to this file by using the `consumer.config` parameter.

Property	Description
`streams.record.strip.streampath`	Set the value of this property to true. In messages that are written to streams, the names of topics include the paths and names of the streams in which those topics are located. Apache Kafka needs only the names of the topics. This parameter removes the path and name of the stream that the topics will be mirrored from.
`streams.consumer.default.stream`	Specifies the path and name of the stream that the topics will be mirrored from.
`group.id`	A unique string that identifies the consumer group the consumers started by MirroMaker belong to.

2. Create a file that contains the required properties and values for producers to use. When you run MirrorMaker, you point to this file by using the `producer.config` parameter.

Property	Description
`bootstrap.servers`	A list of host/port pairs to use for establishing the initial connection to the Kafka cluster. The producers will make use of all servers irrespective of which servers are specified here for bootstrapping—this list only impacts the initial hosts used to discover the full set of servers. This list should be in the form `host1:port1,host2:port2,...`. Since these servers are just used for the initial connection to discover the full cluster membership (which may change dynamically), this list need not contain the full set of servers (you may want more than one, though, in case a server is down).
`producer.type`	Specifies whether the messages are published asynchronously in batches or as data is received by producers. The values are `async` and `sync`.
`compression.codec`	Specifies the compression codec for all messages that are generated by producers. The possible values are `none`, `gzip`, `snappy`, and `lz4`.

3. Run MirrorMaker with this command to start mirroring topics from MapR Event Store For Apache Kafka to Apache Kafka:

   Syntax

```
bin/kafka-mirror-maker.sh

--consumer.config <File that lists consumer properties and values>
--num.streams <Number of consumer threads>
--producer.config <File that lists producer properties and values>
[--whitelist=<Java-style regular expression for specifying the topics to
mirror>]
[--blacklist=<Java-style regular expression for specifying the topics
not to mirror>]
```

Parameter	Description
`consumer.config`	The path and name of the file that lists the consumer properties and their values.
`new.consumer`	Specifies to use consumers that use the Apache Kafka 0.90 API library.
`num.streams`	Use this parameter to specify the number of mirror consumer threads to create. Note that if you start multiple mirror maker processes then you may want to look at the distribution of partitions on the source cluster. If the number of consumption streams is too high per mirror maker process, then some of the mirroring threads will be idle by virtue of the consumer rebalancing algorithm (if they do not end up owning any partitions for consumption).
`producer.config`	The path and name of the file that lists the producer properties and their values.
`whitelist`	A Java-style regular expression for specifying the topics to copy. Commas (',') are interpreted as the regex-choice symbol ('|'). If you use this parameter, do not use the `blacklist` parameter.
`blacklist`	A Java-style regular expression for specifying the topics not to copy. Commas (',') are interpreted as the regex-choice symbol ('|'). If you use this parameter, do not use the `whitelist` parameter.

Example

In this example, the file that lists the properties and values for the consumer that will read messages from the topics in MapR Event Store For Apache Kafka is named `consumers.props`. It contains this list:

```
streams.record.strip.streampath=true
streams.consumer.default.stream=/myStream
group.id=cg1
```

The file that lists the properties and values for the producers that will publish messages to topics in Apache Kafka is named `producers.props`. It contains this list:

```
bootstrap.servers =10.10.83.93:9092
producer.type=sync
compression.codec=none
```

The topics to mirror all have names that begin with `na_west`. When running the command, we can use `"na_west.*"` as the regular expression to use for the whitelist parameter.

```
bin/kafka-mirror-maker.sh --new.consumer
--consumer.config consumers.props --num.streams 2 --producer.config
producers.props
--whitelist="na_west.*"
```


last-updated: Apr 11, 2022

Administering MapR Gateways

A MapR gateway mediates one-way communication between a source MapR cluster and a destination cluster. You can replicate MapR Database tables (binary and JSON) and MapR Event Store For Apache Kafka streams. MapR gateways also apply updates from JSON tables to their secondary indexes and propagate Change Data Capture (CDC) logs.

The initial task for setting up your gateways is to decide where you want to put them:

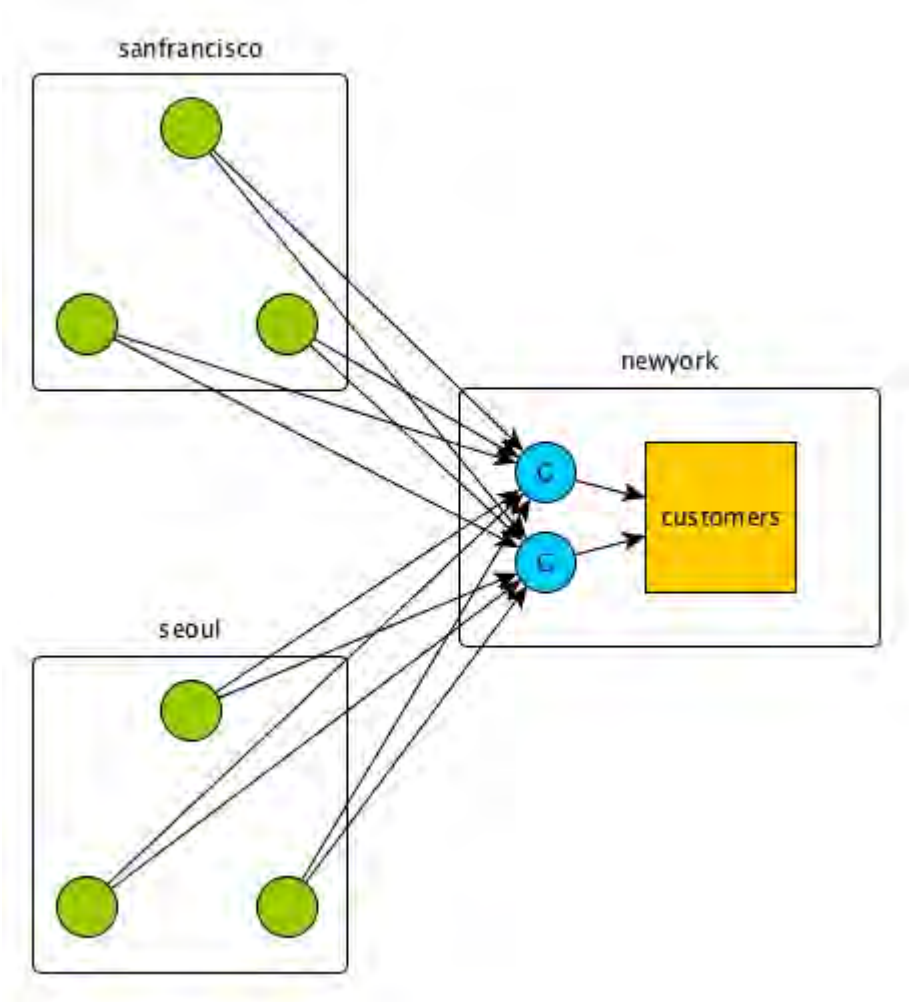
- If you are going to replicate MapR Database tables, see Gateways for Replicating MapR Database Tables on page 621.
- If you are going to replicate streams, see Gateways and Stream Replication on page 662.
- If your MapR Database JSON tables have secondary indexes, see Preparing Clusters for Querying using Secondary Indexes on JSON Tables on page 1089.
- If you are using CDC, see Getting Started with CDC on page 600.

Note: Gateways perform negligible disk I/O and use negligible amounts of memory, though gateways require significant CPU usage.

However, the resource that gateways use the most is network bytes. For example, if the peak network throughput for puts is about 40 MB per second per node, in a 10-node source cluster the peak network throughput will be about 400 MB per second. So, the aggregate network throughput required on the nodes running gateways will be 400 MB per second for both incoming and outgoing traffic. The aggregate network throughput for a 50 node cluster would be 2GB per second.

For another example, in the following diagram there are two source clusters of three nodes each and the clusters are replicating to one destination cluster. The peak traffic on the gateways will be 40MB per second per cluster node, which means that these gateways together will experience a peak network load of 240MB per second.

last-updated: Apr 11, 2022



Although the load is balanced across the two gateways, so that each gateway experiences a peak network load of 120MB per second, each gateway should be able to tolerate the full aggregate network load in case the other gateway fails unexpectedly.

Related concepts

Configuring Gateways for Table and Stream Replication on page 1152
Configuring gateways involves installing the `mapr-gateway` package on nodes on a MapR destination cluster and then configuring the MapR source cluster to communicate with the destination cluster. The MapR source cluster is configured by specifying the destination cluster's CLDB node and gateway nodes.

gateway.conf on page 2197

Gateways for Replicating MapR Database Tables on page 621
In MapR Database table replication, MapR Database replicates updates to tables (binary and JSON) on source MapR clusters to replicas of those tables on destination MapR clusters. Gateways are services that receive these updates and apply them to the replicas. These gateways also propagate updates from JSON tables to their secondary indexes.

Related tasks

Specifying the Location of Gateways on page 782
Describes how to set the location of the MapR gateways using either the Control System or the CLI.

Related reference

cluster gateway delete on page 1564
Deletes the list of MapR gateways from a source MapR cluster.

cluster gateway get on page 1565

last-updated: Apr 11, 2022

Lists the MapR gateways that a source MapR cluster is using.

cluster gateway list on page 1567
Lists all of the gateways that a source MapR cluster is using.

cluster gateway local on page 1569
Lists the gateways that are configured on the MapR cluster where this command is run.

cluster gateway resolve on page 1571
Lists the gateways configured on a MapR cluster that are up and running at the time that the command is issued.

cluster gateway set on page 1573
Specifies the locations of the MapR gateways that a source MapR cluster can use for table replication to a destination MapR cluster, or for indexing table data in an Elasticsearch cluster.

Related information

Managing Gateways on page 1154
Describes the commands for listing gateways, checking status of gateways, managing gateways if they fail, and troubleshooting gateways.

Configuring Gateways for Table and Stream Replication

Configuring gateways involves installing the `mapr-gateway` package on nodes on a MapR destination cluster and then configuring the MapR source cluster to communicate with the destination cluster. The MapR source cluster is configured by specifying the destination cluster's CLDB node and gateway nodes.

How Many Gateways to Configure?

Although it is possible to use a single gateway, the recommended practice is to configure at least two (2) gateways, so that replication can continue if one gateway fails. MapR source clusters distribute requests among the gateways in a round-robin fashion. See Gateways for Replicating MapR Database Tables on page 621, Table Replication on page 610, and Preparing Clusters for Stream Replication on page 1130 for more information about replication.

For more information about setting up cross-cluster security, including cross-cluster security for table and stream replication, see Setting Up Cross-Cluster Security on page 1482.

Default Gateway Configuration for Replication

If you do not perform any of these options, MapR Database uses the configuration from the `mapr-clusters.conf` file. It uses the cluster name specified in that file as the destination cluster, and the CLDB node addresses as the gateway nodes. You must have gateways running on these CLDB nodes.

Configuration Procedure

Tip: To list the current gateways, see cluster gateway get on page 1565.

1. On the destination cluster, install the `mapr-gateway` package on each node where you want to run a gateway. See Installing MapR Software.

   Note: On the gateway nodes in the destination cluster, when you run (or re-run) the `configure.sh` script (in addition to your regular parameters) be sure to specify the `-N` parameter with the name of the cluster to which the gateway belongs. For more information about `configure.sh` usage, options, and examples, see configure.sh on page 2053.

2. To change the port that a gateway uses (by default, gateways use port `7660`):

   a. On the node where the gateway is running, edit the `/opt/mapr/conf/gateway.conf` file, ensure that the line `#gateway.port=7660` is not commented, and change the port number. For more information about `gateway.conf` configuration properties, see gateway.conf on page 2197.

b. After saving the file, restart the gateway by running this command: `maprcli node services -name gateway -action restart`

3. On the source cluster, specify the destination cluster's name and gateway nodes by using one of the following methods:

- Run the `maprcli cluster gateway set` command:

```
maprcli cluster gateway set -dstcluster <cluster name> -gateways
"<space-delimited list of gateways>"
```

The following sample command sets two gateway nodes on the destination cluster `my.cluster.com`:

```
maprcli cluster gateway set -dstcluster my.cluster.com -gateways
"node1.com node2.com"
```

- Add a DNS record to your DNS server's zone file for your domain.

See Managing Gateways on page 1154 for more information about using these methods.

Related concepts

Administering MapR Gateways on page 1150
A MapR gateway mediates one-way communication between a source MapR cluster and a destination cluster. You can replicate MapR Database tables (binary and JSON) and MapR Event Store For Apache Kafka streams. MapR gateways also apply updates from JSON tables to their secondary indexes and propagate Change Data Capture (CDC) logs.

gateway.conf on page 2197

Gateways for Replicating MapR Database Tables on page 621
In MapR Database table replication, MapR Database replicates updates to tables (binary and JSON) on source MapR clusters to replicas of those tables on destination MapR clusters. Gateways are services that receive these updates and apply them to the replicas. These gateways also propagate updates from JSON tables to their secondary indexes.

Related tasks

Specifying the Location of Gateways on page 782
Describes how to set the location of the MapR gateways using either the Control System or the CLI.

Related reference

cluster gateway delete on page 1564
Deletes the list of MapR gateways from a source MapR cluster.

cluster gateway get on page 1565
Lists the MapR gateways that a source MapR cluster is using.

cluster gateway list on page 1567
Lists all of the gateways that a source MapR cluster is using.

cluster gateway local on page 1569
Lists the gateways that are configured on the MapR cluster where this command is run.

cluster gateway resolve on page 1571
Lists the gateways configured on a MapR cluster that are up and running at the time that the command is issued.

cluster gateway set on page 1573
Specifies the locations of the MapR gateways that a source MapR cluster can use for table replication to a destination MapR cluster, or for indexing table data in an Elasticsearch cluster.

last-updated: Apr 11, 2022

Related information

Managing Gateways on page 1154
Describes the commands for listing gateways, checking status of gateways, managing gateways if they fail, and troubleshooting gateways.

Managing Gateways

Describes the commands for listing gateways, checking status of gateways, managing gateways if they fail, and troubleshooting gateways.

You can run the following commands to perform operations on your gateways.

Note: If you have configured an intra-cluster gateway, the source and destination clusters are the same.

- To see a list of the gateways for a particular destination cluster:

  Run the `maprcli cluster gateway get` command on the source cluster. Specify the name of the destination cluster with the `-dstcluster` parameter. If you run the command remotely from your source cluster, specify the name of the source cluster with the `-cluster` parameter.

- To see a list of the gateways for all of the destination clusters to which the source cluster is replicating:

  Run the `maprcli cluster gateway list` command on the source cluster. If you run the command remotely from your source cluster, specify the name of the source cluster with the `-cluster` parameter.

- To remove the list of gateways that you specified for a destination cluster by using the `maprcli cluster gateway set` command:

  Run the `maprcli cluster gateway delete` command on the source cluster. Specify the name of the destination cluster with the `-dstcluster` parameter. If you run the command remotely from your source cluster, specify the name of the source cluster with the `-cluster` parameter.

- To find out how MapR Database or MapR Event Store For Apache Kafka is finding gateways (for example, from DNS records, lists created by the `maprcli cluster gateway set` command or the `mapr-clusters.conf` file):

  Run the command `maprcli cluster gateway resolve` on the source cluster. Specify the name of the destination cluster with the `-dstcluster` parameter. If you run the command remotely from your source cluster, specify the name of the source cluster with the `-cluster` parameter.

- To stop and start one or more gateways:

  Run the `maprcli node services -name gateway -action stop|start` on the clusters where the gateways are running.

  ```
  /opt/mapr/bin/maprcli node services -name gateway -action stop -nodes
  <hostnames or IP addresses separated by spaces>
  ```

  ```
  /opt/mapr/bin/maprcli node services -name gateway -action start -nodes
  <hostnames or IP addresses separated by spaces>
  ```

- To check the status of a gateway service on a particular node:

  Run the command maprcli service list on the clusters where the gateways are running.

last-updated: Apr 11, 2022

Running the `maprcli cluster gateway set` command

The syntax of the `maprcli cluster gateway set` command is:

```
/opt/mapr/bin/maprcli cluster gateway set -dstcluster <cluster
name> -gateways "<space-delimited list of gateways>"
```

To generate a list of the existing gateways in a MapR cluster, use the `maprcli cluster gateway list` command. You can then copy this list and paste it into the `maprcli cluster gateway set` command. Alternatively, to generate a list of the gateways on a local cluster, run the `maprcli cluster gateway local -format text` command. If you want to run the command from a different cluster and point to the cluster where the gateways are located, use the `-cluster` parameter to provide the name of that cluster.

For example, suppose that you are configuring table replication from the cluster `sanfrancisco` to the cluster `newyork`, and want to use two gateways. The nodes on which these gateways are located are named `gw1` and `gw2`.

The command that you run will be as follows:

```
/opt/mapr/bin/maprcli cluster gateway set -dstcluster newyork -gateways
"gw1.bigcompany.com gw2.bigcompany.com"
```

Adding a DNS record to your DNS server's zone file for your domain

In your DNS server's zone file for your domain, you can add a record for the cluster where gateways are located, listing the nodes to use as gateways. You can use the Control System to create a record that you can copy into a DNS configuration file, run a `maprcli` command to generate the record, or create a record manually.

For details, see Specifying the Location of Gateways on page 782.

If a Gateway Fails

If a gateway fails, the warden service tries three (3) times to restart it automatically. After an interval, the warden tries again three times to start the gateway. You can configure the interval by using the parameter `services.retryinterval.time.sec` in the `warden.conf` file. The default is 30 minutes.

During the time that the gateway is down, source clusters will resend updates to other gateways. Source clusters will also ping the failed gateway with an exponentially increasing backoff.

If all of the gateways fail in a destination cluster, source clusters will ping the failed gateways in the same manner. Updates pending replication are stored on disk in an internal data structure until at least one gateway in the destination cluster comes back online. Therefore, you will see additional storage costs during a gateway outage. The Gateway Service Down alarm in the Control System will notify you when none of the gateways in a destination cluster can be reached.

If the additional storage becomes too costly, you can follow either of these procedures:

If you are replicating to a MapR Database binary table:

1. Run the `maprcli table replica remove` command to stop replicating to the replica. This action deletes the pending updates.
2. Resolve the gateway outage.
3. Recreate the replica and start replicating to it by running the `maprcli table replica autosetup` command.

If you are replicating to a MapR Event Store For Apache Kafka stream:

last-updated: Apr 11, 2022

1. Run the `maprcli stream replica remove` command to stop replicating to the replica stream. This action cancels the pending updates to the replica stream.
2. Resolve the gateway outage.
3. Run the command `maprcli stream replica autosetup` to recreate the replica stream and start replicating to it.

Troubleshooting

You can refer to two log files for each gateway when troubleshooting. Both these files are in the `/opt/mapr/logs` directory on the node where the gateway is running:

- `gateway.log`
- `gatewayinit.log`

MapR Database Lookup Order

MapR Database uses the following order to locate the gateways that are running in a destination cluster.

- Look up the destination cluster's name and gateway addresses in the information specified by the `maprcli cluster gateway set` command. If a list of gateways, then a DNS lookup is performed.
- Perform a DNS lookup of the destination cluster and the addresses of the gateways. If no DNS record for the destination cluster is found, then the lookup goes to the `mapr-clusters.conf` file.
- Look up the destination cluster's name and the CLDB node addresses in the `mapr-clusters.conf` file under the assumption that gateways are running on all of the CLDB nodes and only on those nodes.

Related concepts

Administering MapR Gateways on page 1150
A MapR gateway mediates one-way communication between a source MapR cluster and a destination cluster. You can replicate MapR Database tables (binary and JSON) and MapR Event Store For Apache Kafka streams. MapR gateways also apply updates from JSON tables to their secondary indexes and propagate Change Data Capture (CDC) logs.

Configuring Gateways for Table and Stream Replication on page 1152
Configuring gateways involves installing the `mapr-gateway` package on nodes on a MapR destination cluster and then configuring the MapR source cluster to communicate with the destination cluster. The MapR source cluster is configured by specifying the destination cluster's CLDB node and gateway nodes.

gateway.conf on page 2197

Gateways for Replicating MapR Database Tables on page 621
In MapR Database table replication, MapR Database replicates updates to tables (binary and JSON) on source MapR clusters to replicas of those tables on destination MapR clusters. Gateways are services that receive these updates and apply them to the replicas. These gateways also propagate updates from JSON tables to their secondary indexes.

Related tasks

Specifying the Location of Gateways on page 782
Describes how to set the location of the MapR gateways using either the Control System or the CLI.

Related reference

cluster gateway delete on page 1564
Deletes the list of MapR gateways from a source MapR cluster.

cluster gateway get on page 1565
Lists the MapR gateways that a source MapR cluster is using.

cluster gateway list on page 1567

Lists all of the gateways that a source MapR cluster is using.

cluster gateway local on page 1569
Lists the gateways that are configured on the MapR cluster where this command is run.

cluster gateway resolve on page 1571
Lists the gateways configured on a MapR cluster that are up and running at the time that the command is issued.

cluster gateway set on page 1573
Specifies the locations of the MapR gateways that a source MapR cluster can use for table replication to a destination MapR cluster, or for indexing table data in an Elasticsearch cluster.

Administering Services

The various topics in this section describe how to manage (start, stop, restart, etc.) the various services installed on the MapR cluster using the MapR Control System (click **Services**) and the CLI.

Managing Services

Synopsis on managing services.

Once a role is installed on a node and the warden has been restarted, MapR recognizes the role for that node. You can then start the service. Refer to the following topics for information on managing services on a node using the Control System and the CLI.

Viewing the List of Services
Explains how to view the list of services using either the Control System or the CLI.
Viewing the Services Installed on the Cluster Using the Control System

- Log in to the Control System and click **Services**.

  The **Services** pane displays all the services installed on the cluster. On the non-Kubernetes version of the Control System, the pane displays the following:

Column Name	Column Description
Service	The name of the installed service.
Running Nodes	The number of nodes on which the associated service is running. The service can be stopped ( ) or restarted ( ). Click the number in this column to view the nodes on which the service is running.
Standby Nodes	The number of nodes on which the associated service is in standby (available, but not running) state. The service can be started ( ) or restarted ( ). Click the number in this column to view the nodes on which the service is in standby state.
Failed Nodes	The number of nodes on which the service has failed. The service can be started ( ) or restarted ( ). Click the number in this column to view the nodes on which the service has failed.
Stopped Nodes	The number of nodes on which the associated service is stopped (and not running). The service can be started ( ) or restarted ( ). Click the number in this column to view the nodes on which the service has been stopped.
Config Path	The path to the configuration file for the service.
Log Viewer	(Displays only if Kibana is installed on a node) The link ( ) to the Kibana UI.

last-updated: Apr 11, 2022

You can filter the list of services displayed by:

- **EEP**, which includes services such as Hive, Drill, etc.
- **Core**, which includes services such as CLDB, Hoststats, File server, etc.
- **Monitoring**, which includes services such as Grafana, Kibana, etc.

Viewing the Services Running on a Node Using the Control System

1. Log in to the Control System and click **Nodes**.

   Note: The **Nodes** menu is not available in the Kubernetes version of the Control System.

2. You can:
   - Hover the cursor over the number listed in the **Running Services** column in the **Nodes** pane to view the list of services installed on that node.
   - Go to the **Summary** tab in the node information page to view detailed information on the services installed on a node.

   In the **Summary** tab, for each service running on the node, the **Services** pane displays the following:

Column Name	Column Description
Service	The name of the service.
State	The current state of the service. Value can be: • Running • Stopped
Memory Allocated	The amount of system memory allocated to the service.
System Memory Utilized	The percentage of memory utilized by the service.
CPU Usage	The CPU used by the service.
Log Path	The path to the service log file.
Log Viewer	The link to the Kibana UI (only if Kibana is installed).

You can select the checkbox beside one or more services to take the following actions:

- Start Services
- Stop Services
- Restart Services

Note: If Kibana is installed, you can click to view the logs. See Kibana User Guide for more information.

Retrieving the Services Running on a Node Using the CLI or REST API

The command to list all the services on a node is:

```
maprcli service list -node <node name>
```

For complete reference information, see service list on page 1746.

Enabling and Disabling a Service Using the CLI and REST API
Describes how to enable or disable a service using either the REST API or the CLI.

You can disable a service to prevent it from starting or restarting when Warden starts or restarts, and enable a service to allow it to start or restart when Warden starts or restarts.

Disabling a Service Using the CLI or REST API

CLI	Run the following command: `maprcli node services -nodes <hostName> -name <serviceName> -action disable`
REST	Send a request of type POST. For example: `curl -k -X POST 'https://<host>:8443/rest/node/services?nodes=<hostName>&name=<serviceName>&action=disable' --user mapr:mapr`

Note: When you disable a service, the service is stopped and the service is not automatically starrted/restarted when Warden is started/restarted.

See `node services` on page 1730 for more information.

Enabling a Service Using the CLI or REST API

CLI	Run the following command: `maprcli node services -nodes <hostName> -name <serviceName> -action enable`
REST	Send a request of type POST. For example: `curl -k -X POST 'https://<host>:8443/rest/node/services?nodes=<hostName>&name=<serviceName>&action=enable' --user mapr:mapr`

Note: When you enable a service, the service is started/restarted when Warden is started/restarted.

See `node services` on page 1730 for more information.

Related tasks

Restarting the Services on page 831
Describes how to restart a service using either the Control System, the CLI or the REST API.

Starting the Services
Explains how to start services using either the Control System, the CLI or the REST API.

You can start one or more services using the Control System or the CLI if the service is not disabled. If the service is disabled, you must enable the service first, in order to start the service. See Enabling and Disabling a Service Using the CLI and REST API on page 828 for more information.

Starting the Services Running on the Nodes Using the Control System

To start the services running on the nodes:

1. Log in to the Control System and click **Nodes** to display the **Nodes** page.

   Note: The **Nodes** page is not available on the Kubernetes version of the Control System.

2. Select the nodes from the list of nodes in the **Nodes** pane and click **Manage Services** to display the **Manage Services** window.
3. Choose the **Start** radio button for the services you wish to start on the selected nodes and click **Save**.

Starting the Services Running on a Node Using the Control System

To start one or more services running on a node:

1. Go to the **Summary** tab in the node information page.
2. Select the services to start in the **Services** pane.
3. Click **Start Services**.
   The **Start Services** confirmation dialog displays.
4. Verify the list of services to start and click **Start Service**.

Starting the Services on the Cluster Using the Control System

1. Log in to the Control System and click **Services** to display the list of services on the cluster.
2. On the non-Kubernetes version of the Control System, click for the service to start.
   The **Start Service** confirmation dialog displays.
3. Verify the list of nodes on which to start the service and click **Start Service**.

Starting a Service Using the CLI or REST API

The basic command to start a service on a node is:

```
maprcli node services -nodes <node name> -name <service> -action start
```

For complete reference information, see `node services` on page 1730.

Stopping the Services

Describes how to stop services using either the Control System, the CLI or the REST API.

Stopping the Services Running on the Nodes Using the Control System

To stop the services running on the nodes:

1. Log in to the Control System and click **Nodes** to display the **Nodes** page.

   Note: The **Nodes** page is not available in the Kubernetes version of the Control System.

2. Select the nodes from the list of nodes in the **Nodes** pane and click **Manage Services** to display the **Manage Services** window.
3. Choose the **Stop** radio button for the services you wish to stop on the selected nodes and click **Save**.

Stopping the Services on a Node Using the Control System

To stop one or more services running on a node:

1. Go to the **Summary** tab in the node information page.
2. Select the services to stop in the **Services** pane.
3. Click **Stop Services**.
   The **Stop Services** confirmation dialog displays.
4. Verify the list of services to stop and click **Stop Service**.

Stopping a Service on the Cluster Using the Control System

1. Log in to the Control System and click **Services**.
2. On the non-Kubernetes version, click  associated with the service to stop.
   The **Stop Service** confirmation dialog displays.
3. Verify the list of nodes on which to stop the service and click **Stop Service**.

Stopping the Services Using the CLI or REST API

The basic command to stop a service on a node is:

```
maprcli node services -nodes <node name> -name <service> -action stop
```

For complete reference information, see `node services` on page 1730.

Restarting the Services

Describes how to restart a service using either the Control System, the CLI or the REST API.

When a MapR system is rebooted, the following services are automatically restarted:

- mapr-warden
- mapr-zookeeper
- mapr-loopbacknfs
- mapr-posix-client-*

These services are also automatically restarted if they are shut down externally (as opposed to being shut down explicitly via `service` or `sysctl` commands).

Note: This feature is implemented with `systemd` and is only supported on the following operating systems:

- RHEL 7.0, 7.1
- CentOS 7.0, 7.1
- SLES 12

This feature is not supported on any of the Ubuntu versions that MapR currently supports.

You can restart one or more services using the Control System and the CLI if the services are not disabled. However, if a service is disabled, the service cannot be restarted. To restart a service, make sure the service is enabled. See Enabling and Disabling a Service Using the CLI and REST API on page 1159 for more information.

last-updated: Apr 11, 2022

Restarting the Services Running on the Nodes Using the Control System

To restart the services running on the nodes:

1. Log in to the Control System and click **Nodes** to display the **Nodes** page.

   Note: The **Nodes** page is not available on the Kubernetes version of the Control System.

2. Select the nodes from the list of nodes in the **Nodes** pane and click **Manage Services** to display the **Manage Services** window.
3. Choose the **Restart** radio button for the services you wish to restart on the selected nodes and click **Save**.

Restarting one or more Services on a Node Using the Control System

1. Go to the **Summary** tab in the node information page.
2. Select the services to restart in the **Services** pane.
3. Click **Restart Service(s)**.
   The **Restart Service(s)** confirmation dialog displays.
4. Verify the list of services to restart and click **Restart Service**.

Restarting the Services on the Cluster Using the Control System

1. Log in to the Control System and navigate to **Services**.
2. On the non-Kubernetes version, click associated with the service to restart.
   The **Restart Service** confirmation dialog displays.
3. Verify the list of nodes on which to restart the service and click **Restart Service**.

Restarting a Service Using the CLI or REST API

The basic command to restart a service on a node is:

```
maprcli node services -nodes <node name> -name <service> -action restart
```

For complete reference information, see `node services` on page 1730.

Related tasks

Enabling and Disabling a Service Using the CLI and REST API on page 1159
Describes how to enable or disable a service using either the REST API or the CLI.

Viewing a Service Information Page

Describes how to view the information pages for the installed services using the Control System.

1. Log in to the Control System and click **Services**.

   Note: The **Services** page is not available on the Kubernetes version of the Control System.

   The **Services** pane displays the list of services that are installed on the cluster.
2. Click the name of the service from the list.
   The information page for the service displays.

Changing the User for MapR Services from the Command-Line

Explains how use the CLI to change the user that MapR services run as.

All services should run with the same uid/gid on all nodes in the cluster.

Running MapR Services as the Root User

1. Stop Warden.

```
service mapr-warden stop
```

2. If ZooKeeper is installed on the node, stop it.

```
service mapr-zookeeper stop
```

3. Run the script `$INSTALL_DIR/server/config-mapr-user.sh -u root`
4. If Zookeeper is installed, start it.

```
service mapr-zookeeper start
```

5. Start Warden.

```
service mapr-warden start
```

Running MapR Services as a Non-Root User

1. Stop Warden.

```
service mapr-warden stop
```

2. If ZooKeeper is installed on the node, stop it.

```
service mapr-zookeeper stop
```

3. If the MAPR_USER does not exist, create the user/group with the same UID and GID.
4. If the MAPR_USER exists, verify that the uid of MAPR_USER is the same as the value on the CLDB node.
5. Run `$INSTALL_DIR/server/config-mapr-user.sh -u MAPR_USER`.
6. If Zookeeper is installed, start it.

```
service mapr-zookeeper start
```

7. Start Warden.

```
service mapr-warden start
```

8. After clearing NODE_ALARM_MAPRUSER_MISMATCH alarms on all nodes, run `$INSTALL_DIR/server/upgrade2mapruser.sh` on all nodes wherever the alarm is raised.

Viewing the Service Log

Esplains how to view service logs using Kibana.

last-updated: Apr 11, 2022

If Kibana is installed on the node, you can view the service log in the Kibana UI from the Control System. To view the log in the Kibana UI from the Control System:

1. Log in to the Control System and do one of the following:
   - Click **Services** to display the list of services installed on the cluster.
   - Go to the **Summary** tab in the service information page for the service.
2. Click  in the **Log Viewer** column to view the log for the associated service in the Kibana UI. See Kibana User Guide for more information.

Setting Up Central Configuration from the Command-Line

Describes the concept of a central location where customized MapR configuration files for MapR services are stored.

MapR provides a central location where you can place customized configuration files for all the services running on the MapR cluster. As a result, you do not have to edit the configuration files on each node individually.

Default configuration files for each service are stored locally under `/opt/mapr/`. You can edit these files to create customized versions of the configuration files. To push these changes to other nodes in the cluster, copy the customized files to the `mapr.configuration` volume in the MapR filesystem. When the `pullcentralconfig` script runs, it overwrites the local configuration file in `/opt/mapr` with the customized files on all nodes on which the change applies.

The `mapr.configuration` volume is mounted at `maprfs://var/mapr/configuration.`The `mapr.configuration` volume mount location is not configurable and is only used for central configuration.

Using Central Configuration

Lists the steps to configure and use central configuration of files.

MapR provides a central location where you can place customized configuration files for all the services running on the MapR cluster. As a result, you do not have to edit the configuration files on each node individually.

Default configuration files for each service are stored locally under `/opt/mapr/`. You can edit these files to create customized versions of the configuration files. To push these changes to other nodes in the cluster, copy the customized files to the `mapr.configuration` volume in the MapR File System. When the `pullcentralconfig` script runs, it overwrites the local configuration file in `/opt/mapr` with the customized files on all nodes to which the change applies.

The `mapr.configuration` volume is mounted at `maprfs://var/mapr/configuration.`The `mapr.configuration` volume mount location is not configurable and is only used for central configuration.

Complete the following steps to use the Central Configuration (`pullcentralconfig` script) to push a customized configuration file from the `mapr.configuration` volume in MapR File System to nodes in the cluster.

1. Customize a configuration file on any node in the cluster. See Listing the Configuration Files for Each Service to determine which files you can customize.

last-updated: Apr 11, 2022

2. Create the directory structure for the configuration file that is relative to $MAPR_HOME based on the number of nodes to which the file applies:

Scope	MapR File System Location
Use this file on all nodes that use the configuration file unless a node-specific file exists in the `mapr.configuration` volume.	`/var/mapr/configuration/default/<directory path to the configuration file>`
Use this file for a specific node.	`/var/mapr/configuration/nodes/<hostname>/<directory path to the configuration file>`

3. Copy the customized file to the `mapr.configuration` volume in the directory that you created. For example, to create a directory to update the web server configuration file, which is in `/opt/mapr/conf`, you would create directory path `/var/mapr/configuration/default/conf` and then copy the updated file to this directory.

4. Run the `pullcentralconfig` script or wait until the script acknowledges the update. For more information, see About the pullcentralconfig Script.

5. Restart the services associated with the updated configuration files to ensure that the latest version is used for each service.

Scenario

Lists an example scenario to demonstrate customized configuration for a 8-node cluster.

In the following example, you have a cluster with eight nodes, and five of them (host1, host2, host3, host4, and host5) are running the NodeManager service.

You want to create one customized configuration file (`mapred-site.xml`) that applies to host2 through host5 and assign a different customized configuration file to host1.

Hostname	Customized Configuration Files
host1	`/var/mapr/configuration/`**`nodes/host1`**`/hadoop/hadoop-2.7.0/etc/hadoop/mapred-site.xml`
host2	`/var/mapr/configuration/`**`default`**`/hadoop/hadoop-2.7.0/etc/hadoop/mapred-site.xml`
host3	`/var/mapr/configuration/`**`default`**`/hadoop/hadoop-2.7.0/etc/hadoop/mapred-site.xml`
host4	`/var/mapr/configuration/`**`default`**`/hadoop/hadoop-2.7.0/etc/hadoop/mapred-site.xml`
host5	`/var/mapr/configuration/`**`default`**`/hadoop/hadoop-2.7.0/etc/hadoop/mapred-site.xml`
host6	Not applicable. No customized configuration files are used.
host7	Not applicable. No customized configuration files are used.
host8	Not applicable. No customized configuration files are used.

last-updated: Apr 11, 2022

Warning: Do *not* change the name of the new configuration file - it must match the name of the original version.

Step 1. Create a customized configuration file and copy it to a volume
Use the CLI to create a customized configuration file and copy it to a volume.

Complete the following steps to create a customized configuration file for host2 through host5 and copy it to the mapr.configuration volume:

1. Make a copy of the existing default version of the `mapred-site.xml` file (so you can use it as a template), and store it in `/tmp`. You can perform this step on any node in the cluster that contains the configuration file.

```
cp /opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/mapred-site.xml /tmp/
mapred-site.xml
```

2. Edit the copy and put in the changes you want for host2 through host5.
3. Create the directories required to store the file under `/var/mapr/configuration/default`.

```
hadoop fs -mkdir -p /var/mapr/configuration/default/hadoop/
hadoop-2.7.0/etc/hadoop
```

4. Store the new configuration file in the `/var/mapr/configuration/default` directory.

```
hadoop fs -put /tmp/mapred-site.xml /var/mapr/configuration/default/
hadoop/hadoop-2.7.0/etc/hadoop
```

Step 2. Create a node-specific configuration file and copy it to a volume
Use the CLI to create a node-specific configuration file and copy it to a volume.

Complete the following steps to create a node-specific configuration file for host1 and copy it to the mapr.configuration volume:

1. Edit the `mapred-site.xml` configuration file in `/tmp` (or you could copy the default version into `/tmp` again and edit that) and create the node-specific configuration file for host1.
2. Create the directories required to store the file under `/var/mapr/configuration/nodes`:

```
hadoop fs -mkdir -p /var/mapr/configuration/nodes/host1/hadoop/
hadoop-2.7.0/etc/hadoop
```

3. Store the new configuration file for host1 in the node-specific directory you just created.

```
hadoop fs -put /tmp/mapred-site.xml /var/mapr/configuration/nodes/host1/
hadoop/hadoop-2.7.0/etc/hadoop
```

Step 3. Verify the changes
Explains the use of the `pullcentralconfig` script in verifying file changes.

Now that you have two separate customized configuration files for your NodeManager nodes, the `pullcentralconfig` script will detect the new files in `/var/mapr/configuration` the next time it is launched. It overwrites the local version in `/opt/mapr` with the appropriate customized version for each of the five NodeManager nodes.

1. To launch the `pullcentralconfig` script, perform one of the following operations:

- Run `pullcentralconfig` from the command line to overwrite the old files immediately.

```
/opt/mapr/server/pullcentralconfig
```

- Wait five minutes (the interval between successive checks for updated configuration files) for the script to run automatically.

2. Look at the information messages in `pullcentralconfig.log`. Whenever the timestamp comparison is `false`, an INFO message indicates that the older file under `/opt/mapr` is copied into a backup file (`.bkp`) and the newer version is copied to replace it. Sample log output is as follows:

```
cat pullcentralconfig.log
   Header: hostName: doc22.lab, Time Zone: Pacific Standard Time,
processName: CentralConfig, processId: 16701
   2017-07-17 13:13:01,842 INFO
com.mapr.centralconfig.CentralConfigCopyHelper [main]: Copied /var/mapr/
configuration/default///hadoop/hadoop-2.7.0/etc/hadoop/mapred-site.xml
to /opt/mapr//hadoop/hadoop-2.7.0/etc/hadoop/mapred-site.xmlusing
maprexecute.
```

3. Restart the service.

About the pullcentralconfig Script
Provides a brief synopsis about the `pullcentralconfig` script.

The `pullcentralconfig` script runs on each node in the cluster. The `pullcentralconfig` script is launched automatically at specified intervals (the default interval is 300 seconds, which is five minutes). For each service listed in `/opt/mapr/roles`, the `pullcentralconfig` script searches for corresponding configuration files in the central configuration location, `/var/mapr/configuration`. If any configuration files are found, it compares the timestamp (`mtimes`) of the files in the central configuration location to the timestamp of the local version in `/opt/mapr`. If the central configuration version is newer, `pullcentralconfig` overwrites the local version with the central configuration file. To ensure that the newer version gets used, you need to restart the associated service.

Checking for Node-specific vs. Cluster-wide Configuration Files
Explains the order in which the `pullcentralconfig` script checks for central configuration files.

The `pullcentralconfig` script checks for central configuration files in this order:

1. Node-specific configuration files under `/var/mapr/configuration/nodes/<hostname>`. If configuration files are found here, `pullcentralconfig` does not check `/var/mapr/configuration/default`.

2. Cluster-wide configuration files under `/var/mapr/configuration/default`. The `pullcentralconfig` script only searches here if no node-specific configuration files are found.

If no configuration files are found in either location, the script finishes and no changes are made to the files in `/opt/mapr`.

Changing the Polling Frequency
Explains how to change the polling frequency of the `pullcentralconfig` script.

By default, the `pullcentralconfig` script polls the central configuration location every five minutes (300 seconds) to check for configuration files.

1. You can change the polling frequency by editing the following variable in `warden.conf`:

```
pollcentralconfig.interval.seconds
```

2. To make the change take effect, restart Warden:

```
root$> service mapr-warden restart
```

Listing the Configuration Files for Each Service
Explains how to use the CLI to list all the configuration files for each service.

Each service on a node has one or more configuration files associated with it. The default version of each configuration file can be used as a template that you can modify as needed.

Files in the following directory contain the list of configuration files for each installed service: `/opt/mapr/servicesconf/<service>`.

- Use the following command to display the configuration files or the directory that contains the configuration files for each service:

```
cat /opt/mapr/servicesconf/<service>
```

For example, the following command displays the configuration files for NodeManager:

```
cat /opt/mapr/servicesconf/nodemanager
    hadoop/hadoop-2.7.4/etc/hadoop
```

The following table contains the configuration files and configuration file directories available for Hadoop and Hbase:

Service	Pathnames of Configuration Files
cldb	`/opt/mapr/conf/BaseLicense.txt`
	`/opt/mapr/conf/cldb.conf`
	`/opt/mapr/hadoop/hadoop-xxx/etc/hadoop/hadoop-metrics.properties`
	`/opt/mapr/conf/log4j.cldb.properties`
	`/opt/mapr/conf/log4j.properties`
	`/opt/mapr/conf/MapRLicenseIssuerCert.der`
fileserver	`/opt/mapr/conf/mfs.conf`
historyserver	`/opt/mapr/conf/conf.d/warden.historyserver.conf` **Attention:** You can perform additional configuration of the historyserver in `/opt/mapr/hadoop/hadoop-xxx/etc/hadoop/yarn-site.xml`
nfs	`/opt/mapr/conf/nfsserver.conf`
	`/opt/mapr/conf/exports`
nodemanager	`/opt/mapr/conf/conf.d/warden.nodemanager.conf`
	`/opt/mapr/conf.new/hadoop_version`

Service	Pathnames of Configuration Files
resourcemanager	`/opt/mapr/conf/conf.d/warden.resourcemanager.conf` **Attention:** You can perform additional configuration of the resourcemanager in `/opt/mapr/hadoop/hadoop-xxx/etc/hadoop/yarn-site.xml`
webserver	`/opt/mapr/apiserver/conf/properties.cfg`

Preserving Multiple Versions of Configuration Files

- If you want to save multiple versions of customized configuration files, you can take snapshots of `mapr.configuration` to preserve each version you create.

  This maybe helpful if you want to try various versions of the configuration files on the cluster.

  Warning: Before a local configuration file is updated by the `pullcentralconfiguration` script, a backup of the current version is created. The file extension `.bkp` indicates the back-up configuration file.

Disabling Automatic Central Configuration

Explains how to stop automatic runs of the `pullcentralconfig` script.

1. Automatic central configuration is enabled by default. You can configure central configuration to stop automatic runs of the `pullcentralconfig` script by editing `warden.conf` and setting the following variable to *false*: `centralconfig.enabled=false`
2. To make the change take effect, restart Warden:

   ```
   root$> service mapr-warden restart
   ```

   Note: When automatic central configuration is disabled, you can still manually run the `pullcentralconfig` script to push the latest configuration files to nodes in the cluster.

Viewing CLDB Information

Describes how to view CLDB information from the CLDB page, and provides an explanation of each field that the page displays.

The CLDB page on the Control Panel provides information about the Container Location Database (CLDB). The CLDB is a management service that tracks container locations and the root of volumes.

To display the CLDB information page, log in to the Control System and go to the service information page for CLDB. Alternatively, you can use the following links to access the CLDB page:

- For a *secure* cluster, access the CLDB view at `https://<cldbmaster>:7443/cldb.jsp`.
- For an *non-secure* cluster, access the CLDB view at `http://<cldbmaster>:7221/cldb.jsp`.

The CLDB page displays the following information:

Container Location Database	*Description:* This section displays: • CLDB Mode: The CLDB node can be in one of the following modes: MASTER_READ_WRITE or SLAVE_READ_ONLY.

last-updated: Apr 11, 2022

	• CLDB BuildVersion: Lists the build version. • Master for CLDB volume ready: Indicates whether the CLDB volume is ready for use. • CLDB Status: The status of the CLDB node. • Cluster Capacity: The storage capacity for the cluster. • Cluster Used: The amount of storage in use. • Cluster Available: The amount of available storage. • Cluster Used Percentage: The percentage of storage in use.
Active FileServers	*Description:* Displays information about the File Servers in use: • ServerID (Hex): The server's ID in hexadecimal notation. • ServerId: The server's ID in decimal notation. • HostPort: The IP address of the file server. • HostName: The hostname assigned to the file server. • Network Location: The network topology for the file server. • Last Heartbeat (s) : The timestamp for the last received heartbeat. • State: Is the file server ACTIVE (in use at present) • Capacity (MB): Total storage capacity on the file server. • Used (MB): Storage used on the file server. • Available (MB): Storage available on the file server. • In Transit (MB): Amount of data in transit
Active NFSServers	*Description:* Displays information about the NFS Servers in use: • ServerID (Hex): The server's ID in hexadecimal notation. • ServerId: The server's ID in decimal notation. • HostPort: The IP address of the NFS server. • HostName: The hostname assigned to the NFS server. • Last Heartbeat (s) : The timestamp for the last received heartbeat. • State: Is the NFS server ACTIVE (in use at present)

last-updated: Apr 11, 2022

Volumes

Description: Displays information about the volumes on the container:

- Volume Name: The name of the volume. Click the name of the volume to view the containers of the volume (including the container ID, size (in MB), container primary location and container locations, and replication type, which can be S for sequential and C for cascading).
- Mount Point: The path in which the volume is mounted over NFS.
- Mounted: Specifies whether volume is mounted. Value can be Y or N.
- ReadOnly: Specifies whether volume is a read-only or read/write volume. Value can be Y (if volume is read-only) or N (if volume is read/write).
- Volume ID: The Volume ID.
- Volume Topology: The path describing the topology to which the volume is assigned.
- Quota: The total size of the volume's quota. A quota of 0 means no quota is assigned.
- Advisory Quota: The usage level that triggers a disk usage warning.
- Used: Total size of data written to the volume after compression.
- LogicalUsed: Actual size of data written to the volume.
- Root Container ID: The ID of the root container.
- Replication: The number of copies of the volume.
- Guaranteed Replication: The minimum number of copies of the volume.

Accountable Entities

Description: Displays information about users and groups:

- AE Name: The name of the accountable entity.
- AE Type: The type of accountable entity.
- AE Quota: The hard quota allocated to the accountable entity.
- AE Advisory Quota: The advisory quota limit for the accountable entity.
- AE Used: The amount of disk space used by the accountable entity.

Mirrors Information

Description: Contains a link to the Mirrors page, which displays information about volume mirrors:

- Mirror Volume Name: The name of the mirror volume.

last-updated: Apr 11, 2022

- Mirror ID: The ID of the mirror volume used for the last successful mirroring. This ID starts with 1 and is incremented by 1 after each mirroring.
- Mirror NextID: The ID to be used for the next mirroring operation.
- Mirror Status: The status of the mirroring operation.
- Last Successful Mirror Time: The date and time stamp from the last successful mirroring operation.
- Mirror SrcVolume: The source volume for the mirror volume.
- Mirror SrcRootContainerID: The source container ID for the mirror volume.
- Mirror SrcClusterName: The source cluster name for the mirror volume.
- Mirror SrcSnapshot: The source snapshot associated with the mirror volume.
- Mirror DataGenerator Volume: The first volume that generates data (RW Volume) in the mirror chain.

  For example, if the mirror chain is **A** > **B** > **C**, then at C, the Mirror DataGenerator volume is A while the source volume is B.
- Mirror DataGenerator snapshot time: The time at which the last snapshot for the Mirror DataGenerator Volume was generated.

Snapshots Information

Description: Contains a link to the Snapshots page, which displays information about snapshots:

- Snapshot ID: The ID of the snapshot.
- RW Volume ID: The ID of the standard (or read/write) volume associated with the snapshot.
- Snapshot Name: The name of the snapshot.
- Root Container ID: The ID of the container.
- Snapshot Size: The size of the snapshot.
- Snapshot InProgress: The status of the snapshot if it is currently in progress.

The page also displays a list of snapshot containers, and the following information about each:

- Snapshot Container ID: The unique ID of the container.
- Snapshot ID: The ID of the snapshot corresponding to the container.
- RW Container ID: The corresponding source container ID.

last-updated: Apr 11, 2022

	• Latest Epoch: The latest sequence number of the snapshot. Higher the number, the newest and most up-to-date is the snapshot. • SizeMB: The container size, in MB. • Container Master Location: The location of the container's primary replica. • Container Locations: The location of the data containers. • Inactive Locations: The location of inactive containers.
Storage Pools	*Description:* Displays information about storage pools: • SP: The ID of the storage pool. • ServerId: The ID of the server associated with the storage pool. • SpIdx: The index of the storage pool. • Last Heartbeat(s): TBD • Capacity: The total capacity of the storage pool. • Used: The amount of space used on the storage pool. • Disk Fullness Level (percentage): The percentage of disk space (associated with the storage pool) that is full. • InTransit: Indicates that some containers in the storage pool are being resynced. • OutTransit: Indicates that none of the containers in the storage pool are being resynced.
Active Container Moves	*Description:* Displays information about the containers being moved: • Container ID: The ID of the container being moved. • SizeMB: The size (in MB) of the container being moved. • From Location: The location from where the container is being moved. • From SP: The SP out of which the container is being moved. • To Location: The location to which the container is being moved. • To SP: The SP to which the container is being moved.

Managing Drill

Provides a short description on managing Drill services.

last-updated: Apr 11, 2022

You can install and run a Drillbit service on one node or on all of the nodes on a MapR cluster to form a distributed cluster environment. After you have installed Drill and configured connections to your data sources, you can view Drill metrics and manage Drillbits using the Control System.

Viewing Drill Information
Explains how to view Drill information using the Control System.
Viewing Drill Information Using the Control System

1. Log in to the Control System and click **Services**.

   Note: The **Services** page is not available on the Kubernetes version of the Control System.

2. Click **Drill** in the **Services** pane.
   The **Drill** information page displays the following:

Summary

The **Summary** tab displays the following panes:

Information	Displays the number of Drillbits and the number of completed and in-progress queries.
Resource Utilization	The percentage of CPU, memory, and disk space utilized by Drill.
Running Fragments	The number of query fragments running in the Drillbit during the selected date and time range. You can select a preset (last 15 minutes, last 1 hour, last 12 hours, last 7 days, last 30 days, last 90 days) or custom time range and zoom in (by clicking and dragging your mouse in the pane) for a more granular view. For more information on fragments, see Drill Query Execution.
Memory Used	The amount of direct memory used by the JVM during the selected date and time range. You can select a preset (last 15 minutes, last 1 hour, last 12 hours, last 7 days, last 30 days, last 90 days) or custom time range and zoom in (by clicking and dragging your mouse in the pane) for a more granular view. To configure the amount of direct memory allocated to a Drillbit for query processing in a Drill cluster, see Configuring Drill Memory.
Queries	The number of queries during the selected time range. You can select a preset (last 15 minutes, last 1 hour, last 12 hours, last 7 days, last 30 days, last 90 days) or custom time range and zoom in (by clicking and dragging your mouse in the pane) for a more granular view.

Note: The metrics collection infrastructure must be installed during installation to visualize the metrics in the various panes. If the metrics collection infrastructure is not installed, perform an Incremental Install to install the metrics collection infrastructure.

Drill Bits The **Drill Bits** tab displays the list of Drillbits. See Viewing the List of Drillbits on page 1175.

Viewing the List of Drillbits

Explains how to view the list of Drillbits using the Control System.

- Log in to the Control System and go to the **Drill Bits** tab in the service information page for Drill.

  For each Drillbit, the pane displays the following:

Column Name	Column Description
Drill Bit ID	The ID of the Drillbit.
Node	The node on which the Drillbit is installed.
Service State	The status of the service. Value can be: • Running • Stopped
Queries in Progress	The number of queries currently in progress.
Resource Usage - Memory	The percentage of memory being used.
Resource Usage - CPU	The percentage of CPU being used.

  Select the checkbox beside one or more Drillbit IDs to:

  - **Stop the Service(s)**
  - **Start the Service(s)**
  - **Restart the Service(s)**

  For more information, see Stopping, Starting, and Restarting Drillbits on page 1175.

Stopping, Starting, and Restarting Drillbits

Explains how to stop, start and restart Drillbits using the Control System.

Stopping Drillbits

1. Log in to the Control System and go to the **Drill Bits** tab in the service information page for Drill.
2. Select the checkbox associated with the Drillbit(s) and click **Stop Service(s)** to display the **Stop Service** confirmation dialog.

   You can only stop Drillbits that are currently in running state.
3. Review the list of Drillbits and click **Stop Service** to stop the Drillbits.

Starting Drillbits

1. Log in to the Control System and go to the **Drill Bits** tab in the service information page for Drill.

last-updated: Apr 11, 2022

2. Select the checkbox associated with the Drillbit(s) and click **Start Service(s)** to display the **Start Service** confirmation dialog.

   You can only start Drillbits that are currently in stopped state.

3. Review the list of Drillbits and click **Start Service** to start the Drillbits.

Restarting Drillbits

1. Log in to the Control System and go to the **Drill Bits** tab in the service information page for Drill.
2. Select the checkbox associated with the Drillbit(s) and click **Restart Service(s)** to display the **Restart Service** confirmation dialog.
3. Review the list of Drillbits and click **Restart Service** to restart the Drillbits.

Managing the MapR NFS Service

Provides an overview of managing the NFS service on a licensed cluster.

The MapR NFS service lets you access data on a licensed MapR cluster using the NFS protocol:

- Community Edition license: one NFS node allows you to access your cluster as a standard POSIX-compliant filesystem.
- Enterprise Edition or Enterprise Database Edition license: multiple NFS servers allow each node to mount its own MapR filesystem on NFS enabled with VIPs for high availability (HA) and load balancing.

You can mount the MapR cluster using NFS, and use standard shell scripting to read and write live data on the cluster. NFS access to cluster data can be faster than accessing the same data with the `hadoop` commands. To mount the cluster using NFS from a client machine, see Accessing Data with NFS v3.

Managing VIPs for NFS

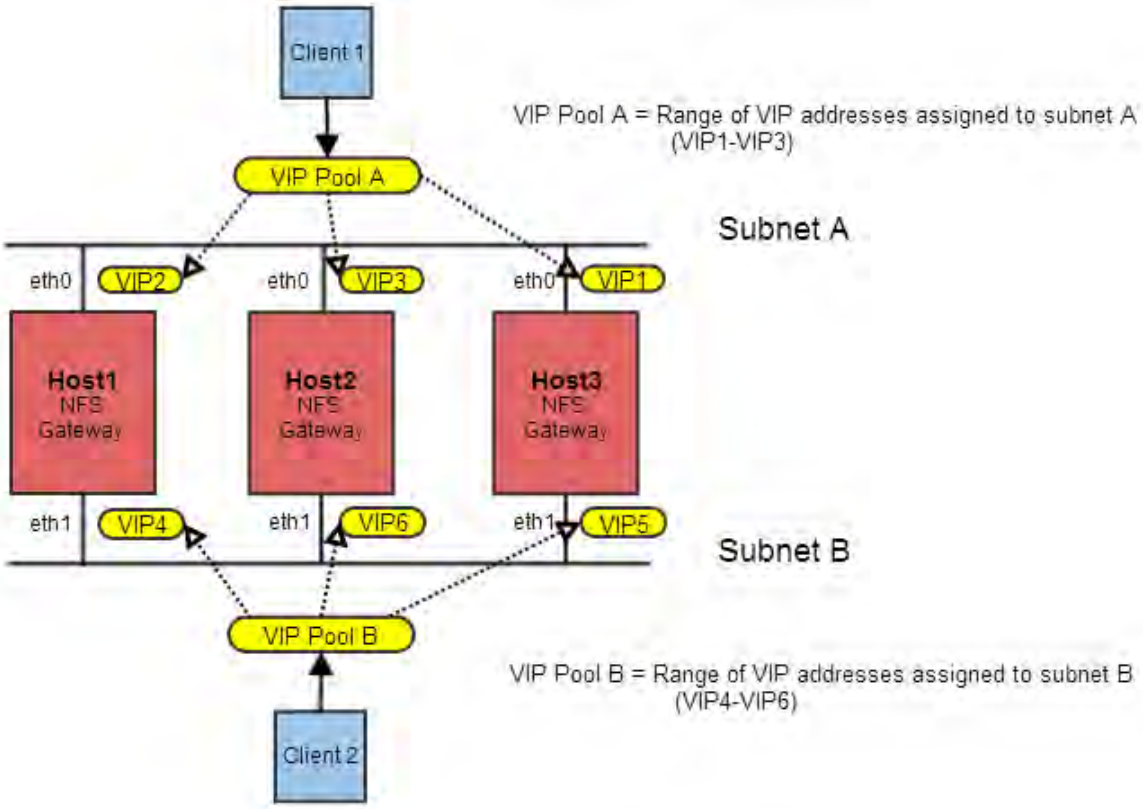
Explains how to use virtual IP addresses (VIPs) on NFS servers.

You can set up a pool of NFS servers on various nodes in your cluster and connect to them using virtual IP addresses (VIPs) to achieve High Availability (HA) with failover; if one node fails, the VIP will automatically be reassigned to another NFS node in the pool. If you do not specify a list of NFS nodes to form a pool, then MapR uses any available node running the MapR NFS Gateway service. VIPs are not assigned to any nodes that are not on the list, regardless of whether they are running NFS.

Note: To add a server to an NFS pool that is not divided into subnets, start the MapR NFS service on that server. The MapR cluster automatically detects it and adds it to the pool.

The following illustration shows three nodes (Host1, Host2, and Host3) acting as NFS servers. Each node has two NICs whose ports are labeled eth0 and eth1. The NICs are grouped into two subnets, called Subnet A and Subnet B. Clients can access any of the NFS servers through a pool (or range) of VIPs assigned to each subnet. MapR assigns each VIP address in the pool at random to a MAC address in the subnet (with its corresponding physical IP address).

last-updated: Apr 11, 2022



The initial VIP assignment shown above is summarized in the following table. If one NFS server becomes unavailable, the VIP assigned to that server is automatically assigned to another server on the same subnet.

Server	Subnet	VIPs (randomly assigned)
NFS1	A	VIP2
NFS2	A	VIP3
NFS3	A	VIP1
NFS1	B	VIP4
NFS2	B	VIP6
NFS3	B	VIP5

If the cluster's NFS nodes have multiple network interface cards (NICs) connected to different subnets, you should restrict VIP assignment to the NICs that are on the correct subnet: for each NFS server, choose whichever MAC address is on the subnet from which the cluster will be NFS-mounted, then add it to the list.

Warning: If you add a VIP that is not accessible on the subnet, then failover will not work. Also, do not use duplicate MAC addresses. For example, for bonding interfaces on the cluster nodes, do not use the same MAC address for bond0 and bond0.X, as then the failover of VIP might not work.

You can only set up VIPs for failover between network interfaces that are in the same subnet. In large clusters with multiple subnets, you can set up multiple groups of VIPs to provide NFS failover for the different subnets.

last-updated: Apr 11, 2022

VIPs are evenly distributed across NFS nodes. For example, if six VIP addresses are available for three NFS servers, two VIPs are assigned to each server. If the previous example did not have two separate subnets, the six VIP addresses might be assigned like this:

Server	VIPs (randomly assigned)
N/FS1	VIP1, VIP3
NFS2	VIP4, VIP5
NFS3	VIP2, VIP6

The following sections describe how to set up, edit, and remove VIPs using the Control System and the CLI.

Using Consistent Export Rules
Lists the default NFS export rules.

Export rules (stored in `conf/exports`) should be the same across all NFS nodes that are in the same VIP pool, and for nodes that are configured for the same VIP failover. The default version of `conf/exports` is as follows:

```
# Sample Exports file

# for non /mapr exports
# <Path>  <comma separated cldb addresses=host:port> <exports_control>

# for /mapr exports
# <Path> <exports_control>

#access_control -> order is specific to default
# list the hosts before specifying a default for all
#  a.b.c.d,1.2.3.4(ro) d.e.f.g(ro) (rw)
#  enforces ro for a.b.c.d & 1.2.3.4 and everybody else is rw

# special path to export clusters in mapr-clusters.conf. To disable
exporting,
# comment it out. to restrict access use the exports_control
#
/mapr (rw)

#to export only certain clusters, comment out the /mapr & uncomment.
# Note: this will cause /mapr to be unexported
#/mapr/clustername (rw)

#to export /mapr only to certain hosts (using exports_control)
#/mapr a.b.c.d(rw),e.f.g.h(ro)

# export /mapr/cluster1 rw to a.b.c.d & ro to e.f.g.h (denied for others)
#/mapr/cluster1 a.b.c.d(rw),e.f.g.h(ro)

# export /mapr/cluster2 only to e.f.g.h (denied for others)
#/mapr/cluster2 e.f.g.h(rw)

# export /mapr/cluster3 rw to e.f.g.h & ro to others
#/mapr/cluster2 e.f.g.h(rw) (ro)
```

Setting Up VIPs for NFSv3
Explains how to set up VIPs for NFS version 3 using either the Control System or the CLI.

last-updated: Apr 11, 2022

Note: The NFSv3 server (`mapr-nfsserver`) nodes cannot failover to NFSv4 server (`mapr-nfs4server`) nodes and vice versa. Ensure that different sets of VIPs are assigned for NFSv3 and NFSv4 server nodes. When running the `maprcli virtualip add` command to set up VIPs, list the MACs of the respective nodes so that the failover works properly (this is necessary when both NFSv3 and NFSv4 are going to be set up on the same cluster). The MACs should be mutually exclusive as both NFSv3 and NFSv4 servers cannot run on the same node.

Adding VIPs Using the Control System

You can use the Control System to specify a range of virtual IP addresses and assign them to the pool of servers that are running the NFS service. You can also restrict the assignment of virtual IP addresses to certain subnets.

Before following this procedure, make sure:

- You have installed NFS on at least three nodes (recommended).
- You have started the NFS gateway service on the servers to which you plan to assign VIPs.

1. Log in to the Control System, click **Services**, and go to the NFS service information page where you can configure VIPs for NFSv3 nodes.

   Note: The **Services** page is not available on the Kubernetes version of the Control System.

2. Click **Add Virtual IP** to display the **Add Virtual IP** page.
3. Enter the start of the VIP range in the **Starting Virtual IP** field.

   MapR distributes the VIPs in this range to the selected network interfaces. VIPs are automatically migrated between the network interfaces when failures occur.
4. Enter the end of the VIP range in the **Ending Virtual IP** field.

   If you are assigning only one VIP, you can leave the field blank. MapR distributes the VIPs in this range to the selected network interfaces. VIPs are automatically migrated between the network interfaces when failures occur.
5. Enter the Netmask for the VIP range in the **Netmask** field.

   For example: `255.255.255.0`. MapR assigns this netmask to the network interfaces along with the VIPs.
6. Specify whether (**Yes**) or not (**No**) to assign a particular VIP address to a specific server or MAC address. If **Yes**, enter the MAC address for the network interface to be assigned to the Starting Virtual IP address. The remaining VIP addresses from the same pool are assigned randomly.
7. Select one of the following:
   - **Use all network interfaces on all nodes that are running the NFS Gateway service** to set up VIPs that use all network interfaces on all the nodes running the NFS Gateway service.

     If additional NFS Gateway services are started, the network interfaces on their nodes will automatically become candidates for the VIPs in this range.
   - **Select network interfaces** to restrict the assignment of virtual IP addresses to certain subnets:

     A list of available and selected node names, physical IP addresses, and MAC addresses displays. Select from the:
     - Available list and click ▶ to move selection to **Selected** VIPs.
     - Selected list and click ◀ to remove from selected list of VIPs.

last-updated: Apr 11, 2022

See Designating NICs for MapR on page 844.

8. Confirm the actual VIP assignment by clicking **Save Changes**.
   It might take up to 40 seconds to assign the VIPs. If necessary, refresh the page in your browser to view the list of VIPs.

Adding VIPs Using the CLI and REST API

CLI

The basic command to set up VIPs is:

```
maprcli virtualip
add -netmask <netmask> -virtualip
<virtualip> -service nfs3 -json
```

REST

Send a request of type POST. For example:

```
curl -k -X POST 'https://<host>:8443/
rest/virtualip/add?
service=nfs3&netmask=<netmask>&virtual
ip=<vip>' --user mapr:mapr
```

For the complete list of required and optional parameters, see `virtualip add` on page 1904.

Setting Up VIPs for NFSv4
Describes how to setup Virtual IPs (VIPs) for high availability of NFSv4 servers, using either the Control System or the CLI.

Virtual IP addresses (VIPs) allow you to achieve high availability with failover when the NFS servers use them to connect to your cluster. When configuring VIPs for NFSv4 servers, ensure that you select NFSv4 server nodes only. MapR does not support failing over between NFSv3 and NFSv4 servers.

Adding VIPs Using the Control System

You can use the Control System to specify a range of virtual IP addresses and assign them to the pool of servers that are running the NFS service. You can also restrict the assignment of virtual IP addresses to certain subnets.

Before following this procedure, make sure that:

- You have installed NFS on at least three nodes (recommended).
- You have started the NFS gateway service on the servers to which you plan to assign VIPs.

1. Log in to the Control System and go to the NFS4 service information page where you can configure VIPs for NFSv4 nodes.
2. Click **Add Virtual IP** to display the **Add Virtual IP** page.
3. Enter the start of the VIP range in the **Starting Virtual IP** field.
   MapR distributes the VIPs in this range to the selected network interfaces. VIPs are automatically migrated between the network interfaces when failures occur.
4. Enter the end of the VIP range in the **Ending Virtual IP** field.
   If you are assigning only one VIP, you can leave the field blank. MapR distributes the VIPs in this range to the selected network interfaces. VIPs are automatically migrated between the network interfaces when failures occur.
5. Enter the Netmask for the VIP range in the **Netmask** field.
   For example: `255.255.255.0`. MapR assigns this netmask to the network interfaces along with the VIPs.

6. Specify whether (**Yes**) or not (**No**) to assign a particular VIP address to a specific server or MAC address. If **Yes**, enter the MAC address for the network interface to be assigned to the Starting Virtual IP address. The remaining VIP addresses from the same pool are assigned randomly.
7. Select one of the following:
   - **Use all network interfaces on all nodes that are running the NFS Gateway service** to set up VIPs that use all network interfaces on all the nodes running the NFS Gateway service.

     If additional NFS Gateway services are started, the network interfaces on their nodes automatically become candidates for the VIPs in this range.
   - **Select network interfaces** to restrict the assignment of virtual IP addresses to certain subnets:

     The system displays a list of available and selected node names, physical IP addresses, and MAC addresses. Do one of the following:
     - Select from the available list and click ▶ to move the selection to **Selected** VIPs.
     - Choose from the selected list and click ◀ to remove the chosen entries from the selected list of VIPs.

     See Designating NICs for MapR on page 844.
8. Confirm the actual VIP assignment by clicking **Save Changes**.

   It might take up to 40 seconds to assign the VIPs. If necessary, refresh the page in your browser to view the list of VIPs.

Setting up VIPs for NFSv4 Server from the Command-Line

To set up VIPs for NFSv4 server:

1. Add VIPs to the NFS server nodes in the cluster by running the `virtualip add` on page 1904 command.

   ```
   maprcli virtualip add -virtualip <vip> -virtualipend <vipend> -service
   nfs4 -netmask <netmask> -macs <mac>
   ```

   For example, for a range of virtual IPs use:

   ```
   maprcli virtualip add -virtualip 10.10.104.203 -virtualipend
       10.10.104.206 -service nfs4 -netmask 255.255.255.0 -macs
      "18:e7:28:2e:b0:80 18:e7:28:2e:2d:a0 18:e7:28:2e:2d:a8"
   ```

   For a single virtual IP, do not include the `-virtualipend` parameter. For example:

   ```
   maprcli virtualip add -virtualip 10.10.104.203 -service nfs4 -netmask
   255.255.255.0 -macs
      "18:e7:28:2e:b0:80 18:e7:28:2e:2d:a0 18:e7:28:2e:2d:a8"
   ```

   For the complete list of required and optional parameters, see `virtualip add` on page 1904.
2. Add the hostname for each VIP in the `/etc/hosts` file on all the nodes in the cluster.
3. Add the principal for each VIP and generate the keytab file. That is, repeat this step in the following order for each VIP:

   a) Add the principals for the following:
      - NFS server hostnames in the kerberos server.

- VIP hostname in the kerberos server.

b) Generate the keytab file, which contains entries for all the NFS server and VIP hostname principals.

4. Restart the `rpc.gssd` service on all the NFSv4 server nodes.
   To restart, run the following command:

```
service rpcgssd start
```

5. Mount the cluster.

Editing a VIP
Explains how to modify the Virtual IP (VIP) range using either the Control System or the CLI.
Editing a VIP Using the Control System

1. Log in to the Control System and go to the service information page for NFS.
2. Click the **VIP Range** to modify.
   The **Edit Virtual IP** page displays.
3. Modify changes to one or more of the following as needed.

Preferred MAC Address	The preferred MAC for this virtual IP. When an NFS server restarts, the MapR system attempts to move all of the virtual IP addresses that list a MAC address on this node as a preferred MAC to this node. If the new value is null, this field resets the preferred MAC value.
Select network interfaces	The list of MAC addresses that represent the NICs on the nodes to which the VIPs in the VIP range can be associated. Use this list to limit VIP assignment to NICs on a particular subnet when your NFS server is part of multiple subnets.

4. Click **Save Changes** for the changes to take effect.

Editing a VIP Using the CLI and REST API

CLI

The basic command to modify a VIP range is:

```
maprcli virtualip edit -netmask
<netmask> -virtualip <virtualip>
```

REST

Send a request of type POST. For example:

```
durl -k -X POST 'https://<host>:8443/
rest/virtualip/edit?
netmast<netmask>&virtualip=<vip>' --us
er mapr:mapr
```

For the complete list of required and optional parameters, see `virtualip edit` on page 1907.

Viewing the List of Virtual IPs
Explains how to view the list of VIPs using either the Control System or the CLI.
Viewing the List of VIPs Using the Control System

- Log in to the Control System and go to the service information page for NFS.
  The page displays the following:

Column Name	Column Description
VIP Range	The Virtual IP range.
Virtual IP	The VIP (in the range) of the associated node.
Node Name	The host name of the node.
Physical IP	The physical IP of the associated node.
MAC Address	The MAC address of the network interface that is assigned to the associated VIP.

You can add and remove VIPs.

Retrieving the List of VIPs Using the CLI and REST API

CLI

The basic command to retrieve a list of VIPs is:

```
maprcli virtualip list
```

REST

Send a request of type GET. For example:

```
curl -k -X GET 'https://<host>:8443/
rest/virtualip/list' --user mapr:mapr
```

For complete reference, see `virtualip list` on page 1908.

Removing a VIP

Explains how to remove a VIP range using either the Control System or the CLI.

Removing a VIP Range Using the Control System

1. Log in to the Control System and go the service information page for NFS.
2. Select the VIP range(s) to remove and click **Remove Virtual IP**.
   The **Remove Virtual IP** confirmation dialog displays.
3. Review the VIP range(s) to remove and click **Remove Virtual IP**.

Removing a VIP Range Using the CLI or REST API

The basic command to remove a VIP range is:

```
maprcli virtualip remove -virtualip <virtual IP>
```

For complete reference information, see `virtualip remove` on page 1912.

Accessing Data with NFS v3

Describes how MapR works with NFS v3.

Unlike other Hadoop distributions that only allow cluster data import or import as a batch operation, MapR lets you mount the cluster itself using NFS so that your applications can read and write data directly. MapR allows direct file modification and multiple concurrent reads and writes using POSIX semantics. With a NFS-mounted cluster, you can read and write data directly with standard tools, applications, and scripts. For example, you could run a MapReduce application that outputs to a CSV file, then import the CSV file directly into SQL using NFS.

MapR exports each cluster as the directory `/mapr/<cluster name>` (for example, `/mapr/my.cluster.com`). If you create a mount point with the local path `/mapr`, then Hadoop FS paths and NFS v3 paths to the cluster will be the same. This makes it easy to work on the same files using NFS v3

last-updated: Apr 11, 2022

and Hadoop. In a multi-cluster setting, the clusters share a single namespace, and you can see them all by mounting the top-level `/mapr` directory.

Warning: MapR uses version 3 of the NFS protocol. NFS version 4 bypasses the port mapper and attempts to connect to the default port only. If you are running NFS on a non-standard port, mounts from NFS version 4 clients time out. Use the `-o nfsvers=3` option to specify NFS v3.

You can mount the cluster on a Linux, Mac, or Windows client. Before you begin, make sure you know the hostname and directory of the NFS v3 share you plan to mount.

Starting, Stopping, and Restarting MapR NFSv3
Explains how to start, stop, and restart NFS version 3 using either the Control System or the CLI.
Starting, Stopping, and Restarting MapR NFSv3 Using the Control System

See:

- Starting the Services on the Cluster Using the Control System on page 830
- Stopping a Service on the Cluster Using the Control System on page 831
- Restarting the Services on the Cluster Using the Control System on page 832

Starting, Stopping, and Restarting MapR NFSv3 Using the CLI and REST API

NFSv3 Server	The command to stop, start, or restart MapR NFSv3 server is: `maprcli node services -nodes <node names> -nfs stop|start|restart`
REST	Send a request of type POST. For example: `curl -k -X POST 'https://<host>:8443/rest/node/services?nodes=<nodeNames>&nfs=stop|start|restart' -- user mapr:mapr`

Note: When NFS server is stopped, the VIPs associated with the server are released, and CLDB attempts to reassign the VIPs to other available NFS servers. For the complete list of parameters, see `node services` on page 1730.

Setting Up Aliases for NFS Exports

When provisioning MapR File System for various tenants, you can set up an alias for the path in MapR File System, rather than exporting the whole path, to mask the path from the users. Once the alias is set up, users will not be able to access or mount the path in MapR File System.

Aliases can be set up for the cluster, volume, and directory, but not for the root of the path in MapR File System (`/mapr`). To set up an alias for a path in MapR File System:

1. Open the NFS exports file in `/opt/mapr/conf/` directory.
2. Specify the alias name for the mount path using the following syntax:

   ```
   <path in MFS> /<alias name> <options>
   ```

   Here:

`<path in MFS>`	Refers to the MapR File System mount path. If this points to a:

	• Volume, the user can access the snapshots associated with the volume. • Directory, the user cannot access the snapshots.
`/<alias name>`	Refers to the alias name to use. If there are duplicate aliases in the file, the last entry will take effect and all other duplicate entries will be ignored. If the alias name is not specified, the path in MapR File System will be exported.
`<options>`	The list of available/supported options.

For example, suppose a MapR File System mount path of `/mapr/samplecluster/samplevolume` for tenant samplecustomer. To set up an alias, add the following to the exports file:

```
/mapr/samplecluster/samplevolume /samplecustomer (rw)
```

For example, to export a certain cluster, volume, or a subdirectory as an alias, comment out `/mapr` and add the following:

```
/mapr/clustername /alias1 (rw)
/mapr/clustername/vol /alias2 (rw)
/mapr/clustername/vol/dir /alias3 (rw)
```

Note: Only the alias will be visible/exposed to the NFS client.

3. Run the following command for the file changes to take effect:

```
/opt/mapr/bin/maprcli nfsmgmt refreshexports
```

4. Run the following command to export the path:

```
mount -t nfs nfsServer:/<alias_name> /localpath
```

Run this command once for each entry in the file.

The same export rules must be set up on all the NFS servers in the cluster to ensure that in the event of a node failure, the same aliases work with VIP failover.

Mounting NFS to MapR File System on a Cluster Node

You can *automatically* or *manually* mount NFS to the MapR File System on a cluster node.

Automatically Mount

Use this procedure to *automatically* mount NFS to MapR File System on the cluster *my.cluster.com* at the `/mapr` mount point.

1. Set up the mount point by creating the directory.

```
sudo mkdir /mapr
```


last-updated: Apr 11, 2022

2. Add the following line to `/opt/mapr/conf/mapr_fstab`:

```
<hostname>:/mapr /mapr hard,nolock
```

Note: The change to `/opt/mapr/conf/mapr_fstab` will not take effect until Warden is restarted.

Every time your system is rebooted, the mount point is automatically re-established according to the `mapr_fstab` configuration file.

Manually Mount

Use this procedure to *manually* mount NFS to MapR File System on the cluster *my.cluster.com* at the `/mapr` mount point.

1. Set up a mount point for a NFS share.

```
sudo mkdir /mapr
```

2. Mount the cluster via NFS.

```
sudo mount -o hard,nolock usa-node01:/mapr /mapr
```

Note: When you mount manually from the command line, the mount point does not persist after a reboot.

Mounting NFS on a Linux Client

Explains how to mount NFS on a Linux client either automatically at start up or manually.

You can *automatically* or *manually* mount NFS on a Linux client when your system starts up.

Automatically Mount

Use this procedure to *automatically* mount to NFS on a Linux client when your system starts up.

Add an NFS mount to `/etc/fstab`.

```
# device          mountpoint      fs-type      options      dump fsckorder
...
usa-node01:/mapr  /mapr_nfs/         nfs          rw           0    0
...
```

Manually Mount

Use this procedure to *manually* mount to NFS on a Linux client.

1. Install the NFS client.
   - `sudo yum install nfs-utils` (Red Hat or CentOS)
   - `sudo apt-get install nfs-common` (Ubuntu)
   - `sudo zypper install nfs-client` (SLES)

2. List the NFS shares exported on the server. For example:

```
showmount -e usa-node01
```

Note: If the NFS protocol is v4 only, the `showmount` command does not return the list of exported NFS shares. Instead, to view the export list, run the following command:

```
/opt/mapr/server/nfs4mgr list-exports
```

3. Set up a mount point for an NFS share. For example:

```
sudo mkdir /mapr_nfs/
```

4. Mount the cluster using NFS. Use the command as in the following example:

```
sudo mount -t nfs -o sec=mode vers=NFS_version usa-node01:/mapr /
mapr_nfs/
```

where mode is one of the following:

- krb5 for Kerberos version 5 authentication service.
- krb5i for Kerberos version 5 with integrity.
- krb5p for Kerberos version 5 with privacy.
- none for no authentication.

and NFS_version is either 3 or 4.

Restriction: You can use the `sec=mode` option only for NFS version 4. NFS version 3 does not support this option.

Tip: For the best performance, use NFS v4.0.

Use the `vers=4.0` parameter in the mount command. For example:

```
mount -t nfs -o sec=krb5,vers=4.0 usa-node01:/mapr /mapr_nfs/
```

For NFS v3, use the command as in the following example:

```
mount -t nfs -o vers=3 usa-node01:/mapr /mapr_nfs/
```

Note: The mount point does not persist after reboot when you mount manually from the command line.

5. List all mounted file systems to verify that the cluster is mounted. For example:

```
$ mount | grep nfs4
  usa-node01:/mapr on /mapr_nfs (nfs, nodev, nosuid, mounted by testUser)
```

Mounting NFS on a Mac Client
Describes how to mount a NFS server on a Mac client.

last-updated: Apr 11, 2022

Use this procedure to mount to the cluster *manually* from the command line:

1. Open a terminal. For example, you can click on **Launchpad > Open terminal**.
2. At the command line, enter the following command to become the root user:
   `sudo bash`
3. List the NFS shares exported on the server. For example:
   ```
   showmount -e usa-node01
   ```
4. Set up a mount point for an NFS share. For example:
   ```
   sudo mkdir /mapr_nfs/
   ```
5. Mount the cluster using NFS. Use the command as in the following example:
   ```
   sudo mount -t nfs -o vers=3 usa-node01:/mapr /mapr_nfs/
   ```
   Note: The mount point does not persist after reboot when you mount manually from the command line.
6. List all mounted file systems to verify that the cluster is mounted. For example:
   ```
   $ mount | grep nfs
   usa-node01:/mapr on /mapr_nfs (nfs, nodev, nosuid, mounted by testUser)
   ```

Mounting NFS on a Windows Client
Describes how to mount an NFS share on a Windows client, and configure the relevant user and group IDs.

To set up the Windows NFS client, mount the cluster, map a network drive, and configure the user ID (UID) and group ID (GID). The Windows client must access NFS using a valid UID and GID from the Linux domain. Mismatched UID or GID results in permission problems when MapReduce jobs try to access files that were copied from Windows over an NFS share.

Due to Windows directory caching, the `.snapshot` directory may not appear in the root directory of each volume. As a workaround, you can force Windows to re-load the volume's root directory by updating its modification time (for example, by creating an empty file or directory in the volume's root directory).

With Windows NFS clients, use the `-o nolock` option on the NFS server to prevent the Linux NLM from registering with the portmapper. The native Linux NLM conflicts with the MapR NFS server.

Complete the following steps to mount NFS on a Windows client:

1. Mount the Cluster.

Windows 10 Enterprise	Complete the following steps for *Windows 10 Enterprise* **a.** Open **Start > Control Panel > Programs**. **b.** Select **Turn Windows features on or off**. **c.** Select **Services for NFS**. **d.** Click **OK**.

e. Enable write permissions for the anonymous user as the default options only grant read permissions when mounting a UNIX share using the anonymous user.

To grant write permissions, make a change to the Windows registry by performing the following steps:

1. Open `regedit` by typing it in the search box and pressing **Enter**.
2. Create a new **New DWORD (32-bit) Value** inside the `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\ClientForNFS\CurrentVersion\Default` folder named `AnonymousUid` and `AnonymousGid` and assign the UID and GID found on the UNIX directory as shared by the NFS system.

f. Restart the NFS client or reboot the machine to apply the changes.

g. Mount the cluster and map it to a drive using the Map Network Drive tool or from the command line.

```
mount -o anon usa-node01:/mapr z:
```

For more information, see step 2.

Windows 7 Enterprise

Complete the following steps for *Windows 7 Ultimate* or *Windows 7 Enterprise*

a. Open **Start > Control Panel > Programs**.

b. Select **Turn Windows features on or off**.

c. Select **Services for NFS**.

d. Click **OK**.

e. Mount the cluster and map it to a drive using the Map Network Drive tool or from the command line.

```
mount -o nolock usa-node01:/mapr
z:
```

For more information, see step 2.

Other Versions of Windows

Complete the following steps for all other versions of Windows:

a. Download and install Microsoft Windows Services for Unix (SFU). You only need to install the NFS Client and the User Name Mapping.

last-updated: Apr 11, 2022

b. Configure the user authentication in SFU to match the authentication used by the cluster (LDAP or operating system users). You can map local Windows users to cluster Linux users, if desired.

c. Once SFU is installed and configured, mount the cluster and map it to a drive using the Map Network Drive tool or from the command line.

```
mount -o nolock usa-node01:/mapr
z:
```

For more information, see step 2.

2. Map a network drive with the Map Network Drive tool.

a) Open **Start > My Computer**.

b) Select Tools > Map Network Drive.

c) In the Map Network Drive window, choose an unused drive letter from the **Drive** drop-down list.

d) Specify the folder by browsing for the MapR cluster, or by typing the hostname and directory into the text field.

e) Browse for the MapR cluster or type the name of the folder to map. This name must follow UNC. Alternatively, click **Browse…** to find the correct folder by browsing available network shares.

f) Select **Reconnect at login** to reconnect automatically to the MapR cluster whenever you log into the computer.

g) Click **Finish**.

3. Configure the UID and GID for NFS access.

System that is part of Active Directory Domain

For a system that is part of the Active Directory Domain, you must instruct the NFS client to access an AD server to get `uidNumber` and `gidNumber`.

a. Ensure that the AD Users schema has auxiliary class `posixAccount`.

b. Populate the AD `uidNumber` and `gidNumber` fields with the matching `uid` and `gid` from Linux.

c. Configure the NFS client to look up `uid` and `gid` in the AD DS store.

d. Refer to details here: `http://technet.microsoft.com/en-us/library/hh509016(v=ws.10).aspx`.

System not using Active Directory

For a standalone Windows 7 or Vista machine (not using Active Directory), Windows always uses its configured anonymous UID and GID for NFS access, which by default are `-2`. However, you can configure Windows to use specific values, which results in being able to access NFS using those values.

last-updated: Apr 11, 2022

The UID and GID values are set in the Windows Registry and are global on the Windows NFS client box. This solution might not work well if your Windows box has multiple users who each need access to NFS with their own permissions, but there is no obvious way to avoid this limitation.

The values are stored in the registry path `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\ClientForNFS\CurrentVersion\Default`. The two DWORD values are `AnonymousUid` and `AnonymousGid`. If they do not exist, you must create them.

Refer to details here: `https://docs.microsoft.com/en-us/archive/blogs/msdn/sfu/can-i-set-up-user-name-mapping-in-windows-vista`.

4. (Optional) Deactivate the `nlockmgr` service.

If the `nlockmgr` service is active on a Windows machine, attempts to mount a MapR NFS share fail with the following message:

```
C:\Users\administrator.Client1>mount -o nolock -u:mapr -p:mapr
ClusterNode1:/mapr / g:
Network Error - 53
Type 'NET HELPMSG 53' for more information.
```

a) Run the `rpcinfo` command to confirm that the `nlockmgr` service is active.

```
C:\Users\administrator.Client1>rpcinfo -p ClusterNode1
program version protocol     port
-------------------------------------------------
100000       4      tcp      111    portmapper
100024       1      udp    60588    status
100007       2      udp      817    ypbind
100021       1      udp    47016    nlockmgr
100021       3      udp    47016    nlockmgr
100021       4      udp    47016    nlockmgr
100021       1      tcp    34254    nlockmgr
100021       3      tcp    34254    nlockmgr
100021       4      tcp    34254    nlockmgr
```

b) Check the output for the presence of `nlockmgr`. To deregister `nlockmgr` services on the node, use the `-d` switch in `rpcinfo` on the MapR node.

```
rpcinfo -d 100021 1
rpcinfo -d 100021 2
rpcinfo -d 100021 3
rpcinfo -d 100021 4
```

c) Re-check `rpcinfo` output to verify that no `nlockmgr` services are registered. The NFS mount completes successfully at this point.

```
C:\Users\administrator.Client1>mount -o nolock -u:mapr -p:mapr
ClusterNode1:/mapr/ Z:
Z: is now successfully connected to ClusterNode1:/mapr/
The command completed successfully.
```

Configuring Access When ACES are set

Some NFS clients, such as the Microsoft native Windows NFSv3 client, check mode bits to determine if access is allowed even before contacting the NFS server. If *ACE*s are set on a directory or file, the client-side permission checks based solely on mode bits prevent the client from accessing the file or directory. You can set the value for the `WindowsAceSupport` property to `true` in the `nfsserver.conf` `on page 2207` file to allow the Windows client access to the file or directory. The default value for this property is `false`, and denies access to the client even before contacting the NFS server.

When the `WindowsAceSupport` property value is set to `true`, MapR returns mode bits 777 to the client if *ACE* is set on the file or directory, thus allowing the client to establish a connection to the server. However, when the client actually tries to read or write from the server, MapR performs permission checks against the mode bits and *ACE*s on the directory and/or file, ensuring proper access.

Note: When the `WindowsAceSupport` property value is set to `true`:

- Tools that visually display access information might show read/write access for users who do not have that access.
- Files that are not executables might appear executable.
- You cannot use the NFSv3 to access an NFSv4 server, because the NFSv4 server only supports the v4 protocol.

Configuring the Linux NFS Client

Describes how to set the optimal number of RPC requests to the NFS server.

The default RPC requests configuration can negatively impact performance and memory. To avoid performance and memory issues, configure the number of outstanding RPC requests to the NFS server to be 128.

Perform the following steps as the `root` user on each NFS client machine:

1. To enable the configuration to persist after a reboot of the NFS client machine, issue the following commands to create the `sunrpc.conf` file under `/etc/modprobe.d` with the recommended configuration:

```
echo "options sunrpc tcp_slot_table_entries=128" >> /etc/modprobe.d/
sunrpc.conf
echo "options sunrpc tcp_max_slot_table_entries=128" >> /etc/modprobe.d/
sunrpc.conf
```

2. To enable the configuration to take effect after you remount the NFS client to the NFS gateway, issue the following echo commands:

```
echo 128 > /proc/sys/sunrpc/tcp_slot_table_entries
echo 128 > /proc/sys/sunrpc/tcp_max_slot_table_entries
```

3. Remount the NFS client to the NFS gateway. For example, the following commands unmount and mount NFS assuming that the cluster is mounted at `/mapr`:

```
umount /mapr
mount -o hard,nolock <hostname>:/mapr /mapr
```

Note: Failure to configure this property may result in the following error in `/opt/mapr/logs/nfsserver.log`:

```
ERROR nfsserver[38960] fs/nfsd/requesthandle.cc:791 0.0.0.0[0]
cannot allocate more OncRpcContexts: [numDropped=2556001]
dropping connection from nfsc=10.13.64.225:0
```

Tip: For CentOS, after the reboot of the node, if the `/proc/sys/sunrpc` directory is not available or if `rpcidmapd` is not running, start the `rpcidmapd` service using the following command:

```
service rpcidmapd start
```

Accessing Data with NFS v4

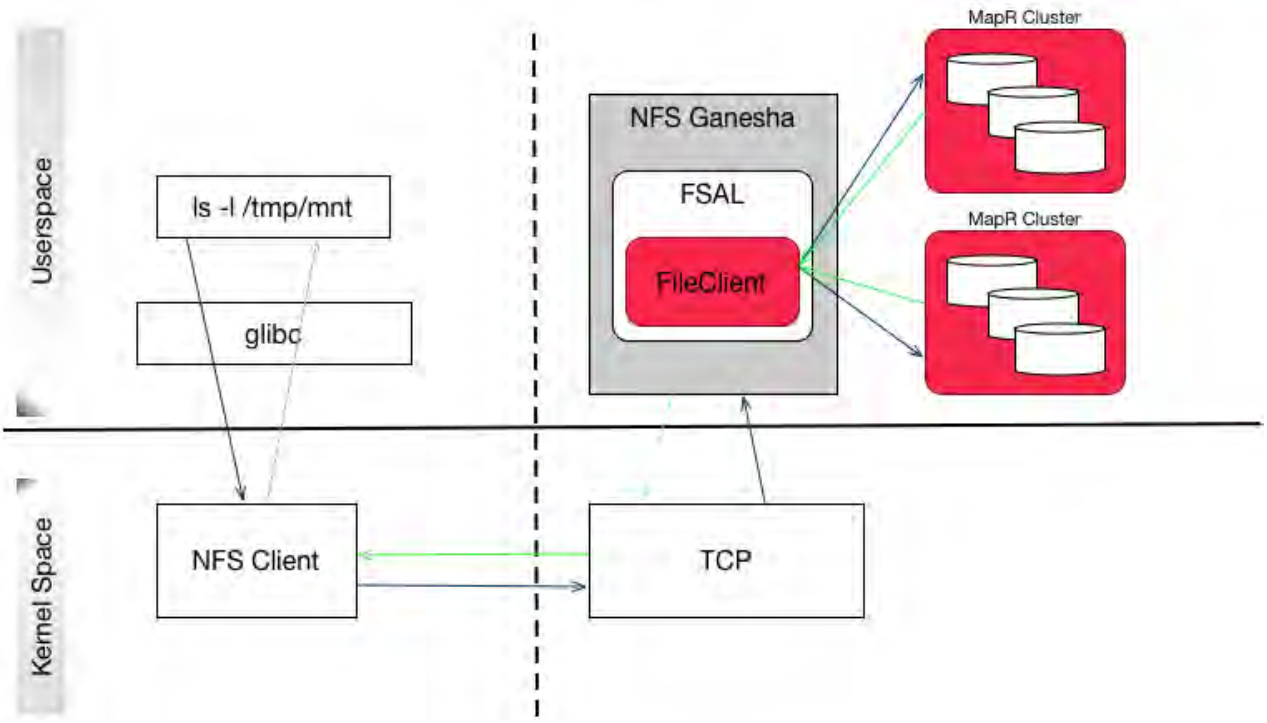
Describes how MapR works with the NFS v4 protocol. Presents an overview of the process flow to read and write MapR processes with NFS v4, and a list of NFS v4 features that MapR does not support.

MapR lets you mount the cluster using NFS v4 so that your applications can read and write data directly. MapR allows direct file modification and multiple concurrent reads and writes using POSIX semantics. With an NFS v4-mounted cluster, you can read and write data directly with standard tools, applications, and scripts. For example, you could run a MapReduce application that outputs to a CSV file, then import the CSV file directly into SQL using NFS v4.

MapR uses NFS Ganesha for supporting NFS v4 features. NFS Ganesha is an Open Source userspace implementation of the NFS v4 server. MapR version 6.1 uses NFS Ganesha version 2.3, while HPE Ezmeral Data Fabric 6.2 uses an upgraded version 3.3 of NFS Ganesha.

The MapR NFS v4, running as a userspace process, registers callbacks with NFS Ganesha through the File System Abstraction Layer (FSAL), which is a shared library (`libfsalmapr.so`). NFS Ganesha loads and uses this library whenever the MapR File System is exported/mounted. The FSAL, in turn, uses FileClient (`libMapRClient.so`) to connect to the cluster.

The following diagram illustrates how the MapR processes read and write operations to the MapR cluster using NFS v4. When the user enters a command (such as `ls`), the NFS client submits the request over TCP to the MapR NFS v4 server. The NFS v4 server uses the MapR FileClient to perform the requested operation on the cluster and returns the response to the NFS v4 client over TCP.



MapR exports each cluster as the directory `/mapr/<cluster name>` (for example, `/mapr/my.cluster.com`). If you create a mount point with the local path `/mapr`, then Hadoop FS paths and NFS v4 paths to the cluster are the same. This makes it easy to work on the same files using NFS v4 and Hadoop. In a multi-cluster setting, the clusters share a single namespace, and you can see them all by mounting the top-level `/mapr` directory.

For NFS v4, MapR also requires alias or pseudo-path, which when specified masks the mount path from the NFS v4 client. MapR's NFS v4 server provides a pseudo-filesystem where only the exported volumes are visible. This is especially useful in scenarios where one or more volumes in the hierarchy should be hidden and not be visible. For more information, see NFS v4 RFC.

Unsupported NFS v4 Features

MapR does not currently support the following NFS v4 features:

- pNFS
- Delegations
- Mandatory locking
- Lock upgrades and downgrades
- Deny share
- *ACL*
- Namespaces
- Persistent reply cache
- Data retention

- Attributes such as time_access, FATTR4_ARCHIVE, FATTR4_FILES_AVAIL, FATTR4_FILES_FREE, FATTR4_FILES_TOTAL, FATTR4_FS_LOCATIONS, FATTR4_MIMETYPE, FATTR4_QUOTA_AVAIL_HARD, FATTR4_QUOTA_AVAIL_SOFT, FATTR4_QUOTA_USED, FATTR4_TIME_BACKUP, and FATTR4_ACL

Configuring NFSv4 Server

You can configure NFSv4 server by setting the values for the parameters in the `/opt/mapr/conf/nfs4server.conf` file. The configuration parameters are defined within blocks in the file. The following sections describe the blocks and required parameters (within each block) for the MapR NFSv4 server.

NFS_CORE_PARAM

Contains the general settings for the daemon. The parameters in this block should not be modified.

`Clustered`	The value is `false`. Do not modify this parameter.
`Plugins_Dir`	The directory for the FSAL libraries. The value is `/opt/mapr/lib`. Do not modify this parameter.
`DRC_TCP_Size`	The maximum number of results stored in the DRC. The default value is `16`.
`DRC_TCP_Recycle_Expire_S`	The amount of time after which to expire results stored in DRC. The default value is `60` seconds.
`Dirent_Entries_Track`	Specifies whether (`true`) or not (`false`) to monitor dirent entries. If `true`, the process restarts if the number of dirent entries exceeds limit.
`Num_Log_Files`	The maximum number of log files. The default value is `1`.
`Max_Logfile_Size`	The maximum amount of space for each log file. The default value is `1073741824`. The total amount of disk space for log files is calculated using the following: `Num_Log_Files * Max_LogFile_Size` For example, suppose `Num_Log_Files = 32` and `Max_LogFile_Size = 1GB`. Then, the total disk space for log files is 32GB.

`Enable_RQUOTA`	Specifies whether (`true`) or not (`false`) to enable support for remote quotas. The default value is `false`.
`NFS_Protocols`	The supported NFS protocols. The only supported value is 4.

NFSV4

Contains settings for NFSv4 protocol. The following parameter should not be modified.

`Delegations`	boolean	Specifies whether delegation is supported. The default value is `false` and should not be modified (cannot be set to `true`) as delegation is not supported.
`Dirent_Cache_Threshold`	128	The threshold for caching directory entries. If directory entries exceed threshold, the entries are not cached; caching is enabled only if entries are below this threshold. **Note:** This should be used only if `readdirplus` is `true`.

EXPORT_DEFAULTS

Contains default values for all subsequent EXPORT blocks. The settings in subsequent EXPORT blocks can override these default values on a per export basis.

`Anonymous_Uid`	The anonymous UID. The default value is `-2`, which is converted to a 32 bit unsigned integer (4294967294) when root squash is enabled.

`Anonymous_Gid`	The anonymous GID. The default value is `-2`, which is converted to a 32 bit unsigned integer (4294967294) when root squash is enabled.
`Protocols`	The supported NFS protocols. The default value is `4`. This cannot be changed.

EXPORT

Contains settings for exporting a filesystem.

`SecType`	The comma-separated list of supported authentication flavors for the export or the type of security. Value can be comma-separated list of: • krb5 — authentication • krb5i — integrity • krb5p — privacy  **Note:** This *must* be specified if you want the clients to use kerberos ticket for secure access.
`Path`	(Required) The (cluster) path to export via NFS. The path should have a leading slash. If just `/mapr` is specified as the path, all the clusters (listed in the `mapr-clusters.conf` file) will be visible. To export only a specific cluster, specify the complete path to the cluster to export.  **Note:** Exporting will not be successful if extra forward slash (`/`) characters are in the path. For example, the following path will not be exported because of the extra slash (shown in bold) in the path: `/mapr/Test3//ATS-VOLUME`

last-updated: Apr 11, 2022

`Pseudo`	(Required for NFSv4 protocol for every directory or volume to export) The pseudo path, which when specified, masks the mount path from the NFS client, for the NFSv4 exports. MapR's NFSv4 server provides a pseudo-filesystem where only the exported volumes are visible. This is especially useful in scenarios where one or more volumes in the hierarchy should be hidden and not be visible. When mounting with NFSv4, use the pseudo path. Value can be the volume or directory path.
`Export_Id`	(Required) The tag used to set the ID for the export or the unique ID to associate with each export. Value should not be 0.  **Note:** The export ID associated with each export must be the same across all NFSv4 servers on the cluster.
`Clients`	The list of clients these export permissions apply to. Clients may be specified by hostname, IP address, netgroup, CIDR network address, host name wild card, or simply "*" to apply to all clients.

`Squash`	Specifies whether to enable or disable root squashing. By default, root squashing is disabled. If root squash is enabled, the values substituted for the root user will be anonymous user (`Anonymous_Uid` and `Anonymouos_Gid`); that is, the UID and GID of a file created will not be `nfsnobody` because the default value of -2 is converted to a 32 bit unsigned integer (4294967294) instead of the 16 bit equivalent (65534), which is the value of nfsnobody.
`Access_Type`	(Required) The type of access on the mount point. Valid values include: • RO — for read-only mount point • RW — for read/write mount point • MDONLY — for read/write access to metadata only • MDONLY_RO — for read-only access to metadata only
FSAL	(Required) The filesystem to use. Value must be MAPR to use the MapR File System `libfsalmapr.so` library, which contains the shared library (`libMapRClient`) and the callbacks.

LOG

Contains configuration for logging. The default log level is INFO. Value can be one of the following:

- FATAL
- MAJ
- CRIT
- WARN
- EVENT
- INFO

last-updated: Apr 11, 2022

- DEBUG
- MID_DEBUG
- FULL_DEBUG

MAPRFS

Contains configurations for NFS gateway access to MapR filesystem.

Parameter	Default Value	Description
`log_path`	`/opt/mapr/logs/nfs4`	Path for the log files.
`ra_sessions`	`5`	Number of parallel read ahead sessions per client library (`libMapRClient.so`). Each open file acts as one read ahead session. For example, if value is set to `5`, up to `5` files can have read ahead sessions per client library (`libMapRClient.so`). To disable read ahead sessions, set value to `0`. **Note:** The number of client libraries is 3 by default and cannot be configured.

last-updated: Apr 11, 2022

Parameter	Default Value	Description
`flush_inline`	`true`	Specifies whether or not to flush all writes inline. Value can be: • `true` — flush all writes inline • `false` — disable inline flushing If enabled (default), writes are sent to server directly. If disabled, for all open files, the buffer is flushed automatically every 3 seconds or when it reaches 64KB.
`fast_local_directio`	`false`	Specifies whether to optimize or disable NFS client for local direct IO. Value can be: • `true` — optimize • `false` — disable
`nfs_track_memory`	`false`	Specifies whether to enable (`true`) or disable (`false`) memory tracking for NFS.

Parameter	Default Value	Description
`hb_interval`	`5`	The interval (in seconds) for sending heartbeat to CLDB to allow CLDB to determine whether server is running. The CLDB will declare the NFS gateway dead when it loses about 8 heartbeats in a row and will trigger a failover.
`req_threshold`	`5`	The amount of time (in seconds) for processing requests. If the threshold is exceeded, warnings will be logged.

Parameter	Default Value	Description
`client_lib_path`	`/tmp/nfs4`	The location for the client library (`libMapRClient`). **Note:** To install and use NFSv4 and FUSE-based POSIX client on the same node, ensure that the path for the client library for the NFSv4 and FUSE-based POSIX client is not `/tmp`. Specify a different location for the client libraries. For example, `/tmp/nfs4lib`.
`readdirplus`	`true`	Specifies whether (`true`) or not (`false`) to enable extended read from the directory. If enabled (`true`), each entry returns the name, the file ID, attributes (including the field), and file handle.

last-updated: Apr 11, 2022

Note:

- The `libnfsidmap` must be configured to use `nsswitch`, a translation mechanism for mapping names to IDs, in the `/etc/idmapd.conf` file.
- The NFSv3 (mapr-nfsserver) nodes cannot failover to NFSv4 server nodes and vice versa. Ensure that different set of VIPs are assigned for NFSv3 and NFSv4 server nodes. When running the `maprcli virtualip add` command to set up VIPs, list the MACs of the respective nodes so that the failover works properly (this is necessary if both NFSv3 and NFSv4 are going to be set up for same cluster). The MACs should be mutually exclusive as both NFSv4 and NFSv3 servers cannot run on the same node.

Sample Configurations

You can refer to the following sections, which contain blocks for the various required configurations using sample values.

Configuration for Supported NFS Protocols

To specify the supported NFS protocols, set the value for the `NFS_Protocols` parameter in the `/opt/mapr/conf/nfs4server.conf` file. For example, for NFSv4 protocol, in the `/opt/mapr/conf/nfs4server.conf` file, set the value for the `NFS_Protocols` parameter as shown (in bold) below.

```
NFS_CORE_PARAM
{
        Plugins_Dir = /opt/mapr/lib;
        NFS_Protocols = 4;
        Clustered = false;
}
```

Note: The only supported protocol is `4`.

For NFSv4, the `showmount` command does not return the list of exported NFS shares.

Configuration for Mounting the Cluster

Add the `MAPRFS` block in the `/opt/mapr/conf/nfs4server.conf` file as shown below.

```
MAPRFS
{
  #Directory path where nfsv4 logs should be stored
  log_path = /opt/mapr/logs/nfs4;

  #Set number of readahead sessions
  ra_sessions = 5;

  #Flush all writes inline
  flush_inline = true;

  #Optimize for local direct writes
  fast_local_directio = false;

  #Set security ticket file
  tkt_location = /tmp/maprticket_XXX;

  #Hearbeat interval for NFSv4 (in seconds)
  hb_interval = 5;

  #Request threshold, logs warning if any request takes more time (in
seconds)
```

```
  req_threshold = 5;
}
```

Configuration for Exporting the File System

Modify the EXPORT block in the `/opt/mapr/conf/nfs4server.conf` file as shown (in bold) below. The following sample block shows a standard configuration where the exported path and the actual path are the same.

```
EXPORT
{
     # Export Id (mandatory, each EXPORT must have a unique Export_Id)
     Export_Id = 77;

     # Exported path (mandatory)
     Path = /mapr;

     # Pseudo Path (required for NFS v4)
     Pseudo = /mapr;
     Squash = No_Root_Squash;

    # Required for access (default is None)
    # Could use CLIENT blocks instead
     Access_Type = RW;

    # Security type (krb5,krb5i,krb5p)
     SecType = krb5;

    # Exporting FSAL
      FSAL {
      Name = MAPR;
    }

}
```

Note: If you change anything in the export block, restart NFSv4 service and remount the path.

Configuration for Pseudo Path

To mask the path to the volume from the client, set the pseudo path. For the pseudo path, you can specify a value that is different from the path parameter to hide the true path name. To hide the full path to volumes and/or directories, specify the complete path in the EXPORT block.

For example, modify the EXPORT block in the `/opt/mapr/conf/nfs4server.conf` file as shown (in bold) below to mask the path and show only the name of the volume to the client. Note that the following sample block shows a pseudo path that is different from the exported path.

```
EXPORT
{
    # Export Id (mandatory, each EXPORT must have a unique Export_Id)
    Export_Id = 77;

    # Exported path (mandatory)
    Path = /mapr;

    # Pseudo Path (required for NFS v4)
    Pseudo = /vol1;
    Squash = No_Root_Squash;

    # Required for access (default is None)
    # Could use CLIENT blocks instead
```

```
     Access_Type = RW;

    # Security type (krb5,krb5i,krb5p)
     SecType = krb5;

   # Exporting FSAL
      FSAL {
      Name = MAPR;
    }

}
```

Configuration for Security

The NFS client to NFS Server can be secured using Kerberos. Before configuring Kerberos to work with MapR, modify the `/opt/mapr/conf/nfs4server.conf` file to specify the security type. For example, modify the `EXPORT` block in the `/opt/mapr/conf/nfs4server.conf` file as shown (in bold) below.

```
EXPORT
{
    # Export Id (mandatory, each EXPORT must have a unique Export_Id)
    Export_Id = 77;

    # Exported path (mandatory)
    Path = /mapr;

    # Pseudo Path (required for NFS v4)
      Pseudo = /vol1;
      Squash = No_Root_Squash;

    # Required for access (default is None)
    # Could use CLIENT blocks instead
      Access_Type = RW;

    # Security type (krb5,krb5i,krb5p)
    SecType = krb5;

    # Exporting FSAL
       FSAL {
        Name = MAPR;
      }

}
```

The NFSv4 server uses the ticket in `/opt/mapr/conf` directory, if it is present, to secure communication between the NFS server and the MapR cluster.

Configuration for Clients

You can add a client block to the NFSv4 server configuration specifying the list of clients to which the export permissions apply. Clients may be specified by hostname, IP address, netgroup, CIDR network address, host name wild card, or simply "*" to apply to all clients. For example:

```
EXPORT
{
     # Export Id (mandatory, each EXPORT must have a unique Export_Id)
     Export_Id = 77;

     # Exported path (mandatory)
     Path = /mapr;
```


last-updated: Apr 11, 2022

```
     # Pseudo Path (required for NFS v4)
     Pseudo = /mapr;

    # Defining the clients who are allowed to export
      CLIENT
    {
     # Required for access (default is None)
     Clients=192.168.0.10, 192.168.1.0/8;
     Access_Type = RW;
     Squash = No_root_squash;
     SecType=krb5;
    }
     # Exporting FSAL
     FSAL{
         Name = MAPR;
     }
}
```

Configuration for NFS Ganesha Debug Logging

Add the following block in the `/opt/mapr/conf/nfs4server.conf` file.

```
LOG {
  COMPONENTS {
    ALL = DEBUG;
  }
}
```

Default NFSv4 Server Configuration File

The `nfs4server.conf` file is available in `/opt/mapr/conf` directory.

```
      LOG
{
  COMPONENTS {
    ALL = INFO;
  }

  FORMAT {
    EPOCH = false;
    CLIENTIP = true;
    HOSTNAME = false;
    PROGNAME = false;
    FILE_NAME = false;
    LINE_NUM = true;
    FUNCTION_NAME = true;
    COMPONENT = false;
    LEVEL = false;
    time_format = syslog_usec;
  }
}

NFSV4
{
  #Delegation is not supported.
  Delegations = false;

  #Dirent cache threshold. Use only when readdirplus is true
  #Dirent_Cache_Threshold = 128;
}
```


last-updated: Apr 11, 2022

```
NFS_CORE_PARAM
{
  Plugins_Dir = /opt/mapr/lib;

  Clustered = false;

  # Max number of results stored in DRC
  DRC_TCP_Size = 16;

  # Expire DRC after 60 seconds (if refcount is zero)
  DRC_TCP_Recycle_Expire_S = 60;

  # Only NFSv4 is supported. showmount will not work
  NFS_Protocols = 4;

  # RQUOTA protocol is not supported
  Enable_RQUOTA = false;

  # To set number of Nfs4server logs
  Num_Log_Files = 1;

  # Total disk space usage for logs = Num_Log_Files * Max_LogFile_Size
  # If Num_Log_Files = 32 and Max_LogFile_Size = 1GB,then disk space used
for logs = 32 GB.
  Max_Logfile_Size = 1073741824;

  # Monitor dirent entries (process restarts if number of entries beyond
limit, if true
  Dirent_Entries_Track = true;
}

MAPRFS
{
  #Set number of readahead sessions
  #ra_sessions = 5;

  #Flush all writes inline
  #flush_inline = true;

  #Optimize for local direct writes
  #fast_local_directio = false;

  #Enable/Disable memory tracking for nfs
  nfs_track_memory = false;

  #Sets client debug level, values are fatal, error, warn, info, debug
  mapr_log_debug_level = error;

  #Hearbeat interval for NFSv4 (in seconds)
  #hb_interval = 5;

  #Request threshold, logs warning if any request takes more time (in
seconds)
  #req_threshold = 5;

  #Specify the folder to copy libMapRClient
  #client_lib_path="/tmp/nfs4";

  #Readdirplus support
  #readdirplus = true;
}

#EXPORT_DEFAULTS
#{
```


last-updated: Apr 11, 2022

```
  #Default value for anonymous uid/gid is -2. Should be configured to
  #nfsnobody/nobody uid/gid if required
  #Anonymous_Uid = -2;

  #Anonymous_Gid = -2;

  #Supported NFS protocols. Currently only v4 is supported.
  #Protocols = 4;
#}

EXPORT
{
  # Export Id (mandatory, each EXPORT must have a unique Export_Id)
  Export_Id = 30;

  # Exported path (mandatory)
  Path = /mapr;

  # Pseudo Path (required for NFS v4)
  Pseudo = /mapr;

  Squash = No_Root_Squash;

  # Required for access (default is None)
  # Could use CLIENT blocks instead
  Access_Type = RW;

  # Security type (krb5,krb5i,krb5p)
  #SecType = krb5;

  # Exporting FSAL
  FSAL {
    Name = MAPR;
  }

  #SuperUser_Uid = 0;
}
```

Configuring NFSv4 Server for Kerberos
Describes how to configure and use NFSv4 on Kerberos.

You can configure MapR NFSv4 server to use Kerberos-based authentication. MapR supports configuration of NFSv4 server for Kerberos with Active Directory server and Kerberos with LDAP. You can also configure MapR NFSv4 server to work with other Kerberos installations. Before configuring MapR NFSv4 server for Kerberos, you must have performed the following:

- Installed packages for Kerberos server.
- Installed NFSv4 server. See Installing MapR NFS on page 386 for more information.
- Installed packages for Kerberos client.

Note: The steps in this section assume a Linux-based Kerberos environment, and the specific commands for your environment may vary. Please consult with your Kerberos administrator for assistance.

Configure NFSv4 Server for Kerberos with Active Directory Server

The following procedure describes how to configure the MapR NFSv4 server to work with the Kerberos available with Active Directory server. Before configuring the MapR NFSv4 server, ensure that Active

last-updated: Apr 11, 2022

Directory server is installed and all the nodes on the cluster have joined that Active Directory server. The following procedure requires the NFSv4 server to run under user `mapr` and group `maprgrp`.

1. In an Active Directory server environment, join the cluster nodes to the Active Directory server.

   Follow the sample procedure here or consult with your system administrator for assistance with installing and joining the nodes to Active Directory server.

2. Check if Kerberos tickets for host and NFS service principal are present, by running the following command:

```
# klist
klist: No credentials cache found (filename: /tmp/krb5cc_0)
```

3. Ensure host principal is available by checking to see if existing keys are present on the node.

   For example, when you run the following command, the output should look similar to the following output for `nfs4ad.com` domain:

```
# klist -kt
Keytab name: FILE:/etc/krb5.keytab
KVNO Timestamp           Principal
---- ------------------- ------------------------------------------------------

   2 04/10/2018 23:51:24 host/atsqa4-161.nfs4ad.com@NFS4AD.COM
   2 04/10/2018 23:51:24 host/ATSQA4-161@NFS4AD.COM
   2 04/10/2018 23:51:24 host/atsqa4-161.nfs4ad.com@NFS4AD.COM
   2 04/10/2018 23:51:24 host/ATSQA4-161@NFS4AD.COM
   2 04/10/2018 23:51:24 host/atsqa4-161.nfs4ad.com@NFS4AD.COM
   2 04/10/2018 23:51:24 host/ATSQA4-161@NFS4AD.COM
   2 04/10/2018 23:51:25 host/atsqa4-161.nfs4ad.com@NFS4AD.COM
   2 04/10/2018 23:51:25 host/ATSQA4-161@NFS4AD.COM
   2 04/10/2018 23:51:25 host/atsqa4-161.nfs4ad.com@NFS4AD.COM
   2 04/10/2018 23:51:25 host/ATSQA4-161@NFS4AD.COM
   2 04/10/2018 23:51:25 ATSQA4-161$@NFS4AD.COM
   2 04/10/2018 23:51:25 ATSQA4-161$@NFS4AD.COM
   2 04/10/2018 23:51:25 ATSQA4-161$@NFS4AD.COM
   2 04/10/2018 23:51:25 ATSQA4-161$@NFS4AD.COM
   2 04/10/2018 23:51:25 ATSQA4-161$@NFS4AD.COM
```

4. Generate the host ticket by running the `kinit` command.

   For example:

```
[root@atsqa4-161 ~]# kinit -k ATSQA4-161$
[root@atsqa4-161 ~]# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: ATSQA4-161$@NFS4AD.COM
Valid starting       Expires              Service principal
04/11/2018 03:04:38  04/11/2018 13:04:38  krbtgt/NFS4AD.COM@NFS4AD.COM
        renew until 04/18/2018 03:04:38
```

5. Add NFS service principal entry for the host in the AD server by running the `setspn` command.
   For example, for `nfs4ad.com` domain, run the following command:

```
C:\Users\Administrator>setspn -A nfs/atsqa4-161.nfs4ad.com mapr
Checking domain DC=nfs4ad,DC=com
Registering ServicePrincipalNames for CN=mapr,CN=Users,DC=nfs4ad,DC=com
        nfs/atsqa4-164.nfs4ad.com
Updated object
```

6. Get the latest service ticket for the host from the AD server by running the `kvno` command.
   For example:

```
# kvno nfs/atsqa4-164.nfs4ad.com@NFS4AD.COM
nfs/atsqa4-164.nfs4ad.com@NFS4AD.COM: kvno = 46
kvno nfs/qa108-43.nfs4ad.com@NFS4AD.COM
```

7. Add entry for NFS service principal key in the Kerberos keytab file, `/etc/krb5.keytab`:

```
# ktutil
ktutil:  addent -password -p nfs/atsqa4-164.nfs4ad.com@NFS4AD.COM -k
46 -e RC4-HMAC
Ex: addent -password -p nfs/qa108-43.nfs4ad.com@NFS4AD.COM -k 46 -e
RC4-HMAC
Password for nfs/atsqa4-164.nfs4ad.com@NFS4AD.COM:
( Give mapr user password i.e nfs4AD123 )
ktutil:  l
slot KVNO Principal
---- ---- ---------------------------------------------------------------------

   1   46      nfs/atsqa4-164.nfs4ad.com@NFS4AD.COM
ktutil:  wkt /etc/krb5.keytab
ktutil:  q
```

8. Verify that NFS service principal and host principal are in the `/etc/krb5.keytab` file by running the `klist` command.

   For example, for domain `nfs4ad.com`, run the following command and verify the entries in the file:

   ```
   # klist -kt /etc/krb5.keytab
   Keytab name: FILE:/etc/krb5.keytab
   KVNO Timestamp           Principal
   ---- ------------------- ------------------------------------------------------

      4 08/01/2018 00:29:21 host/atsqa4-161.nfs4ad.com@NFS4AD.COM
      4 08/01/2018 00:29:21 host/ATSQA4-161@NFS4AD.COM
      4 08/01/2018 00:29:21 host/atsqa4-161.nfs4ad.com@NFS4AD.COM
      4 08/01/2018 00:29:21 host/ATSQA4-161@NFS4AD.COM
      4 08/01/2018 00:29:21 host/atsqa4-161.nfs4ad.com@NFS4AD.COM
      4 08/01/2018 00:29:21 host/ATSQA4-161@NFS4AD.COM
      4 08/01/2018 00:29:21 host/atsqa4-161.nfs4ad.com@NFS4AD.COM
      4 08/01/2018 00:29:21 host/ATSQA4-161@NFS4AD.COM
      4 08/01/2018 00:29:21 host/atsqa4-161.nfs4ad.com@NFS4AD.COM
      4 08/01/2018 00:29:21 host/ATSQA4-161@NFS4AD.COM
      4 08/01/2018 00:29:21 ATSQA4-161$@NFS4AD.COM
      4 08/01/2018 00:29:21 ATSQA4-161$@NFS4AD.COM
      4 08/01/2018 00:29:22 ATSQA4-161$@NFS4AD.COM
      4 08/01/2018 00:29:22 ATSQA4-161$@NFS4AD.COM
      4 08/01/2018 00:29:22 ATSQA4-161$@NFS4AD.COM
     46 08/01/2018 02:58:01 nfs/atsqa4-161.nfs4ad.com@NFS4AD.COM
   ```

9. Ensure that `/etc/krb5.keytab` file is owned by user `mapr` and if necessary, change ownership to user `mapr`.

   For example:

   ```
   [root@qa108-41 ~]# chown mapr:root /etc/krb5.keytab
   [root@qa108-41 ~]# ls -l /etc/krb5.keytab
   -rw------- 1 mapr root 4175 Jul 22 23:53 /etc/krb5.keytab
   ```

10. Restart the `rpcgssd` service on the host to establish GSS security contexts.

CentOS	`service rpcgssd start`
Ubuntu	`service gssd restart`

11. Enable security variable, `SecType`, in the NFSv4 server configuration file at `/opt/mapr/conf/nfs4server.conf`.

    For example:

    ```
      # Security type (krb5,krb5i,krb5p)
      SecType = krb5;
    ```

12. Start the NFSv4 server.

    For more information, see Starting, Stopping, and Restarting MapR NFSv4 on page 1216.

13. List the shares exported on the server by running `showmount -e` command.

    If the protocol is v4 only, the `showmount` command will not return the list of exported NFS shares. Instead, to view the export list, run the following command:

    ```
    /opt/mapr/server/nfs4mgr list-exports
    ```

14. Ensure that the `list-exports` command runs successfully.

    For example:

    ```
    #  maprcli nfs4mgmt list-exports
    Export Id     Path
    30            /mapr
    0             /
    ```

15. (Troubleshooting) Run the following command to restart the services if you see security-related issues.

    CentOS

    ```
    maprcli node services -nfs4
    stop -nodes `hostname` ; service
    rpcgssd restart; sleep 1; service
    rpcbind  restart ; sleep 1;
    service nfs restart ; service
    nfs stop ; sleep 2; maprcli
    node services -nfs4 start -nodes
    `hostname`
    ```

    Ubuntu

    ```
    maprcli node services -nfs4
    stop -nodes `hostname` ; service
    gssd restart; sleep 1; service
    rpcbind  restart ; sleep 1; service
    nfs-kernel-server restart ; service
    nfs-kernel-server stop ; sleep
    2; maprcli node services -nfs4
    start -nodes `hostname`
    ```

16. Set up VIPs for the NFSv4 servers:

    a) Add entries for IPs and names of VIPs in the `/etc/hosts` file on the NFSv4 server host first and then on the AD server host.

       For example:

       ```
       10.10.88.14 nfsvirtualip1
       10.10.88.15 nfsvirtualip2
       ```

    b) Add NFS service principal for the virtual IP by running the `setspn` command.

       For example:

       ```
       C:\Users\Administrator>setspn -A host/nfsvirtualip1 nfsserver
       C:\Users\Administrator>setspn -A nfs/nfsvirtualip1 nfsserver

       C:\Users\Administrator>setspn -A host/nfsvirtualip2 nfsserver
       C:\Users\Administrator>setspn -A nfs/nfsvirtualip2 nfsserver
       ```


last-updated: Apr 11, 2022

c) Restart the `rpcgssd` service on the host to re-establish GSS security contexts.
For example:

```
service rpcgssd restart
```

Configuring NFSv4 Server for Other Kerberos Installations

1. Configure NFS server for Kerberos.
Consult with your system administrator for assistance with the commands for configuring the NFS server for Kerberos-based authentication. For example, you must do the following:
   - Create a service principal with `nfs` as the service name.
   For example: `nfs/host.domain.com@REALM`
   - Generate a keytab for the NFS service principal, store it in the `/etc/krb5.keytab` file, and set correct permissions on the file.
2. Enable the security variable, `SecType`, in the NFSv4 server configuration file at `/opt/mapr/conf/nfs4server.conf`.
For example:

```
# Security type (krb5,krb5i,krb5p)
SecType = krb5;
```

3. Start the NFSv4 server.
For more information, see Starting, Stopping, and Restarting MapR NFSv4 on page 1216.
4. List the shares exported on the server by running `showmount -e` command.
If the protocol is v4 only, the `showmount` command will not return the list of exported NFS shares. Instead, to view the export list, run the following command:

```
/opt/mapr/server/nfs4mgr list-exports
```

5. Ensure that the `list-exports` command runs successfully.
For example:

```
#  maprcli nfs4mgmt list-exports
Export Id     Path
30            /mapr
0             /
```

Configuring NFSv4 Client

1. Ensure that NFS client has a `/etc/krb5.keytab` file with a valid principal similar to one of the following: `nfs/<client_fqdn>@<domain>@<REALM>`, `host/<client_fqdn>@<domain>@<REALM>`, or `<HOSTNAME>$@<REALM>`.
If the principal is not present, create the `keytab` file with the principal, which will be used to mount the share, for the OS (as mentioned in the OS vendor documentation).

2. Mount the cluster by running the `mount` command.
   For example:

   ```
   mount -t nfs4 -o sec=<security-type> <nfs4-server-hostname>:/
   <pseudo-path> <mount-point>
   ```

   For example:

   ```
   # mount -t nfs4 -o sec=krb5 <FQDN>:/mapr /mnt/nfs4mnt
   ```

3. Generate user ticket for the user to access the mount path.
   For example, for user `mapr` on domain `nfs4ad.com`, run one of the following commands to generate the ticket:

   - ```
     kinit mapr@NFS4AD.COM
     <Enter password>
     ```

   - ```
     echo usr2AD123 | kinit user2@NFS4AD.COM
     ```

   Note: You must renew the user ticket before it expires; otherwise, the mount path returns permissions denied error after the ticket expires.

4. (Troubleshooting) Restart the services and mount again to avoid security-related issues.

CentOS	`service rpcgssd restart; sleep 1;`
	`service rpcbind  restart ; sleep 1;`
	`service nfs stop`

Ubuntu	`service rpcgssd restart; sleep 1;`
	`service rpcbind  restart ; sleep 1;`
	`service nfs stop`

Troubleshooting: Any running IO on NFSv4 mount (with Kerberos) is stuck if the krb5 ticket expires for the current user. The mount point also hangs and becomes inaccessible.

Workaround: Restart the `rpcgssd` service with the new ticket to make the mount point accessible and re-trigger the IO to proceed.

Configuring NFSv4 Server Without Kerberos

To start using NFSv4 server without Kerberos, do the following:

1. Start the NFSv4 server.
   For more information, see Starting, Stopping, and Restarting MapR NFSv4 on page 1216.

last-updated: Apr 11, 2022

2. Verify that the list-exports command runs successfuly.
   For example:

```
#  maprcli nfs4mgmt list-exports
Export Id     Path
30            /mapr
0             /
```

3. Mount the cluster by running the `mount` command.
   For example:

```
mount -t nfs4 <nfs4-server-hostname>:/<pseudo-path> <mount-point>
```

Starting, Stopping, and Restarting MapR NFSv4
Describes how to start, stop and restart the NFS version 4 service using either the Control System, the CLI, or the REST API.
Starting, Stopping, and Restarting MapR NFSv4 Using the Control System

See:

- Starting the Services on the Cluster Using the Control System on page 830
- Stopping a Service on the Cluster Using the Control System on page 831
- Restarting the Services on the Cluster Using the Control System on page 832

Starting, Stopping, and Restarting MapR NFSv4 Using the CLI and REST API

Note: On Ubuntu 20, run the following commands before starting NFS4, Else, NFS fails to start as it cannot find the jemalloc library.

```
apt-get install libjemalloc2

ln /usr/lib/x86_64-linux-gnu/libjemalloc.so.2 /opt/mapr/lib/
libjemalloc.so.1
```

CLI	To stop, start, or restart the MapR NFSv4 server, run: `maprcli node services -nodes <node names> -nfs4 stop|start|restart`
REST	Send a request of type POST. For example: `curl -k -X POST 'https://<host>:8443/rest/node/services?nodes=<nodeNames>&nfs4=stop|start|restart' --user mapr:mapr`

Note: When the NFS server is stopped, the VIPs associated with the server are released, and CLDB attempts to reassign the VIPs to other available NFS servers.

For the complete list of parameters, see `node services` on page 1730.
Mounting NFS on a Linux Client
Describes how to mount a NFS server on a Linux client.

You can manually mount NFS on a Linux client when your system starts up.

Note: On nodes running CentOS, use the VIP for mounting because, by default, the mount command will use the physical IP of the node.

1. List the NFS shares exported on the server.

   For example, run the following command for NFS version 4 servers:

   ```
   /opt/mapr/server/nfs4mgr list-exports
   ```

   If the NFS protocol is not version 4 only, use the `showmount` command to retrieve the list of exported NFS shares. For example:

   ```
   showmount -e usa-node01
   ```

2. Mount the cluster using NFS.

   For example:

   ```
   mount -t nfs4 -o sec=krb5 usa-node01:/<psuedo_mapr> /mapr
   ```

   Tip: For the best performance, use NFS v4.0.

   Use the `vers=4.0` parameter in the mount command. For example:

   ```
   mount -t nfs4 -o sec=krb5,vers=4.0 usa-node01:/<psuedo_mapr> /mapr
   ```

Note: When you mount manually from the command line, the mount point does not persist after a reboot.

Configuring the Linux NFS Client
Describes how to set the optimal number of RPC requests to the NFS server.

The default RPC requests configuration can negatively impact performance and memory. To avoid performance and memory issues, configure the number of outstanding RPC requests to the NFS server to be 128.

Perform the following steps as the `root` user on each NFS client machine:

1. To enable the configuration to persist after a reboot of the NFS client machine, issue the following commands to create the `sunrpc.conf` file under `/etc/modprobe.d` with the recommended configuration:

   ```
   echo "options sunrpc tcp_slot_table_entries=128" >> /etc/modprobe.d/
   sunrpc.conf
   echo "options sunrpc tcp_max_slot_table_entries=128" >> /etc/modprobe.d/
   sunrpc.conf
   ```

2. To enable the configuration to take effect after you remount the NFS client to the NFS gateway, issue the following echo commands:

   ```
   echo 128 > /proc/sys/sunrpc/tcp_slot_table_entries
   echo 128 > /proc/sys/sunrpc/tcp_max_slot_table_entries
   ```


last-updated: Apr 11, 2022

3. Remount the NFS client to the NFS gateway. For example, the following commands unmount and mount NFS assuming that the cluster is mounted at `/mapr`:

```
umount /mapr
mount -o hard,nolock <hostname>:/mapr /mapr
```

Note: Failure to configure this property may result in the following error in `/opt/mapr/logs/nfsserver.log`:

```
ERROR nfsserver[38960] fs/nfsd/requesthandle.cc:791 0.0.0.0[0]
cannot allocate more OncRpcContexts: [numDropped=2556001]
dropping connection from nfsc=10.13.64.225:0
```

Tip: For CentOS, after the reboot of the node, if the `/proc/sys/sunrpc` directory is not available or if `rpcidmapd` is not running, start the `rpcidmapd` service using the following command:

```
service rpcidmapd start
```

Advisory Locking in NFS v4

Explains NFS v4 support for Advisory Locking.

MapR NFS v4 service includes support for (advisory) file locking. MapR keeps track of the locks on a file in the NFS gateway, but does not prevent a client process from writing to a file that is locked by another process. The locks are not shared with other NFS gateways. Since the locks are enforced locally, it is the responsibility of the client process to check for write locks on a file before attempting to perform write operations on the file.

To ensure that a file is locked and not available for changes by other processes and to ensure that the lock on a file by a process is honored by other processes, add a program similar to the following for the process.

Sample Program Description	The following program demonstrates how to open a file, check if the file has a write lock, and wait if another process currently has locked the file. Before running this application, ensure that you have access to a cluster running MapR File System.
Opens a file	`if (argc > 1) { int fd = open(argv[1], O_WRONLY); if(fd == -1) { printf("Unable to open the file\n"); exit(1); }`
Checks if the file is locked for a write operation	`lock.l_type = F_WRLCK; lock.l_start = 0;`

```
lock.l_whence =
SEEK_SET;
    lock.l_len =
0;
    lock.l_pid =
getpid();
```

Gets lock, else waits

```
    int ret
= fcntl(fd,
F_SETLKW,
&amp;lock);

printf("Return
value of
fcntl:%d\n",ret);
    if(ret==0) {
      while (1) {

scanf("%c",
NULL);
      }
    }
```

Sample Program Code

```
#include <stdio.h>
#include <fcntl.h>

int main(int argc, char **argv) {
  if (argc > 1) {
    int fd = open(argv[1], O_WRONLY);
    if(fd == -1) {
      printf("Unable to open the
file\n");
      exit(1);
    }
    static struct flock lock;

    lock.l_type = F_WRLCK;
    lock.l_start = 0;
    lock.l_whence = SEEK_SET;
    lock.l_len = 0;
    lock.l_pid = getpid();

    int ret = fcntl(fd, F_SETLKW,
&lock);
    printf("Return value of
fcntl:%d\n",ret);
    if(ret==0) {
      while (1) {
        scanf("%c", NULL);
      }
    }
  }
}
```

NFSv4 Troubleshooting

Issues with listing the exports:

If `nfs4mgr list-exports` command returns the error, `Error org.freedesktop.DBus.Error.ServiceUnknow`

`n: The name org.ganesha.nfsd was not provided by any .service files`, do the following to resolve the error:

1. Open the `/etc/dbus-1/system.conf` file.
2. Find and replace the following in the file:
   a. Find:

```
<deny
send_destination="org.freedeskt
op.DBus"

send_interface="org.freedesktop
.DBus"

send_member="UpdateActivationEn
vironment"/>
```

   b. Replace with:

```
<!-- Allow anyone to talk to
the message bus -->
<allow
send_destination="org.freedeskt
op.DBus"/>
<!-- But disallow some
specific bus services -->
<allow send_interface="*"/>
<allow receive_interface="*"/>
<allow own="*"/>
```

3. Save and close the file.
4. Restart the messagebus and NFSv4 service.

   To restart the messagebus service, run the following command:

```
service messagebus restart
```

   To restart NFSv4 service, see Starting, Stopping, and Restarting MapR NFSv4 on page 1216.

Lock expires as a result of problems in node connection or node failover:

If the lock expires as a result of problems in node connection or node failover, all IOs from the application will fail with EIO message to prevent the file from getting corrupted. You can tune the kernel to reclaim lost locks and not fail writes on lock lease expiration. To tune the kernel, run the following command:

```
echo Y > /sys/module/nfs/parameters/
recover_lost_locks
```

Warning: This kernel tuning this might result in data corruption.

User ID mapping is not working when NFSv4 server and client are in two different sub-domains:

If user ID (with FQDN user names) mapping is not working with the installed `libnfsidmap` library (for

example, `libnfsidmap-0.25-19` on Centos), do the following:

1. Download the latest `libnfsidmap` library for the OS.

   For CentOS:

   a. Download the library source with the patches.

   b. Apply the patches till `libnfsidmap-0.25-dns-resolved.patch`.

   c. Build the new library.

2. Stop NFSv4 service and replace the old library with the new library.

3. Replace the old `nsswitch.so` file with the new `nsswitch.so` file that is compiled.

4. Restart rpcidmapd service and NFSv4 service.

NFS Ganesha crashes when unmounting the cluster:

NFS Ganesha crashes when unmounting the cluster as a non-root user. Download and install the following Kerberos 16 package:

- `sssd-krb5-common-1.16.0-19.el7.x86_64`
- `sssd-krb5-1.16.0-19.el7.x86_64`

Discover realm command fails:

If the `realm discover` command fails with following error: `realm: Couldn't connect to realm service: Error calling StartServiceByName for org.freedesktop.realmd: GDBus.Error:org.freedesktop.DBus.Error.TimedOut: Activation of org.freedesktop.realmd timed out`, reboot the node and add Active Directory information in the `/etc/resolv.conf` again. For example, your entry in the `/etc/resolv.conf` file should look similar to the following:

```
# cat /etc/resolv.conf
nameserver 10.10.111.41
domain nfs4ad.com
search nfs4ad.com lab qa.lab
scale.lab perf.lab ipmi.lab
```

Unable to start the service:

If you or the warden is unable to restart the NFS service, do the following:

1. Review the warden logs (in `$MAPR_HOME/logs/warden.log` file) to determine when the NFSv4 Service Alarm on page 2232 was raised.

last-updated: Apr 11, 2022

2. Review the following NFSv4 server logs in `$MAPR_HOME/logs/nfs4/nfs4server.log-0` and `$MAPR_HOME/logs/nfs4/fsal.log-0`, and other logs under `$MAPR_HOME/logs/nfs4` on the node where the service went down to determine the cause for the error.

   The following example logs show some common causes for NFSv4 service shutting down such as license not present or an issue in the configuration.

```
# tail -f /opt/mapr/logs/nfs4/
fsal.log-0
2018-08-10 04:04:20,3058
FATAL FuseOps fs/client/fuse/cc/
fuse_ops_ll.c:505 Thread: 1209 No
license found. Shutting down

2018-08-10 04:04:34,6487 ERROR
FuseAPI fc/fuse_api.cc:1384
Thread: 8877 Shmid to be used by
fcdebug 1003847690, guts 0
2018-08-10 04:04:34,7749 ERROR
Cidcache fc/cidcache.cc:5448
Thread: 8877 License not found.
Shutting down
2018-08-10 04:04:34,7749
FATAL FuseOps fs/client/fuse/cc/
fuse_ops_ll.c:505 Thread: 8877 No
license found. Shutting down

2018-08-10 04:04:48,6729 ERROR
FuseAPI fc/fuse_api.cc:1384
Thread: 15412 Shmid to be used by
fcdebug 1005748236, guts 0
2018-08-10 04:04:48,7412 ERROR
Cidcache fc/cidcache.cc:5448
Thread: 15412 License not found.
Shutting down
2018-08-10 04:04:48,7413
FATAL FuseOps fs/client/fuse/cc/
fuse_ops_ll.c:505 Thread: 15412 No
license found. Shutting down
```

```
# tail -f /opt/mapr/logs/nfs4/
nfs4server.log-0
10/08/2018 T05:58:06.410328-0700
8163[none] [main]
713 :export_commit_common :Exportin
g to NFSv4 but not Pseudo path
defined
10/08/2018 T05:58:06.410338-0700
8163[none] [main]
2267 :fsal_put :FSAL MAPR now
unused
10/08/2018 T05:58:06.410369-0700
8163[none] [main]
1443 :build_default_root :Export 0
(/) successfully created
```

```
10/08/2018 T05:58:06.410373-0700
8163[none] [main] 476 :main :No
export entries found in
configuration file !!!
10/08/2018 T05:58:06.410380-0700
8163[none] [main]
219 :config_errs_to_log :Config
File (/opt/mapr/conf/
nfs4server.conf:104): Syntax error
in statement
10/08/2018 T05:58:06.410384-0700
8163[none] [main]
219 :config_errs_to_log :Config
File (/opt/mapr/conf/
nfs4server.conf:65): Unknown
parameter (nfs_track_memory)
10/08/2018 T05:58:06.410387-0700
8163[none] [main]
219 :config_errs_to_log :Config
File (/opt/mapr/conf/
nfs4server.conf:68): Unknown
parameter (mapr_log_debug_level)
10/08/2018 T05:58:06.410389-0700
8163[none] [main]
219 :config_errs_to_log :Config
File (/opt/mapr/conf/
nfs4server.conf:95): 1 validation
errors in block EXPORT
10/08/2018 T05:58:06.410392-0700
8163[none] [main]
219 :config_errs_to_log :Config
File (/opt/mapr/conf/
nfs4server.conf:95): Errors
processing block (EXPORT)
10/08/2018 T05:58:06.411681-0700
8163[none] [main]
1040 :cache_inode_lru_pkginit :Sett
ing the system-imposed limit on
FDs to 65536.
```

3. Take corrective action to rectify the cause for the error.

Viewing the List of NFS Servers

Explains how to view the list of NFS servers using the Control System.

- Log in to the Control System and go to the service information page for CLDB.
  The **Active NFS Servers** section displays the following:

Column Name	Column Description
Server ID-HEX	The server's ID in hexadecimal notation.
Server ID	The server's ID in decimal notation.
Host Port	The IP address of the NFS server host.
Hostname	The hostname of the NFS server host.
Last heartbeat	The timestamp for the last received heartbeat.

Column Name	Column Description
State	The status of the NFS server. Value can be: • Active

Handling Heavy Write Loads on Red Hat Enterprise Linux
Describes a fix to mitigate resource contention between NFS Clients and the NFS Server on Red Hat Linux.

If you are operating on RHEL and have a heavy NFS write load, you might experience resource contention between the NFS client and the NFS server. This resource contention can cause the NFS server to be unresponsive. To avoid this potential problem, try one of following approaches. These approaches work on all versions of Red Hat (5.x, 6.x and 7.x).

- Edit `/etc/sysctl.conf` and apply these settings on each NFS server:

  ```
  vm.dirty_ratio=10
  vm.dirty_background_ratio=5
  ```

  Reboot the server so the changes will take effect. To make the settings take effect immediately, issue the `echo` command as shown:

  ```
  % echo 10 > /proc/sys/vm/dirty_ratio
  % echo 5 > /proc/sys/vm/dirty_background_ratio
  ```

- Separate the NFS client from the NFS server so they do not compete for memory on the same system.

Configure NFS Write Performance
Describes how to set the optimal value for outstanding Remote Procedure Call (RPC) requests to the NFS server.

The default Remote Procedure Call (RPC) requests configuration can negatively impact performance and memory. To avoid performance and memory issues, configure the number of outstanding RPC requests to the NFS server to be 128, for optimal performance. The NFS client uses this value to determine when to send requests to the NFS server, along with the number of parallel requests to send.

- If the value is too small, the NFS client does not send many parallel requests. This scenario results in decreased performance.
- If the value is too high, the NFS client sends a lot of parallel requests, but the NFS server discards some requests, as it has a limit on the number of requests it can handle. This scenario causes the NFS client to resend the requests, and negatively affects performance.

The kernel tunable value `sunrpc.tcp_slot_table_entries` represents the number of simultaneous RPC requests. The default value of this tunable is 16 (on Red Hat versions prior to version 6.3). On Red Hat versions 6.3 and above, the default value of this tunable is set at 65536. Increasing or decreasing this value to 128 (depending on the Red Hat version in use), may improve write speeds. Use the command `sysctl -w sunrpc.tcp_slot_table_entries=128` to set the value. Add an entry to your `sysctl.conf` file to make the setting persist across reboots.

Perform the following steps as the root user, on each NFS client machine:

1. Issue the following commands to create the `sunrpc.conf` file under `/etc/modprobe.d` with the recommended configuration. These commands enable the configuration to persist after a reboot of the NFS client machine.

```
echo "options sunrpc tcp_slot_table_entries=128" >> /etc/modprobe.d/
sunrpc.conf
echo "options sunrpc tcp_max_slot_table_entries=128" >>  /etc/modprobe.d/
sunrpc.conf
```

2. Issue the following `echo` commands. These commands enable the configuration to take effect after you remount the NFS client to the NFS gateway.

```
echo 128 > /proc/sys/sunrpc/tcp_slot_table_entries
echo 128 > /proc/sys/sunrpc/tcp_max_slot_table_entries
```

3. Remount the NFS client to the NFS gateway. Mount the MapR NFS server with a rsize and wsize of 128K, as this value significantly cuts down NFS server requests for a given transfer, and improves the overall performance. For example, the following commands unmount and mount the NFS server, assuming that the cluster is mounted at `/mapr`.

```
umount /mapr
mount -o nolock,rsize=131072,wsize=131072 <hostname>:/mapr /mapr
```

4. After rebooting the node, if the `/proc/sys/sunrpc` directory is not available, or if `rpcidmapd` is not running, start the `rpcidmapd` service, using the following command: `service rpcidmapd start`.

Failure to set this tunable to an optimum value, may result in the following error in the `/opt/mapr/logs/nfsserver.log` file:

```
ERROR nfsserver[38960] fs/nfsd/requesthandle.cc:791 0.0.0.0[0] cannot
allocate more OncRpcContexts: [numDropped=2556001] dropping connection from
nfsc=10.13.64.225:0
```

NFS write performance varies between different Linux distributions. The recommended value of this tunable may have no effect, or even a negative effect on your particular cluster.

Adjusting NFS Memory Settings

The memory allocated to each MapR service is specified in the `/opt/mapr/conf/warden.conf` file, which MapR automatically configures based on the physical memory available on the node. You can adjust the minimum and maximum memory used for NFS, as well as the percentage of the heap that it tries to use, by setting the `percent`, `max`, and `min` parameters in the `warden.conf` file on each NFS node. Example:

```
...
service.command.nfs.heapsize.percent=3
service.command.nfs.heapsize.max=1000
service.command.nfs.heapsize.min=64
...
```

The percentages need not add up to 100; in fact, you can use less than the full heap by setting the `heapsize.percent` parameters for all services to add up to less than 100% of the heap size. In general, you should not need to adjust the memory settings for individual services, unless you see specific memory-related problems occurring.

Running NFS on a Non-standard Port

1. To run NFS on an arbitrary port, modify the following line in `warden.conf`:

   ```
   service.command.nfs.start=/etc/init.d/mapr-nfsserver start
   ```

   Add `-p <portnumber>` to the end of the line, as in the following example:

   ```
   service.command.nfs.start=/etc/init.d/mapr-nfsserver start -p 12345
   ```

2. After modifying `warden.conf`, restart the MapR NFS server by issuing the following command:

   ```
   maprcli node services -nodes <nodename> -nfs restart
   ```

3. You can verify the port change with the `rpcinfo -p localhost` command.

   Warning: MapR uses version 3 of the NFS protocol. NFS version 4 bypasses the port mapper and attempts to connect to the default port only. If you are running NFS on a non-standard port, mounts from NFS version 4 clients time out. Use the `-o nfsvers=3` option to specify NFS version 3.

Enabling Debug Logging for NFS Using the CLI

Debug-level logging is available to help you isolate and identify NFS-related issues.

Enabling Debug Logging for NFSv3

1. To enable logging at the debug level, enter this command at the command line:

   ```
   maprcli trace setlevel -port 9998 -level debug
   ```

   where `-port 9998` indicates NFS.

   Warning: The `debug` log level provides much more information than the default log level of `info`.

2. In default mode, information is logged to a buffer and dumped periodically. To display information immediately instead, enable `continuous` mode by entering:

```
maprcli trace setmode -port 9998 -mode continuous
```

Sample log output from an `ls` command is shown here:

From `/opt/mapr/logs/nfsserver.log`:

```
2013-06-10 16:13:27,2278 DEBUG nfsserver[30283] fs/nfsd/nfsserver.cc:555
127.0.0.1[0x5d349889] NFS Proc=NFSPROC3_GETATTR
2013-06-10 16:13:27,2278 DEBUG nfsserver[30283] fs/nfsd/
nfsserver.cc:1022 127.0.0.1[0x5d349889] NFS FileHandle:
2.1012313856.2.2.2
2013-06-10 16:13:28,3774 DEBUG nfsserver[30283] fs/nfsd/nfsserver.cc:555
127.0.0.1[0x5e349889] NFS Proc=NFSPROC3_ACCESS
2013-06-10 16:13:28,3774 DEBUG nfsserver[30283] fs/nfsd/
nfsserver.cc:1022 127.0.0.1[0x5e349889] NFS FileHandle:
2.1012313856.2.2.2
2013-06-10 16:13:28,3775 DEBUG nfsserver[30283] fs/nfsd/nfsserver.cc:555
127.0.0.1[0x5f349889] NFS Proc=NFSPROC3_GETATTR
2013-06-10 16:13:28,3775 DEBUG nfsserver[30283] fs/nfsd/
nfsserver.cc:1022 127.0.0.1[0x5f349889] NFS FileHandle:
2.1012313856.2.2.2
2013-06-10 16:13:28,3776 DEBUG nfsserver[30283] fs/nfsd/nfsserver.cc:555
127.0.0.1[0x60349889] NFS Proc=NFSPROC3_READDIRPLUS
2013-06-10 16:13:28,3783 INFO nfsserver[30283] fs/nfsd/mount.cc:822
Cluster my.cluster.com, Setting myTopology to /default-rack/
ubuntu-n3.jon.prv
2013-06-10 16:13:28,3784 DEBUG nfsserver[30283] fs/nfsd/cache.cc:659
127.0.0.1[0x60349889] Sending CLDB Lookup for cid=3410106368.2049
(sleep=0) ip= cldb=10.10.80.41:7222
2013-06-10 16:13:28,3906 DEBUG nfsserver[30283] fs/nfsd/nfsserver.cc:555
127.0.0.1[0x61349889] NFS Proc=NFSPROC3_LOOKUP
2013-06-10 16:13:28,3906 DEBUG nfsserver[30283] fs/nfsd/attrs.cc:1032
127.0.0.1[0x61349889] Lookup: my.cluster.com
2013-06-10 16:13:28,3906 DEBUG nfsserver[30283] fs/nfsd/cache.cc:449
127.0.0.1[0x61349889] using existing RpcBinding
2013-06-10 16:13:28,3927 DEBUG nfsserver[30283] fs/nfsd/nfsserver.cc:555
127.0.0.1[0x62349889] NFS Proc=NFSPROC3_GETATTR
2013-06-10 16:13:28,3927 DEBUG nfsserver[30283] fs/nfsd/
nfsserver.cc:1022 127.0.0.1[0x62349889] NFS FileHandle:
2.1012313856.2.2.2
2013-06-10 16:13:28,8755 DEBUG nfsserver[30283] fs/nfsd/nfsserver.cc:555
127.0.0.1[0x63349889] NFS Proc=NFSPROC3_GETATTR
2013-06-10 16:13:28,8755 DEBUG nfsserver[30283] fs/nfsd/
nfsserver.cc:1022 127.0.0.1[0x63349889] NFS FileHandle:
0.3410106368.2049.16.2
2013-06-10 16:13:28,8755 DEBUG nfsserver[30283] fs/nfsd/cache.cc:449
127.0.0.1[0x63349889] using existing RpcBinding
2013-06-10 16:13:28,8759 DEBUG nfsserver[30283] fs/nfsd/nfsserver.cc:555
127.0.0.1[0x64349889] NFS Proc=NFSPROC3_ACCESS
2013-06-10 16:13:28,8759 DEBUG nfsserver[30283] fs/nfsd/
nfsserver.cc:1022 127.0.0.1[0x64349889] NFS FileHandle:
0.3410106368.2049.16.2
2013-06-10 16:13:28,8759 DEBUG nfsserver[30283] fs/nfsd/cache.cc:449
127.0.0.1[0x64349889] using existing RpcBinding
2013-06-10 16:13:28,8763 DEBUG nfsserver[30283] fs/nfsd/nfsserver.cc:555
127.0.0.1[0x65349889] NFS Proc=NFSPROC3_GETATTR
2013-06-10 16:13:28,8763 DEBUG nfsserver[30283] fs/nfsd/
nfsserver.cc:1022 127.0.0.1[0x65349889] NFS FileHandle:
0.3410106368.2064.16.2
```


last-updated: Apr 11, 2022

```
2013-06-10 16:13:28,8763 DEBUG nfsserver[30283] fs/nfsd/cache.cc:659
127.0.0.1[0x65349889] Sending CLDB Lookup for cid=3410106368.2064
(sleep=0) ip= cldb=10.10.80.41:7222
2013-06-10 16:13:28,8886 DEBUG nfsserver[30283] fs/nfsd/nfsserver.cc:555
127.0.0.1[0x66349889] NFS Proc=NFSPROC3_GETATTR
2013-06-10 16:13:28,8886 DEBUG nfsserver[30283] fs/nfsd/
nfsserver.cc:1022 127.0.0.1[0x66349889] NFS FileHandle:
0.3410106368.2049.44.66108
2013-06-10 16:13:28,8886 DEBUG nfsserver[30283] fs/nfsd/cache.cc:449
127.0.0.1[0x66349889] using existing RpcBinding
2013-06-10 16:13:28,8889 DEBUG nfsserver[30283] fs/nfsd/nfsserver.cc:555
127.0.0.1[0x67349889] NFS Proc=NFSPROC3_GETATTR
2013-06-10 16:13:28,8890 DEBUG nfsserver[30283] fs/nfsd/
nfsserver.cc:1022 127.0.0.1[0x67349889] NFS FileHandle:
0.3410106368.2537.16.2
2013-06-10 16:13:28,8890 DEBUG nfsserver[30283] fs/nfsd/cache.cc:659
127.0.0.1[0x67349889] Sending CLDB Lookup for cid=3410106368.2537
(sleep=0) ip= cldb=10.10.80.41:7222
2013-06-10 16:13:28,9185 DEBUG nfsserver[30283] fs/nfsd/nfsserver.cc:555
127.0.0.1[0x68349889] NFS Proc=NFSPROC3_GETATTR
2013-06-10 16:13:28,9186 DEBUG nfsserver[30283] fs/nfsd/
nfsserver.cc:1022 127.0.0.1[0x68349889] NFS FileHandle:
0.3410106368.2050.16.2
2013-06-10 16:13:28,9186 DEBUG nfsserver[30283] fs/nfsd/cache.cc:659
127.0.0.1[0x68349889] Sending CLDB Lookup for cid=3410106368.2050
(sleep=0) ip= cldb=10.10.80.41:7222
2013-06-10 16:13:28,9312 DEBUG nfsserver[30283] fs/nfsd/nfsserver.cc:555
127.0.0.1[0x69349889] NFS Proc=NFSPROC3_GETATTR
2013-06-10 16:13:28,9312 DEBUG nfsserver[30283] fs/nfsd/
nfsserver.cc:1022 127.0.0.1[0x69349889] NFS FileHandle:
0.3410106368.2536.16.2
2013-06-10 16:13:28,9312 DEBUG nfsserver[30283] fs/nfsd/cache.cc:659
127.0.0.1[0x69349889] Sending CLDB Lookup for cid=3410106368.2536
(sleep=0) ip= cldb=10.10.80.41:7222
2013-06-10 16:13:28,9432 DEBUG nfsserver[30283] fs/nfsd/nfsserver.cc:555
127.0.0.1[0x6a349889] NFS Proc=NFSPROC3_GETATTR
2013-06-10 16:13:28,9432 DEBUG nfsserver[30283] fs/nfsd/
nfsserver.cc:1022 127.0.0.1[0x6a349889] NFS FileHandle:
0.3410106368.2535.16.2
2013-06-10 16:13:28,9432 DEBUG nfsserver[30283] fs/nfsd/cache.cc:659
127.0.0.1[0x6a349889] Sending CLDB Lookup for cid=3410106368.2535
(sleep=0) ip= cldb=10.10.80.41:7222
```

The log shows every operation sent to and received from an NFS client.

3. To return to the default log level and log mode, enter:

```
maprcli trace setlevel -port 9998 -level default
maprcli trace setmode -mode default
```

Enable Debug Logging for NFSv4

1. Modify `core-site.xml` file to add the following:

```
<property>
   <name>fs.mapr.trace</name>
   <value>DEBUG</value>
   <description> </description>
</property>
```

2. Save and close the file.

Run the `/opt/mapr/server/nfs4mgr` command for debugging NFS Ganesha.

Unmounting the MapR Cluster from the Command-Line

- To unmount the MapR cluster, run the `umount` command.

  For example, to unmount the cluster in `/mapr`, run the following command:

  ```
  umount /mapr
  ```

  If a process is busy on the mount point, the umount command will fail. To unmount the cluster after the process completes, run the following command:

  ```
  umount -l /mapr
  ```

Managing MapR POSIX Clients

Provides a brief synopsis of MapR POSIX clients.

The MapR POSIX clients allow app servers, web servers, and other client nodes and apps to read and write data directly and securely to a MapR cluster. The following topics describe the steps for configuring and managing loopback NFS POSIX and FUSE-based POSIX clients.

Apart from the clients that are EOL, all others are supported. However, newer clients might have features that may not be supported in older clients.

Difference between the POSIX loopback NFS client and the FUSE-based POSIX Basic and Platinum clients

The following table summarizes the differences between the POSIX loopback NFS client and the FUSE-based POSIX Basic and Platinum clients:

	MapR POSIX Loopback NFS Client	MapR FUSE-based POSIX Basic/ Platinum Client
Throughput	• 500MB/s for remote read/write • 1G/s for local read/write	Greater than 2G/s for remote and local read/write
Client OS	Supported Linux and Ubuntu distributions only.	
Installs On Node Type	• Client node • Cluster node	
Access to Cluster	Must have direct network access to all MapR cluster nodes.	Must have direct network access to all MapR cluster nodes. However, each client only supports up to 16 clusters.
Connection to File System	• Proxied on host to regular MapR client traffic • Direct, no NFS gateway • No single point of failure	
Security	Fully secured.	
Caching	Buffered writes are cached in the kernel.	Buffered writes are cached (only in kernel >= 3.15) if writeback option is enabled.

last-updated: Apr 11, 2022

MapR loopbacknfs POSIX Client
Explains the differences between the MapR POSIX client and the Linux native NFS client.

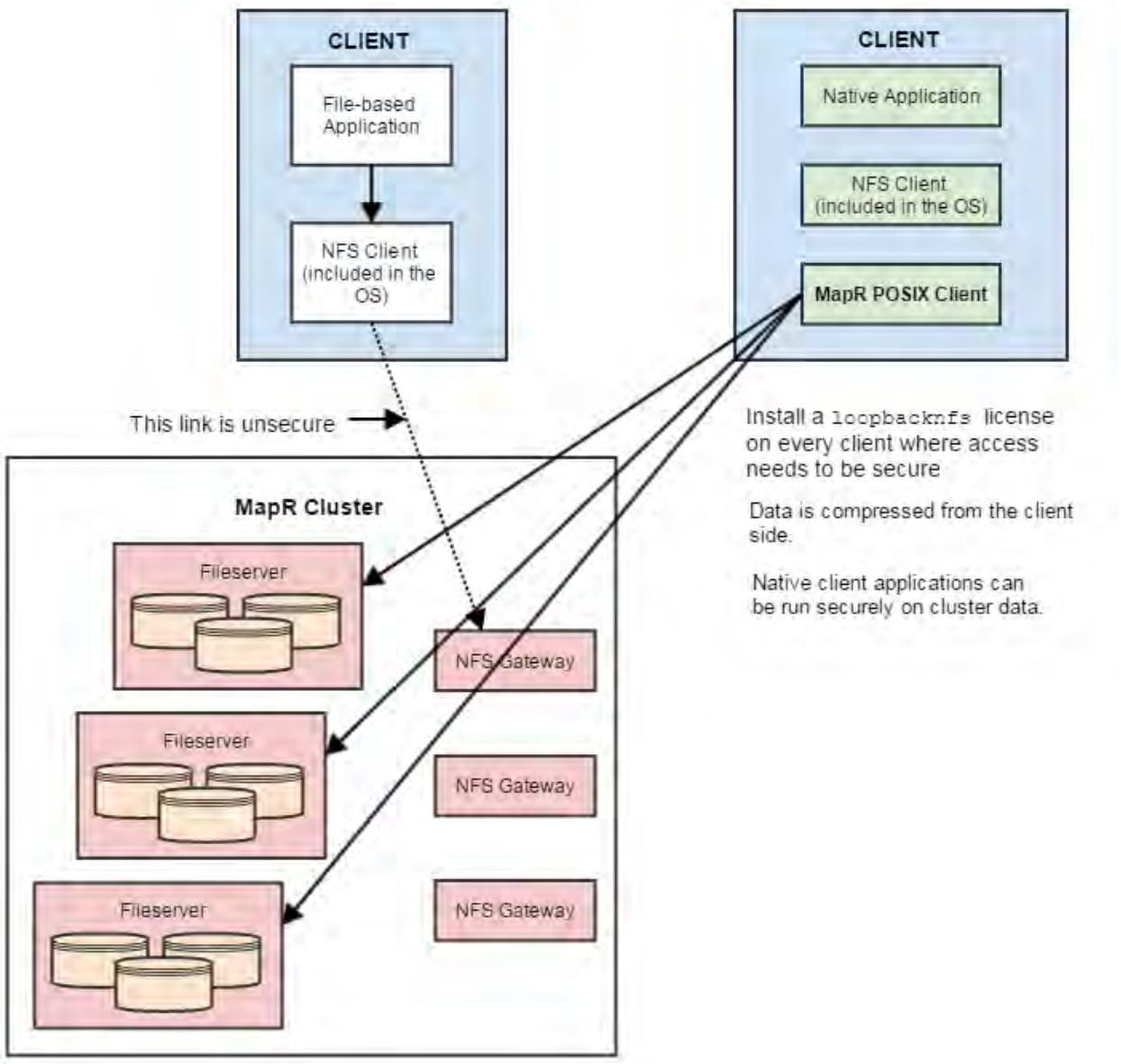
The MapR POSIX Client feature allows app servers, web servers, and other client nodes and apps to read and write directly to a MapR cluster. Starting with the 4.0.2 release, MapR provides single-user `loopbacknfs` licenses that give access to one or more clusters.

The table below summarizes the differences between the basic Linux OS NFS client and the MapR POSIX client:

	Linux OS Client	MapR POSIX Client
Client OS	• Supported Linux distributions and desktop systems (Mac OS X and Windows)	• Supported Linux distributions only • No version for Mac OS X
Installs On Node Type	• Client node - not part of the MapR cluster • No mapr-fileserver or other Hadoop services	• Client node • Cluster node
Access to Cluster	• Must have direct network access to NFS Gateways.	• Must have direct network access to all MapR cluster nodes
Supported Interfaces	• POSIX-NFS	• POSIX-NFS
Connection to File System	• Point to point • Via an NFS gateway • Single point of failure	• Proxied on host to regular MapR client traffic • Direct, no NFS gateway • No single point of failure
Security	• Link to NFS gateway is insecure	• Fully secured

The Linux OS NFS client must go through an NFS gateway, the link to the gateway is not secured, and transmitted data is not compressed.

The following diagram illustrates how the MapR POSIX client (`mapr-loopbacknfs`) works, in comparison with the Linux OS NFS client (left).



The instructions on this page are for the MapR POSIX client. For instructions on setting up NFS on a MapR cluster, see Managing the MapR NFS Service on page 1176.

The table below summarizes the differences in the MapR POSIX client deployment behavior when installed with a MapR cluster where security is disabled or enabled:

	Cluster Security Disabled	Cluster Security Enabled
Client Node	• MapR cluster looks exactly like network attached storage (NAS) • POSIX permissions are enforced	• Single-user authentication • Write access is supported only for applications with UID matching authenticated user
Cluster Node	• MapR cluster looks exactly like NAS • POSIX permissions are enforced	• Secure cluster access is key • Best Practice: Use ticket from *mapr* user

last-updated: Apr 11, 2022

Specifying Environment Variables

Explains how to set environment variables on a client node.

A subset of the environment variables defined on the servers for the MapR cluster must be defined, with the same values, on the client. You can add environment variables directly to the startup script, or create a local `env.sh` file in `/usr/local/mapr-loopbacknfs/conf`. You cannot simply copy the `env.sh` file from a server node in the cluster because the MAPR_HOME setting would be different.

Complete the following steps to specify environment variables:

1. On a server node in the MapR cluster, locate the `env.sh` and `env_override.sh` files in the `/opt/mapr/conf` directory. If the `env_override.sh` file is not present, use the `env.sh` file. For more information about these files, see About env_override.sh on page 2290.
2. Retrieve the MAPR_SUBNETS and JAVA_HOME settings from the server files and clone them to `/usr/local/mapr-loopbacknfs/conf/env.sh` on the client node.
3. (Optional) Set the `NFS_LOOPBACK_HONOUR_SUBNETS` environment variable to avoid re-registration whenever there is a change in any network interface. Value can be:
   - `true` to consider the `MAPR_SUBNETS` while registering with CLDB. If set to `true`, re-registration does not happen whenever there is a change in any network interface.
   - `false` to ignore the `MAPR_SUBNETS`. If set to `false`, re-registration happens whenever there is a change in any network interface.

   For example:

   ```
   # export NFS_LOOPBACK_HONOUR_SUBNETS=true
   # export MAPR_SUBNETS=10.10.104.0/24

   # env | grep SUBNET
   NFS_LOOPBACK_HONOUR_SUBNETS=true
   MAPR_SUBNETS=10.10.105.0/24,10.10.104.0/24
   ```
4. Change the JAVA_HOME setting to point to the location where Java is installed on the client.
5. Add the following lines to the client node `env.sh` file:

   ```
   export MAPR_HOME=/usr/local/mapr-loopbacknfs
   export MAPR_TICKETFILE_LOCATION=/usr/local/mapr-loopbacknfs/initscripts/
   mapr-loopbacknfs/longlived_ticket
   ```

   Note: To allow impersonation, set the value for `MAPR_TICKETFILE_LOCATION` to the path to the mapr user ticket.
6. Save and close the `env.sh` file.
7. Restart the loopbacknfs service for the changes to take effect.

Copying Configuration Files from a Server Node

Settings in the `nfsserver.conf` and `mapr-clusters.conf` files on server nodes in the MapR cluster are also needed by the POSIX client. Complete the following steps to copy configuration files from a server node:

1. On a server node in the MapR cluster, locate the `nfsserver.conf` and `mapr-clusters.conf` files in the `/opt/mapr/conf/` directory.

last-updated: Apr 11, 2022

2. Copy both of those files to the `/usr/local/mapr-loopbacknfs/conf/` directory on the client machine.

Starting the mapr-loopbacknfs Service to Access a Cluster
Describes the prerequisites and the process of starting the `mapr-loopbacknfs` service to access a secure cluster.

The following instructions explain how to start the loopbacknfs service so you can access either a non-secure or secure cluster.

To access multiple clusters, ensure that the first cluster that you configure is a MapR 4.0.2 or later cluster, with available POSIX client licenses.

Prerequisites for accessing a secure cluster:

- Ensure that the stock Linux NFS service is not running. Linux NFS and MapR NFS cannot run concurrently.
- Disable the lock manager (`nlockmgr`).
- Check that the `rpcbind` service is running on RHEL and CentOS v6.0 and higher. You can use the command `ps ax | grep rpcbind` to check.
- Check that the `portmapper` service is running on RHEL and CentOS v5.x and lower, and on Ubuntu and SLES. You can use the command `ps ax | grep portmap` to check.
- Make sure you have applied a Community Edition (M3) license or an Enterprise Edition (M5) license (paid or trial) to the cluster. See Adding a License on page 777.
- Enable security for the cluster. See Enabling Wire-level Security on page 1411 and Disabling Wire-level Security on page 1412 wire-level security.
- Generate a user ticket. See Generating a MapR User Ticket on page 1426 for instructions. If you do not already have a MapR user ticket, with full control *ACL* authorization on the cluster, you must have a cluster administrator do this for you.
  - In the MapR cluster, navigate to the server node to which you want to connect.
  - First, run `maprlogin password` to login. The user who logs in must be a privileged user, such as the `mapr` superuser.
  - Next, run `maprlogin generateticket -type service -user <user> -duration 365:0:0 -out <file>` to generate the user ticket. The <user> for whom the ticket is generated can be any user. If the service ticket expires, the POSIX client:
    - Automatically uses the renewed service ticket without requiring a restart, if the ticket is replaced before expiration (ticket expiry time + grace period of 55 minutes). If the ticket is replaced after expiration (which is ticket expiry time + grace period of 55 minutes), the POSIX loopbacknfs client does not refresh the ticket as the mount becomes stale.
    - Allows impersonation if a service ticket is replaced before ticket expiration (which is ticket expiry time + grace period of 55 minutes) with a `servicewithimpersonation` ticket.
    - Honor all changes in user/group IDs of the renewed ticket.
- Copy the user ticket file from the cluster server node where you generated it to the `/usr/local/mapr-loopbacknfs/conf` directory on the client machine where the MapR POSIX client runs.

last-updated: Apr 11, 2022

Note: Since the NFS server runs based on a single user's ticket, it can act on behalf of only one user. Therefore, the UID or GID associated with the ticket must match the UID or GID of any user who accesses the NFS server through the MapR POSIX Client.

Note: Securing the cluster so that only one user can have secure access provides tight control over cluster access, but it also means that any user on the client who is able to read the generated ticket has read access to all data in the cluster.

Start the mapr-loopbacknfs service and mount the volume

Complete the following steps from your client node, except where noted, to start the `mapr-loopbacknfs` service and mount the volume:

Note: If cluster security is enabled, the ticket that you generated using the preceding procedure, must be available or the NFS server does not start.

1. Start the `mapr-loopbacknfs` service from the command line.

```
service mapr-loopbacknfs start
```

2. Create a mount point at `/mapr` and mount the client node to it.

```
mkdir /mapr
mount localhost:/mapr /mapr
```

3. You can also automate the mounting of the volume with every launch of the `mapr-loopbacknfs` service. On the POSIX client node, create `/usr/local/mapr-loopbacknfs/conf/mapr_fstab` and add the following line:

```
localhost:/mapr /mapr hard,nolock
```

Securing the Mountpoint

POSIX permissions are the only limitation on read access by the MapR POSIX client, whether the cluster connected to has security enabled or disabled. By securing the mountpoint, you can limit access to a single user.

Complete the following steps to secure the mountpoint:

1. On the client system, create `/mapr/<clustername>`:

```
mkdir -p /mapr/<clustername>
```

2. Set ownership and permissions:

```
chown user1:<posix_user> /mapr
chmod 700 /mapr
```

3. Mount the cluster:

```
mount localhost:/mapr/<clustername>  /mapr/<clustername>
```

Now only the `<posix_user>` can access the cluster with the POSIX client.

Registering a POSIX Client with Additional Clusters

The first time you start the `loopbacknfs` service, you edit the `mapr-loopbacknfs` init script by defining the `CLUSTER_NAME` and `CLDB_IPS` variables, then run the script. These actions update the `/usr/local/mapr-loopbacknfs/conf/mapr-clusters.conf` file.

However, when you want to register a client with a new cluster or an additional cluster, you must add entries directly to the `/usr/local/mapr-loopbacknfs/conf/mapr-clusters.conf` file. Editing the `mapr-loopbacknfs` script and restarting the `loopbacknfs` service does not update the `mapr-clusters.conf` file.

Configuring the MapR POSIX Client

Explains how to set the number of RPC requests that POSIX clients send to a cluster.

The default RPC requests configuration can negatively impact performance and memory. To avoid performance and memory issues, configure the number of outstanding RPC requests to the cluster to be 128.

Perform the following steps as the root user on each POSIX client machine:

1. Issue the following commands to create the *sunrpc.conf* file under */etc/modprobe.d* with the recommended configuration:

   ```
   echo "options sunrpc tcp_slot_table_entries=128" >> /etc/modprobe.d/
   sunrpc.conf
   echo "options sunrpc tcp_max_slot_table_entries=128" >> /etc/modprobe.d/
   sunrpc.conf
   ```

   These commands enable the configuration to persist after a reboot of the NFS client machine.

2. Issue the following echo commands:

   ```
   echo 128 > /proc/sys/sunrpc/tcp_slot_table_entries
   echo 128 > /proc/sys/sunrpc/tcp_max_slot_table_entries
   ```

   The commands enable the configuration to take effect after you remount the POSIX client to the MapR cluster.

3. Remount the POSIX client to the MapR cluster. For example, the following commands unmount and mount the NFS assuming that the cluster is mounted at */mapr*:

   ```
   umount /mapr
   mount -o hard,nolock 127.0.0.1:/mapr /mapr
   ```

   Note: Failure to configure this property may result in the following error in */usr/local/mapr-loopbacknfs/log*: `ERROR nfsserver[38960] fs/nfsd/requesthandle.cc:791 0.0.0.0[0] cannot allocate more OncRpcContexts: [numDropped=2556001] dropping connection from nfsc=10.13.64.225:0`

   #### CentOS Troubleshooting Tip

   After the reboot of the node, if the */proc/sys/sunrpc* directory is not available, or if `rpcidmapd` is not running, start the `rpcidmapd` service using the following command: `service rpcidmapd start`.

Verifying MapR POSIX Client Licenses

Use the Control System, to check how many MapR POSIX client licenses are available and being used.

1. Log in to the Control System and click **Admin** > **Cluster Settings**.

last-updated: Apr 11, 2022

2. Look at the **LICENSES** pane for the number of POSIX Client nodes that are available and currently being used.

Managing the mapr-loopbacknfs Service
Explains how to start/stop and manage the `loopbacknfs` service from the command line.

To manually start or stop the service:

```
service mapr-loopbacknfs [start|stop]
```

To have the service start automatically when the OS starts up:

```
systemctl enable mapr-loopbacknfs
```

To monitor the service:

```
service mapr-loopbacknfs status
showmount -e localhost
```

The showmount command displays:

```
Export list for <host>
/mapr               127.0.0.1
/mapr/<clustername> 127.0.0.1
```

Setting Up Aliases for NFS Exports

When provisioning MapR File System for various tenants, you can set up an alias for the path in MapR File System, rather than exporting the whole path, to mask the path from the users. Once the alias is set up, users will not be able to access or mount the path in MapR File System.

Aliases can be set up for the cluster, volume, and directory, but not for the root of the path in MapR File System (`/mapr`). To set up an alias for a path in MapR File System:

1. Open the NFS exports file in `/opt/mapr/conf/` directory.
2. Specify the alias name for the mount path using the following syntax:

   ```
   <path in MFS> /<alias name> <options>
   ```

   Here:

`<path in MFS>`	Refers to the MapR File System mount path. If this points to a: • Volume, the user can access the snapshots associated with the volume. • Directory, the user cannot access the snapshots.
`/<alias name>`	Refers to the alias name to use. If there are duplicate aliases in the file, the last entry will take effect and all other duplicate entries will be ignored. If the alias name is not specified, the path in MapR File System will be exported.
`<options>`	The list of available/supported options.

   For example, suppose a MapR File System mount path of `/mapr/samplecluster/samplevolume` for tenant samplecustomer. To set up an alias, add the following to the exports file:

   ```
   /mapr/samplecluster/samplevolume /samplecustomer (rw)
   ```

For example, to export a certain cluster, volume, or a subdirectory as an alias, comment out `/mapr` and add the following:

```
/mapr/clustername /alias1 (rw)
/mapr/clustername/vol /alias2 (rw)
/mapr/clustername/vol/dir /alias3 (rw)
```

Note: Only the alias will be visible/exposed to the NFS client.

3. Run the following command for the file changes to take effect:

```
/opt/mapr/bin/maprcli nfsmgmt refreshexports
```

4. Run the following command to export the path:

```
mount -t nfs nfsServer:/<alias_name> /localpath
```

Run this command once for each entry in the file.

The same export rules must be set up on all the NFS servers in the cluster to ensure that in the event of a node failure, the same aliases work with VIP failover.

Troubleshooting mapr-loopbacknfs Service Issues

Describes solutions for mapr-loopbacknfs issues.

To debug authentication issues, follow these steps:

1. If you receive a standard error (stderr):
   - Make sure `rpcinfo/portmap` is installed and/or run service portmap start.
   - Run service `rpcbind restart`.
2. Examine the log files for error messages:

```
/usr/local/mapr-loopbacknfs/logs/loopbacknfs.log
/usr/local/mapr-loopbacknfs/logs/mount_local_fs.log
```

Error messages in `loopbacknfs.log` file:

Error Message	Solution
`Refresh User tickets failed as security layer could not be initialized with user ticket /tmp/maprticket_0`	`Unset MAPR_TICKETFILE_LOCATION in initscripts/mapr-loopbacknfs.`
`exiting: license only allows 10 NFS/mfs server(s), currently alive=10`	`If you have multiple clusters listed in the mapr-clusters.conf file on the client, make sure the first one listed is a MapR 4.0.2 or later cluster. If that is not the problem, you will probably need to purchase additional licenses, or reduce number of installations of the mapr-loopbacknfs service.`
`mount.nfs: Protocol not supported`	`The NFS directory with mapr-loopbacknfs is already mounted.`

3. Verify that settings in configuration files are correct.
   - For all clusters: `/usr/local/mapr-loopbacknfs/conf/mapr-clusters.conf`

last-updated: Apr 11, 2022

- For secure clusters: `MAPR_TICKETFILE_LOCATION`

4. Check for “stale” mounts.
   - The `mount_local_fs.pl` script can cause the initscript wrapper to force unmounts of the mounted file systems.
   - Always check for stale mounts after stopping the service:
     - `df -k` should return instantly.
     - Use `umount -f <mount_point>` to force the unmount.
     - Use `ps -ef | grep mount_local` to confirm that the script is not stuck.

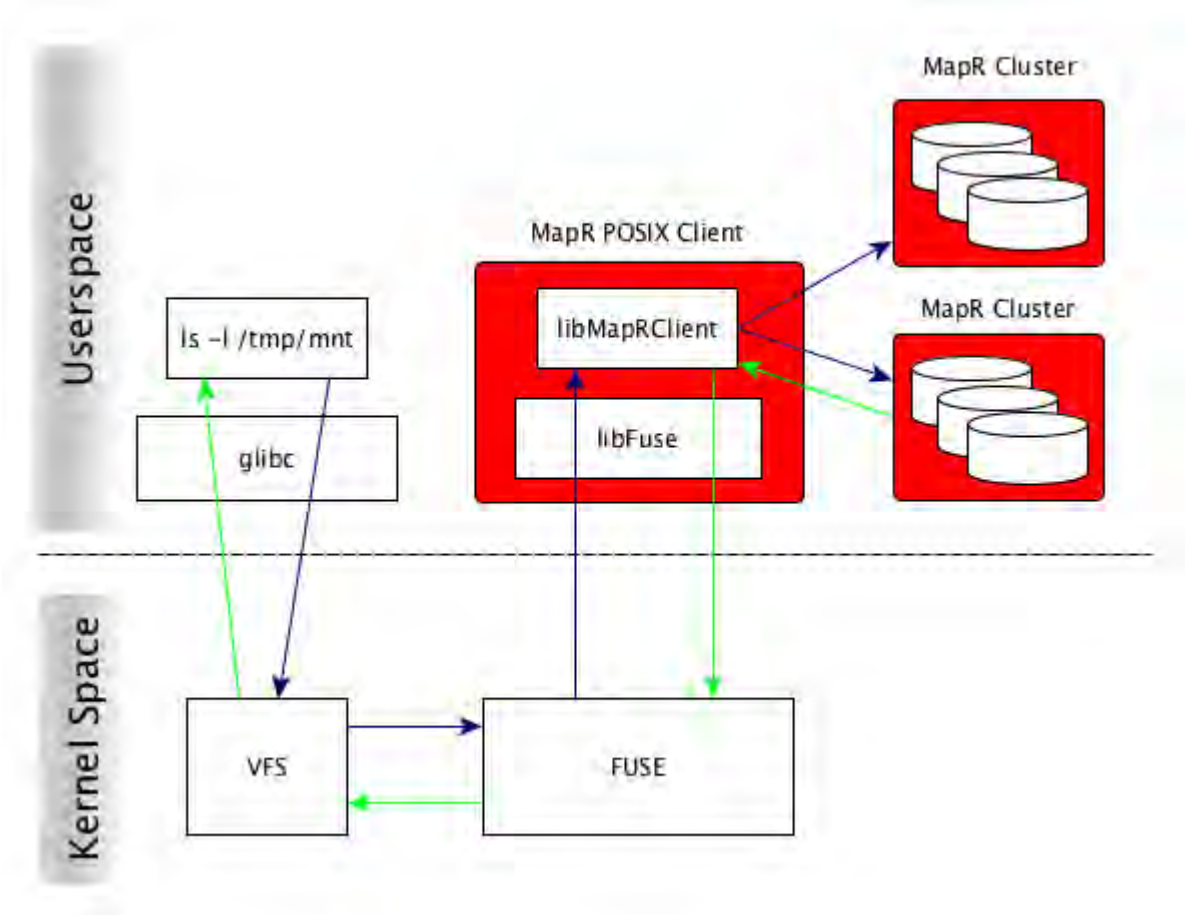
MapR FUSE-Based POSIX Client
Provides a brief description of the FUSE-based POSIX client.

The MapR FUSE-based POSIX client (either *mapr-posix-client-basic* or *mapr-posix-client-platinum*) allows app servers, web servers, and other client nodes and apps to read and write data directly and securely to a MapR cluster like a Linux filesystem.

The same MapR client can access both secure and nonsecure clusters; however, a MapR client that is configured to access a secure cluster can access a nonsecure cluster only if these conditions are met:

1. The secure cluster must be listed in the `mapr-clusters.conf` file.
2. A user must obtain a ticket for the secure cluster even if the user wants to access only the nonsecure cluster.

The FUSE-based MapR POSIX client runs as a userspace process to connect to one or more MapR clusters. The necessary FUSE (Filesystem in Userspace) library (`libfuse`) is bundled with the POSIX client package. With the installation of the POSIX client package, the MapR POSIX client performs operations such as read and write on the filesystem exposed by FUSE. The following diagram illustrates how the MapR FUSE-based POSIX client works.



Example of Mounting FUSE

This example shows you how to mount FUSE and perform operations as a regular user.

The following example is a quick introduction to mounting FUSE and accessing the mount point as a regular user.

Assume that you have a mount point */mapr* that you want to mount on FUSE and access as user *kate*.

Perform the following steps:

1. Create the user *kate*: Run

   ```
   adduser kate
   ```

2. Generate a ticket with impersonation as user *kate*. You will use this ticket to mount and access FUSE. Run:

   ```
   maprlogin generateticket -type servicewithimpersonation -user
   kate -out /var/tmp/sample_ticket
   ```

   Note: For more information on generating a ticket with impersonation, see How Impersonation Works on page 1478 and Generating a Service with Impersonation Ticket on page 1428

3. Edit the `/opt/mapr/conf/fuse.conf` file and set `fuse.ticketfile.location=/var/tmp/sample_ticket`

4. The mount point */mapr* is already set in the `fuse.conf` file.

   Note: Change `fuse.mount.point=/mapr` if your mount point is different from */mapr*.

5. Create the */mapr* directory. Run:

   ```
   mkdir /mapr
   ```

6. Start the MapR FUSE POSIX client, either *basic* or *platinum* as per your licence. For example:

   ```
   service mapr-posix-client-basic start
   ```

The */mapr* directory is now mounted on FUSE. You can perform all operations on */mapr/* as the user *kate*.

Related concepts

Configuring the MapR FUSE-Based POSIX Client on page 1240
Lists FUSE configuration parameters.

Managing the FUSE-Based POSIX Client on page 1255
Describes how to use the FUSE-based POSIX client.

How Impersonation Works on page 1478
Introduces impersonation functionality, limitations, and core requirements.

Generating a Service with Impersonation Ticket on page 1428

Configuring the MapR FUSE-Based POSIX Client
Lists FUSE configuration parameters.

FUSE Parameters

You can set the POSIX client configuration values in the `/opt/mapr/conf/fuse.conf` file. After installing the FUSE-based POSIX client, you can edit the configuration file to define the values for the following parameters and save the file.

To retrieve the list of configuration parameters, run the following command:

```
/opt/mapr/bin/posix-client-* --help
```

Here * refers to the basic or platinum client package installed on the system. If necessary, set the shared `LD_LIBRARY_PATH` environment variable to run the `help` option with the command. For example:

```
export LD_LIBRARY_PATH=/usr/lib/jvm/
java-1.7.0-openjdk-1.7.0.79.x86_64/jre/lib/amd64/server/:/opt/mapr/lib
```

Note: The MapR FUSE-based POSIX clients support only the configuration parameters in the `fuse.conf` file. All other FUSE configuration parameters are not supported. For more information on the non-mapr configuration parameters, refer to FUSE documentation.

`fuse.access.type`	*Default Value:* rw Sets the type of access on the mount point. Value can be: • `ro` — Read only • `rw` — Read and write
`fuse.affinity`	*Default Value:* 0 (Disabled)

Specifies whether to enable (`1`) or disable (`0`) NUMA affinity. If enabled, sets the NUMA affinity for the POSIX client.

`fuse.allow.other`

Default Value: 1

Allow other users to access the mount point. Value can be one of:

- `0` - do not allow other users
- `1` - allow other users

Set this to `1` if the `root` user starts the FUSE service. Set to `0` or comment out this parameter if a non-root user starts the FUSE service . If set to `1`, also add the `user_allow_other` parameter to the `/etc/fuse.conf` file.

`fuse.asyncdirect.io`

Default Value: 1

Specifies whether to enable asynchronous direct IO. Value can be one of:

- `0` - disable
- `1` - enable

`fuse.attr.timeout`

Default Value: 3.0

The timeout value in seconds for file/directory (regular) attributes (such as file size, UID, and GID, which are normally stored inside the inode) cache. This value is used to determine whether to use the cached attribute information (only if within the specified timeout window) or fetch attribute information again. The default is 3.0 seconds, which specifies that cached attribute information must be considered stale and refreshed after 3.0 seconds. You can assign fractions of a second as well (for example, `fuse.attr.timeout=2.8`).

Set the value for this parameter to `0` to compare POSIX (pjd) compliance with the ext3/4 file system. A value of 0 disables caching. For better performance, avoid disabling caching.

`fuse.auto.inval.data`

Default Value: 1

Specifies whether (1) or not (0) to automatically invalidate the kernel FUSE cache for any data change that causes mtime change, on the files. If set to 1, when the file is read, the correct file data is returned. If set to 0, the kernel cache of the data, which might not have the most current change, is returned.

`fuse.auto.unmount`

Default Value: 1

Specifies whether to automatically unmount the filesystem when the process is terminated. Value can be one of:

- `0` - disable
- `1` - enable

`fuse.big.writes`

Default Value: 1

Specifies whether to enable writes larger than 4KB. Value can be one of:

last-updated: Apr 11, 2022

- `0` - disable
- `1` - enable

Sets the size of the data/buffer that can be transferred from the kernel to the FUSE library, per request. If enabled, FUSE allows writes of 128KB from the kernel. If disabled, FUSE allows writes of 4KB from the kernel.

`fuse.client.lib.path`

Default Value: `/tmp`

Specifies the path to store the client libraries.

Note: To install and use FUSE-based POSIX client and NFS v4 on the same node, ensure that the path for both the client library for the FUSE-based POSIX client, and NFS v4 is not `/tmp`, which is the default. Specify a different location for the client libraries. For example, `/tmp/fuselib`.

`fuse.cluster.conf.location`

Default Value: `/opt/mapr/conf/mapr-clusters.conf`

The path to the configuration file to use.

`fuse.congestion.threshold`

Default Value: 10

Specifies the kernel's congestion threshold.

`fuse.disable.shardcache`

Default Value: 0 (false)

Specifies whether to disable shard cache, which is a cache of lookups. Value can be:

- `0` - false
- `1` - true

If true, more number of lookup calls are used. The FUSE client uses the shard cache to ensure that requests for data related to the same file are served by the same library. This is done using hash to improve performance. In very rare circumstances, it might make sense to disable this cache in conjunction with MapR support.

`fuse.disable.writeback`

Default Value: 0

Specifies whether (`1`) or not (`0`) to disable the writeback cache. This parameter is applicable only in kernel versions >= 3.15. By default, in kernel versions >= 3.15, writeback is enabled. To disable writeback cache, set the value for this parameter to `1`. If enabled, the writes are buffered in the kernel. However, when multiple FUSE clients work on the same file, writes to a file by one FUSE client might never be seen by other FUSE clients performing a read because the kernel does not update the attributes of the file unless the file is modified locally. You can disable the writeback cache to allow the kernel to perform a write through.

`fuse.enforce.core.pattern`

Default Value: false

Specifies whether (`true`) or not (`false`) to write to `/proc/sys/kernel/core_pattern` file when the FUSE-based POSIX starts. The default value is `false`. If `true`, the `core_pattern` file contains

an `/opt/cores/%e.core.%p.%h` entry and if `false`, the file is not touched.

`fuse.entry.timeout`

Default Value: 3

The timeout value in seconds for the name lookup cache. Use this parameter to determine whether to use the cached entry for the name lookup (if within the specified timeout window) or lookup the name again. The default is 3 seconds, which specifies that cached name lookup information must be considered stale and refreshed after 3 seconds. For this option, it is possible to give fractions of a second as well (for example, `fuse.entry.timeout=2.8`).

Set the value for this parameter to `0` to compare POSIX (pjd) compliance with the ext3/4 file system. Avoid retaining this value as 0 as it disables the cache, and impacts performance.

`fuse.evenly.spread.data`

Default Value: 0

Specifies whether (`1`) or not (`0`) to evenly spread writes across the nodes on the cluster. If set to `0`, writes are always sent to the local primary node, from where data is replicated on all the other nodes. If set to `1`, writes are distributed across different nodes. Set the value to `1` in case of reduced performance resulting from a large number of writes on the local primary node.

`fuse.export`

Default Value: `/mapr`

Denotes the fully-qualified cluster path to the volume or directory under the mount point.

When you do not specify a value, all clusters found in `mapr-clusters.conf` are mounted under the entity specified by the `fuse.mount.point` property (`/mapr` by default). If `mapr-clusters.conf` contains two clusters A and B, there are directories pointing to the root directories of those clusters, for example `/mapr/A` and `/mapr/B`.

When you specify a value, it overrides the default behavior, and causes exactly one path from one cluster to be exposed at the entity specified by the `fuse.mount.point` property. You can either fully expose a single cluster, or expose only a subset of a single cluster.

If you set `fuse.export` to the name of a cluster, enclosed within `/`, then that cluster is mounted at `/mapr`. For example if `fuse.export=/A/`, then the path `/mapr` shows the root directory of cluster A.

If you set `fuse.export` to a path within a cluster, then `/mapr` points to that path. For example, if `fuse.export=/A/var/`, then `/mapr` displays the directory contents of `/var` from the MapR cluster A.

Note: If the value is not a valid path to the name of a volume or directory, the FUSE service does not start. The value *cannot* be the path to a file.

`fuse.fast.local.directio`

Default Value: 0

Specifies whether to optimize (1) or disable (0) FUSE client for local direct IO. Value can be one of:

last-updated: Apr 11, 2022

- `0` - disable
- `1` - optimize

`fuse.flush.inline`

Default Value: 0

Specifies whether (1) or not (0) to flush all writes inline. Value can be one of:

- `0` - disable inline flushing
- `1` - flush all writes inline

If disabled, for all open files, by default, the buffer is flushed automatically every 3 seconds or when it reaches 64KB. If enabled, writes are sent to server directly.

`fuse.fsname`

Default Value: FUSE mount point

Specifies the filesystem source, which is the first field in the `/etc/mtab` file. The default value is the FUSE mount point that is denoted by the parameter `fuse.mount.point`.

`fuse.hb.interval`

Default Value: 5

Specifies the heartbeat interval (in seconds) for the FUSE-based POSIX client.

`fuse.log.debug_level`

Default Value: error

The FUSE-based POSIX client log level. The value can be one of:

- `fatal`
- `error`
- `warn`
- `info`
- `debug`

`fuse.log.path`

Default Value: `/opt/mapr/logs`

Specifies the path to store the log files.

`fuse.max.background`

Default Value: 64

Specifies the maximum number of asynchronous requests that can be submitted. IO requests beyond the maximum limit are blocked.

`fuse.max.cache.pages`

Default Value: 1048576 (1 Million pages)

Specifies the maximum number of pages (each page is 8KB) in the page cache that each MapR Client library in FUSE process can use when working with a large number of open files. This setting limits the amount of memory consumed by FUSE.

`fuse.max.read`

Default Value: 131072

Specifies the maximum size (in bytes) of read requests.

`fuse.max.readahead`

Default Value: 131072

Specifies the maximum number of bytes to read ahead.

`fuse.max.write`

Default Value: 131072

Specifies the maximum number of bytes that is allowed in a single write request.

`fuse.mount.point`

Default Value: `/mapr`

This parameter is mandatory. Specifies the mount point where the MapR filesystem must be mounted. Ensure that the specified mount point is empty before starting the service. Once mounted, the POSIX client has access to all the clusters specified in `/opt/mapr/conf/mapr-clusters.conf` file. The value should not be `/mapr` if you wish to mask MapR branding.

Note: If NFS server is also running on this node, ensure that the FUSE mount point is different from the NFS server mount point.

`fuse.mount.setuid`

Default Value: `0`

By default, FUSE mounts with the `nosuid` option. This prevents users other than root from running executable files with the SUID bit set, on FUSE. Enable this parameter (set to 1), to allow users other than root to run executable files with the SUID bit enabled, on the MapR Fuse FileSystem.

This parameter works in conjunction with the `allowreadforexecute` parameter in volume create on page 1931 and volume modify on page 2005 commands.

The following table describes how both parameters work together to permit running SUID binaries:

Table

fuse.mount.setuid	allowreadforexecute	Result
Disabled	Does not matter	SUID binaries cannot be executed by users other than *root*.
Enabled	Disabled	Users other than *root* can run the SUID binaries only when the binary has **both read and execute permissions**.
Enabled	Enabled	Users other than *root* can execute the SUID binaries either when the binary has **both read and execute permissions** OR **execute permission alone**.

last-updated: Apr 11, 2022

fuse.negative.timeout

Default Value: 3

Applicable for the Container, Basic, and Platinum POSIX clients.

Indicates the duration in seconds to cache negative lookup results.

Negative lookup results that are returned when a file does not exist (lookup retuned ENOENT), are cached for the specified number of seconds. The lookup is performed again, only after this period elapses. The file is deemed to be non-existent till this period elapses.

The default value of 3 indicates that negative lookup results are cached for 3 seconds.

Set this value to `0` to disable the negative lookup cache.

When patching or upgrading the client from an older release, this parameter is automatically applied. However, new parameters are not automatically written to `fuse.conf`. Make sure to copy this parameter from `fuse.conf.new` to `fuse.conf`, only if you want to change the default value, or disable this cache.

fuse.nonempty

Default Value: 0

Specifies whether FUSE can be mounted on a non-empty mount point (1) or on an empty mount point (0). Value can be:

- 0 - indicates that mount point should be empty
- 1 - indicates that mount point need not be empty

fuse.num.libs

Default Value:

- Container - 1
- Basic - 1
- Platinum - 5

Specifies the number of client libraries to run with. For:

- Container client, value must be 1.
- Basic client, value must be 1.
- Platinum client, default value is 5 and can be set to a value greater than 5.

More than one library allows for more than 1GB/sec throughput on remote operations as each additional library increases the throughput by sharding operations across libraries (for parallelism).

Note: Each additional library will consume additional memory and CPU.

fuse.num.threads

Default Value: 64

Specifies the number of FUSE threads in userspace per mount point. A higher number allows parallel processing of multiple operations. Recommended value is only up to 64.

fuse.ra.sessions

Default Value:

- Container - 1
- Basic - 1
- Platinum - 5

Specifies the number of parallel read ahead sessions per library. Each open file acts as one read ahead session. For example, for the default value of 5, up to 5 files can have read ahead sessions per library. If value is set to `0`, readahead is disabled.

Note: A greater value allows larger number of parallel read ahead sessions, which is useful if more number of files need to be opened simultaneously. However, each additional read ahead session consumes additional memory (512K per read ahead session) and threads.

`fuse.readdirplus`

Default Value: 1

Enables (1) or disables (0) `readdirplus` functionality for high latency networks. The `readdirplus` attribute returns the file handle and attribute information such as the name and the file ID, along with the directory entries, unlike the `readdir` attribute that requires the client to query the server separately for each directory entry. For the best performance, do not disable this parameter.

`fuse.sync.read`

Default Value: 0

Specifies whether to enable or disable synchronized reads. Value can be:

- `0` - disable
- `1` - enable

`fuse.ticketfile.location`

Default Value: `/opt/mapr/conf/maprfuseticket`

Specifies the ticket to use to start the service in secure mode. Generate the required ticket and place it in `/opt/mapr/conf/<maprfuseticket>`.

Note: To support impersonation, provide the mapr user ticket file location or the user's servicewithimpersonation ticket file location. You can use the mapr user ticket on the server node, and service with impersonation ticket on client node. The FUSE service must be started by the root user if servicewithimpersonation ticket is specified. In case of non-impersonated ticket, the ticket credentials becomes the identity for all the requests, no matter which user is accessing the fuse mount point.

See also: Setting up a Ticket for the POSIX Client.

`fuse.track.memory`

Default Value: false

Specifies whether to enable (`true`) or disable (`false`) memory tracking for FUSE.

`fuse.use.compressed.inode.format`

Default Value: 0

Specifies whether or not to use compressed inode format. When enabled, a 16-bit unique identifier is used to avoid inode cache collisions when multiple

last-updated: Apr 11, 2022

clients are modifying (creating, deleting, and similar operations) the same directories/files. The value can be one of:

- `0` — (default) do not use compressed inode format
- `1` — use compressed inode format including unique identifier

Note: Even when set to `1`, EBUSY errors are returned if client accesses more than 32k volumes at the same time.

Enabling this flag may not completely avoid inode cache collisions when too many modifications such as creation, and deletion are performed on the same directories or files. Give the kernel sufficient time to purge inode cache entries between modifications.

`fuse.xattr.enable`

Default Value: 0 (false)

Specifies whether (true) or not (false) to enable extended attributes through the FUSE client. Value can be one of:

- 0 - false
- 1 - true

The default value is 0 (false). This is disabled by default because if enabled, during operations, the kernel might make a lot of extended attribute calls for security checks resulting in performance degradation even when there are no extended attributes on the inode. When disabled, extended attributes can still be added using the `hadoop fs` command; however, this must be enabled to perform any operations on extended attributes using the FUSE-based POSIX client.

Note: Of the five types of extended attribute namespaces in Linux, system, trusted, user, raw, and security, only user namespace is supported. For all other namespaces, EINVAL is returned.

You must start/restart the FUSE-based POSIX client for the changes to take effect. See Starting and Stopping the POSIX Client for more information.

Configuration Backup When Installing/Upgrading POSIX Clients

When you install a patch, the `/opt/mapr/conf/fuse.conf.new` file contains the new settings. You can copy the new parameters (with default values) to your existing `fuse.conf` file and restart FUSE for the settings to take effect.

When you upgrade from a prior release, on all supported OS other than Ubuntu, the old `fuse.conf` file is backed up as `fuse.conf.backup`, before being overwritten with the new settings. This backup is available in the `/opt/mapr/conf` directory.

On Ubuntu, the upgrade process does not create a backup copy of the file. You need to manually backup the `fuse.conf` file before upgrading, as this file is overwritten with the new settings after upgrading.

last-updated: Apr 11, 2022

To continue using FUSE with your custom settings, and take advantage of the new settings, manually copy your custom settings in the `fuse.conf.backup` file to the `fuse.conf` file, set custom values for the new parameters in the `fuse.conf` file where necessary, and restart FUSE for the settings to take effect.

To restart FUSE, use one of the following commands depending on the POSIX client of your choice:

- For POSIX container: `service mapr-posix-client-container restart`
- For POSIX basic: `service mapr-posix-client-basic restart`
- For POSIX platinum: `service mapr-posix-client-platinum restart`

Optimizing FUSE performance when running the Flexible I/O tester (fio tool)

Performance Tips

- With Linux kernels prior to version 4.8, size extending writes are serialized by the kernel, and result in degraded write performance. For optimized write performance, ensure that the Linux kernel in use, is at least version 4.8.
- With kernel 4.8 and above, fio performance improves when using larger block sizes and larger number of jobs (numjobs). Keep numjobs constant and use larger blocksizes (>128k) for enhanced performance.

For example, for optimised performance, the `fio` command could be as follows:

```
fio --ioengine=libaio --direct=1 --gtod_reduce=1 --name=perftest --filename=
perfile
          --bs=16m --iodepth=64 --size=4G --rw=write --numjobs=4
```

Configuring Timeout for Inactive Connections
In cases where the file client connects infrequently to a remote CLDB node that is firewalled, TCP segments on the connection are silently dropped by the firewall due to the long idle time. However, the client keeps waiting for the response till RPC times out. To mitigate this scenario, you can now configure the timeout for inactive connections. Use the `fs.mapr.binding.inactive.threshold` parameter in the *core-site.xml* file to set this threshold in seconds. For example:

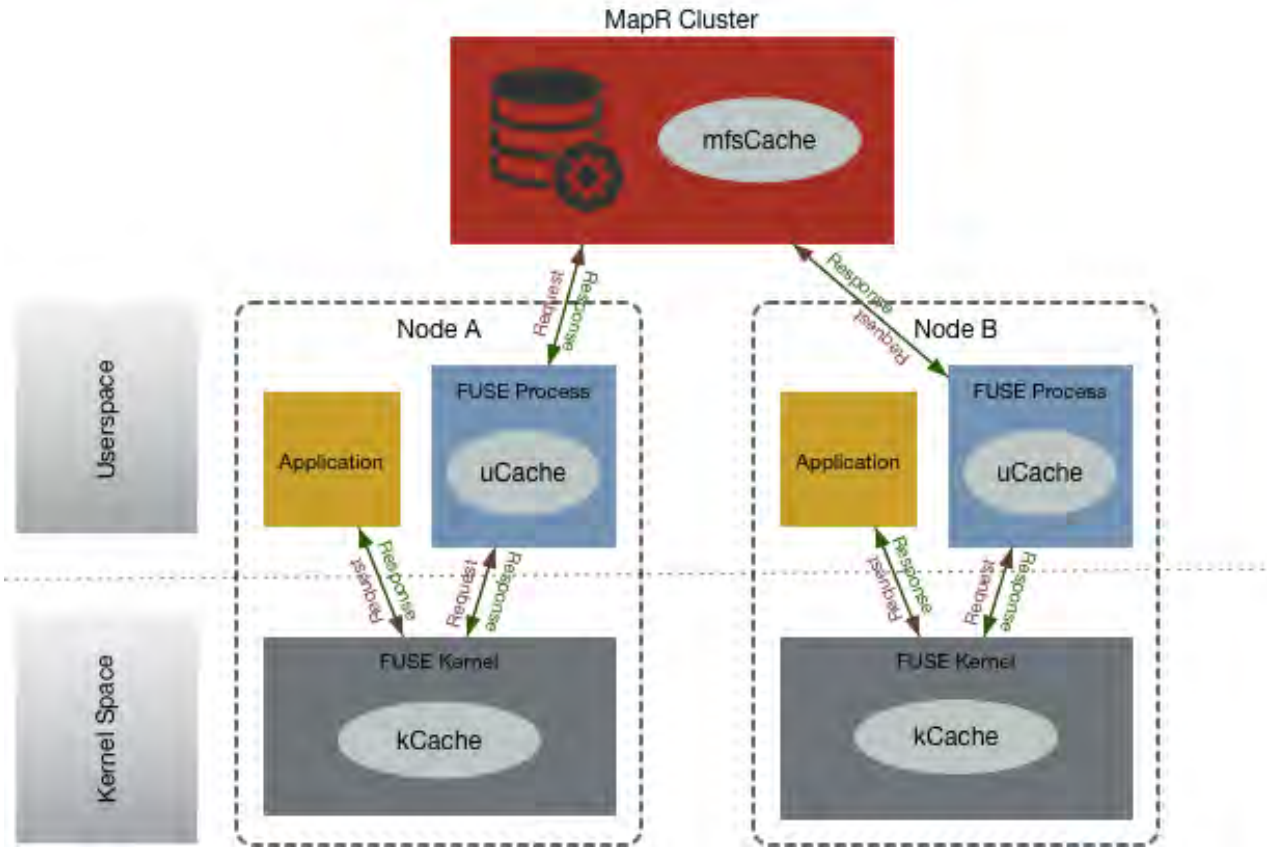
```
<property>
<name>fs.mapr.binding.inactive.threshold</name>
<value>600</value>
</property>
```

In this example, when the client tries to send data to the CLDB after a certain idle time, the system checks if the specified time (here 600 seconds, that is 10 minutes) is crossed after the previous request was sent. If so, the system tears down the existing TCP connection and creates a new TCP connection for the file client and CLBD to use for communication.

Tuning the Cache for FUSE-Based POSIX Clients
Describes performance tuning measures for FUSE clients.

The FUSE kernel and the FUSE userspace process caches both data and metadata. When an application performs a read of a file using the FUSE-based POSIX client, data is generally returned from the local FUSE kernel cache if that portion of the file resides in cache from a previous read or write operation. However, if the file has been modified on the MapR cluster by a different client, the data in the local kernel cache may be stale. The following illustration shows the layers of cache — FUSE kernel cache (referred to as kCache), the FUSE userspace cache (referred to as uCache), MapR filesystem cache (referred to as mfsCache) — and the following sections describe how these affect reads and writes and how the FUSE attributes can be used to tune the caching behavior.

last-updated: Apr 11, 2022



Inode Attribute Cache

Inode attributes are cached in the FUSE kernel cache (kCache) and in the userspace cache (uCache). When the same file is accessed from multiple FUSE clients, writes on the file through one mountpoint may not be seen by other applications performing a read on the file (through a different client). The inode attributes are cached in the kernel because of which another application modifying the file might not see the updates instantly. For example, the inode attributes, to name a few, such as size, or modification timestamp (mtime) of the file, in the kCache and uCache on Node A might be stale if the file is being modified concurrently by an application on Node B.

Tuning the attribute timeout in kCache	The FUSE kernel refreshes inode attributes from the userspace FUSE process once every 3 seconds by default. This can be tuned through the `fuse.attr.timeout` parameter in the `fuse.conf` file. The `fuse.attr.timeout` parameter specifies the interval of time at which to refresh the inode attributes in kernel and can be used to minimize the amount of time it takes to refresh the cache. Even if `fuse.attr.timeout` is set to `0`, Node A might still not see the latest writes from Node B because there is cached metadata in the userspace FUSE process on Node A; metadata can be served from the uCache and the application might not see current updates for attributes like size. To see the latest changes on a file, see Tuning the attribute timeout in uCache on page 1250.
Tuning the attribute timeout in uCache	The userspace FUSE process caches both data and metadata and refreshes the inode attributes from the MapR filesystem once every 3 seconds; *this is not tunable*. Even if the `fuse.attr.timeout` is set to 0, because there is cached data and metadata in

the userspace FUSE process (uCache), stale data or metadata can be served from the userspace FUSE process, which only refreshes inode attributes every 3 seconds. However, the userspace FUSE process updates the inode attributes every time a file is opened. To see the latest changes on a file, applications, especially readers on a file, that require to see updates on the file within 3 seconds can close and open the file to refresh the attributes and see instant updates.

Readdir Cache

Directory entries (files within a directory) are not cached in the uCache, but are cached in the kCache. The kCache can be stale on Node A if there are files being created in the directory by an application from the mountpoint on Node B. That is, a user listing directory entries on Node A for a directory (using a command like ls) might not see the files that were just created from Node B.

Tuning the entry timeout in kCache

The `fuse.entry.timeout` parameter specifies the interval of time at which to refresh the readdir (or lookup) cache in the kernel (kCache). The default value is 3 seconds and this can be configured in the `fuse.conf` file.

Data Cache

By default:

- Reads are buffered both in the kCache and the uCache.
- Writes are not buffered in the kCache, but are buffered in the uCache.

An application trying to read a file on Node A might not see the latest updates to the file (written from node B) for the following reasons:

- Reads on Node A might have been served either from the kCache or the uCache of Node A.

  See Tuning the cache for reads on page 1251 for information on invalidating the cache.

- The writes might have been buffered in the uCache of Node B.

  See Tuning the cache for writes on page 1252 for information on disabling buffering at a file level or altogether.

Tuning the cache for reads

Tuning the kCache

The `fuse.auto.inval.dat` a parameter specifies whether or not to automatically invalidate the kCache for any data change, which causes mtime change, on the files. If enabled, any mtime update on the file automatically invalidates the page cache of the file. The mtime is an inode attribute; for information on refreshing the cache for inode attributes, see Inode Attributes.

last-updated: Apr 11, 2022

	Tuning the uCache	Every read cache page is valid for 3 seconds. After 3 seconds, the read cache page is dropped from uCache. *This is not tunable*.
Tuning the cache for writes	**Tuning the kCache**	The `fuse.writeback.cache` parameter specifies whether to buffer writes in the kernel or to perform a write through. If enabled, the writes are buffered in the kernel. If disabled, writes are not buffered in the kernel and are directly sent to the FUSE process. By default, writeback caching is disabled; that is, the kernel sends all writes to the FUSE process directly. If an application does small writes, then the FUSE process might run out of CPU because of the overhead involved in small writes. To mitigate this, the FUSE kernel can be configured to enable caching in the kernel using the `fuse.writeback.cache` attribute. However, this can be enabled only on kernels running version >= 3.15.
	Tuning the uCache	The `fuse.flush.inline` attribute can be used to disable data buffering in the uCache. By default, the userspace FUSE process caches the writes locally. This parameter specifies whether to cache writes (for up to 3 seconds or 64KB in size) or write directly to server. This can be disabled at both the file and FUSE process levels. If the application does buffering before writing to filesystem, to avoid redundant buffering, you can disable buffering at the FUSE level by setting the

	`fuse.flush.inline` parameter value to `1` in the `fuse.conf` file. Caching can be disabled at a file-level by opening the file in `O_DIRECT` mode.

Caching Negative Lookup Results

FUSE issues thousands of lookup calls for the file, even when the initial lookup call has returned ENOENT, indicating that the file that does not exist, This behaviour is in contrast to NFS, which caches the lookup result for a specified time. For example, when running a `git clone` operation on the *mapr-core* repository, testing indicated that FUSE issued 870k calls, while NFS issued 82k calls, for files that are non-existent.

To reduce the number of negative lookups, and optimize performance, the FUSE configuration contains the `fuse.negative.timeout` parameter.

By default, this parameter is set to 3 seconds. Negative lookup results that are returned when a file does not exist (lookup retuned ENOENT), are cached for 3 seconds. The lookup is performed again, only after this period elapses. The file is deemed to be non-existent till this period elapses.

For more information on this parameter, check the FUSE configuration.

Configuring MapR FUSE-based POSIX Client for Tenant Environment
Explains the parameters to set to enable tenant user access to tenant shares from the FUSE-based POSIX client.

To enable tenant users to access the tenant share from the FUSE-based POSIX client, set the following configuration parameters in the `fuse.conf` file on the tenant host:

fuse.mount.point	Specifies the local mount path to where the cluster filesystem is going to mount. To mask the MapR branding from the tenant user, do not specify `/mapr` as the value for this parameter.
fuse.export	Specifies the path to the tenant volume mount path or directory under the tenant volume mount point. This is disabled by default, allowing users to access all the clusters specified in the `/opt/mapr/conf/mapr-clusters.conf` file. This must be set to enable the user on the tenant host to directly access only the specified volume or directory.
fuse.ticketfile.location	Specifies the path to the tenant ticket file to start the service in secure mode.

For more information on all other available and supported parameters, see Configuring the MapR FUSE-based POSIX Client.

Sample MapR FUSE-Based POSIX Client Configuration File

```
#Set path to the mount point
fuse.mount.point=/mapr

#Set path where logs shall be stored
fuse.log.path=/opt/mapr/logs

#Set path where client libraries shall be stored
fuse.client.lib.path=/tmp
```


last-updated: Apr 11, 2022

```
#Allow all users to access the filesystem
fuse.allow.other=1

#Enable larger than 4kB writes
fuse.big.writes=1

#Enable NUMA affinity
fuse.affinity=0

#Auto unmount on process termination
fuse.auto.unmount=1

#Set number of libMapRClient libraries to run with
#fuse.num.libs=DEFAULT_NUM_LIBS

#Set number of readahead sessions
#fuse.ra.sessions=1

#Enable/Disable memory tracking for fuse
fuse.track.memory=false

#Set number of FUSE threads
#fuse.num.threads=64

#Enable async direct io
fuse.asyncdirect.io=1

#Set the maximum size of read requests
#fuse.max.read=128

#Set the maximum bytes to readahead
#fuse.max.readahead=128k

#Set the maximum size in a single write request
#fuse.max.write=128

#Enable sync reads
#fuse.sync.read=0

#Set number of maximum background requests
fuse.max.background=64

#Set kernel's congestion threshold
#fuse.congestion.threshold=10

#Flush all writes inline
#fuse.flush.inline=0

#Optimize for local direct writes
#fuse.fast.local.directio=0

#Optimize by evenly distribute data across cluster
#fuse.evenly.spread.data=0

#Disable shard cache
#fuse.disable.shardcache=0

#Sets the filesystem  source  (first  field  in  /etc/mtab).
#The default is the mount program name.
#fuse.fsname=NAME

#Set fuse ticket file
fuse.ticketfile.location=/opt/mapr/conf/maprfuseticket
```

```
#fuse nonempty option to enable mounting at nonempty mount point
#fuse.nonempty=0

#by default, we support user namespace xattr.
#setting below option to 1 will enable the user xattr.
#fuse.xattr.enable=1

#Attribute timeout for inodes
#fuse.attr.timeout=3.0

#Entry timeout for inodes
#fuse.entry.timeout=3.0

#Hearbeat interval for FUSE in seconds
#fuse.hb.interval=5

#fuse sub exports
#fuse.export=/clus.default/vol1

#fuse core pattern
#fuse.enforce.core.pattern=true

#Readonly or readwrite, values are ro,rw
#fuse.access.type=rw

#Auto invalidation of data on mtime change
fuse.auto.inval.data=1

#Disable writeback cache
#If multiple fuse servers are operating on the same file then enabling
#this option will break consistency among different fuse servers i.e.
#writes to file1 on server1 will not be seen by an application reading
#file1 on server2 forever.
fuse.disable.writeback=1

#Set cluster configuration file
#fuse.cluster.conf.location=/opt/mapr/conf/mapr-clusters.conf

#Sets client debug level, values are fatal, error, warn, info, debug
fuse.log.debug_level = error;

#Inode compressed format
fuse.use.compressed.inode.format=0
```

Managing the FUSE-Based POSIX Client
Describes how to use the FUSE-based POSIX client.

Ports Needed for POSIX Clients and File System to Communicate With Each Other

POSIX clients communicate with the CLDB and server components of the MapR filesystem. You need to open the relevant ports for **TCP** connectivity from POSIX clients to the MapR file-system cluster nodes. Open the CLDB, file-system server, and file-system server instances ports.

- CLDB - Ports 7222 and 7223.
- Filesystem Server - 5660, 5692, 5724, 5756, and 6660.

You can open additional CLDB ports for better performance. See Ports Used by MapR Software on page 2290 for more information.

When using Multi-MFS instances, open the relevant ports for these instances to work. For example, instance 0 will use four ports from 5660, 5692 (5660+32), 5724 (5660+64), and 5756 (5660+96), instance

1 will use four ports from 5661, 5693, 5725, 5757, and so on for every additional instance. See Working with Multiple Instances of the File System on page 790 for more information.

Setting up a Ticket for the POSIX Client

The POSIX client can be accessed using user and service tickets. The service tickets have long lifetimes, by default, for ease of administration. This is useful for running application processes that should not be bounded by the CLDB duration (`cldb.security.user.ticket.duration.seconds`) and renewal (`cldb.security.user.ticket.max.duration.seconds`) properties, which limit the lifetime of user tickets. If you plan to use a user ticket, ensure that the user has read permissions on the ticket file.

Irrespective of the ticket type, after generating the ticket, set the `fuse.ticketfile.location` parameter value in the `fuse.conf` file to point to the ticket file to use.

For more information, see:

- Generating a MapR User Ticket on page 1426
- Generating a Service Ticket on page 1428

Note: If the UID/GID in the ticket (without impersonation capability) is different from the UID/GID of the logged-in user, all operations are performed using the UID/GID of the ticket and not that of the logged-in user.

Starting and Stopping the POSIX Client

To ensure that the service can be started and stopped and to run the help option, set the shared `LD_LIBRARY_PATH` environment variable. Update the shared library environment variable to include the paths to the following:

- Full path to the directory containing `libjvm.so` file
- `$MAPR_HOME/lib` (that is, `/opt/mapr/lib` directory)

For example:

```
export LD_LIBRARY_PATH=/usr/lib/jvm/
java-1.7.0-openjdk-1.7.0.79.x86_64/jre/lib/amd64/server/:/opt/mapr/lib
```

To allow a non-root user to start this service, as administrator or `root` user, run the following command:

```
chmod u+s /opt/mapr/bin/fusermount
```

Verify that permissions have been set for non-root user to start the service. For example:

```
ls -l /opt/mapr/bin/fusermount
```

Your output should look similar to the following:

```
-rwsr-xr-x 1 root root 39704 Feb 16 19:41 /opt/mapr/bin/fusermount
```

Ensure that the non-root user has full permissions on the mount point and log files.

To manually start or stop the service:

```
service mapr-posix-client-* [start|stop|status]
```

When you run the command, replace * with basic or platinum, which corresponds with the package that is installed on the system.

Note: The POSIX client service cannot be stopped or restarted if any other process is accessing the mount point. With `systemd` (on CentOS/RH 7.x and SLES 12), the service will enter failed state (if you tried to stop) or activating state (if you tried to restart) and you must manually kill the client processes.

Running the POSIX Client in Secure Mode

The POSIX client reads the `mapr-clusters.conf` file to determine whether the process must start in secure or non-secure mode. If security is configured, the servicewithimpersonation ticket file must be present in the default `/tmp` directory. If the ticket file is not in the default `/tmp` directory, specify the location of the ticket file using the `fuse.ticketfile.location` configuration parameter in the `fuse.conf` file

Tip: See also: Enabling Impersonation for the MapR Superuser on page 1480 and Enabling Impersonation for any User on page 1480.

If the ticket expires after a connection has been established between the POSIX client and the cluster, the connection can stay alive for up to an hour. No new connections will be allowed. If the ticket was blacklisted, restart POSIX client to refresh the ticket.

Mounting the MapR File System

To mount the MapR filesystem at the mount point specified in the `/opt/mapr/conf/fuse.conf` file, create the mountpoint specified in the `fuse.conf` file and start the service. For example:

```
mkdir /mapr
service mapr-posix-client-* start
```

Note: When you run the command, replace * with basic or platinum, which corresponds with the package that is installed on the system.

Attention: Remember the following points when using a FUSE mounted filesystem:

- When trying to open a FIFO on a FUSE mounted filesystem, permissions to perform the operation are not checked.
- Any user can set time using `touch -t` for any file on a FUSE mounted filesystem.

See also: Enabling Soft Mount and Setting the Timeout on page 420

Monitoring the POSIX Client

To determine whether the POSIX client is running, run the following command:

```
service mapr-posix-client-* status
```

Note: When you run the command, replace * with basic or platinum, which corresponds with the package that is installed on the system.

Adding and Removing Users

Before you add and/or remove users using the POSIX client, make a note of the following:

- Invalid UID/GID cannot perform operations on the system.
- When you add or remove users, it may take up to 30 minutes for the changes to take effect.

last-updated: Apr 11, 2022

By default, the UID cache will expire in 30 minutes. To disable UID cache, set the value (in seconds) for `fs.mapr.uid.cache.timeout.seconds` parameter in the `core-site.xml` file. You can set the value to:

- `0` to fetch the GID information from the idstore directly
- `>0` to specify the amount of time to keep the UID information in cache

For example, your `core-site.xml` entry would look similar to the following for:

- Disabling cache:

```
<property>
   <name>fs.mapr.uid.cache.timeout.seconds</name>
   <value>0</value>
   <description>disable UID cache</description>
</property>
```

- Setting 3 minutes as the amount of time to keep the UID information in cache:

```
<property>
   <name>fs.mapr.uid.cache.timeout.seconds</name>
   <value>180</value>
   <description>UID cached for 3 minutes</description>
</property>
```

Registering POSIX Client with Additional Clusters

To register the POSIX client with additional clusters, you must add entries directly to the `/opt/mapr/conf/mapr-clusters.conf` file. The clusters will be visible after few minutes.

Note: Each client supports up to 16 clusters.

Configuring the FUSE Read Chunk Size

The POSIX FUSE platinum client can break large reads into multiple pieces to be handled in parallel, if you set the FUSE read chunk size of a file. This process is called sharding.

By default, the FUSE read chunk size is set to 2 MB. To change the chunk size used by the FUSE platinum client for parallel reads, set the value (in bytes) for the `fs.mapr.fuseshard.chunksize` configuration field in the `core-site.xml` file. To set the chunksize to 5 MB (5242880 bytes), use:

```
<property>
        <name>fs.mapr.fuseshard.chunksize</name>
        <value>5242880</value>
        <description>setting chunk size</description>
        </property>
```

For example, if the FUSE read chunk size is set to 1 MB, and the FUSE platinum client is configured with 5 libraries, then the platinum client reads 5 MB in parallel.

Unmounting the FUSE Mount

To unmount the mountpoint and kill the FUSE process, run the following command:

```
service mapr-posix-client-* stop
```

Note: When you run the command, replace * with basic or platinum, which corresponds with the package that is installed on the system.

Troubleshooting the FUSE-Based POSIX Client
Explains how to enable and collect the stack trace to troubleshoot POSIX client issues.

This section contains information for troubleshooting the FUSE-based POSIX client.

Enabling Traces

To enable traces at system startup, set the property `fs.mapr.trace` in the `core-site.xml` file. For example:

```
<property>
   <name>fs.mapr.trace</name>
   <value>DEBUG | INFO | WARN | ERROR | CRITICAL | OFF</value>
   <description> </description>
</property>
```

Collecting the Stack Trace

If the mountpoint is not responding or if the filesystem operations are taking too much time, collect the stack trace of all the threads to debug. To collect the stack trace of all threads, run the following command:

```
gstack <fuse-process-id> > ./gstack.log
```

If the filesystem commands fail, repeat the filesystem command with `strace` and collect the log file:

```
strace <filesystem command> > ./strace.log
```

Managing the MAST Gateway

You can start, stop, and restart the MAST Gateway using the MapR Control System and the CLI. You can also configure how frequently MapR runs the load balancer to balance the load on the MAST Gateway.

Configuring the MAST Gateway Service

After installing MAST Gateway service, perform the following steps on the node if MapR File System is not installed on the node. If MapR File System is (also) installed on the node, start at step 4:

1. Run `configure.sh` on page 2053 utility.
   For example:

   ```
   /opt/mapr/server/configure.sh -C <CLDB nodes> -Z <Zookeeper nodes> -N
   <ClusterName>
   ```

2. Start Warden if it is already not running.

   ```
   service mapr-warden start
   ```

3. Run `jps` or `/etc/init.d/mapr-mastgateway status` to check whether MAST Gateway is running on the node.

4. Open the `/opt/mapr/conf/mastgateway.conf` file and set values for the following parameters:

last-updated: Apr 11, 2022

Parameter	Default Value	Description
`mastgateway.port`	8660	The port on which the MAST Gateway process runs. Default value is 8660.
`mastgateway.worker.numthreads`	16	The number of threads to execute tiered data operations such as read and modify part of the offloaded data. The default value is 16. You can modify this based on the machine's configuration.
`mastgateway.cntr.worker.numthreads`	16	The number of threads to use to execute container-based tiered data operations such as offload and recall of file-level and volume-level data in parallel. The default value is 16. **Tip:** For faster offload, modify this value based on the machine's configuration.
`mastgateway.logfile.size.mb`	1024	The maximum size (in MB) of the MAST Gateway log file. When the size limit is reached, the logs get rolled over.

Note: If you modify the `mastgateway.conf` file, you must restart the MAST Gateway for the changes to take effect.

5. (Optional) Add the following parameters in the `/opt/mapr/conf/mastgateway.conf` file only if you wish to customize libcurl.

   The MAST Gateway uses libcurl to perform tiering-related operations. The following table lists the customizable libcurl options and their default values. If these are not set in the `mastgateway.conf` file, the default values are used.

   Note: In the `mastgateway.conf` file, add only the parameters that you wish to customize.

Parameter	Default Value	Description
`mastgateway.curl.timeout`	300000	Timeout for the entire request.
`mastgateway.curl.connecttimeout`	60000	Timeout for the connection phase.
`mastgateway.curl.nosignal`	0	Do not install signal handlers.
`mastgateway.curl.followlocation`	0	Follow HTTP redirects.
`mastgateway.curl.maxsendspeed`	0	Limit on data upload speed.
`mastgateway.curl.maxrecvspeed`	0	Limit on data download speed.
`mastgateway.curl.maxconnects`	5	Maximum number of connections in the connection pool.
`mastgateway.curl.dnsservers`	null	Preferred DNS servers.
`mastgateway.curl.interface`	null	Bind connection locally to this.

Parameter	Default Value	Description
mastgateway.curl.verifypeer	1	Verify the SSL certificate.
mastgateway.curl.verifyhost	2	Verify the host name in the SSL certificate.
mastgateway.curl.cainfo	null	CA cert bundle.
mastgateway.curl.issuercert	null	Issuer certificate.
mastgateway.curl.sslcert	null	Client cert.
mastgateway.curl.sslcerttype	null	Client cert type.
mastgateway.curl.sslkey	null	Client key.
mastgateway.curl.sslkeytype	null	Client key type.
mastgateway.curl.sslkeypasswd	null	Client key password.
mastgateway.curl.proxy	null	Proxy to use. **Note:** Specify a value for this parameter to configure MAST Gateway to use a proxy server. See example below this table for more information.
mastgateway.curl.preproxy	null	Socks proxy to use.
mastgateway.curl.proxyport	0	Proxy port to use. **Note:** Specify a value for this parameter to configure MAST Gateway to use a proxy server. See example below this table for more information.
mastgateway.curl.proxytype	0	Proxy type. **Note:** Specify a value for this parameter to configure MAST Gateway to use a proxy server. See example below this table for more information.
mastgateway.curl.httpproxytunnel	0	Tunnel through the HTTP proxy. **Note:** Specify a value for this parameter to configure MAST Gateway to use a proxy server. See example below this table for more information.
mastgateway.curl.proxyuser	null	Proxy user name. **Note:** Specify a value for this parameter to configure MAST Gateway to use a proxy server. See example below this table for more information.

last-updated: Apr 11, 2022

Parameter	Default Value	Description
`mastgateway.curl.proxypasswd`	null	Proxy password. **Note:** Specify a value for this parameter to configure MAST Gateway to use a proxy server. See example below this table for more information.
`mastgateway.curl.proxyauth`	1	HTTP proxy authentication methods. **Note:** Specify a value for this parameter to configure MAST Gateway to use a proxy server. See example below this table for more information.
`mastgateway.curl.proxyverifypeer`	1	Verify the proxy's SSL certificate.
`mastgateway.curl.proxyverifyhost`	2	Verify the proxy certificate's name against host.
`mastgateway.curl.proxycainfo`	null	Path to proxy Certificate Authority (CA) bundle.
`mastgateway.curl.proxysslcert`	null	SSL proxy client certificate.
`mastgateway.curl.proxysslcerttype`	null	Type of the proxy client SSL certificate.
`mastgateway.curl.proxysslkey`	null	Private keyfile for TLS and SSL proxy client cert.
`mastgateway.curl.proxysslkeytype`	null	Type of proxy private key file.
`mastgateway.curl.proxysslkeypasswd`	null	Passphrase for proxy private key.

For example, to configure the MAST Gateway for proxy server, your `mastgateway.conf` file settings for proxy server should look similar to the following:

```
mastgateway.curl.proxy=10.20.30.140

mastgateway.curl.preproxy=null

mastgateway.curl.proxyport=3128

mastgateway.curl.proxytype=0

mastgateway.curl.httpproxytunnel=0

mastgateway.curl.proxyuser=proxyuser

mastgateway.curl.proxypasswd=proxyuserpwd
```

```
mastgateway.curl.proxyauth=1
```

6. Save and close the `/opt/mapr/conf/mastgateway.conf` file.
7. (Optional) Configure memory for the MAST Gateway in the `/opt/mapr/conf/conf.d/warden.mastgateway.conf` file by setting values for the following parameters:

Parameter	Default Value	Description
`service.heapsize.min`	2048	The minimum amount of node memory (in MB) to allocate.
`service.heapsize.max`	20480	The maximum amount of node memory (in MB) allocate.
`service.heapsize.percent`	10	The percentage of node memory to allocate.

   By default, 10% of the node memory or 20GB, whichever is lower, is allocated to MAST Gateway. If the MAST Gateway is processing jobs for both warm and cold tiers, memory consumption can increase up to 7GB or more. If you see high memory alarms for small memory consumption also, tune the percentage of memory allocated for MAST Gateway. Ensure that the percentage of memory allocated through `service.heapsize.percent` is available for MAST Gateway.

8. (Optional) Set the value for `fs.mapr.pool.queue.max_size` parameter to 20000 in the `/opt/mapr/conf/dbclient.conf` file.

   If compression is enabled on the data in a tiering-enabled volume, tiering jobs can fail and return errors because of the large number of operations sent to the DB (where metadata for offloaded data is stored). To prevent errors, add the `fs.mapr.pool.queue.max_size` parameter to the `/opt/mapr/conf/dbclient.conf` file and set the value for this parameter to a large number, such as 20000. For example, your entry in the `/opt/mapr/conf/dbclient.conf` file should look similar to the following:

   ```
   fs.mapr.pool.queue.max_size = 20000
   ```

9. Restart the MAST Gateway for the changes to take effect.

   See Starting, Stopping, and Restarting the MAST Gateway on page 1263 for more information.

Configuring Secure Access

If the MapR cluster is a secure cluster and the MAST Gateway is installed on a cluster node, no additional configuration is needed for the MAST Gateway to access data. On the other hand, if the MapR cluster is a secure cluster and if MAST Gateway is installed on an edge node, to enable the MAST Gateway to communicate with the secure MapR cluster, do the following:

1. Copy the `/opt/mapr/conf/maprserverticket`, `/opt/mapr/conf/ssl_keystore`, and `/opt/mapr/conf/ssl_truststore` files on the CLDB node to the `/opt/mapr/conf` directory on the edge node.
2. Run `configure.sh` as shown below:

   ```
   /opt/mapr/server/configure.sh -C <cldb-node-IP-addresses> -Z
   <zookeeper-node-IP-addresses> -secure -N <cluster-name>
   ```

   See `configure.sh` on page 2053 for more information.

Starting, Stopping, and Restarting the MAST Gateway

You can start, stop, and restart the MAST Gateway using the Control System and the CLI.

Starting, Stopping, and Restarting the MAST Gateway Using the Control System

See:

- Starting the Services on the Cluster Using the Control System on page 830
- Stopping a Service on the Cluster Using the Control System on page 831
- Restarting the Services on the Cluster Using the Control System on page 832

Starting, Stopping, and Restarting the MAST Gateway Using the CLI

- Run the following command to:
  - Start the MAST Gateway:

```
maprcli node services -nodes <nodename|IP_address> -name
mastgateway -action start
```

  - Stop the MAST Gateway:

```
maprcli node services -nodes <nodename|IP_address> -name
mastgateway -action stop
```

  - Restart the MAST Gateway:

```
maprcli node services -nodes <nodename|IP_address> -name
mastgateway -action restart
```

Balancing Gateway Load

Explains how CLDB balances MAST Gateway loads.

CLDB assigns volumes to MAST Gateways so that any tiering-related operation for a volume is performed by the MAST Gateway assigned to the volume. This assignment is sticky and the volume remains assigned to the same gateway across CLDB, MAST Gateway, and cluster restarts. When a MAST Gateway goes down, volumes assigned to the MAST Gateway are not re-assigned immediately. Instead, when a tiering operation needs to be run and the assigned MAST Gateway is down, CLDB assigns a new MAST Gateway to the volume, and the operation is performed using the newly assigned MAST Gateway.

Volumes are assigned to gateways with the lowest load (or lowest number of volumes currently assigned to it) to ensure equal distribution. Whenever a volume is created or removed or whenever a MAST Gateway is added or removed, the load on the gateways require rebalancing. MapR automatically balances the load on the gateways after a certain (configurable) amount of time since the occurrence of the event that necessitates a rebalance. The delay after which MapR tries to rebalance varies based on the type of event that necessitates a rebalance. See Configuring the Delay After Which Load Balancer runs for Events on page 1265 for more information.

Each volume in the cluster is assigned a weight, which is 1 for all volumes. The load on a gateway is the sum of weights of all the volumes that are assigned to the gateway. Load balancer tries to ensure that the load on a gateway is at least the average weight.

```
avg weight = (sum(weight of all tiered vols) + num active gws - 1) / num
active gws
```

When the balancer needs to pick volumes from a gateway for reassignment, it first picks the volume with max weight and one that currently has no activity (volume level offload, recall, or compaction). To minimize the interruptions in gateway activity, the balancer first considers idle volumes and picks volumes with active

tasks after. However, any balancing and/or reassignment is skipped if there is already assignment related flux in the cluster (such as volumes with gateway assignment currently in progress). If this happens, load balancer runs again with a shorter time delay of 10 minutes.

MapR performs load balancing in batches of 5 volumes per run. That is, it assigns 5 volumes to gateways and then after a delay of 10 minutes by default, runs the load balancer again to distribute other volumes (in batches of 5). MapR figures out the next batch of volumes by re-evaluating the current assignment state. When the load on the gateway is balanced, the balancer is disabled and the load balancer is run again only by any of the 4 events.

Configuring Interval Between Load Balancer Runs

- MapR assigns 5 volumes to gateways and then after a delay of 10 minutes by default, runs the load balancer again to distribute other volumes (in batches of 5). Run the following command to configure the interval between runs:

```
maprcli config save -values
{"cldb.tier.gw.balance.delay.recheck":"<time-in-seconds>"}
```

Configuring the Delay After Which Load Balancer runs for Events

MapR runs the load balancer automatically for any of the following events. You can configure the delay after which MapR runs the load balancer.

Create Volume	When you create a tiering-enabled volume, it is assigned the gateway with the lowest load. If gateways are not available for the volume at the time of creation, the volume might stay unassigned. To ensure that the volume has an associated gateway for handling the tiering operations, MapR runs the gateway load balancer after a delay of 2 hours (7200 seconds) by default for this event. To configure the delay, set the value for `cldb.tier.gw.balance.delay.vol.create` property (in seconds) using the `config save` on page 1586 command. For example: `maprcli config save -values {"cldb.tier.gw.balance.delay.vol.create":"<time-in-seconds>"}`
Remove Volume	When you remove a volume, the distribution of volumes across gateways can become uneven, and the balancer must be run to redistribute the volumes. For this event, MapR runs the load balancer to redistribute the volumes and rebalance the gateways after a delay of 2 hours (7200 seconds) by default for this event. To configure the delay, set the value for the `cldb.tier.gw.balance.delay.vol.delete` property (in seconds) using the `config save` on page 1586 command. For example: `maprcli config save -values {"cldb.tier.gw.balance.delay.vol.delete":"<time-in-seconds>"}`
Add Gateway	When you install a new gateway on a cluster, volumes are assigned to the new gateway only if a new volume is created or an existing volume has a pending

last-updated: Apr 11, 2022

	task, but no active assigned gateway. Volumes that have active gateways are not re-assigned by default. For this event, MapR runs the load balancer to re-distribute the volumes and rebalance the gateway load after a delay of 2 hours (7200 seconds) by default. To configure the delay, set the value for the `cldb.tier.gw.balance.delay.new.gw` property (in seconds) using the `config save` on page 1586 command. For example: `maprcli config save -values {"cldb.tier.gw.balance.delay.new.gw":"<time-in-seconds>"}`
Remove Gateway	When you remove a gateway or when a gateway goes down, all the volumes assigned to the gateway are not re-assigned by default; only volumes with pending tasks are assigned to new gateways. For this event, MapR runs the load balancer to re-distribute the volumes after 6 hours (21600 seconds) by default. To configure the delay, set the value for the `cldb.tier.gw.balance.delay.dead.gw` property (in seconds) using the `config save` on page 1586 command. For example: `maprcli config save -values {"cldb.tier.gw.balance.delay.dead.gw":"<time-in-seconds>"}`

Determining the Volumes Assigned to MAST Gateways

- Run one of the following commands to determine the volumes that are assigned to MAST Gateways:
  - `/opt/mapr/server/mrconfig info mastgateway`
  - `/opt/mapr/server/mrconfig info volume mastgateway`

  For more information, see `mrconfig info` on page 2152.

Enabling Debug Logging for MAST Gateway

The MAST Gateway service log file contains alarm messages, error codes, and information on the errors. When the `mastgateway.log` file reaches `mastgateway.logfile.size.mb/5`, a roll over happens and the `mastgateway.log` file is renamed as `mastgateway.log.1`; also, a new `mastgateway.log` file is created. When the newly created `mastgateway.log` reaches `mastgateway.logfile.size.mb/5`, a roll over happens again and:

- The `mastgateway.log.1` is renamed as `mastgateway.log.2`
- The `mastgateway.log` is renamed as `mastgateway.log.1`
- A new `mastgateway.log` is created

This process continues and up to 5 files, whose size is `mastgateway.logfile.size.mb/5`, are created before the oldest log file, `mastgateway.log.4`, is deleted.

- Run the following command to enable debug logging for MAST Gateway:

```
maprcli trace setlevel -module MASTGateway -level Debug -port 8660
```

Configuring NodeManager Restart

NodeManager restart is enabled by default. Active containers will keep running in the event that the NodeManager shuts down.

When the NodeManager restart is enabled, it stores the container state of active containers in a recovery directory;when the NodeManager restarts, it retrieves the container state from the recovery directory.

If you disable NodeManager restart, active containers are shut down when the NodeManager shuts down and containers need to be reallocated when the NodeManager starts again.

To configure NodeManager restart, enable the NodeManager recovery and also specify a port that can be dedicated to run the NodeManager service.

1. Add the following parameters to the `yarn-site.xml` on each NodeManager node:
   a) Set `yarn.nodemanager.recovery.enabled` to true.
   b) Set `yarn.nodemanager.address` to include a port that is dedicated to run the NodeManager on this node.
   c) Optionally, set `yarn.nodemanager.recovery.dir` to a different recovery directory for this node.
      By default, the recovery directory is set to`$hadoop.tmp.dir/yarn-nm-recovery` which resolves to `tmp/hadoop-mapr/nm-local-dir/yarn-nm-recovery`. See the following example configuration:

```
<property>
  <name>yarn.nodemanager.recovery.enabled</name>
  <value>true</value>
</property>
<property>
  <name>yarn.nodemanager.address</name>
  <value>0.0.0.0:8099</value>
</property>
```

2. Restart the NodeManager Service.
   For more information, see Managing Services on page 827.

Managing Jobs and Applications

If you have used Hadoop in the past to run MapReduce applications, then running jobs on the MapR Data Platform platform will be very familiar to you. MapR Data Platform is a full Hadoop distribution, API-compatible with all versions of Hadoop. MapR Data Platform provides additional capabilities not present in any other Hadoop distribution.

You can perform the following procedures to manage applications in a MapR Data Platform cluster:

Job Scheduling

You can use job scheduling to prioritize the YARN applications that run on your MapR cluster.

The MapReduce system supports a minimum of one queue, named `default`. Hence, this parameter's value should always contain the string `default`. Some job schedulers, like the Capacity Scheduler, support multiple queues.

last-updated: Apr 11, 2022

The default job scheduler is the Fair Scheduler, which is designed for a production environment with multiple users or groups that compete for cluster resources.

The MapR Converged Data Platform supports these job schedulers:

- **FIFO queue-based scheduler**: The FIFO queue scheduler runs jobs based on the order in which the jobs were submitted. You can prioritize a job by changing the value of the `mapred.job.priority` property or by calling the `setJobPriority()` method.
- **Fair Scheduler**: This is the default scheduler. The Fair Scheduler allocates a share of cluster capacity to each user over time. The design goal of the Fair Scheduler is to assign resources to jobs so that each job receives an equal share of resources over time. The Fair Scheduler enforces fair sharing within each queue. Running jobs share the queue's resources.
- **Capacity Scheduler**: The Capacity Scheduler enables users or organizations to simulate an individual hadoop cluster with FIFO scheduling for each user or organization. You can define organizations using *queues*.

The following sections provide more information about job scheduling:

Hadoop 2.x Fair Scheduler

The FairScheduler is a pluggable scheduler for Hadoop that allows YARN applications to share resources in a large cluster fairly. Fair scheduling is a method of assigning resources to applications such that all applications get, on average, an equal share of resources over time. Hadoop 2.x is capable of scheduling multiple resource types.

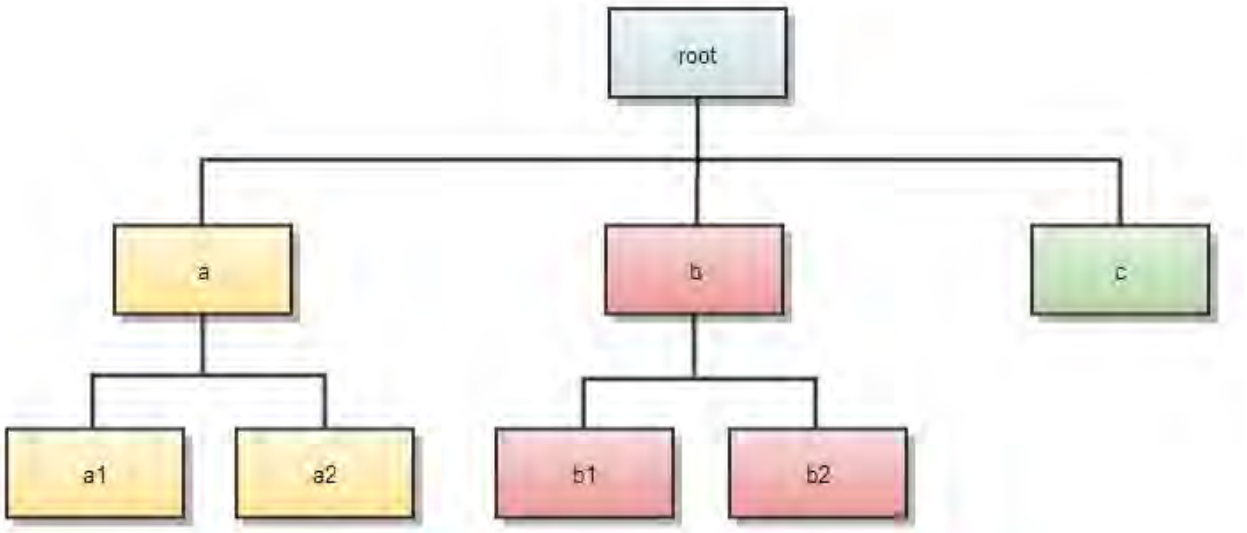
By default, the Fair Scheduler bases scheduling fairness decisions only on memory. It can be configured to schedule resources based on memory, CPU, and disk usage. When only one application is running, that application uses the entire cluster. When other applications are submitted, resources that free up are assigned to the new applications, so that each application eventually gets approximately the same amount of resources. Unlike the default Hadoop scheduler, which forms a queue of applications, this lets short applications finish in reasonable time while not starving long-lived applications. It is also a reasonable way to share a cluster between a number of users. Finally, fair sharing also uses priorities applied as weights to determine the fraction of total resources that each application should get.

For additional information about Hadoop Fair Scheduler, you can also refer to the open source documentation.

Scheduling Queues

The scheduler organizes applications further into *queues*, and shares resources fairly between these queues. By default, all users share a single queue, named `default`. If an application specifically lists a queue in a container resource request, the request is submitted to that queue. You can also assign queues based on the user name included with the request through configuration. Within each queue, a scheduling policy is used to share resources between the running applications. The default is memory-based fair sharing, but FIFO and multi-resource with Dominant Resource Fairness can also be configured.

Queues can be arranged in a hierarchy to divide resources, and they can be configured with weights to share the cluster in specific proportions. The Fair Scheduler uses a concept called a *queue path* to configure a hierarchy of queues. The queue path is the full path of the queue's hierarchy, starting at *root.* The following example has three top-level child-queues a, b, and c and some sub-queues for a and b:



In addition to providing fair sharing, the Fair Scheduler allows assigning guaranteed minimum shares to queues, which is useful for ensuring that certain users, groups or production applications always get sufficient resources. When a queue contains apps, it gets at least its minimum share, but when the queue does not need its full guaranteed share, the excess is split between other running apps. This lets the scheduler guarantee capacity for queues while utilizing resources efficiently when these queues do not contain applications.

Configuring the Fair Scheduler

The Fair Scheduler lets all applications run by default, but you can also limit the number of running applications per user and per queue through the configuration file. This can be useful when a user must submit hundreds of applications at once, or in general to improve performance if running too many applications at once would cause too much intermediate data to be created or too much context-switching. Limiting the applications does not cause any subsequently submitted applications to fail; it only causes them to wait in the scheduler's queue until earlier applications finish.

To customize the Fair Scheduler, set the configuration properties in `yarn-site.xml` and update the allocation file to list existing queues and their respective weights and capacities. The allocation file is automatically created during MapR installation in the following directory:

```
{$MAPR_HOME}/hadoop/hadoop-2.x/etc/hadoop/fair-scheduler.xml
```

The allocation file is reloaded every 10 seconds to refresh the scheduler with any modified settings that are specified in the file.

Specifying Fair Scheduler Configuration Properties in yarn-site.xml
Lists the properties in the *yarn-site.xml* file for Fair Scheduler.

The `yarn-site.xml` file contains the following parameters that determine scheduler-wide options.

yarn.scheduler.fair.allocation.file	*Default Value*: `fair-scheduler.xml` *Description*: Specifies the path to the allocation file. If a relative path is given, the file is searched for on the classpath.
yarn.scheduler.fair.user-as-default-queue	*Default Value*: `true` *Description*: Determines whether to use the username associated with the allocation file as the default queue name, if a queue name is not specified.

last-updated: Apr 11, 2022

	Note: If a queue placement policy is given in the allocations file, this property is ignored.
yarn.scheduler.fair.preemption	*Default Value*: `false` *Description*: Indicates whether to use preemption. **Note:** Do not use preemption when `FairScheduler DominantResourceFairness` is in use and node labels are present.
yarn.scheduler.fair.sizebasedweight	*Default Value*: `false` *Description*: Indicates whether to assign shares to individual applications based on their size, rather than providing an equal share to all applications regardless of size. When set to `true`, applications are weighted by *(ln 1 + <application's total requested memory>)/ ln 2*.
yarn.scheduler.fair.assignmultiple	*Default Value*: `false` *Description*: Indicates whether to allow multiple container assignments in one heartbeat.
yarn.scheduler.fair.resources-based-on-labels-enabled	*Default Value*: `false` *Description*: Indicates whether to allow container allocation on all nodes by recomputing fair shares based on labels.
yarn.scheduler.fair.preemption.cluster-utilization-threshold.based-on-labels-enabled	*Default Value*: `false` *Description*: Indicates whether to enable/disable preemption of the threshold per-label.

Fair Scheduler Allocation File
Describes an allocation file and the entities within an allocation file.

An allocation file is an XML manifest that describes queues and their properties, as well as certain policy defaults. The allocation file is automatically created during MapR installation in the following directory:

```
{$MAPR_HOME}/hadoop/hadoop-2.x/etc/hadoop/fair-scheduler.xml
```

The allocation file is reloaded every 10 seconds to refresh the scheduler with any modified settings that are specified in the file.

The allocation file contains the following types of elements:

Queue Elements

Queue elements represent queues and can contain the following elements:

- `minResources`
- `maxResources`
- `maxRunningApps`

  Tip: The `queueMaxAppsDefault` value is used for any parent queue that does not set a value for the `maxRunningApps` element.

- `weight`

last-updated: Apr 11, 2022

- `schedulingPolicy`
- `aclSubmitApps`
- `aclAdministerApps`
- `minSharePreemptionTimeout`
- `maxContainerAllocation`

  Tip: The `maxContainerAllocation` property sets a limit on the resources a queue can allocate for a single container. The value cannot exceed `maxResources`. If you do not set `maxContainerAllocation`, the value is inherited from a parent queue. The default values are set through the `yarn.scheduler.maximum-allocation-mb`, `yarn.scheduler.maximum-allocation-vcores`, and `yarn.scheduler.maximum-allocation-disks` properties, which you can modify in `/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop/yarn-site.xml`. The `maxContainerAllocation` element is not valid for the root queue.

For more information on these elements, see Hadoop: Fair Scheduler.

User Elements

User elements represent settings that govern the behavior of individual users. They can contain a single property, `maxRunningApps`, which limits the number of running applications for a particular user. It contains the following elements:

- `userMaxAppsDefault`
- `queueMaxAppsDefault`
- `fairSharePreemptionTimeout`
- `defaultQueueSchedulingPolicy`
- `queuePlacementPolicy`

For more information on these elements, see Hadoop: Fair Scheduler.

Tip: If you set a value for `queueMaxAppsDefault` and do not set a value for `maxRunningApps` for the root queue, the value of `queueMaxAppsDefault` sets the application limit for all queues under the root queue hierarchy.

Example Allocation File

See example allocation file in Hadoop: Fair Scheduler.

Queue Access Control Lists

Queue Access Control Lists (*ACL*s) allow administrators to control who may take actions on particular queues. They are configured with the `aclSubmitApps` and `aclAdministerApps` properties, which can be set per queue. Currently, the only supported administrative action is killing an application. Anyone who has permission to administer a queue may also submit applications to it. These properties take values in a format such as user1,user2 group1,group2 or group1,group2. An action on a queue is permitted if its user or group is in the *ACL* of that queue or in the *ACL* of any of that queue's ancestors. Therefore, if queue2 is inside queue1, and user1 is in queue1's *ACL*, and user2 is in queue2's *ACL*, then both users may submit to queue2.

last-updated: Apr 11, 2022

For more information, see Hadoop: Fair Scheduler.

The `yarn.admin.acl and yarn.acl.enable` Properties	By default, on a secure cluster, users cannot kill jobs that do not belong to them.
	On a secure cluster, you do not need to set the `yarn.acl` or the `yarn.admin.acl` properties. By default, they are set as follows. On unsecured clusters, these properties are not set by default.
	`<property>`
	`<name>yarn.acl.enable</name> >`
	`<value>true</value> >`
	`</property>`
	`<property>`
	`<name>yarn.admin.acl</name> >`
	`<value> </value> >`
	`</property>`
	The `yarn.admin.acl` property is set by default to " ", meaning that an administrator is not specified on a cluster.
	To allow users to kill jobs that do not belong to them, or to get access to their logs, you need to set the `yarn.admin.acl` property with the user or group name.

Fair and Capacity scheduler root queue admins	For both the Fair scheduler and Capacity scheduler, the default value of the administrators for the root queues is " " .

Hadoop 2.x Capacity Scheduler

The `CapacityScheduler` is a pluggable scheduler for Hadoop that allows multiple tenants to securely share a large cluster. Resources are allocated to each tenant's applications in a way that fully utilizes the cluster, governed by the constraints of allocated capacities.

Queues are typically set up by administrators to reflect the economics of the shared cluster. The Capacity Scheduler supports hierarchical queues to ensure that resources are shared among the sub-queues of an organization before other queues are allowed to use free resources.

The following sections provide more information about the CapacityScheduler:

Capacity Scheduler Features

The CapacityScheduler supports these features:

- **Hierarchical Queues** Hierarchical queues ensure that resources are shared among the sub-queues of an organization before other queues are allowed to use free resources, thereby providing more control and predictability.
- **Capacity Guarantees** Queues are allocated a fraction of the capacity of the grid, which means that a certain capacity of resources are at their disposal. All applications submitted to a queue have access to the capacity allocated to the queue. Administrators can configure soft limits and optional hard limits on the capacity allocated to each queue.
- **Security** Each queue has strict Access Control Lists (ACLs). The ACLs control which users can submit applications to individual queues. Also, safeguards ensure that users cannot view or modify applications from other users. Per-queue and system administrator roles are also supported.

- **Elasticity** Free resources can be allocated to any queue beyond its capacity allocation. As tasks scheduled on these resources complete, the resources become available to be reassigned to applications on queues running below their capacity. (Note that pre-emption is not supported.) This ensures that resources are available in a predictable and elastic manner to queues, thus preventing artificial silos of resources in the cluster and improving cluster utilization.
- **Multi-tenancy** A comprehensive set of limits is provided to prevent a single application, user, or queue from monopolizing resources of the queue or the cluster as a whole. This ensures that the cluster is not overwhelmed.
- **Operability**
  - **Runtime Configuration** The queue definitions and properties, such as capacity or ACLs, can be changed in a secure manner by administrators at runtime, which minimizes disruption to users. Also, a console is provided for users and administrators to view the current allocation of resources to various queues in the system. Administrators can add queues at runtime, but queues cannot be deleted at runtime.
  - **Drain applications** Administrators can stop queues at runtime to ensure that while existing applications run to completion, no new applications can be submitted. If a queue is in the STOPPED state, new applications cannot be submitted to that queue or any of its child queues. Existing applications continue to completion, so the queue can be drained gracefully. Administrators can also start the stopped queues.
- **Resource-based Scheduling** Support for resource-intensive applications, where an application can optionally specify higher resource requirements than the default, thereby accommodating applications with differing resource requirements. Currently, *memory* is the only supported resource requirement.

Setting Up ResourceManager to Use CapacityScheduler

To configure the ResourceManager to use the CapacityScheduler, set the following property in the `yarn-site.xml` file:

Property Name	Value
`yarn.resourcemanager.scheduler.class`	`org.apache.hadoop.yarn.server.resourcemanager.scheduler.capacity. CapacityScheduler`

Setting Up Queues

The ResourceManager uses the configuration file `capacity-scheduler.xml`, where you can configure various scheduling parameters related to queues. These parameters include:

- the short queue name
- the full queue path name
- a list of associated child queues and applications
- the guaranteed capacity (expressed as a percentage of total resources in the cluster) available to the jobs in the queue
- the maximum capacity of the queue
- a list of active users and their resource allocation limits
- the state of the queue (running or stopped)
- access control lists that determine who can access the queue

The CapacityScheduler has a pre-defined queue called *root*. All queues in the system are children of the `root` queue. Further queues can be set up by configuring `yarn.scheduler.capacity.root.queues` with a list of comma-separated child queues.

Queue Properties

The `capacity-scheduler.xml` file includes three types of queue properties:

Resource Allocation Properties

The following table lists resource allocation properties:

Property	Description
`yarn.scheduler.capacity.<queue-path>.capacity`	Queue *capacity* in percentage (%) expressed as a float (for example, 12.5). The sum of capacities for all queues, at each level, must equal 100. Applications in the queue may consume more resources than the queue's capacity if there are free resources, which provides elasticity.
`yarn.scheduler.capacity.<queue-path>.maximum-capacity`	Maximum queue capacity in percentage (%) expressed as a float. This property limits the elasticity for applications in the queue. The default is -1 which disables it.
`yarn.scheduler.capacity.<queue-path>.minimum-user-limit-percent`	Sets the minimum value, expressed as an integer, on the percentage of resources allocated to a user, if there is a demand for resources. A value of 100 implies no user limits are imposed. The default is 100. The maximum value depends on the number of users who have submitted applications. For example, if this property is set to 25 and two users have submitted applications to a queue, the maximum percent of queue resources for each user is 50%. If a third user submits an application, no single user can use more than 33% of the queue resources. With four or more users, no user can use more than 25% of the queue resources.
`yarn.scheduler.capacity.<queue-path>.user-limit-factor`	The multiple of the queue capacity that can be configured to allow a single user to acquire more resources. Value is specified as a float. The default is 1, which ensures that a single user can never take more than the queue's configured capacity no matter how idle the cluster is.

last-updated: Apr 11, 2022

Property	Description
`yarn.scheduler.capacity.resource-calculator`	Specifies the `ResourceCalculator` implementation to be used to compare resources in the scheduler. The default value is `DiskBasedResourceCalculator`, which uses memory, CPU and disk. Other values for this parameter include: • `DefaultResourceCalculator`, which uses memory only • `DominantResourceCalculator`, which uses dominant-resource to compare multi-dimensional resources such as memory and CPU • `DiskBasedDominantResourceCalculator`, which uses dominant-resource to compare multi-dimensional resources, such as memory, CPU and disk

Running and Pending Application Limits

Applications are considered active if they are either running or pending. The following table lists properties that specify running and pending application limits:

Property	Description
`yarn.scheduler.capacity.maximum-applications`	Maximum number of applications in the system that can be concurrently active, both running and pending. Limits on each queue are directly proportional to their queue capacities and user limits. This is a hard limit; any applications submitted when this limit is reached will be rejected. The default is 10000. This applies to all queues.
`yarn.scheduler.capacity.<queue-path>.maximum-applications`	Overrides `yarn.scheduler.capacity.maximum-applications` on a per queue basis.
`yarn.scheduler.capacity.maximum-am-resource-percent`	Maximum percent of resources in the cluster that can be used to run application masters - controls the number of concurrent active applications. Limits on each queue are directly proportional to their queue capacities and user limits. Specified as a float. For example, 0.5 = 50%. The default is 0.1. This can be set for all queues with yarn.scheduler.capacity.maximum-am-resource-percent
`yarn.scheduler.capacity.<queue-path>.maximum-am-resource-percent`	Overrides `yarn.scheduler.capacity.maximum-am-resource-percent` on a per queue basis.

Queue Administration and Permissions

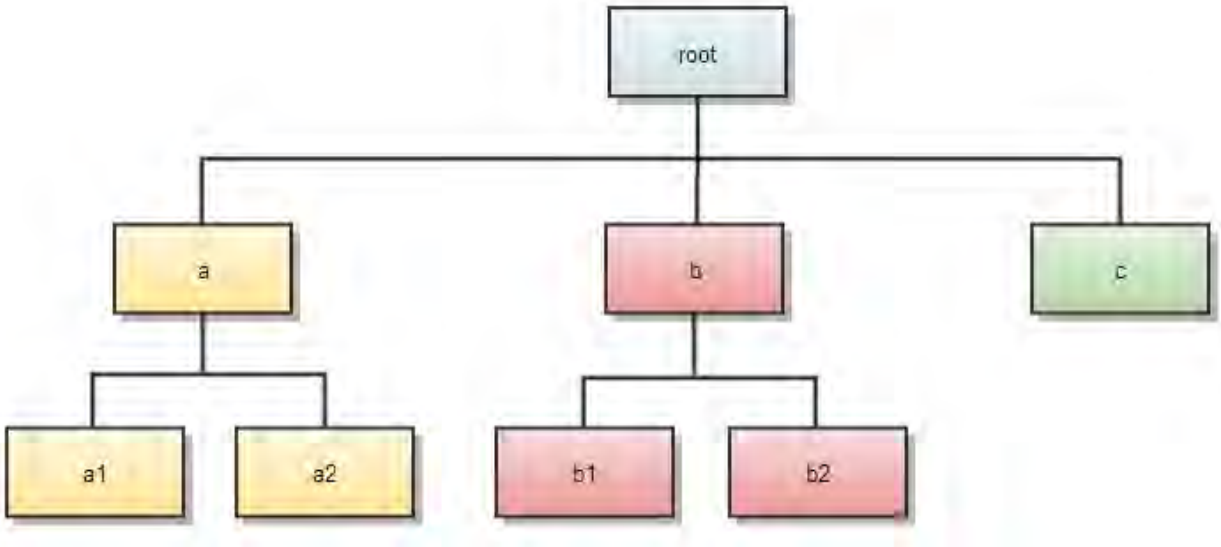
The following table lists queue administration and permission properties:

last-updated: Apr 11, 2022

Property	Description
`yarn.scheduler.capacity.<queue-path>.state`	The *state* of the queue. Possible values are RUNNING or STOPPED. If a queue is in the STOPPED state, new applications cannot be submitted to that queue or any of its child queues. If the *root* queue is STOPPED, no applications can be submitted to the entire cluster. Existing applications continue to completion, so the queue can be *drained* gracefully.
`yarn.scheduler.capacity.root.<queue-path>.acl_submit_applications`	The *ACL* that controls who can *submit* applications to the given queue. If the given user/group belongs to the *ACL* on a given queue or one of the parent queues in the hierarchy, they can submit applications. *ACLs* for this property *are* inherited from the parent queue if not specified.
`yarn.scheduler.capacity.root.<queue-path>.acl_administer_queue`	The *ACL* that controls who can *administer* applications on the given queue. If the given user/group has the necessary ACLs on the given queue or one of the parent queues in the hierarchy, they can administer applications. *ACLs* for this property *are* inherited from the parent queue if not specified.

Setting Up a Hierarchy of Queues

CapacityScheduler uses a concept called a *queue path* to configure a hierarchy of queues. The queue path is the full path of the queue's hierarchy, starting at *root.* The following example has three top-level child-queues a, b, and c and some sub-queues for a and b:



Queue paths are defined for each level under the `root` queue. A queue's children are defined with the parameter `yarn.scheduler.capacity.<queue-path>.queues`, where `<queue-path>` takes the form `root.<child>`, `root.<child>.<child>`, and so on. For example, the queue path to a2 is designated as `root.a.a2`.

Warning:

Children do not inherit properties directly from the parent unless otherwise noted.

last-updated: Apr 11, 2022

The corresponding queue definition block of the `capacity-scheduler.xml` file is shown below.

```
<property>
  <name>yarn.scheduler.capacity.root.queues</name>
  <value>a,b,c</value>
  <description>The queues at this level (root is the root queue).
  </description>
</property>

<property>
  <name>yarn.scheduler.capacity.root.a.queues</name>
  <value>a1,a2</value>
  <description>The queues at this level (root is the root queue).
  </description>
</property>

<property>
  <name>yarn.scheduler.capacity.root.b.queues</name>
  <value>b1,b2,b3</value>
  <description>The queues at this level (root is the root queue).
  </description>
</property>
```

Changing Queue Configuration

You can change queue properties and add new queues by editing `capacity-scheduler.xml`. Make sure that the updated queue configuration is valid and that the queue-capacity at each level equals 100%.

For the changes to take effect, run the following command:

```
yarn rmadmin -refreshQueues
```

Warning:

Queues cannot be deleted, only added.

Queue Access Control Lists
Describes how to restrict access to queues using Access Control Lists (ACLs).

Queue *ACL*s allow administrators to control who may take actions on particular queues. They are configured with the following properties:

```
yarn.scheduler.capacity.root.support.acl_submit_applications
yarn.scheduler.capacity.root.support.acl_administer_queue
```

You can set these properties for each queue. Currently, the only supported administrative action is killing an application. Anyone who has permission to administer a queue may also submit applications to it. These properties take values in a format such as `user1,user2 group1,group2` or `group1,group2`. An action on a queue is permitted if its user or group is in the *ACL* of that queue, or in the *ACL* of any of that queue's ancestors. So if queue2 is inside queue1, and user1 is in queue1's *ACL*, and user2 is in queue2's *ACL*, then both users may submit to queue2.

The root queue's *ACL*s are "*" by default. Since *ACL* are passed down, by default, everyone may submit to, and kill applications from every queue. To restrict access, change the root queue's *ACL*s to something other than `*`.

last-updated: Apr 11, 2022

By default, the yarn.admin.acl property in yarn-site.xml is also set to *, which means that any user can be the administrator. If queue *ACL*s are enabled, you also need to set the yarn.admin.acl property to the correct admin user for the YARN cluster. For example:

```
<property>
<name>yarn.admin.acl</name> >
<value>mapr</value> >
</property>
```

If you do not set this property correctly, users can kill YARN jobs even when they do not have access to the queues for those jobs.

Label-based Scheduling

Label-based scheduling provides a way to allocate shared cluster resources on particular nodes in a cluster. First, you assign node labels in a text file. The node labels map to one or more nodes. Next, you can create queue labels and job labels based on the node labels.

When you run jobs, you can place them on specified nodes on a per-job basis (using a job label) or at the queue level (using a queue label). This feature is used in conjunction with schedulers, such as the Fair Scheduler or the Capacity Scheduler.

The following sections provide more information about label-based scheduling:

Label-based Scheduling for YARN Applications

Label-based scheduling provides job placement control on a multi-tenant hadoop cluster. Using label-based scheduling, an administrator can control exactly which nodes are chosen to run jobs submitted by different users and groups. This is useful for data locality and multi-tenancy use cases.

To use label-based scheduling, an administrator assigns node labels in a text file, then composes queue labels or job labels based on the node labels. When you run jobs, you can place them on specified nodes on a per-job basis (using a job label) or at the queue level (using a queue label).

This feature is used in conjunction with schedulers, such as the Fair Scheduler or the Capacity Scheduler.

YARN Resource Calculation Based on Labels

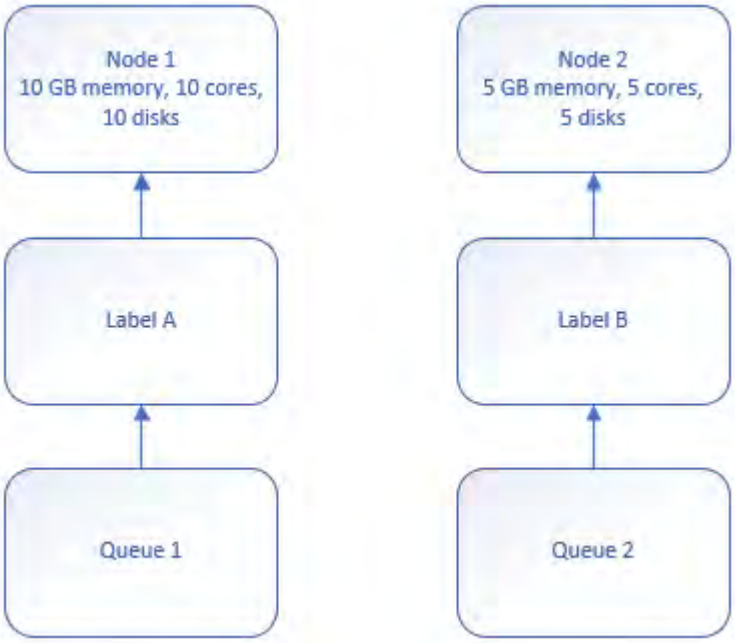
MapR 6.1.0 implements correct steady/instantaneous Fair Scheduler shares, headroom, and maximum resource calculation for queues with label-based scheduling (LBS).

This new approach to YARN resource allocation enables the cluster LBS configuration to compute the resources that are allowed for each queue. It also uses LBS to assign containers to the correct queues to preempt containers, and to prevent resource overuse. The LBS approach to YARN resource allocation relies on:

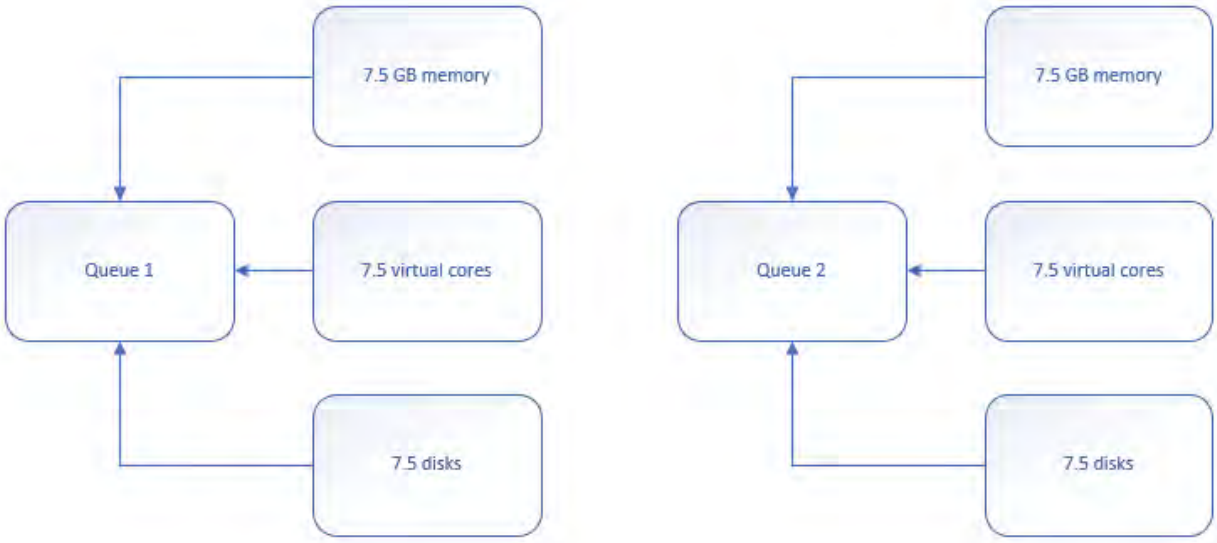
1. Resource computation (memory, CPU, disks) based on labels for each queue.
2. Per-label preemption threshold.

Example of LBS Resource Computation

For example, imagine that you have the following resources:

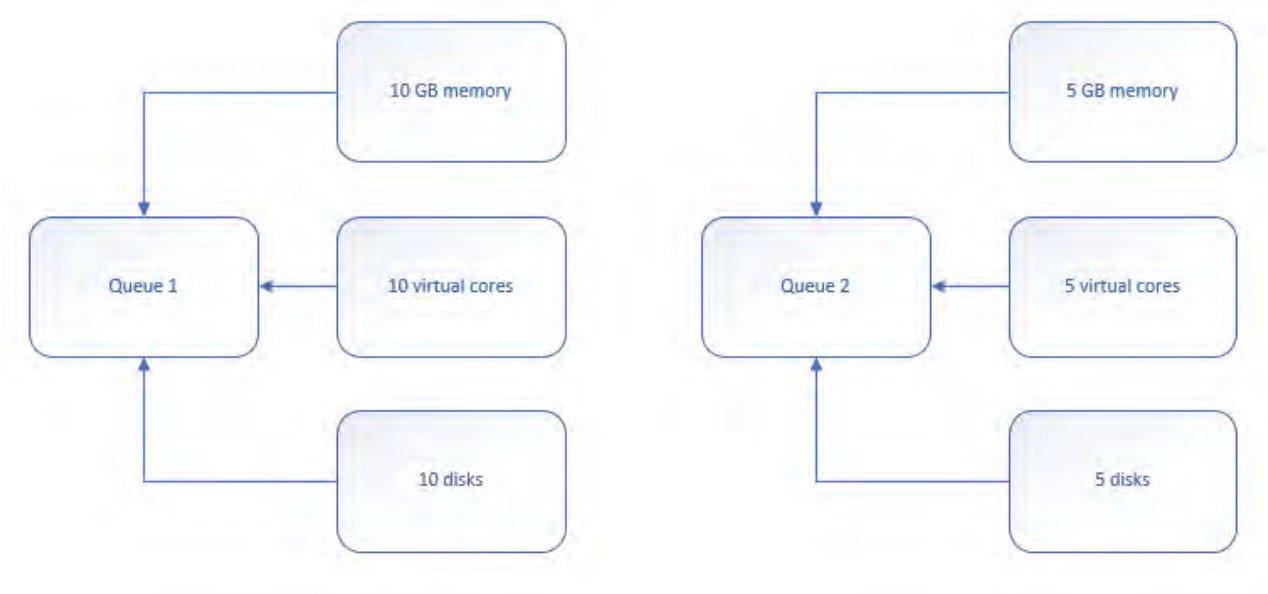


Using the former fair-share resource distribution results in the following:



Using the new LBS resource distribution results in the following:

last-updated: Apr 11, 2022



Understanding LBS preemption thresholds

Before LBS, preemption occurred by default when an entire cluster became 80% full (`yarn.scheduler.fair.preemption.cluster-utilization-threshold=0.8f`). With LBS, preemption occurs per labeled resource, as that resource becomes 80% full.

Descriptions of New Properties When Using LBS

- `defaultQueueLabel`

  Assigned to all new queues and existing queues that do not have a label (excluding the `root` queue).

  For example, `root.%username%` queue is created if you submit a new job without queue information and property `yarn.scheduler.fair.user-as-default-queue` is true.

  You can specify this property in `fair-scheduler.xml`:

```
<allocations>
  <defaultQueueLabel>LabelA</defaultQueueLabel>
  <queue name="root">
          ...
  <queue name="queue1">
  </queue>
  <queue name="queue2">
    <label>LabelB</label>
   </queue>
          ...
  </queue>
</allocations>
```

  See Specifying Fair Scheduler Configuration Properties in yarn-site.xml on page 1269 for more information.

- `yarn.scheduler.fair.resources-based-on-labels-enabled`

  Used to enable or disable recomputing of fair shares based on labels. It allows container allocation on all nodes.

  You can specify this property in `yarn-site.xml`:

  ```
  <property>
    <name>yarn.scheduler.fair.resources-based-on-labels-enabled</name>
    <value>false</value>
  </property>
  ```

- `yarn.scheduler.fair.preemption.cluster-utilization-threshold.based-on-labels-enabled`

  Allows enabling or disabling (default) preemption of the threshold per-label. To overcome the default and start container preemption when the threshold of the label is exceeded, change this property to `true`.

  You can specify this property in `yarn-site.xml`:

  ```
  <property>

  <name>yarn.scheduler.fair.preemption.cluster-utilization-threshold.based-on-labels-enabled</name>
    <value>false</value>
  </property>
  ```

See Specifying Fair Scheduler Configuration Properties in yarn-site.xml on page 1269 for more information.

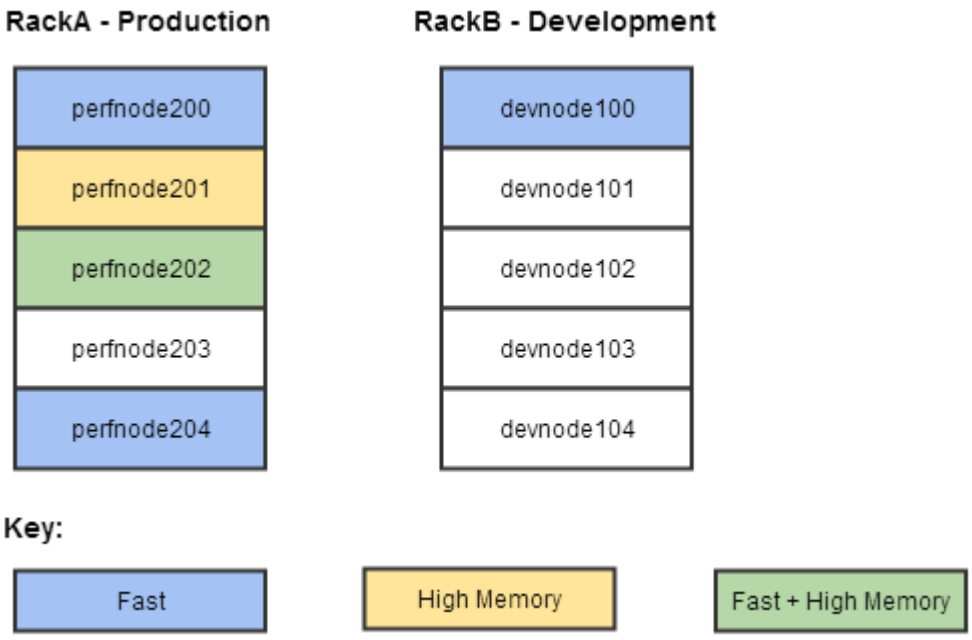
LBS Requirements and Restrictions?

Before you adopt LBS, you should be aware of two issues:

- Multiple labels for a node and label expression at the queue level are not supported.
- Except for `root`, all queues including the default, must be labeled, either independent or through inheritance.

Sample Cluster Configuration

To illustrate the concept of label-based scheduling, consider a cluster with two racks: RackA and RackB. The nodes in RackA are dedicated to the Production group, and the nodes in RackB are dedicated to the Development group. In addition, some nodes are configured with a fast CPU, one node is configured with high memory, and one node is configured with both a fast CPU and high memory. The following diagram illustrates the cluster configuration:



Creating a Node Labels File

The node labels file is a text file stored in MapR filesystem that maps labels to nodes. A node label applies a name to a cluster node, to identify it for the purpose of specifying where to run MapReduce YARN applications (MRv2).

Syntax for Node Labels File

Each line in the node labels file consists of an identifier (a regular expression that identifies one or more nodes), whitespace to separate the identifier from the labels, then one or more labels (separated by commas, whitespace, or both) to apply to the specified nodes. If a label contains two or more words (such as "High Memory"), enclose the name in single or double quotation marks so the whitespace will not be interpreted as a delimiter between two labels.

```
<identifier> <label1>[,<label2>,...,<labeln>]
```

Warning:

Labels must *not* start with a digit.

The identifier specifies nodes by matching the node names or IP addresses in one of two ways:

- Unix-style glob which supports the ? and * wildcards
- Java regular expressions (refer to http://docs.oracle.com/javase/6/docs/api/java/util/regex/Pattern.html for more information)

Warning:

The identifier *must* match the fully qualified domain name (FQDN). To determine the FQDN, run the command `hostname --fqdn`.

Sample Node Labels File

last-updated: Apr 11, 2022

The following node labels file is written for the sample cluster configuration and uses glob identifiers. Note that the FQDNs include the realm `company.com`.

```
perfnode20*.company.com RackA
perfnode200.company.com Fast
perfnode201.company.com 'High Memory'
perfnode202.company.com Fast
perfnode204.company.com Fast

devnode10*.company.com RackB
devnode100.company.com Fast
```

Using Node Labels to Schedule YARN Applications

To set up node labels for the purpose of scheduling YARN applications (including MapReduce applications) on a specific node or group of nodes:

1. Create a text file and specify the labels you want to use for the nodes in your cluster. In this example, the file is named `node.labels`.
2. Copy the file to a location on MapR filesystem where it will not be modified or deleted, such as `/var/mapr`.

   ```
   hadoop fs -put ~/node.labels /var/mapr
   ```

3. Edit `yarn-site.xml` on all ResourceManager nodes and set the `node.labels.file` parameter and the optional `node.labels.monitor.interval` parameter as shown:

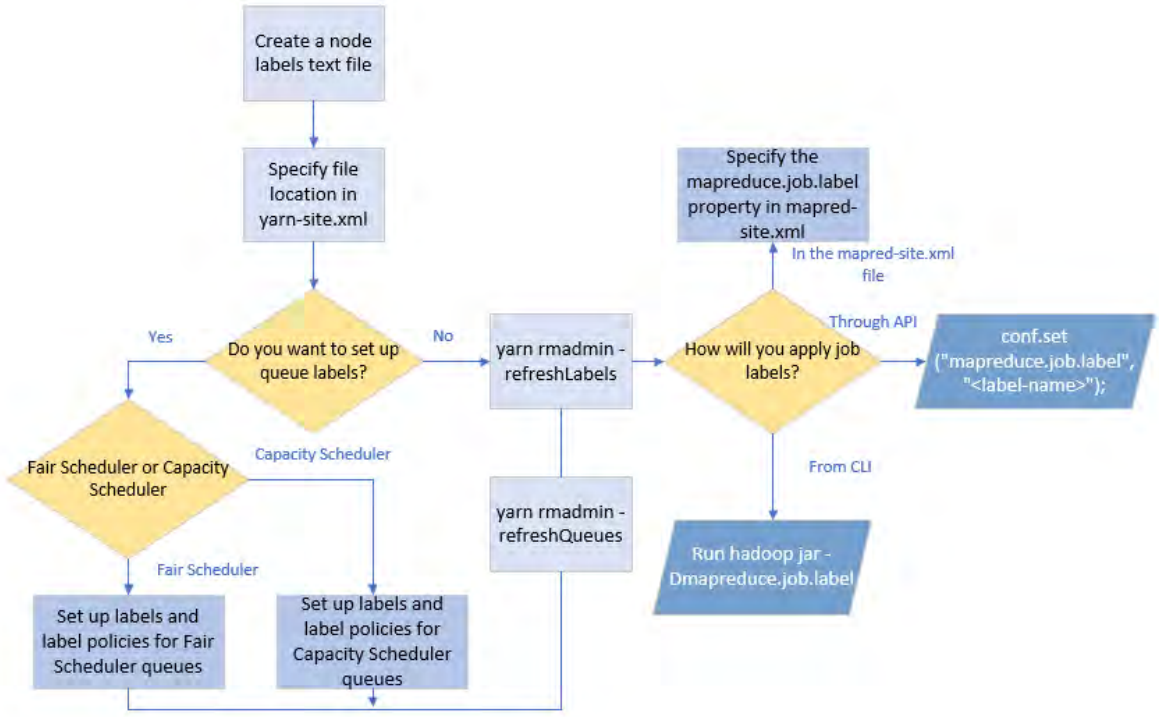
   ```
   <property>
      <name>node.labels.file</name>
      <value>/var/mapr/node.labels</value>
      <description>The path to the node labels file.</description>
   </property>

   <property>
      <name>node.labels.monitor.interval</name>
      <value>120000</value>
      <description>Interval for checking the labels file for updates
   (default is 120000 ms)</description>
   </property>
   ```

4. For this and subsequent changes to take effect, issue either of the following commands to manually tell the ResourceManager to reload the node labels file:
   - For any YARN applications, including MapReduce applications, enter `yarn rmadmin -refreshLabels`
   - For MapReduce applications, enter `mapred job -refreshLabels`
5. Verify that labels are implemented correctly by running either of the following commands:

   ```
   yarn rmadmin -showLabels
   mapred job -showlabels
   ```

The following flowchart summarizes these steps. In addition, the flowchart introduces the concept of queue labels for the Fair Scheduler and the Capacity Scheduler.



Creating Queue Labels

Queue labels are optional with label-based scheduling. You can use queue labels to determine which nodes an application or job can run on (subject to the queue label policy). A queue label is created from node labels as explained below.

Defining Queue Labels for Fair Scheduler

Explains how to customize the Fair Scheduler.

By default, all users share a single queue, named *default*.

To customize the Fair Scheduler, create an allocation file that lists existing queues and their respective weights and capacities, as explained in Hadoop 2.x Fair Scheduler. In the allocation file, add the following property within the `queue` section:

```
<label>labelname</label>
```

For example:

```
<queue name="Customer Data Analysis">
   <weight>2.0</weight>
   <label>Fast</label>
</queue>
```

For a hierarchical queue, note that labels and label policies can be defined on any level of the queue. If a child queue does not have its own labels or label policies, the labels and label policies of the closest level are used.

Defining Queue Labels for Capacity Scheduler

The Capacity Scheduler has a pre-defined queue called *root*. All queues in the system are children of the root queue. Further queues can be set up by configuring `yarn.scheduler.capacity.root.queues` with a list of comma-separated child queues.

If a parent queue has child queues, this is how labels and label policies are applied to the child queues:

last-updated: Apr 11, 2022

- If the child queues have their own labels or label policies, they are used.
- If the child queues do not have their own labels or label policies, the parent queue's labels and label policies apply.

To use label-based scheduling with the Capacity Scheduler, add the following property to the `capacity-scheduler.xml` file:

```
yarn.scheduler.capacity.root.<queue-name>.label
```

For example, for a queue named `alpha`, the label could be defined like this:

```
<property>
   <name>yarn.scheduler.capacity.root.alpha.label</name>
   <value>Fast||Development</value>
</property>
```

When you make changes to queue labels or queue policies, remember to refresh them by running the following command:

```
yarn rmadmin -refreshQueues
```

Defining a Queue Label Policy

A queue label policy determines whether an application label or a queue label prevails when there is a conflict between the two. For YARN applications, you can set the following queue label policies:

- PREFER_QUEUE — always use label set on queue
- PREFER_APP — always use label set on application
- AND (default) — application label AND queue label
- OR — application label OR queue label

See the following sections for directions on setting the queue label policy for Fair Scheduler and Capacity Scheduler.

Setting Queue Label Policies for Fair Scheduler

To set a queue label policy for a Fair Scheduler queue, specify the label policy in the corresponding `queue` section of the allocation file, as shown here:

```
<queue name="CustomerDataAnalysis">
      <weight>2.0</weight>
      <labelPolicy>OR</labelPolicy>
      <label>Fast</label>
</queue>
```

Setting Queue Label Policies for Capacity Scheduler

To set a queue label policy for a capacity queue, add the following property to the `capacity-scheduler.xml` file:

```
yarn.scheduler.capacity.root.<queue-name>.label-policy
```

For example:

```
<property>
  <name>yarn.scheduler.capacity.root.alpha.label-policy</name>
```

```
  <value>PREFER_APP</value>
</property>
```

Examples of Queue Policy Behavior

The following examples show the job placement policy behavior in various scenarios, based on the sample node labels file:

Application Label	Queue Label	Queue Policy	Outcome
Fast	High Memory	PREFER_APP	The job runs on nodes labeled **Fast** (hostnames match perfnode200, perfnode202, perfnode204, or devnode100)
Fast	High Memory	PREFER_QUEUE	The job runs on nodes labeled **High Memory** (hostnames match perfnode201 or perfnode202)
Fast	High Memory	AND	The job runs on nodes only if they are labeled both **Fast** *and* **High Memory** (hostname matches perfnode202)
Fast	High Memory	OR	The job runs on nodes if they are labeled either **Fast** *or* **High Memory** (hostnames match perfnode200, perfnode201, perfnode202, perfnode204, or devnode100)

Creating Job Labels

Explains the use of job labels to place jobs on particular nodes.

To place an individual job on a particular node, use a *job label*. You can apply job labels in three ways:

- Use set() from the Hadoop configuration API in your Java application. For example:

```
conf.set("mapreduce.job.label","Production");
```

- At the command line, pass the label in `-Dmapreduce.job.label` when you run the job with the hadoop jar command. For example:

```
hadoop jar /opt/mapr/hadoop/hadoop-2.4.1/share/hadoop/mapreduce/
hadoop-mapreduce-examples-2.4.1-mapr-4.0.1-20140804.191359-4.jar \
teragen -Dmapreduce.job.label=Production 100000000 /teragen
```

- Set the `mapreduce.job.label` parameter in `mapred-site.xml`. For example, to configure an application label expression for a MapReduce application that can run on any Development node or Fast node (including a Fast node from the Production rack), set the `mapreduce.job.label` parameter as shown:

```
<property>
   <name>mapreduce.job.label</name>
   <value>Development || Fast</value>
   <description>Label expression for MapReduce application</description>
</property>
```

Warning:

If an application is submitted with a label that does not correspond to any nodes, the application will run as though no label had been specified. Consult the ResourceManager log for more information (look for '`invalid label`' error messages).

Submitting Jobs and Applications to the Cluster

You can submit YARN applications (MapReduce version 2 and other applications that run on YARN) to the same cluster. An application can be submitted to the cluster in the following ways:

- The `hadoop jar` command submits an MRv2 application.
- The `yarn jar` command submits an application.
- An external application submits an application.
- An ecosystem component generates and submits an application.
- The `hadoop job` command submits an MRv2 application.
- The `mapred job` command submits an MRv2 application.

When you submit a non-MapReduce application to the cluster, such a Spark application, it is automatically processed using `yarn` mode (ResourceManager, NodeManager, and MapReduce ApplicationMaster).

Note: The method to submit Hadoop commands from a Windows or Mac client is different, For details, see Setting Up the Client.

Configuration Files for Jobs and Applications

Lists the locations of the MapReduce configuration files.

To override the MapReduce default configuration, use the following MapReduce configuration files:

MapReduce Version	Configuration File Locations
MapReduce Version 2	• `/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop/mapred-site.xml` • `/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop/yarn-site.xml`

To override the default configuration for applications, use the `yarn-site.xml` file that is present in `/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop`.

YARN Container Resources

Provides an overview of YARN.

last-updated: Apr 11, 2022

A YARN application can be a MapReduce version 2 (MRv2) application or a non-MapReduce application. The Warden on each node calculates the resources that can be allocated to process YARN applications. Each application has an ApplicationMaster that negotiates YARN container resources. For MapReduce applications, YARN processes each map or reduce task in a container. The ApplicationMaster requests resources from the ResourceManager based on memory, CPU, and disk requirements for the YARN containers. For YARN containers that process MRv2 tasks, there are additional considerations. See YARN Container Resources for MapReduce Version 2 Applications on page 1289 for details.

The ApplicationMaster requests YARN container resources based on the values of the following parameters:

yarn.scheduler.minimum-allocation-mb

Default: 1024

Description: Defines the minimum memory allocation available for a container in MB.

To change the value, edit the yarn-site.xml file for the node that runs the ResourceManager. Assign the new value to this property, then restart the ResourceManager.

yarn.scheduler.maximum-allocation-mb

Default: 8192

Description: Defines the maximum memory allocation available for a container in MB.

To change the value, edit the yarn-site.xml file for the node that runs the ResourceManager. Assign the new value to this property, then restart the ResourceManager.

yarn.nodemanager.resource.memory-mb

Default: Variable. This value is calculated by Warden.

Description: Defines the memory available to processing Yarn containers on the node in MB.

Warden uses the following formula to calculate this value: `[total physical memory on node] - [memory required by the operating system, MapR File System, and MapR services installed on the node]`.

To determine the value, go to the ResourceManager UI and view the memory available for that node.

yarn.nodemanager.resource.cpu-vcores

Default: Variable. This value is calculated by Warden.

Description: Defines the number of CPUs available to process YARN containers on this node.

Warden uses the following formula to calculate this value: `[Number of CPU cores on node] - [Number of CPU cores assigned to MapR File System]`.

To determine the value, go to the ResourceManager UI or the YARN pane on the Control System and view the number of CPUs available for that node.

To change the value, edit the yarn-site.xml file for the node, assign the new value to this property, then restart the NodeManager.

yarn.nodemanager.resource.io-spindles

Default: Variable. This value is calculated by Warden.

Description: Defines the number of disks available to process YARN containers. Warden uses the following formula to calculate this value: `[Number of of disks on the node]`.

last-updated: Apr 11, 2022

	To determine the value, go to the ResourceManager UI or the YARN pane on the Control System and view the disk information for this node.

YARN Container Resources for MapReduce Version 2 Applications

In addition to the YARN container resource allocation parameters, the MapReduce ApplicationMaster also considers the following container requirements when it sends requests to the ResourceManager for containers to run MapReduce applications:

Parameter	Default	Description
`mapreduce.map.memory.mb`	1024	Defines the container size for map tasks in MB.
`mapreduce.reduce.memory.mb`	3072	Defines the container size for reduce tasks in MB.
`mapreduce.reduce.java.opts`	-Xmx2560m	Java options for reduce tasks.
`mapreduce.map.java.opts`	-Xmx900m	Java options for map tasks.
`mapreduce.map.disk`	0.5	Defines the number of disks a map task requires. For example, a node with 4 disks can run 8 map tasks at a time. **Note:** If I/O intensive tasks do not run on the node, you may want to change this value.
`mapreduce.reduce.disk`	1.33	Defines the number of disks that a reduce task requires. For example, a node with 4 disks can run 3 reduce tasks at a time. **Note:** If I/O intensive tasks do not run on the node, you might want to change this value.

You can use one of the following methods to change the default configuration:

- Provide updated values in the `mapred-site.xml` file on the node that runs the job. You can use central configuration to change this value on each node that runs the NodeManager in the cluster. Then, restart NodeManager on each node in the cluster. The `mapred-site.xml` file for MapReduce ve applications is located in the following directory: `opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop`
- Override the default values from the command line for each application that requires a non-default value.

Monitoring the Cluster

This section describes how to monitor the health and performance of a MapR cluster.

Monitoring Using the Control System and the CLI

Describes the Overview page in the Control System, which displays information about the cluster.

The **Overview** displays a summary of information about the cluster in six panes including information on cluster health, activity, and usage.

Note: During installation using the MapR Installer, you can configure metrics and logging using settings on the **Monitoring** page of the MapR Installer user interface. The metrics collection infrastructure must be installed because the Control System relies on these metrics to provide graphs and charts in the **Overview** page. If the metrics collection infrastructure is not installed, you cannot visualize the metrics in the **Overview** page. If you want, you can install metrics collection or logging at any time by selecting the feature during an Incremental Install.

The following sections provide information about each pane.

Setting the Refresh Rate on the Control System
Explains how to configure how frequently you want to refresh the graphs and data in the panes.

To set the refresh rate:

1. Log in to the Control System and click **Settings** from the drop-down menu.
2. Enter the refresh rate in seconds for the following.
   - Data refresh rate
   - Metrics and charts refresh rate

   The default refresh rate is 30 seconds. The minimum refresh rate is:
   - 5 seconds for data.
   - 30 seconds for metrics and charts.
3. Set the User Session Inactivity Timer in minutes. The user is logged out from the Control System after the specified number of minutes if there is no activity on the Control System. The default value is 30 minutes.
4. Click **Save Changes** for the changes to take effect.

Customizing the List of Metric Charts and Columns on the Control System
Explains how to customize the list of metric charts and columns on the Control System.

You can customize the list of table metric charts and columns that you see on the Control System.

Adding and Removing Charts from the Charts View

1. Log in to the Control System and go to one of the following pages:
   - **Metrics** tab of the node details page.
   - **Metrics** tab in the **Data** > **Tables** page.
   - **Metrics** tab of the table information page.
   - **Metrics** tab of the secondary index details page.
2. Click to display the **Customize Active Charts** window.
   In the **Customize Active Charts** window, the:
   - **Available** list displays the charts available, but currently not displayed.
   - **Selected** list displays the charts currently displayed in the page.
3. Select the charts from the:
   a) Selected list of charts and click to remove selected charts.
   b) Available list of charts and click to move selection to **Selected** charts to display on page.

You can select up to six charts to display at a time on the page. For the list of charts that can be viewed, see Viewing Table Metrics in the Control System on page 1306.

4. (Optional) Click ^ and/or down v arrows to sort the order of charts.
5. Click **Save** to view the selected charts.

Adding and Removing Columns from the List View

1. Log in to the Control System and do one of the following:
   - Go to the **Metrics** tab of the node details page and select **Activity by Tables** from the drop-down menu to view the metrics for all table-related activities on the node.
   - Go to the **Metrics** tab of the table information page and select:
     - **Activity by Nodes** (default selection) from the drop-down menu to view the metrics for table activity across nodes.
     - **Activity by Indexes** from the drop-down menu to view metrics for all index-related activity on the table.
   - Go to the **Metrics** tab of the secondary index details page and select **Activity by Nodes** to view metrics for index-related activity across nodes.
2. Click  to switch to a list view.
3. Click  to display the **Customize Columns** window.
   In the **Customize Columns** window, the:
   - **Available** list displays the columns available, but currently not displayed.
   - **Selected** list displays the columns currently displayed in the page.
4. Select the columns from the:
   a) Selected list of columns and click < to remove selected columns from the view.
   b) Available list of columns and click > to move selection to **Selected** columns (for display).
   For the list of metrics that can be viewed in the columns, see Viewing Table Metrics in the Control System on page 1306
5. (Optional) Click ^ and/or down v arrows to sort the order of columns.
6. Click **Save** to view the selected columns.

Monitoring the Cluster

Explains how to view the cluster health, disk, memory, CPU utilization metrics, and alarms on the cluster using either the Control System or the CLI.

Monitoring Cluster Health Using the Control System

- Log in to the Control System and click **Overview**.
  The **Overview** page displays the following panes:
  - Node Health — the health of the nodes on the cluster, by service (default) or topology
  - Active Alarms — a summary of active alarms for the cluster
  - Cluster Utilization — CPU, memory, and disk space usage

last-updated: Apr 11, 2022

- Yarn — the number of running and queued applications, number of Node Managers, and percent of memory and CPU's used relative to the amount configured

Note: During installation using the MapR Installer, you can configure metrics and logging using settings on the **Monitoring** page of the MapR Installer user interface. The metrics collection infrastructure must be installed because the Control System relies on these metrics to provide graphs and charts in the panes. If the metrics collection infrastructure is not installed, you cannot visualize the metrics in the various panes. If you want, you can install metrics collection or logging by selecting the feature during an Incremental Install.

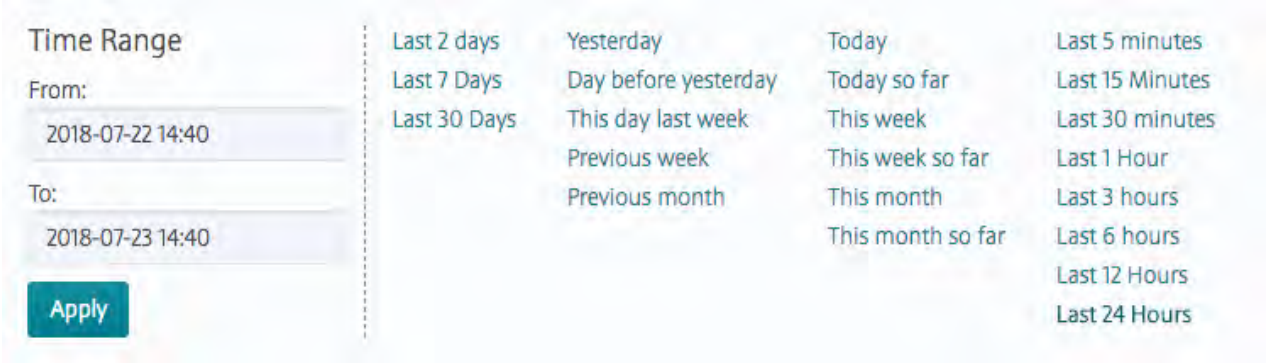
Viewing Cluster Utilization Information on the Control System

The **Cluster Utilization** pane in the **Overview** page displays the following for:

- CPU — Percentage of cores currently utilized and total cores
- Memory — Percentage of memory (in GB) currently utlized and total memory (in GB)
- Disk — Percentage of space (in GB) currently utilized and total disk space (in GB)

The **Cluster Utilization** pane also shows the amount of raw data and the savings (in percentage) after compression.

The **Utilization Trend** pane shows CPU, memory, and disk usage trend for the last 24 hours by default. You can select a preset (shown in the following screenshot) or specify a custom time range (shown in the following screenshot).



You can zoom in (by clicking and dragging the cursor in the pane) for a more granular view. Click **Reset Zoom** to zoom out and return to selected date/time range view. If there were any alarms during the selected date/time range, the **Alarms** pane above shows:

- When the alarm was raised
- The severity of the alarm
  - — an error
  - — a warning
  - — information

Monitoring Cluster Alarms on the Control System

See Viewing Active Cluster Alarms on page 1316 for more information.

Retrieving Cluster Information Using the CLI or REST API

The basic command to retrieve cluster health and disk space information is:

```
maprcli dashboard info -cluster <cluster>
```

The `utilization` field in the output shows the total and utilized amount of disk space, memory, and CPU for the cluster, which can also be visualized on the Control System. For example:

```
# /opt/mapr/bin/maprcli dashboard info -json
{
    "timestamp":1525230746268,
    "timeofday":"2018-05-01 08:12:26.268 GMT-0700 PM",
    "status":"OK",
    "total":1,
    "data":[
        {
            ...
            "utilization":{
                "cpu":{
                    "util":7,
                    "total":8,
                    "active":0
                },
                "memory":{
                    "total":15886,
                    "active":11281
                },
                "disk_space":{
                    "total":273,
                    "active":0
                },
                "compression":{
                    "compressed":0,
                    "uncompressed":0
                },
                "tiering":{
                    "logicalUsed":0,
                    "replicatedLogicalUsed":0,
                    "replicatedTotalUsed":0,
                    "ecTotalUsed":0,
                    "cvTotalUsed":0,
                    "offloaded":0,
                    "recalled":0
                }
            },
            ...
        }
    ]
}
```

For information on all the fields returned by this command, see `dashboard info` on page 1587.

Monitoring Nodes

Explains how to monitor nodes using either the Control System or the CLI.

You can check the health of the nodes on the cluster in the Control System, organized by service or by topology, or by using the CLI.

Note: The metrics collection infrastructure must be installed during installation to visualize the graphs and charts. If the metrics collection infrastructure is not installed, perform an Incremental Install to install the metrics collection infrastructure.

last-updated: Apr 11, 2022

Note: The **Nodes** page is not available on the Kubernetes version of the Control System.

Monitoring Node Health Using the Control System

To monitor the health of nodes:

1. Log in to the Control System and click:
   - **Overview** to view the health of the nodes in the **Node Health** pane.
   - **Nodes** to view the health of the nodes in the **Node Health** pane.
2. Select one of the following from the drop-down menu in the **Node Health** pane.
   - **By Service** to organize the display of nodes by services.

     This is the default view in the **Overview** page. This view contains the list of services and the nodes on which the service is running ( ) and is down ( ).

     Note: The color of the node (which reflects the status of the service) is even when a service is stopped (not running) on the node.
   - **By Topology** to view the display of nodes by topology.

     This is the default view in the **Nodes** page. This view contains the list of topologies and the health of the nodes (as shown in the following table) in the topology.

	Indicates the node is healthy.
	Indicates the node is degraded and/or may need attention. A node is considered to be in degraded state if: • There is no heartbeat from the MapR filesystem/NFS node for over 60 seconds. • One or more services are down on the node. • One or more alarms are raised on the node.
	Indicates the node is in maintenance mode.
	Indicates critical issue(s) on the node. A node is considered to be in critical state if: • There is no heartbeat from the node for more than 5 minutes. • All MapR files system disks on the node are dead or are offline. • All containers on the node are being re-replicated because either the node was removed, unregistered, or there was no heartbeat from the node for more than 1 hour. • File server is dead/inactive because there is no heartbeat for a long time. • NFS server on node is dead. • MapR install directory is full. • Node reported high MapR filesystem memory usage.

Monitoring Node Resource Utilization from the Control System

- Log in to the Control System and click **Nodes** to view the nodes that consumed the most CPU and memory (in percentage) in the **Current Resource Utilization** pane. The shade of the bubble indicates node resource utilization with the darker shade indicating the nodes that are nearing disk capacity.

last-updated: Apr 11, 2022

Monitoring Active Node Alarms from the Control System

See Viewing Active Node Alarms on page 1317 for more information.

Monitoring Node Health Using the CLI or REST API

You can check general health of the nodes with the following command:

```
maprcli node heatmap -cluster <cluster>
```

This command displays a heatmap for the nodes on the specified cluster; a subset of the output can also be visualized on the Control System. For complete reference information, see `node heatmap` on page 1702.

Viewing Node Metrics

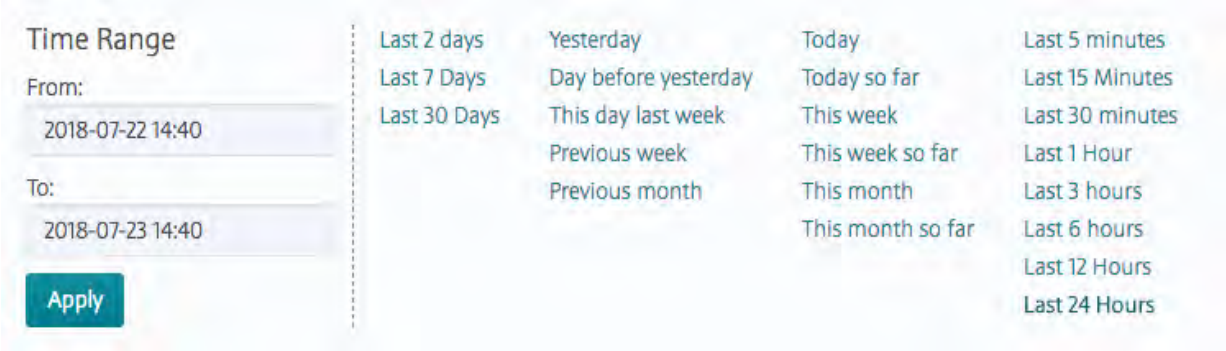
Explains how to view node metrics using the Control System.

Note: The metrics collection infrastructure must be installed during installation to visualize the metrics in the various panes. If the metrics collection infrastructure is not installed, perform an Incremental Install to visualize the metrics that are desribed in the following section.

Monitoring Node Metrics Using the Control System

- Log in to the Control System and go to the **Metrics** tab in the node information page.

  By default, the page displays charts that show metrics for the last 24 hours. You can select a preset or specify a custom time range.



  You can also zoom in (by clicking and dragging the cursor in the pane) for a more granular view. Click **Zoom Out** to expand time window or click:

  - to shift time window forwards.
  - to shift time window backwards.

  Click associated with the chart to view information about the graph. Click to display the **Customize Active Charts** window. You can select charts to display and remove from the **Available** and **Selected** lists in the **Customize Active Charts** window. You can view up to 6 charts at a time in the page.

  Use the following table when selecting the charts to view in the page. In the following table, the Charts column lists the charts that are available and the Metric column describes that type of metric that can be visualized in the chart:

Metric	Charts
CPU Usage	• Node Active CPU Usage • Node CPU Usage** • Node CPU Usage IDLE • Node CPU Usage NICE • Node CPU Usage SYSTEM • Node CPU Usage USER • Node CPU Usage WAIT • MFS CPU Usage • Allocated vs Available CPU Cores • MapR Process CPU Usage • MAST Gateway CPU Usage • DB Gateway CPU Usage • Data Access Gateway CPU Usage
Memory Usage	• Node Free Memory • Node Utilized Memory*** • Node Memory Free vs Used* • MFS Process Memory Usage • MapR Process Memory Usage
SWAP Space	• Node Swap Free • Node Swap Used • Node Swap Space Available vs Used* • Node Swap IO
Node IOs	• Node Network IO* • Node Network Interface Input • Node Network Interface Output • Node Network Interface Error Input • Node Network Interface Error Output
System Disk Throughput	• Disk Read Ops • Disk Write Ops • Disk Reads and Writes*

Metric	Charts
System Disk Latency	• Disk Avg Read Latency • Disk Avg Write Latency • Disk Read and Write Times
MFS Throughput	• MFS Read Throughput • MFS Write Throughput • MFS Read and Write Throughput • MFS System Disk Activity in Bytes*

* This metric is diplayed in the default chart view for a node.

** This metric is diplayed in the default chart view for a node and in the default list view for a table.

*** This metric is diplayed in the default list view for a table.

For information on viewing metrics for:

- All table activities on a node, see Viewing Per Node Metrics for Table Activities on page 1297.
- All stream activities on a node, see Monitoring Streams Operations Using the Control System on page 1313.

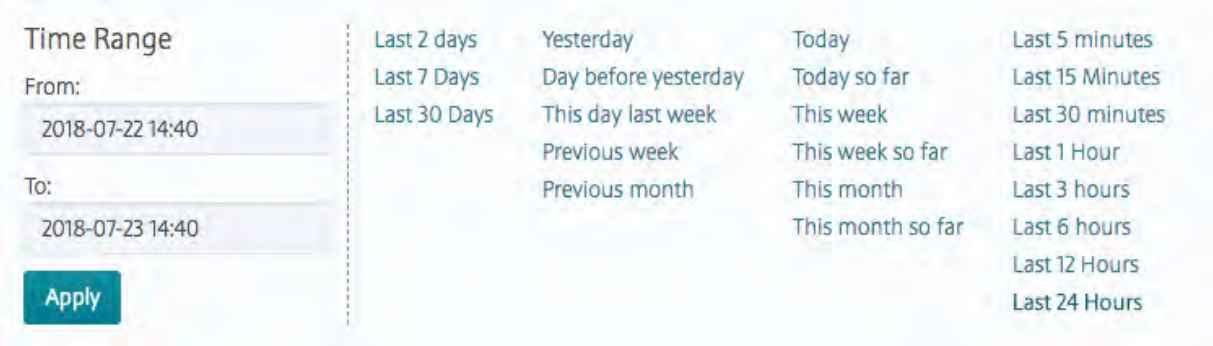
Viewing Per Node Metrics for Table Activities
Describes how to view per node metrics for table activities using the Control System and Grafana.

This section describes how to view the metrics as charts and lists in the Control System. For information on visualizing the metrics in the Grafana UI, see Metric Visualization on page 1381

Note: The metrics collection infrastructure must be installed to visualize the metrics in the various panes. If the metrics collection infrastructure is not installed, perform an Incremental Install to visualize the metrics as described below.

1. Log in to the Control System and go the **Metrics** tab in the node information page.
2. Select **Activity by Tables** from the drop-down menu.

   By default, the page displays charts that show metrics for the last 24 hours. You can select a preset or specify a custom time range.



   You can also zoom in (by clicking and dragging the cursor in the pane) for a more granular view. Click **Zoom Out** to expand time window or click:

   - › to shift time window forwards.

last-updated: Apr 11, 2022

- to shift time window backwards.

The charts show metrics for the node across tables. Click associated with the chart to view information about the graph. Click to display the **Customize Active Charts** window, where you can select charts to display and remove from the **Available** and **Selected** lists. You can view up to 6 charts at a time in the page.

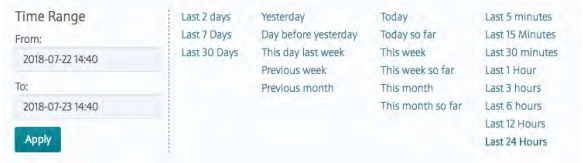
You can switch to the list view by clicking . In the list view, you can:

- Click the table name to go to the metrics page for the table.
- Select one or more tables and switch to charts view (by clicking) to visualize metrics for the selected tables only.

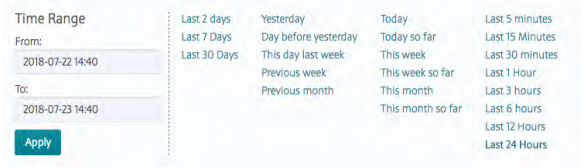
For the complete list of metrics that you can view at both the node and table levels, see Viewing Table Metrics in the Control System on page 1306.

Monitoring YARN

- Log in to the MapR Control System and:
  - Click **Overview** to view the following in the **YARN** pane:
    - Number of node managers, running applications, and queued jobs.
    - Memory and CPU utilization metrics.
  - Go to the service information page for YARN. The YARN information page displays the following panes:

Summary	Displays: • Total number of nodes and number of active and unhealthy nodes. • Allocated, pending, and reserved number of Resource Manager containers.
Top Queues by CPU Utilization	Displays the queues that utilize the most CPU (in percentage).
Top Queues by Memory Consumption	Displays the queues that consume the most memory (in percentage).
Applications	Displays the number of submitted, completed, running, pending, and failed applications during a selected date and time range. You can select a preset or specify a custom time range.  You can zoom in (by clicking and dragging the cursor in the pane) for a more granular view.
Resource Manager (CPU)	The number of cores allocated and used by Resource Manager during a selected date and time range. You can select a preset or specify a custom time range.

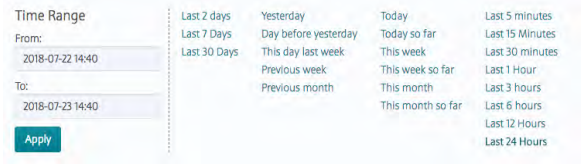
last-updated: Apr 11, 2022



You can zoom in (by clicking and dragging the cursor in the pane) for a more granular view.

Resource Manager (Memory)

The amount (in MB) of memory allocated and used by Resource Manager during a selected date and time range. You can select a preset or specify a custom time range.



You can zoom in (by clicking and dragging the cursor in the pane) for a more granular view.

Note: The metrics collection infrastructure must be installed during installation to visualize the metrics in the various panes. If the metrics collection infrastructure is not installed, perform an Incremental Install to install the metrics collection infrastructure.

Monitoring Volumes

Explains how to monitor volume parameters using the Control System.

You can monitor volume:

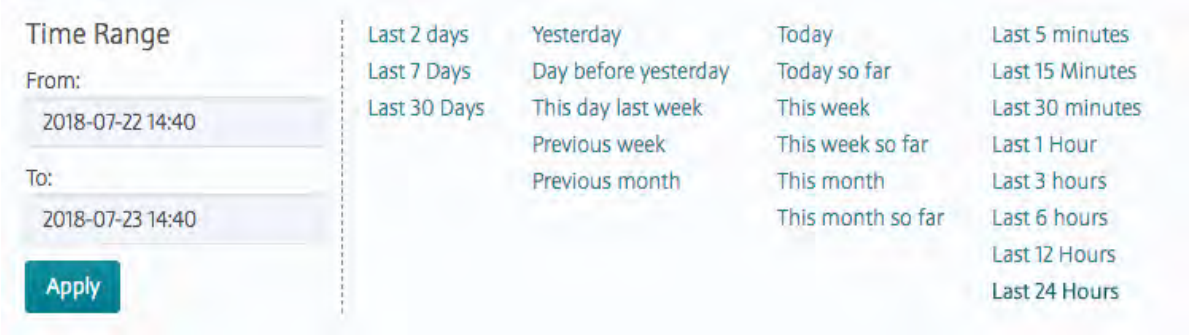
- Disk usage
- Alarms

Monitoring Volume Disk Usage Using the Control System

- You can view:
  - Volumes that use the most amount of allocated disk space in the **Top Volume Utilization** pane under **Data** > **Volumes**.

    Note: The **Volumes** page is under the **Volumes** menu on the Kubernetes version of the Control System.

  - Disk usage trend for a volume in the **Usage Trends** pane in the **Summary** tab of the volume information page. You can select a preset (shown in the following screenshot) or specify a custom time range (shown in the following screenshot).



You can zoom in (by clicking and dragging the cursor in the pane) for a more granular view. Click **Reset Zoom** to zoom out and return to selected date/time range view.

last-updated: Apr 11, 2022

Note: The metrics collection infrastructure must be installed during installation to visualize the metrics in the various panes. If the metrics collection infrastructure is not installed, perform an Incremental Install to view the charts described here.

Monitoring Volume Alarms in the Control System

See Viewing Active Volume Alarms on page 1317.

Monitoring Local and Remote Storage for Volumes

You can view the following storage utilization metrics for tiered volumes in the **Data** > **Volumes** > **Local & Remote Usage** pane.

Note: The **Volumes** page is under the **Volumes** menu on the Kubernetes version of the Control System.

- Local — The total disk space (in GB) used (before compression) for the volumes in the MapR cluster. This value does not include erasure-coded backend volumes and cache-volumes because their logical usage is already accounted for by the front-end and parent volumes respectively.
- Cold Offloaded — The total physical data (in GB) offloaded to the cold tier. This is calculated using the following: *amount of data purged from the hot tier (MapR cluster) + amount of data recalled to the hot tier (MapR cluster)*.
- Warm Offloaded — The total physical data (in GB) offloaded to the erasure coded volume (warm tier). This is calculated using the following: *amount of data purged from the hot tier (MapR cluster) + amount of data recalled to the hot tier (MapR cluster)*.

Collecting Volume Metrics

Describes how to enable and collect operational metrics for volumes.

You can collect the following volume metrics on MapR File System after you enable metrics collection as described in Enabling Volume Metric Collection on page 1301:

Read I/Os	Number of reads
Write I/Os	Number of writes
Read throughput	Amount of data read
Write throughput	Amount of data written
Read latency	Time taken by read operations
Write latency	Time taken by write operations

If you enable metrics collection on a volume, for each MapR File System instance, on every node where the volume containers reside, metrics for the volume (for a day) are captured every 10 seconds and logged to files in a local volume, which is two way replicated. The metrics log file, `Metrics.log-<date>-<n>.json`, is available at `/var/mapr/local/<hostname>/audit/<mfs-port>` directory. Here:

- `mfs-port` is the port on which the MapR File System instance listens.
- `date` is the record date in the format yyyy-mm-dd. A new file is created at the beginning of each day.
- n is the iteration of the log file represented by 3 digits. A new file is created every time Warden is restarted on the node. For the first file, <n> is 001 and <n> is incremented every time warden restarts. For example: `Metrics.log-2017-08-18-001.json` and `Metrics.log-2017-08-18-002.json` When a new file is created, the old file is purged based on the CLDB audit log retention period, which is 30 days by default.

Each record in the file looks similar to the following:

```
{
 "ts":1503048590000,
  "vid":35211529,
  "RDT":0.0,
  "RDL":0.0,
  "RDO":0.0,
  "WRT":308085.8,
  "WRL":2434.0,
  "WRO":2192.0
}
```

Here:

- `ts` — timestamp in milliseconds
- `vid` — volume ID
- `RDT` — read throughput in KB (cumulative for 10 seconds)
- `RDL` — amount of time taken by read operations (average for 10 seconds)
- `RDO` — number of read operations (cumulative for 10 seconds)
- `WRT` — write throughput in KB (cumulative for 10 seconds)
- `WRL` — amount of time taken by write operations (average for 10 seconds)
- `WRO` — number of write operations (cumulative for 10 seconds)

The `collectd` service reads up to 16 MB of data every ten seconds from each file, then aggregates and writes one minute worth of data to OpenTSDB. When reading the file, the `collectd` service stores offsets (as to how much has been read) as extended attributes (`trusted.disptachedOffset`) on the file. In addition to the default tags assigned to each metric when collectd writes metrics to OpenTSDB, the following tags are assigned to volume metrics:

- `mapr.volmetrics.[read_|write_][throughput|latency|ops]` — Displays the type of metric
- `volume_name` — Displays the name of the volume

For more information on the default tags, see Metric Collection on page 1334.

For each metrics file, MapR also creates a file, `Vollist_metrics.log-<date>-<n>`, in the `/var/mapr/local/<hostname>/audit/<mfs-port>/` directory. This file is purged based on the CLDB audit log retention period, which is 30 days by default. This file contains a comma separated list of volume name and volume ID (for volumes for which metrics are captured) and is associated with the `Metrics.log-<date><n>.json` file. For example, the record in the file looks similar to the following:

```
<volumeid>,<volume name>,<volumeid>,<volume name>,...
```

You can visualize the metrics in the dashboards on Grafana. See Metric Visualization on page 1381 for more information.

Enabling Volume Metric Collection
Describes how to enable metric collection for a volume, and gather volume statistics.

On all new installations, this feature is enabled by default and you only need to enable metrics collection for each volume. If you are upgrading from a version prior to MapR version 6.0, to start collecting volume

last-updated: Apr 11, 2022

metrics that you can then visualize in Grafana, you must enable the metrics collection feature and then enable metrics collection for each volume.

You can enable metrics collection from the command-line and using the Control System.

Enabling the Metric Collection Feature

- Enable the feature by running the following command:

```
/opt/mapr/bin/maprcli cluster feature enable -name
mfs.feature.metrics.support
```

Enabling Metric Collection for New Volumes Using the Control System

1. Log in to the Control System and click **Data** > **Volumes**.

   Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

2. Click **Create Volume** to display the **Create New Volume** page.
3. Enter values for the required settings and move the slider for **Collect Metrics** to **Yes** to enable metric collection for the volume.

   For more information, see Creating a Volume on page 864.
4. Click **Create Volume** to create the volume.

Enabling Metric Collection for Existing Volumes Using the Control System

1. Log in to the Control System and click **Data** > **Volumes**.

   Note: The **Volumes** page is under the **Volumes** menu in the Kubernetes version of the Control System.

2. Select the **Volumes** to edit and **Edit Volume** from the **Actions** drop-down menu or go the volume info page and click **Edit Volume**.
3. Move the slider for **Collect Metrics** to **Yes** to enable metric collection for the volume.

   For more information, see Creating a Volume on page 864.
4. Click **Save Changes** for the changes to take effect.

Enabling Metric Collection for Volumes Using the CLI

- Enable metrics collection on:
  - A new volume by running the following command:

    ```
    /opt/mapr/bin/maprcli volume create -name <volume-name> -path
    <volume-mount-path> -type <volume-type> -metricsenabled true
    ```

    For more information, see `volume create` on page 1931.
  - An existing volume by running the following command:

    ```
    /opt/mapr/bin/maprcli volume modify -name <volume-name> -metricsenabled
    true
    ```

    For more information, see `volume modify` on page 2005.

- All volumes that do not have mapr in the name by running the following script:

```
for i in `/opt/mapr/bin/maprcli volume list -columns n | grep -v
mapr | grep -v volumename`; do echo $i; /opt/mapr/bin/maprcli volume
modify -name $i -metricsenabled true; done
```

Monitoring Tables
Explains how to monitor table alarms and view table metrics in the Control System.
Monitoring Table Alarms in the Control System

See Viewing Active Table Replication Alarms on page 1318.

Viewing Throughput by Operation Type Using the Control System

- Log in to the Control System and do one of the following:
  - Go to the **Summary** tab in the table information page.

    The **Throughput - By Op Type** pane displays a graph for the following operations on the table in the last hour:
    - Gets
    - Puts
    - Scans
    - Increments
    - Appends
    - Checks and Puts
    - Updates and Gets
  - Go to the **Summary** tab in the index details page.

    The **Throughput - By Op Type** pane displays a graph for the following operations on the index in the last hour:
    - Puts
    - Scans

  You can move the cursor over the graph to view the number of operations on the table (across nodes).

Visualizing Table Metrics in the Control System

In the Control System, you can visualize node-level metrics for table operations on a node. You can view charts that show metrics for all tables across all nodes, metrics per table aggregated across nodes, metrics per table per node, and metrics per node across tables. In addition, you can view metrics for activity by indexes and for the table and its secondary indexes.

For the full list of metrics (and associated charts/columns) that you can view in the Control System, see Viewing Table Metrics in the Control System on page 1306.

- Log in to the Control System and do one of the following to view table metrics:
  - Go to the **Metrics** tab in the node information page and select **Activity by Tables** from the drop-down menu.

    In this tab, you can view charts/columns that show metrics for all table operations on the node, operations on streams, and other node metrics described here.

- Go to the **Metrics** tab in the **Data** > **Tables** page.

  In this tab, you can view charts/columns that show metrics per node for all tables.

- Go to the **Metrics** tab in the table information page.

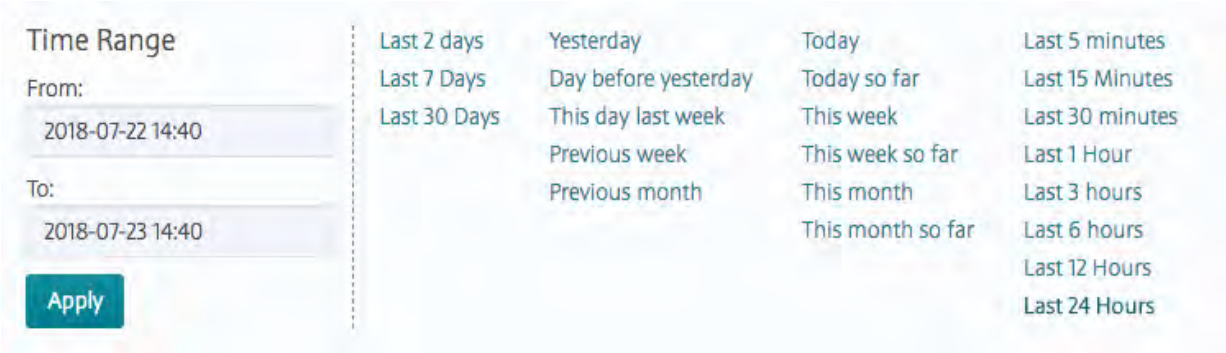
  In this tab, you can view charts/columns that show operations on the table and its secondary indexes across nodes.

- Go to the **Metrics** tab in the index details page and select **Activity by Nodes** from the drop-down menu.

  In this tab, you can view charts that show all index operations and index operations per node.

You can select the charts to view by clicking . See Adding and Removing Charts from the Charts View on page 1290 for more information.

The charts on the page show the metrics for the last 24 hours by default. You can select a preset or specify a custom time range.



You can also zoom in (by clicking and dragging the cursor in the pane) for a more granular view. Click **Zoom Out** to expand time window or click:

- to shift time window forwards.
- to shift time window backwards.

Note: When you select a granular view, the chart might not show the most accurate data because of the difference between the interval at which the metrics are logged and the downsampling (lowering the sampling rate) for the interval being viewed.

You can switch to a list view, where available, by clicking . In this view, you can select metrics to view by clicking . See Adding and Removing Columns from the List View on page 1291 for more information.

In the list view, you can:

- Filter the list by clicking .
- Click the node name to go to the node-level metrics page.
- Select one or more nodes and switch to charts view by clicking to visualize metrics for the selected nodes only. The legends for the charts reflect the selected nodes.

Note: Certain grouping of metrics, available in the charts view, are not available in the list view. All the metric grouping available in the list view are available in the charts view as well. When you switch from charts to list view and vice versa, your selection of metrics in one view is not carried over to the other view.

last-updated: Apr 11, 2022

The chart and list views allow you to detect and diagnose bottlenecks and performance issues on individual tables and nodes. You can use the charts for measuring the throughput and latency of different RPC operations on a table and for determining which operations on a table are slow or which tables are most frequently accessed.

For example:

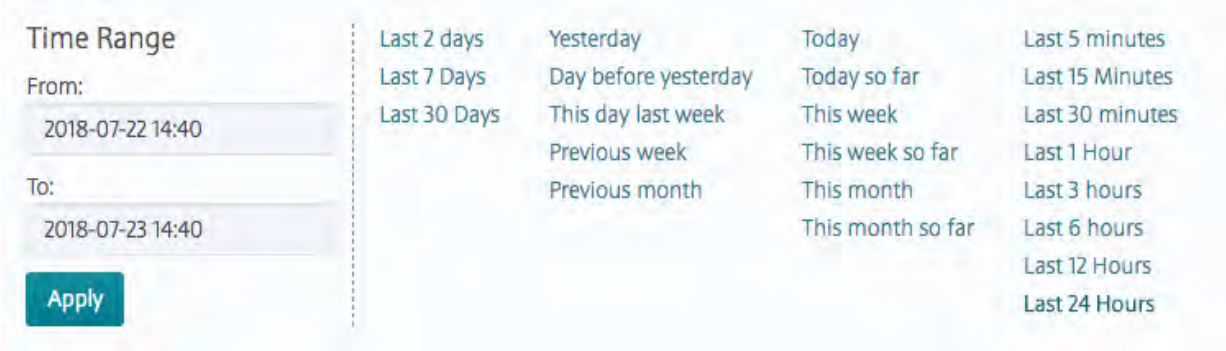
- Suppose your node is busy and you are noticing intermittent latency spikes on your table. You can compare throughput and latency in the **Metrics** tab of the table information page and investigate if the latency spike is due to node being very busy or node having high CPU utilization by switching to the list view from where you can navigate to the **Metrics** tab of the node details page.
- Suppose your index queries take minutes instead of seconds to complete. You can compare the get latency percentile with the scan read/response in the **Metrics** tab of the table information page by zooming in to the area where you see the spike. Switch to **Activity by Index** to view the index vs primary table scans where you can determine whether excessive scan load went to the primary table.
- Suppose you are noticing latency spikes on one of your table as a result of a lot of activity on another table. Observe the get latency percentile spikes in the **Metrics** tab of the table information page and switch to **Activity by Node** list view to identify the nodes with high overall aggregate table RPC load and node IOps. Select the saturated nodes in the list view and switch to the chart view. Go back to the list view and click the saturated node to navigate to the **Metrics** tab of the node details page. Switch to **Activity by Tables** to determine the most active table.

Viewing Secondary Index Metrics

You can visualize the secondary index metrics in the Control System.

- Log in to the Control System and do one of the following:
  - Go to the **Metrics** tab in the table information page and select **Activity by Indexes** from the drop-down menu to view all index-related activity on the table.
  - Go to the **Metrics** tab in the secondary index page to view metrics for index-related activity across tables or per node.

  By default, the page displays metrics for the last 24 hours. You can select a preset or specify a custom time range.



  You can also zoom in (by clicking and dragging the cursor in the pane) for a more granular view. Click **Zoom Out** to expand time window or click:

  - > to shift time window forwards.
  - < to shift time window backwards.

last-updated: Apr 11, 2022

The page displays charts by default. When viewing activity by nodes, you can switch to a list view by clicking and return to charts view by clicking .

Charts view	You can select the charts to view by clicking . See Adding and Removing Charts from the Charts View on page 1290 for more information. Click the associated with the chart to view information about the graph.
List view	The list view shows the metrics in the columns. You can customize the columns by clicking . See Adding and Removing Columns from the List View on page 1291 for more information. In addition, you can: • Click the column name to sort the table by that column. • Click the node name to go to the metrics page for the node. • Click one or more checkboxes next to the node name and switch to the charts view to visualize metrics for the secondary index activities on those nodes only.

You can use the charts to diagnose and troubleshoot bottlenecks and performance issues. For the complete list of metrics that you can view for secondary indexes, see Viewing Table Metrics in the Control System on page 1306.

Viewing Region Distribution

- Log in to the Control System and go to one of the following pages:
  - **Summary** tab in the table information page to view the region distribution for a table.
  - **Summary** tab in the index details page to view the region distribution for a secondary index.

  The **Region distribution** pane shows the distribution of the table or secondary index regions across the nodes in the cluster. The shade of the node reflects the sum of the physical size of data on the node with the darker shade indicating increased resource utilization on the node. You can move the cursor over a node to view the following:
  - Hostname of the node
  - Number of regions on the node
  - Total size of data (across regions) on the node

  You can click a node to go to the node information page.

Viewing Table Metrics in the Control System

Explains how to view primary and secondary table index metrics using the Control System.

A subset of the following primary table and secondary index metrics are available as charts and lists in the Control System. For information on how to:

- View these metrics in the Control System, see Monitoring Tables on page 1303.
- Customize the charts you see on the page, see Adding and Removing Charts from the Charts View on page 1290.

last-updated: Apr 11, 2022

- Customize columns you see on the page, see Adding and Removing Columns from the List View on page 1291.

Chart/Column Name	Metric	Description
Table Bytes Read Per Node	Throughput - bytes read	The number of bytes read from the primary table per node for all RPC types.
Table and Index Bytes Read		The number of bytes read across a primary table and its secondary indexes for all RPC types.
Table Bytes Written Per Node	Throughput - bytes written	The number of bytes written to the primary table per node for all RPC types.
Table and Index Bytes Written		The number of bytes written across a primary table and its secondary indexes for all RPC types.
Table Rows Read Per Node	Throughput - rowCount read	The number of rows read from the primary table per node for all RPC types.
Table and Index Rows Read		The number of rows read across a primary table and its secondary indexes for all RPC types. This is displayed in the default list view for a node.
Table Rows Written Per Node	Throughput - rowCount written	The number of rows written to the primary table per node for all RPC types.
Table and Index Rows Written		The number of rows written across a primary table and its secondary indexes for all RPC types.
All Tables Written Rows Throughput		Number of rows written by RPC operation type.
Table Rows Responded Per Node	Throughput - rowCount returned	The number of rows returned from the primary table per node for all RPC types.
Table and Index Rows Responded		The number of rows returned across a primary table and its secondary indexes for all RPC types. This is displayed in the default list view for a node.
Scan Throughput		Compares the scan throughput for rows read versus rows returned. This is displayed in the default chart view for a node and for a table.

last-updated: Apr 11, 2022

Chart/Column Name	Metric	Description
All Tables Throughput	Throughput - rpcCount/second	Number of RPC operations by type for all tables in the cluster. This is displayed in the default chart view for all the tables.
Throughput by Rpc Type		The combined RPC load for the primary table and its indexes.
All Tables RPC Byte Throughput		The number of bytes processed by RPC operation type.
All Tables Read Rows Throughput		Number of rows processed by RPC operation type.
All Tables Returned Rows Throughput		Number of rows returned by RPC operation type.
Put and Append Operation Throughput		Number of put and append RPC operations for the table, including its indexes.
Table Check and Put Ops Per Node		The number of check and put operations completed for a primary table and for a node.
Table Update and Get Ops Per Node		The number of update and get operations completed for a primary table and for a node.
Table Get and Index Scans		The number of get and index scans completed for a primary table and for a node.
Table Write and Index Maintenance Activity		The number of table writes (puts, appends, increments, check and puts, update and gets) that require puts to the index.
Table Get Ops Per Node		The number of get operations completed for a primary table and for a node. This is displayed in the default list view for a node and for a table.
Table Get Throughput Per Node		The number of get operations completed per second for a table, excluding its secondary indexes, per node. This is displayed in the default chart view for a node and for a table.
Table Increment Ops Per Node		The number of increment operations completed for a primary table and for a node.
Table Put Ops Per Node		The number of put operations completed for a primary table and for a node. This is displayed in the default list view for a node and for a table.
Table Scan Ops Per Node		The number of scan operations completed for a primary table and for a node.
Table Append Ops Per Node		The number of append operations completed for a primary table and for a node.
Table Write Throughput Per Node		The number of put and append

Chart/Column Name	Metric	Description
Table and Index Scan Latency	Latency	The 99th percentile latency of all scan operations across the primary table and its secondary indexes. A bad ratio between rows read and responded with high scan latency may indicate a poorly configured index.
Table and Index Scan Latency Per node		The 99th percentile latency of scan operations completed across the primary table and secondary index per node. Large scans may hit the disks and result in poor performance, or a degrading disk may spike the latency.
Table Append Latency Per Node		The 99th percentile latency of append operations on the primary table per node.
Table Increment Latency Per Node		The 99th percentile latency of increment operations on the primary table per node.
Table Put Latency Per Node		The 99th percentile latency of put operations on the primary table per node. This is displayed in the default list view for a node and for a table.
Table Get Latency Per Node		The 99th percentile latency of get operations on the primary table per node. This is displayed in the default list view for a node and for a table.
Table Scan Latency Per Node		The 99th percentile latency of scan operations on the primary table per node.
Table Get Latency Percentiles*		The get operation latency by percentile for the primary table and its secondary indexes.
Primary Table Put & Append Latency Percentiles*		The pure write operation latency by percentile for the primary table and its secondary indexes.
Table Write Throughput Latency Percentiles		The 99th percentile latency of put operations on the primary table per node. This is displayed in the default chart view for a node and for a table.
Table Get Throughput Latency Percentiles		The 99th percentile latency of get operations on the primary table per node. This is displayed in the default chart view for a node and for a table.
Table Check and Put Latency Per Node		The 99th percentile latency of check and put operations on the primary table per node.
Table Update and Get Latency Per Node		The 99th percentile latency of update and get operations on the primary table per node.
Index Put Latency		The 99th percentile latency of put operations per secondary index of the

Chart/Column Name	Metric	Description
All Tables Replication Sent Bytes	Replication - rows/bytes sent	The number of bytes of replication data sent.
All Tables Replication Pending Bytes	Replication - rows/bytes pending	The number of bytes of replication data not yet sent.
All Tables Replication Activity	Replication - rows/bytes sent/pending	Number of bytes of replication data sent vs not yet sent. This is displayed in the default chart view for all the tables.
Index Throughput by RPC Type	Index - Throughput	The combined RPC load for the primary table and its secondary indexes.
Index Put Ops		The number of put operations completed per secondary index of the primary table.
Index Put Ops Per Node		The number of put operations completed per secondary index of the primary table per node.
Index Scan Ops		The number of scan operations completed per secondary index of the primary table.
Index Scan Ops Per Node		The number of scan operations completed per secondary index of the primary table per node.
Index Bytes Read	Index - rows/bytes read	The number of bytes read per secondary index of the primary table for all RPC types.
Index Bytes Read Per Node		The number of bytes read per secondary index of the primary table per node for all RPC types.
Index Bytes Written Per Node		The number of bytes written per secondary index of the primary table per node for all RPC types.
Index Rows Read		The number of rows read per secondary index of the primary table for all RPC types.
Index Rows Read Per Node		The number of rows read per secondary index of the primary table per node for all RPC types.
Index Rows Responded	Index - rows/bytes returned	The number of rows returned per secondary index of the primary table for all RPC types.
Index Rows Responded Per Node		The number of rows returned per secondary index of the primary table per node for all RPC types.
Index Scan Read vs Returned Rows	Index - rows/bytes read	The number of secondary index rows that were read versus returned.

last-updated: Apr 11, 2022

Chart/Column Name	Metric	Description
Index Bytes Written	Index - rows/bytes write	The number of bytes written per secondary index of the primary table for all RPC types.
Index Rows Written		The number of rows written per secondary index of the primary table for all RPC types.
Index Rows Written Per Node		The number of rows written per secondary index of the primary table per node for all RPC types.
All Tables Index Sent Bytes	Index - rows/bytes sent	The number of bytes sent for secondary index updates.
All Tables Index Pending Bytes	Index - rows/bytes pending	The number of bytes of secondary index data remaining to be sent.
All Index Maintenance Activity		Number of bytes of index data sent vs not yet sent. This is displayed in the default chart view for all the tables.
All Tables CDC Sent Bytes	CDC - rows/bytes sent	The number of bytes of CDC data sent.
All Tables CDC Pending Bytes	CDC - rows/bytes pending	The number of bytes of CDC data per node not yet sent.
All Tables CDC Propagation Activity		The number of bytes of CDC data sent vs not yet sent. This is displayed in the default chart view for all the tables.
All Streams Producer Ops	Streams Throughput, RPCs	The number of Streams producer RPCs.
All Streams Consumer Ops		The number of Streams consumer RPCs.
All Streams Producer Messages	Streams Throughput, messages	The number of Streams messages produced.
All Streams Consumer Messages		The number of Streams messages read by consumers.
Table Value Cache All Lookups	Value Cache Lookups	All operations for a primary table and for a node that performed a cache lookup.
Table Value Cache Lookups		The number of get operations for a primary table and for a node that performed a cache lookup.
Table Value Cache Lookups Per Index		The number of get operations for a primary table and for a node that performed a cache lookup.

last-updated: Apr 11, 2022

Chart/Column Name	Metric	Description
Table and Index Value Cache All Lookups	Value Cache Hits	All operations across the primary table and its secondary indexes that performed a cache lookup.
Table and Index Value Cache Lookups Per Index		The number of get operations across the primary table and its secondary indexes that performed a cache lookup per secondary index.
Table and Index Value Cache Lookups		The number of get operations across the primary table and its secondary indexes that performed a cache lookup.
Table Value Cache All Hits		All operations for a primary table and for a node that resulted in a cache hit.
Table Value Cache Hits		The number of get operations for a primary table and for a node that resulted in a cache hit.
Table Value Cache Hits Per Index		The number of get operations for a primary table and for a node that resulted in a cache hit.
Table and Index Value Cache All Hits		All operations across the primary table and its secondary indexes that resulted in a cache hit.
Table and Index Value Cache Hits		The number of get operations across the primary table and its secondary indexes that resulted in a cache hit.
Table and Index Value Cache Hits Per Index		The number of get operations across the primary table and its secondary indexes that resulted in a cache hit per secondary index.
Value Cache Utilization	Value Cache	Compares the relative distribution of get operations that either hit the cache or require a lookup. This is displayed in the default chart view for a node and for a table.
All Tables Flushes	Bucket Flushes	The number of table flushes that were manually and automatically triggered. Table flushes reorganize data from bucket files (unsorted data) to spill files (sorted data) when the bucket size exceeds a threshold. This is displayed in the default chart view for all the tables.
All Tables Flushes		The number of total table flushes that were manually versus automatically triggered.
All Tables Force Flushes	Bucket Force Flushes	The number of table flushes that were not automatically triggered.

last-updated: Apr 11, 2022

Chart/Column Name	Metric	Description
All Tables Compactions	Compaction	Number of table compactions. Compactions combine multiple MapR Database data files containing sorted data (known as spills) into a single spill file. This is displayed in the default chart view for all the tables.
All Tables Full Compactions		Number of full compactions. Compactions combine multiple MapR Database data files containing sorted data (known as spills) into a single spill file. Full compactions improve read performance because after compaction, MapR Database needs to read only the single resulting sorted spill file. But they incur I/O costs because the compaction must read, sort, and rewrite all data in the spill files.
All Tables Mini Compactions		Number of partial compactions. Compactions combine multiple MapR Database data files containing sorted data (known as spills) into a single spill file. After a mini compaction, MapR Database needs to read only two spill files.
All Tables TTL Compactions		Number of compactions that result in reclamation of disk space after removal of stale data. You can control the frequency of TTL compactions by configuring the TTL for a table's column families.
All Tables Free Index Memory	Memindex Usage	The number of available MB in the in-memory bucket file cache.

* Percentiles are estimated by linearly interpolating between fixed buckets sizes.

Monitoring Streams
Explains how to monitor streams using either the Control System or the CLI.

The speed at which messages flow from producers to partitions, and from partitions to consumers depends on the performance of your producers, the cluster nodes hosting partitions, and your consumers.

Monitoring Streams Operations Using the Control System

- Log in to the Control System and go to the **Metrics** tab in the node information page to select the charts that show the following when you filter the list of charts by table activities:
  - **All Streams Producer Messages**: The number of Streams messages produced on the node
  - **All Streams Consumer Messages**: The number of Streams messages on the node read by consumers
  - **All Streams Producer Ops**: The number of Streams producer RPCs on the node
  - **All Streams Consumer Ops**: The number of Streams consumer RPCs on the node

  For more information, see Adding and Removing Charts from the Charts View on page 1290.

last-updated: Apr 11, 2022

Monitoring Active Stream Alarms

See Viewing Active Stream Alarms on page 1318.

Monitoring Cluster Nodes that Host Partitions

You can find out which nodes in a MapR cluster are being used for topics in a stream by running the command `maprcli stream topic info`. The nodes are listed in the `servers` field.

The `guts` utility can show you whether there are any I/O bottlenecks on these nodes. This utility can also show you whether there is any capacity on other nodes in the cluster that you can take advantage of by creating additional partitions for topics.

To run this utility, issue this command after logging into the MapR cluster that you want statistics for:

```
/opt/mapr/bin/guts
```

You can also use the `guts` utility to show only these statistics from MapR Event Store For Apache Kafka:

Table

Name	Description
`mpr`	The number of RPCs from MapR Event Store For Apache Kafka producers to the server.
`mpm`	The number of messages that have been published to the server.
`mpMB`	The total size in MB of the messages that have been published to the server.
`mlr`	The number of RPCs from MapR Event Store For Apache Kafka consumers to the server.
`mlm`	The number of messages that have been read from the server.
`mcl`	The number of concurrent RPCs from consumers to the server.
`mlMB`	The total size in MB of the messages that have been read from the server.

To see these statistics, run this command:

```
/opt/mapr/bin/guts streams:all
```

Note: These statistics are for the most recent sample period at the time the command is run, and are not cumulative. Sample periods are one second.

Monitoring Producers

To get a sense of how quickly producers are sending messages to the producer client library, you can run the command `stream topic info` at intervals.

Doing so will show you changes over time in the rate at which the values for `maxoffset` and `maxtimestamp` increase for the partitions in the topics that your producers are publishing to.

For example, if you have a script that runs the command at intervals of 10 seconds, the change per second would be `(Value at first run - Value at second run)/10`.

This is an indirect measure of the speed of the producers because the producer client library batches messages before publishing them to partitions. The faster the producers send messages to the client, the faster the client publishes message batches, and the greater the change per second.

last-updated: Apr 11, 2022

If producers do not seem to be sending messages quickly enough, and this problem is not caused by server-side I/O bottlenecks, you can spawn more producer threads.

See `stream topic info` for the syntax of this command.

Monitoring Consumers
There are two commands that you can run at intervals to get a sense of how far behind a consumer is in a partition. The consumer must belong to a consumer group, even if the consumer is the only member of that group.

To find the lag in milliseconds between the timestamp of the most recently published message in a stream, topic, or partition and the timestamp of a consumer's most recently committed cursor, run the command `stream cursor list`. The lag is the value of the `consumerlagmillis` parameter.

To find the timestamp of the most recently committed cursor for the consumer that is furthest back in a partition compared to all other consumers reading from the same partition, run the command `stream topic info`. This timestamp is the value of the `mintimestampacrossconsumers` parameter. Use this timestamp together with the values of the following parameters to get a sense of where this cursor is in the partition:

`mintimestamp`	This parameter shows the timestamp of the oldest message in the partition.
`maxtimestamp`	This parameter shows the timestamp of the most recently published message in the partition.

If a consumer's configuration for cursor commits is the default (the configuration parameter `enable.auto.commit` is set to `true` and `auto.commit.interval.ms` is set to 1000 milliseconds), the consumer will be only about one second ahead of the offset and timestamp reported for the consumer's most recently committed cursor.

If it seems that consumers are falling behind and that this problem is not caused by server-side I/O bottlenecks, you can start more consumer threads.

If the current number of consumers in a consumer group is equal to the number of partitions in the topic with the fewest partitions to which the consumer group is subscribed, add a partition to this topic before adding a consumer. The consumer client library dynamically reassigns the existing partitions in the topic to the consumers in the consumer group, as well as assigning the new partition to a consumer.

If the current number of consumers in a consumer group is less than the number of partitions in the topic with the fewest partitions to which the consumer group is subscribed, you don't need to add any partitions before adding a consumer.

Monitoring Alarms
Provides an overview of how to monitor alarms using the Control Panel and the CLI.

On a cluster with an Enterprise Edition or Enterprise Database Edition license, MapR raises alarms to alert you to information about the cluster:

- Cluster health, including disk failures
- Volumes that are under-replicated or over quota
- Services not running

You can see any currently raised alarms in the **Active Alarms** pane and the **Alarm Summary** page of the Control System (click  in the Control System) or using the `alarm list on page 1545` command. For a list of all alarms, see the Alarms Reference.

When you click  , the **Alarm Summary** page displays all active alarms and for each alarm, the **All** active alarms pane displays the following:

last-updated: Apr 11, 2022

Column Name	Column Description
Severity	The severity of the alarm. Value can be: • — Critical • — Warning • — Information
Status	The status of the alarm. Value can be: • Active • Muted
Time	The date and time stamp from when the alarm was raised.
Name	The name of the alarm.
Info	The notes associated with the alarm, which contains description of the alarm and the recommended action to take.
Entity	The entity on which the alarm was raised.
Type	The type of alarm. Value can be: • CLUSTER • NODE • VOLUME • USER/GROUP

You can select the checkbox beside one or more alarms to:

- Dismiss the alarms
- Mute the alarms

Viewing Active Cluster Alarms
Describes how to view cluster alarms using the Control System and the CLI.

You can view cluster alarms using the Control System and the CLI.

Viewing Active Cluster Alarms in the Control System

- Log in to the Control System and click:
  - **Overview** to view all the alarms on the cluster in the **Active Alarms** pane. To view only the cluster alarms, select **Cluster Alarm** from the drop-down menu in the **Active Alarms** pane.
  - (Alarm Summary) to view all the alarms on the cluster in the **All** alarms pane. To view only the cluster alarms, select **Cluster Alarm** from the drop-down menu in the **All** alarms pane.

  You can:
  - View alarm information
  - Dismiss an alarm

last-updated: Apr 11, 2022

- Mute an alarm

Retrieving Cluster Alarms Using the CLI or REST API

The basic command to retrieve the alarms for a cluster is:

```
maprcli alarm list -cluster <cluster name> -type cluster
```

For complete reference information, see `alarm list` on page 1545.

Viewing Active Node Alarms
Describes how to view active node alarms using the Control System and the CLI.
Viewing Active Node Alarms in the Control System

- **Note:** The **Nodes** page is not available in the Kubernetes version of the Control System.

  Log in to the Control System and:

  - Click **Nodes** to view all the node alarms on the cluster in the **Active Alarms** pane.
  - Go to the node information page to view alarms in the **Alarms** pane for the selected node.
  - Click (in the top navigation bar) to display the **Alarm Summary** page and select **Node Alarms** from the drop-down menu in the **All** alarms pane.
  - Click **Overview** and select **Node Alarms** from the drop-down menu in the **Active Alarms** pane to view all the node alarms on the cluster.

  You can:

  - View alarm notes
  - Mute an alarm
  - Dismiss an alarm

Retrieving Active Node Alarms Using the CLI or REST API

The basic command to retrieve node alarms is:

```
maprcli alarm list -cluster <cluster name> -type node
```

For complete reference information, see `alarm list` on page 1545.

Viewing Active Volume Alarms
Describes how to view volume alarms using the Control System and the CLI.

You can view volume alarms in the Control System and using the CLI.

Viewing Active Volume Alarms in the Control System

- Log in to the Control System and:
  - Click **Data** > **Volumes** to view all active volume alarms in the **Active Alarms** pane.

    Note: The **Volumes** page is under the **Volumes** menu on the Kubernetes version of the Control System.

  - Go to the **Summary** tab in the volume information page to view the recent and active alarms for the selected volume in the **Alarms** pane.

last-updated: Apr 11, 2022

- Click  (in the top navigation bar) and select **Volume Alarms** from the drop-down menu in the **All** alarms pane to view all the active volume alarms.
- Click **Overview** and select **Volume Alarms** from the drop-down menu in the **Active Alarms** pane to view all active volume alarms.

You can:

- View information related to the alarm.
- Dismiss an alarm.
- Mute an alarm.

See Volume Alarms on page 2240 for more information on the volume alarms.

Retrieving Active Volume Alarms Using the CLI or REST API

The basic command to retrieve node alarms is:

```
maprcli alarm list -cluster <cluster name> -type volume
```

For complete reference information, see `alarm list` on page 1545.

Viewing Active Table Replication Alarms
Describes how to view active table replication alarms using the Control System and the CLI.

You can view table replication alarms using the Control System, the log files, and the CLI.

Viewing Active Table Alarms in the Control System

- Log in to the Control System and click **Data** > **Tables** to view table replication alarms in the **Active Alarms** pane.

  You can:

  - View information related to the alarm.
  - Dismiss an alarm.
  - Mute an alarm.

  See Table-Replication Alarms on page 2237 for more information on the table alarms.

Retrieving Active Table Replication Alarms Using the CLI or REST API

Alarms for replication are issued per volume rather than per source table. To retrieve table replication alarms, run the following command:

```
maprcli alarm list -cluster <cluster name> -type volume
```

For complete reference information, see `alarm list` on page 1545.

Viewing Table Replication Alarms in the Log Files

The log files `mfs.log-5` and `cldb.log` display these alarms. These files are located in the `/opt/mapr/logs` directory.

Viewing Active Stream Alarms
Describes how to view active stream alarms using the Control System and the CLI.

You can view stream alarms using the Control System and the CLI.

Viewing Active Stream Alarms in the Control System

- Log in to the Control System and:
  - Click **Data** > **Streams** to view all stream alarms in the **Active Alarms** pane.
  - Go to the **Summary** tab in the stream information page to view the recent and active alarms for the selected stream in the **Active Alarms** pane.

  You can:
  - View information related to the alarm.
  - Dismiss an alarm.
  - Mute an alarm.

  See Alarms Reference on page 2221 for more information on the stream alarms.

Retrieving Active Stream Alarms Using the CLI or REST API

The basic command:

```
maprcli alarm list -cluster <cluster name> -type volume
```

For complete reference information, see `alarm list` on page 1545.

Monitoring Errors
Explains how to monitor errors using the Control System.

To view the errors on the cluster:

1. Log in to the Control System and click  to display the **Recent Errors** window.
   The **Recent Errors** window displays only if there are any errors. For each error, the window displays the following:

Column Name	Column Description
Name	The name of the error.
Type	The type of error.
Action	The type of operation during which the error occurred.
Date & Time	The date and time when the error occurred.
Description	A brief description of the error.

2. Choose one of the following from the drop-down menu to filter the list of errors by a specific type:
   - All Types — to display all types of errors.
   - Data — to display errors related to data
   - Disk Usage — to display errors related to disk usage
   - Nodes — to display errors on nodes
   - Schedules — to display errors related to schedules
   - Services — to display service-related errors
   - Snapshots — to display errors during snapshot

last-updated: Apr 11, 2022

- Volumes — to display errors on volumes
- Tables — to display errors on tables

Metering
The Control System includes a metering feature that can be used to collect cluster storage and compute usage details for consumption-based billing, for actual monthly storage and computations.

Understanding MapR Metering

The metering feature relies on collecting data for cluster storage and compute usage to produce reports. The metering reporting process is secure, reliable, and auditable.

The metrics are collected every 15 minutes and output in JSON format. This JSON file is written to `/var/mapr/metering/` with READ-ONLY permission. For archival and audit purposes, the metering files are saved in MapR filesystem, `/var/mapr/metering/zips` and `/var/mapr/metering/sent`. It is recommended to keep these files for a period of at least 2 years.

For a description of each metric collected by default, see Metering Data Descriptions on page 1322.

For install instructions, see Installing Metering on page 283.

Note: If more than one API server is running on your cluster, the Control System automatically turns on High Availability (HA) for your cluster.

Enabling or Disabling Upload to MapR Server
You can agree to (enable) or turn off (disable) automatic upload of metrics to the MapR server using the Control System and the CLI.

If the cluster is connected to the internet, automatic upload of metrics is enabled by default. If the cluster is not connected to the Internet, disable automatic upload of data and follow the steps to upload metering data manually.
Enabling or Disabling Using the Control System

1. Log in to the Control System and go to the **Auditing and Metering** tab in the **Admin** > **Cluster Settings** page.
2. Move the slider to **Yes** to enable or **No** to disable in the **Send Metering Metrics to MapR** section.

Enabling or Disabling Using the CLI

- Set the value for `mcs.metering.upload.data` to `0` to disable and `1` to enable by running the following command.

```
maprcli config save -values '{"mcs.metering.upload.data":"1"}'
```

Uploading Metering Measurement Results to MapR Server
You can choose to automatically send metrics to MapR server or upload metrics manually, depending on whether your cluster has internet connectivity.
Uploading Automatically While Connected to the Internet

If the cluster is connected to the internet, automatic upload of metrics is enabled by default. To disable automatic upload, see Enabling or Disabling Upload to MapR Server on page 1320.

Uploading Manually While Connected to the Internet

If you do not want to send your metrics every 30 minutes and instead want to send metrics on demand, perform the following steps:

1. Disable automatic upload of metrics in the MapR Control System web interface.
   See Enabling or Disabling Upload to MapR Server on page 1320 for the steps.

last-updated: Apr 11, 2022

2. Run the following command in the MapR CLI to create a .zip file of the metrics:

```
mapr-apiserver-cli.sh --prepareForOfflineSend
```

   The .zip file is stored in the `/var/mapr/metering/zips/` directory.

3. Download the following `tar` or `zip` file:

Linux and MAC	`https://package.mapr.com/releases/metering/mapr-apiserver-utils/mapr-apiserver-utils-1.0.201809050108.tar`
Windows	`https://package.mapr.com/releases/metering/mapr-apiserver-utils/mapr-apiserver-utils-1.0.201809050108.zip`

4. Untar the file into any target directory and `cd` into the that directory:

```
$ tar -xvf mapr-apiserver-utils-1.0.201809050108.tar -C <TARGET_DIR>
$ cd <TARGET_DIR>
```

5. Change directory into `./mapr-apiserver-utils/bin`:

```
$ cd ./mapr-apiserver-utils/bin/
```

6. Run the upload command:

```
$ ./uploadMetrics.sh -u -z <ZIP>
```

Uploading Using a Standalone Utility on an Internet-Connected Computer

You must transfer the metrics file to an internet-connected computer and then upload manually when your cluster is not connected to the internet.

1. Run the following command to create a .zip file of the metrics:

```
mapr-apiserver-cli.sh --prepareForOfflineSend
```

   The .zip file is stored in the `/var/mapr/metering/zips/` directory.

2. Copy the .zip file from the MapR cluster onto an internet-connected computer:

```
$ hadoop fs -copyToLocal /var/mapr/metering/zips/<ZIP>
```

3. On the internet-connected computer, download the following `tar` or `zip` file:

Linux and MAC	`https://package.mapr.com/releases/metering/mapr-apiserver-utils/mapr-apiserver-utils-1.0.201809050108.tar`

last-updated: Apr 11, 2022

Windows	`https://package.mapr.com/releases/metering/mapr-apiserver-utils/mapr-apiserver-utils-1.0.201809050108.zip`

4. Untar the file into any target directory and `cd` into the that directory:

```
$ tar -xvf mapr-apiserver-utils-1.0.201809050108.tar -C <TARGET_DIR>
$ cd <TARGET_DIR>
```

5. Change directory into `./mapr-apiserver-utils/bin`:

```
$ cd ./mapr-apiserver-utils/bin/
```

6. Run the upload command:

Linux	`$ ./uploadMetrics.sh -u -z <ZIP>`
Windows	`$ ./uploadMetrics.bat -u -z <ZIP>`

Calculating Metering Metrics

The metering metrics of tiered data is based on collection of total gigabytes of data offloaded with object tiering to cloud storage and the total amount of data (without replication) sitting in the cold tier.

- Total gigabytes of data offloaded with object tiering to cloud storage.

  This metric is computed by picking the "offloaded" value, in gigabytes, in the collected dashboard info on page 1587 stats from the tiering section. At any time, it gives the total amount of data offloaded to cold tiers for that cluster in gigabytes. The dashboard information is collected every 15 mins by the metering daemon.

- Total amount of data (without replication) sitting in the cold tier.

  This metric is also computed from the same series of collected dashboard info on page 1587 stats from the tiering section. It is the difference between consecutive values of "offloaded" in the series, ignoring negative values, and summing the series of positive values over the billing period.

Metering Data Descriptions

This table lists the metrics collected by the metering feature.

Table

JSON Field Name	Description
clusterId	Randomly generated ID created when the cluster is installed
collectionDate	Epoch timestamp for the latest metric collection
mapRCoreBuildVersion	MapR core software build version
signature	Digital signature of the metering JSON file
version	Version of the metering program used to capture this JSON result

Table (Continued)

JSON Field Name	Description
isSecure	Identifies whether the cluster is secure or not Options: `true` or `false`.
numberOfNodes	Number of nodes in the cluster
clusterDiskCapacityInGB	Total data in the cluster at that time
clusterDiskCapacityUsedInGB	Total data consumed in the cluster at that time
totalAmountOfDataOffloadedToColdTiersInGB	Data offloaded to the cloud, using Object Tiering You can display this metric using the `maprcli dashboard info` command.
cpuCoreInSeconds	CPU core hours consumed by all MapR Services across all nodes Also, CPU core-hours used by all Apache ecosystem components, as well as MapR File System, CLDB, API servers, REST gateways, replication gateways, and so on.
clients	Number of active instances at that moment
numberOfactiveUniquePlatinumPosixClients	Number of monthly active unique Platinum POSIX clients You can display this metric using the `maprcli node list -clientsonly` command.
numberOfactiveUniqueDSROrPACCInstances	Number of monthly active unique DSR instances, using Gold POSIX clients

Sample JSON File for Metering

A sample metering JSON file for an 8-node cluster with no workloads enabled.


```
{
   "id":"metering-1529939795-b82b1db90a364a0d9af5b35328db133f",
   "clusterId":"8036442972050269505",
   "collectionDate":"1529939795",
   "mapRCoreBuildVersion":"6.1.0.20180621112549.GA",

"signature":"8c9820b3eaac45c45952635a0733ba82c1f5e1d6ddbfa31602acff351bfe6f2
a",
   "version":2.0,
   "isSecure":true,
   "numberOfNodes":8,
   "storage":{
      "clusterDiskCapacityInGB":1374,
      "clusterDiskSpaceUsedInGB":68,
      "totalAmountOfDataOffloadedToColdTiersInGB":0
   },
   "nodes":[
      {
         "id":"5411072923155745779",
         "yarn":{
            "allocatedVcores":0.0,
            "availableVcores":4.0
         },
         "processes":[
            {
               "name":"warden",
               "cpuCoreInSeconds":44854
```

```
            },
            {
               "name":"mfs",
               "cpuCoreInSeconds":7386580
            },
            {
               "name":"collectd",
               "cpuCoreInSeconds":10633592
            },
            {
               "name":"zookeeper_server",
               "cpuCoreInSeconds":255249
            },
            {
               "name":"data-access-gateway",
               "cpuCoreInSeconds":890914
            },
            {
               "name":"hbase-mapr-rest",
               "cpuCoreInSeconds":33099
            },
            {
               "name":"yarn-mapr-nodemanager",
               "cpuCoreInSeconds":116368
            },
            {
               "name":"hoststats",
               "cpuCoreInSeconds":31701
            },
            {
               "name":"apiserver",
               "cpuCoreInSeconds":138405
            },
            {
               "name":"gateway",
               "cpuCoreInSeconds":44854
            },
            {
               "name":"mastgateway",
               "cpuCoreInSeconds":187132
            },
            {
               "name":"ganesha",
               "cpuCoreInSeconds":49779
            },
            {
               "name":"nfs4server",
               "cpuCoreInSeconds":49779
            },
            {
               "name":"drillbit",
               "cpuCoreInSeconds":97031
            },
            {
               "name":"cldb",
               "cpuCoreInSeconds":501441
            },
            {
               "name":"fluentd",
               "cpuCoreInSeconds":10191
            }
         ],
         "clients":[
            {
```

```
               "id":"1842047781700531199",
               "clienttype":"posixclientplatinum",
               "clienthealth":"Active"
            }
         ],
         "dsr_service_configured":false
      },
      {
         "id":"6250302233036420992",
         "yarn":{
            "numberOfCPUsAllocated":0.0,
            "numberOfCPUsAvailable":4.0
         },
         "processes":[
            {
               "name":"warden",
               "cpuCoreInSeconds":53348
            },
            {
               "name":"mfs",
               "cpuCoreInSeconds":9973216
            },
            {
               "name":"collectd",
               "cpuCoreInSeconds":10868359
            },
            {
               "name":"zookeeper_server",
               "cpuCoreInSeconds":322375
            },
            {
               "name":"data-access-gateway",
               "cpuCoreInSeconds":24619
            },
            {
               "name":"hbase-mapr-rest",
               "cpuCoreInSeconds":36581
            },
            {
               "name":"yarn-mapr-nodemanager",
               "cpuCoreInSeconds":132342
            },
            {
               "name":"hoststats",
               "cpuCoreInSeconds":36616
            },
            {
               "name":"apiserver",
               "cpuCoreInSeconds":164456
            },
            {
               "name":"gateway",
               "cpuCoreInSeconds":91140
            },
            {
               "name":"mastgateway",
               "cpuCoreInSeconds":236735
            },
            {
               "name":"opentsdb",
               "cpuCoreInSeconds":3492717
            },
            {
               "name":"ganesha",
```

```
               "cpuCoreInSeconds":68427
            },
            {
               "name":"nfs4server",
               "cpuCoreInSeconds":68427
            },
            {
               "name":"drillbit",
               "cpuCoreInSeconds":91138
            },
            {
               "name":"elasticsearch",
               "cpuCoreInSeconds":3060305
            },
            {
               "name":"cldb",
               "cpuCoreInSeconds":326018
            },
            {
               "name":"fluentd",
               "cpuCoreInSeconds":10965
            }
         ],
         "clients":[

         ],
         "dsr_service_configured":true
      },
      {
         "id":"2948891216591685686",
         "yarn":{
            "numberOfCPUsAllocated":0.0,
            "numberOfCPUsAvailable":4.0
         },
         "processes":[
            {
               "name":"warden",
               "cpuCoreInSeconds":77977
            },
            {
               "name":"mfs",
               "cpuCoreInSeconds":257182
            },
            {
               "name":"collectd",
               "cpuCoreInSeconds":12581361
            },
            {
               "name":"zookeeper_server",
               "cpuCoreInSeconds":505537
            },
            {
               "name":"data-access-gateway",
               "cpuCoreInSeconds":35647
            },
            {
               "name":"hbase-mapr-rest",
               "cpuCoreInSeconds":39213
            },
            {
               "name":"yarn-mapr-nodemanager",
               "cpuCoreInSeconds":185212
            },
            {
```

```
                "name":"hoststats",
                "cpuCoreInSeconds":101114
            },
            {
                "name":"apiserver",
                "cpuCoreInSeconds":4237000
            },
            {
                "name":"gateway",
                "cpuCoreInSeconds":903572
            },
            {
                "name":"mastgateway",
                "cpuCoreInSeconds":256831
            },
            {
                "name":"opentsdb",
                "cpuCoreInSeconds":3492194
            },
            {
                "name":"ganesha",
                "cpuCoreInSeconds":149134
            },
            {
                "name":"nfs4server",
                "cpuCoreInSeconds":149134
            },
            {
                "name":"drillbit",
                "cpuCoreInSeconds":102115
            },
            {
                "name":"elasticsearch",
                "cpuCoreInSeconds":3887653
            },
            {
                "name":"cldb",
                "cpuCoreInSeconds":110166
            },
            {
                "name":"yarn-mapr-resourcemanager",
                "cpuCoreInSeconds":347747
            },
            {
                "name":"fluentd",
                "cpuCoreInSeconds":16277
            }
        ],
        "clients":[

        ],
        "dsr_service_configured":true
    },
    {
        "id":"952332856626659546",
        "yarn":{
            "numberOfCPUsAllocated":0.0,
            "numberOfCPUsAvailable":4.0
        },
        "processes":[
            {
                "name":"warden",
                "cpuCoreInSeconds":54471
            },
```

```
            {
               "name":"mfs",
               "cpuCoreInSeconds":340097
            },
            {
               "name":"collectd",
               "cpuCoreInSeconds":11863923
            },
            {
               "name":"hbase-mapr-rest",
               "cpuCoreInSeconds":30307
            },
            {
               "name":"yarn-mapr-nodemanager",
               "cpuCoreInSeconds":150883
            },
            {
               "name":"hoststats",
               "cpuCoreInSeconds":31853
            },
            {
               "name":"mastgateway",
               "cpuCoreInSeconds":202457
            },
            {
               "name":"opentsdb",
               "cpuCoreInSeconds":3626835
            },
            {
               "name":"ganesha",
               "cpuCoreInSeconds":36490
            },
            {
               "name":"nfs4server",
               "cpuCoreInSeconds":36488
            },
            {
               "name":"drillbit",
               "cpuCoreInSeconds":76016
            },
            {
               "name":"elasticsearch",
               "cpuCoreInSeconds":2804536
            },
            {
               "name":"fluentd",
               "cpuCoreInSeconds":10227
            },
            {
               "name":"mapred-mapr-historyserver",
               "cpuCoreInSeconds":37070
            },
            {
               "name":"httpfs",
               "cpuCoreInSeconds":47311
            },
            {
               "name":"grafana",
               "cpuCoreInSeconds":14206
            },
            {
               "name":"hue",
               "cpuCoreInSeconds":4261
            },
```

```
            {
               "name":"spark-mapr-org",
               "cpuCoreInSeconds":61948
            },
            {
               "name":"kibana",
               "cpuCoreInSeconds":101479
            },
            {
               "name":"hive-mapr-hiveserver2",
               "cpuCoreInSeconds":105652
            },
            {
               "name":"hive-mapr-metastore",
               "cpuCoreInSeconds":39584
            },
            {
               "name":"oozie",
               "cpuCoreInSeconds":717478
            },
            {
               "name":"hbase-mapr-thrift",
               "cpuCoreInSeconds":32331
            }
         ],
         "clients":[

         ],
         "dsr_service_configured":false
      },
      {
         "id":"8619138289230167562",
         "yarn":{
            "numberOfCPUsAllocated":0.0,
            "numberOfCPUsAvailable":4.0
         },
         "processes":[
            {
               "name":"warden",
               "cpuCoreInSeconds":41172
            },
            {
               "name":"mfs",
               "cpuCoreInSeconds":189731
            },
            {
               "name":"collectd",
               "cpuCoreInSeconds":27735
            },
            {
               "name":"hbase-mapr-rest",
               "cpuCoreInSeconds":31519
            },
            {
               "name":"yarn-mapr-nodemanager",
               "cpuCoreInSeconds":150749
            },
            {
               "name":"hoststats",
               "cpuCoreInSeconds":34185
            },
            {
               "name":"apiserver",
               "cpuCoreInSeconds":27253
```

```
            },
            {
               "name":"mastgateway",
               "cpuCoreInSeconds":194093
            },
            {
               "name":"ganesha",
               "cpuCoreInSeconds":36815
            },
            {
               "name":"nfs4server",
               "cpuCoreInSeconds":36815
            },
            {
               "name":"drillbit",
               "cpuCoreInSeconds":85679
            },
            {
               "name":"fluentd",
               "cpuCoreInSeconds":9464
            }
         ],
         "clients": [

{ "id": "3893819064903061731", "clientType": "posixclientplatinum",
"clientHealth": "Active" },

{ "id": "6476832847333424974", "clientType": "posixclientplatinum",
"clientHealth": "Active" },

{ "id": "4713798361306456121", "clientType": "posixclientplatinum",
"clientHealth": "Active" },

{ "id": "2714377982799294814", "clientType": "posixclientplatinum",
"clientHealth": "Active" }

],
      }
   ]
}
```


Using MapR Data Platform Monitoring (Spyglass Initiative)

MapR Data Platform Monitoring (part of the Spyglass initiative) provides the ability to collect, store, and view metrics and logs for nodes, services, and jobs/applications.

Metric Monitoring

Administrators can monitor the current status of the cluster and anticipate future cluster requirements with dashboards. For example, you can use metrics dashboards to visualize the following:

Storage Utilization	Use metrics dashboards to monitor storage trends. For example, you can compare the volume of MapR File System usage at different times to the MapR File System capacity and then allocate resources to the MapR File System accordingly.
Node Utilization	Use metrics dashboards to check for node overload. For example, if the CPU usage is high on a few nodes, you may want to distribute the load across more nodes for better performance and efficiency.
MapR Database Operational Trends	Use metrics dashboards to display historical trends for MapR Database operations. For example, if a user

	reports MapR Database slowness, the historical trends associated with row scans, get, and put operations can be used to identify the node(s) on which the performance degradation occurs.

Log Monitoring

Administrators can use dashboards to visualize, search, and review logs when troubleshooting issues. For example, you can use log dashboards to troubleshoot the following issues:

Service Failures	When metrics indicate that one or more services are down, use log dashboards to check the logs for each failed service and drill-down to each associated node.
Application Failures	When an application or job fails, use log dashboard to identify possible bottlenecks. For example, you can search the logs for a given application ID across all the nodes in the cluster.
MapR File System Performance	When users experience MapR File System or NFS slowness, use log dashboards to search the MapR filesystem logs for service errors or application issues.

Related Information

- Using MapR Data Platform Monitoring (Spyglass Initiative) on page 1330
- MapR Monitoring Storage Options on page 116
- Step 8: Install Metrics Monitoring on page 162
- Step 9: Install Log Monitoring on page 165

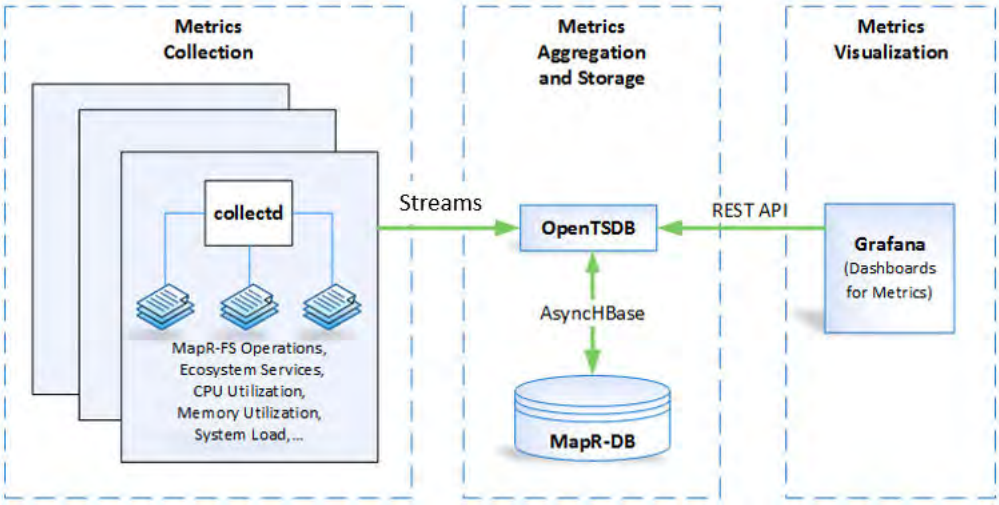
MapR Monitoring Architecture

MapR Monitoring integrates with open-source components to collect, aggregate, store, and visualize metrics and logs.

Note: The MapR Monitoring architecture is designed for use on MapR cluster nodes. Installing monitoring components on client nodes or edge nodes is not supported.

Metric Monitoring Architecture

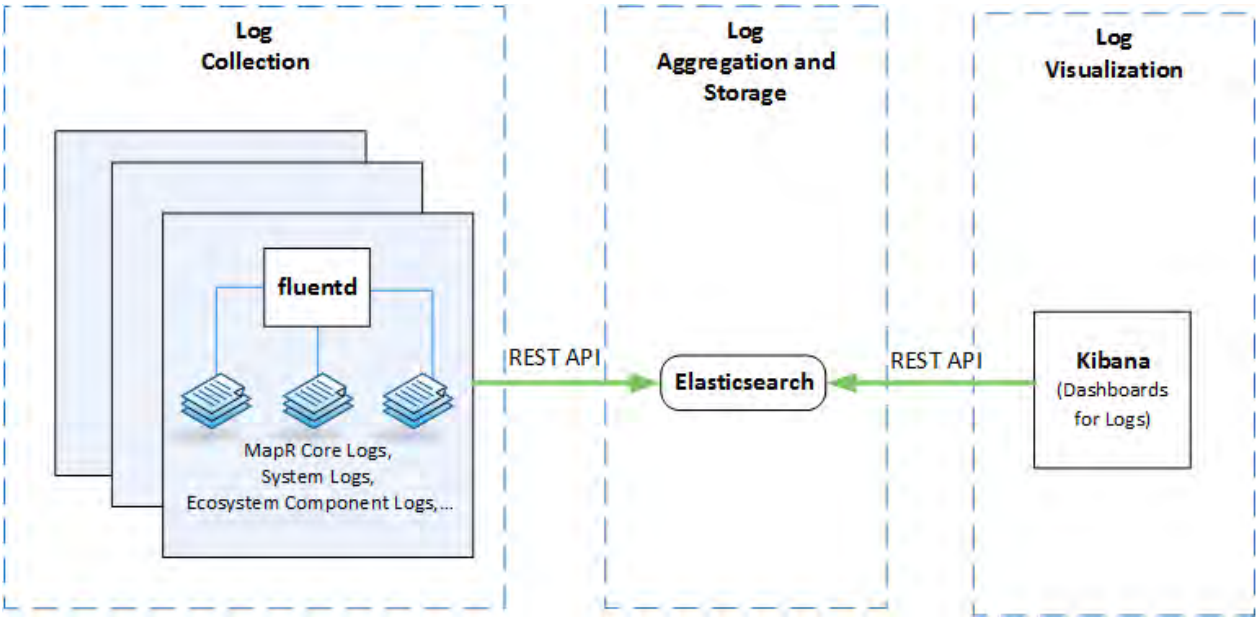
To visualize cluster metrics, MapR Monitoring integrates with the following components:



collectd	The collectd service runs on each node in the cluster to collect metrics. It uses streams to send the metrics to opentsdb. For more information, see Metric Collection on page 1334.
OpenTSDB	OpenTSDB aggregates the metrics and runs as the time-series database on top of MapR Database, the metrics data store. Based on your cluster requirements, it runs on one or more nodes in the cluster.
Grafana	Grafana uses REST API to access metrics data from OpenTSDB. Using a single instance of Grafana, users can build custom dashboards or use sample dashboards to visualize the metric. For more information about dashboards, see Metric Visualization on page 1381.

Log Monitoring Architecture

To visualize logs, MapR Monitoring integrates with the following components:



fluentd	fluentd runs on each node in the cluster to collect and parse logs. It uses REST API to send the logs to ElasticSearch. For more information, see Log Collection on page 1386.
Elasticsearch	Elasticsearch indexes the logs so that they are easily accessed and searchable. Based on your cluster requirements, it runs on one or more nodes in the cluster. For more information, see Log Aggregation and Storage on page 1391.
Kibana	Kibana uses REST API to access and search the logs available in Elasticsearch. Using a single instance of Kibana, users can create visualizations and dashboards to analyze their logs. For more information, see Log Visualization on page 1396.

Spyglass on Streams

Release 6.0 of the MapR Data Platform introduced Spyglass on Streams. When you install release 6.0 or later, Streams is the default mechanism through which metrics flow from the Collectd service to

OpenTSDB. Moving metrics through streams secures the data and provides a mechanism to perform real-time data analytics.

Note: Currently, Spyglass on Streams is not available for logs. Fluentd continues to use the REST API to send logs to ElasticSearch for the indexing of logs.

The Flow of Metrics via Streams

The Collectd service collects node-level and service-level metrics from each node in the cluster. The Collectd service hashes metrics to a stream and writes the metrics into topics in that stream.

In release 6.1.0 and later, Collectd creates one stream per cluster: `/var/mapr/mapr.monitoring/metricstreams/0`. Topic names use the format `<hostname>`. For example: `mfs81.qa.lab`.

The Streams server distributes metrics to the available OpenTSDB nodes, and OpenTSDB consumes the metrics.

Note: Writing to an external OpenTSDB is not supported from release 6.0 onwards. In addition, inserting non-MapR data into the provided OpenTSDB is not supported. Any custom data added to the provided OpenTSDB will be removed by the purge script (`tsdb_cluster_mgmt.sh`) that runs periodically.

Determining How Many OpenTSDB Nodes to Install

Having multiple OpenTSDB nodes in the cluster distributes the workload. The number of partitions and OpenTSDB nodes determines the level of parallelism for consumption.

Each OpenTSDB node can consume one partition at a time. By default, metrics data is divided across 12 partitions in each topic and optimal parallelism is reached if there are five OpenTSDB nodes to consume the partitions. See Parallelism When Consuming Messages. Note that the term “consumer” in the topic equates to an OpenTSDB node in Spyglass on Streams.

A general guideline for the minimum number of OpenTSDB nodes in a cluster is one for every 10x increase in nodes beyond 10, for example:

- Three OpenTSDB nodes in a 10-node cluster
- Four OpenTSDB nodes in a 100-node cluster
- Five OpenTSDB nodes in a 1000-node cluster

If your cluster has 10 or more nodes, at least three OpenTSDB nodes should be available to consume metrics. Typically, for every 10x increase in nodes, you should add another OpenTSDB node. For example, if your cluster reaches a size of 100 nodes, have four OpenTSDB nodes available for consumption.

These guidelines do not guarantee optimal performance. Evaluate the performance of the streams to determine if your cluster would benefit from additional OpenTSDB nodes.

Note: If all configured OpenTSDB nodes have been offline for several hours, you may notice an initial spike in memory and CPU usage by OpenTSDB processes as they aggressively try to keep up with the metrics. You can reduce the number of AsynchHBase threads to reduce the CPU and memory usage. By default, AsynchHBase runs 128 threads. To modify the number of threads, add or modify the following property in the `/opt/mapr/asynchbase/asynchbase-<version>/conf/asynchbase.conf` file on the OpenTSDB nodes:

```
"fs.mapr.async.worker.threads=<value>"
```


last-updated: Apr 11, 2022

Increasing the Number of Streams

For release 6.1 and later, the default setting for the number of streams is one. Even if your cluster grows to 1000 nodes or more, you do not need to increase the number of streams. For release 6.0.x, increasing the number of streams is recommended as you add more nodes (see the release 6.0 documentation), but this practice is not required in release 6.1 and later.

Changing the Automatic Stream Cursor Commits

You can adjust the frequency of automatic stream cursor commits for OpenTSDB. Modify the `tsd.streams.autocommit.interval` in `opentsdb.conf` The unit is thousands of seconds. The default value is '60000' which is 60 secs. For a system with heavy loads, consider changing the value to something like 5 minutes.

Metric Collection

Metrics are collected from each node in the cluster so that administrators can use the data to monitor the cluster. In general, the `collectd` service collects metrics every 10 seconds. The exception is volume metrics which are collected every 10 minutes.

When `collectd` writes metrics to streams, tags are assigned to each metric so that administrators can filter metric data to create dashboards that are specific to their needs.

By default, each metric contains the following tags:

- `fqdn`: Displays values for a specified node.
- `clusterid`: Displays values for a specific cluster.
- `clustername`: As of EEP 3.0, displays values for a specific cluster.

However, many metrics have additional tags that you can use to filter metric data.

Streams store metrics in OpenTSDB with the following schema:

```
<metrictype.name> <fqdn:fqdnvalue> <clusterid:clusteridvalue>
<clustername:clusternamevalue>[<AdditionalTagA:AdditionalTagAvalue>
<AdditionalTagB:AdditionalTagBvalue>...] <metricvalue> <timestamp>
```

Note: A negative value shown in the metrics indicates that the maximum value configured for that metric is exceeded. The maximum value for GUT metrics is int32 (2^31-1).

For more information on using tags and dashboards, see Metric Visualization on page 1381.

Configure Metric Retention

By default, OpenTSDB stores two weeks of metrics. Based on your requirements, you can change metric retention period.

The following cron job runs each day to purge metrics based on the retention period.

```
$min $hour * * * $OTSDB_HOME/bin/tsdb_cluster_mgmt.sh -purgeData >>
$OTSDB_HOME/var/log/opentsdb/purgeData.log 2>&1
```

Complete the following steps to edit the metric retention period:

1. Open the `/opt/mapr/opentsdb/opentsdb-<version>/bin/tsdb_cluster_mgmt.sh` file.

2. In the following line, update the value of `'2 weeks ago'` to the new retention period.

```
$OT_HOME/bin/tsdb scan --delete 2000/01/01 $(date --date='2 weeks ago'
+'%Y/%m/%d') sum $metric
```

For example, to delete metrics from 1/1/2000 to [current date - 2 days]:

```
$OT_HOME/bin/tsdb scan --delete 2000/01/01 $(date --date='2 days ago'
+'%Y/%m/%d') sum $metric
```

Warning: MapR monitoring uses 2 MB disk space per minute per node when MapR Event Store For Apache Kafka metrics is enabled. This is approximately 3 GB per day on a single node or 7 GB per node per day with a 3X replication. This stream metrics data is automatically deleted every 30 days.

Note: For more information, see the OpenTSDB scan command documentation.

Configure Queue Filters for `mapr.rm.<value>` Metrics

The YARN application metrics that are collected by JMX have the metric name syntax `mapr.rm.<metric_name>` and the metric values are aggregated among all the queues in the default queue. However, you can configure collectd to create a filter for each queue. As an alternative, you can use the REST API queue metrics (`mapr.rm_queue.<metric_name>`) which are by default set up for filtering by queue.

To configure collectd to create queue filters for `mapr.rm.*` metrics, define each queue that you want to create filters for in the `/opt/mapr/collectd/collectd-<version>/etc/collectd.conf` file. You can configure collectd to generate filters for every queue or only for specific queues. Changes that you make to the collectd.conf file only apply to metrics collected after you restart the collectd service.

1. Open the collectd.conf file and locate the MBean "QueueMetrics" block.

```
<MBean "QueueMetrics">
      ObjectName
"Hadoop:service=ResourceManager,name=QueueMetrics,q0=root"
      InstancePrefix "rm"

      <Value "AppsRunning">
        Type "apps_running"
        InstancePrefix "default-queue"
      </Value>

      <Value "ActiveApplications">
        Type "active_applications"
        InstancePrefix "default-queue"
      </Value>

      ...
    </MBean>
```

This block specifies that there is one queue named `root` and that the filter for this queue is named `default-queue`.

2. Create copy of the MBean "QueueMetrics" block.

3. Configure the ObjectName option in the MBean "QueueMetrics" block copy, with the queue path for the queue that you want to create a filter for.

last-updated: Apr 11, 2022

- To define the a child queue named `alpha` under the `root` queue:

```
ObjectName
"Hadoop:service=ResourceManager,name=QueueMetrics,q0=root,q1=alpha"
```

- To define a child queue named beta which is under a child queue named alpha:

```
ObjectName
"Hadoop:service=ResourceManager,name=QueueMetrics,q0=root,q1=alpha,q2=b
eta"
```

4. For each Value block within the MBean "QueueMetrics" block you are defining, replace `default-queue` with the queue name that you want to create a filter for.
   - To define filter value `alpha` for the rm_queue tag, set the InstancePrefix to `alpha:`

```
 <MBean "QueueMetrics">
      ObjectName
"Hadoop:service=ResourceManager,name=QueueMetrics,q0=root,q1=alpha"
      InstancePrefix "rm"
<Value "AppsRunning">
        Type "apps_running"
        InstancePrefix "alpha"
      </Value>
<Value "ActiveApplications">
        Type "active_applications"
        InstancePrefix "alpha"
      </Value>
...
    </MBean>
```

   - To define a filter value `beta` for the rm_queue tag, set the InstancePrefix to `beta:`:

```
 <MBean "QueueMetrics">
      ObjectName
"Hadoop:service=ResourceManager,name=QueueMetrics,q0=root,q1=alpha,q2=b
eta"
      InstancePrefix "rm"
<Value "AppsRunning">
        Type "apps_running"
        InstancePrefix "beta"
      </Value>
<Value "ActiveApplications">
        Type "active_applications"
        InstancePrefix "beta"
      </Value>
...
    </MBean>
```

5. Repeat steps 2 and 3 for each queue that you want to create a filter value for.
6. Save the collectd.conf file.
7. Repeat steps 1 through 6 on each ResourceManager node.

8. Restart the collectd service.

```
maprcli node services -name collectd -nodes <space separated list of
ResourceManager Nodes> -action restart
```

In the following example, rm_queue tag will have the following filter values: `alpha` , `beta` (child of alpha), and `highpriority` (child of root):

```
<MBean "QueueMetrics">
      ObjectName "Hadoop:service=ResourceManager,name=QueueMetrics,q0=root
q1=alpha"
      InstancePrefix "rm"

      <Value "AppsRunning">
        Type "apps_running"
        InstancePrefix "alpha"
      </Value>

      <Value "ActiveApplications">
        Type "active_applications"
        InstancePrefix "alpha"
      </Value>
      ...
      <Value "ReservedVCores">
        Type "reserved_vcores"
        InstancePrefix "alpha"
      </Value>
    </MBean>

<MBean "QueueMetrics">
      ObjectName "Hadoop:service=ResourceManager,name=QueueMetrics,q0=root
q1=alpha q2=beta"
      InstancePrefix "rm"

      <Value "AppsRunning">
        Type "apps_running"
        InstancePrefix "beta"
      </Value>

      <Value "ActiveApplications">
        Type "active_applications"
        InstancePrefix "beta"
      </Value>
      ...
      <Value "ReservedVCores">
        Type "reserved_vcores"
        InstancePrefix "beta"
      </Value>
    </MBean>

<MBean "QueueMetrics">
      ObjectName "Hadoop:service=ResourceManager,name=QueueMetrics,q0=root
q1=highpriority"
      InstancePrefix "rm"

      <Value "AppsRunning">
        Type "apps_running"
        InstancePrefix "highpriority"
      </Value>

      <Value "ActiveApplications">
        Type "active_applications"
```


last-updated: Apr 11, 2022

```
        InstancePrefix "highpriority"
      </Value>
      ...
      <Value "ReservedVCores">
        Type "reserved_vcores"
        InstancePrefix "highpriority"
      </Value>
    </MBean>
```

Configure the Collectd Service Heap Size

The `collectd` service uses an embedded JVM when it gathers metrics from the CLDB, Node Manager, Resource Manager, and Drill. You can edit the Plugin Java section of *collectd.conf* to configure limits to the collectd virtual memory footprint.

Note: The Plugin Java section of the collectd.conf file may be commented or uncommented. The configure.sh utility will uncomment the Plugin Java section when `collectd` runs on a node that requires an embedded JVM. Therefore, when you update the file, do not add or remove comment symbols (#) in the Plugin java section

Complete the following steps on each `collectd` node:

1. Open the `/opt/mapr/collectd/collectd-<version>/etc/collectd.conf` file.
2. Look for the following section:

```
## **** MAPR_CONF_JMX_TAG: MAPR CONFIGURATION - DO NOT EDIT or REMOVE
TAG/BLOCK ***
 <Plugin java>
JVMArg "-Djava.class.path=......
```

3. Update `Xms` and `Xmx` options in the Plugin java section.

   `Xms` defines the amount of memory allocated to the service when it starts. `Xmx` defines the maximum amount of memory allocated to the service.

   If the <Plugin java> section is not commented out, the configuration may look like this:

```
## **** MAPR_CONF_JMX_TAG: MAPR CONFIGURATION - DO NOT EDIT or REMOVE
TAG/BLOCK ***
<Plugin java>
JVMArg "-Djava.class.path=......
JVMArg "-Xms32m"
JVMArg "-Xmx128m"
```

   If the <Plugin java> section is commented out, the configuration may look like this:

```
### **** MAPR_CONF_JMX_TAG: MAPR CONFIGURATION - DO NOT EDIT or REMOVE
TAG/BLOCK ***
#<Plugin java>
# JVMArg "-Djava.class.path=......
# JVMArg "-Xms32m"
# JVMArg "-Xmx128m"
```

4. Restart the `collectd` service.

```
maprcli node services -name collectd -nodes <space separated list of
hostname/IPaddresses> -action restart
```

CPU Metrics

Every 10 seconds, the collectd service uses the cpu plugin to gather the following CPU metrics on each node in the cluster.

Name	Description	Additional Tag(s)
`cpu.percent`	The aggregate percentage of all CPUs.	• `cpu_core`: Display values specified core. Core values: 0,1, and 2 • `cpu_class`: Display values for a specified class of CPU. CPU class values: idle, user, nice, system.
`thread.cpu_usage`	The percentage of CPU used by each thread.	• `thread_name`: Indicates if thread belongs to RPC, MapR Database, MapR File System, or an instance of MapR File System.

Disk Free Metrics

Every 10 seconds, the collectd service uses the df plugin to gather the following disk free metrics on each node in the cluster.

Name	Description	Additional Tag(s)
df.df_complex	The aggregate number of bytes for disk partitions.	• `df_partition`: Display values for a specified disk. Disk values: dev or boot. • `df_type`: Display values for a certain type. Type values: free, reserved, and used.
df.percent_bytes	The aggregated percentage of disk partitions.	• `df_partition`: Display values for a specified disk. Disk values: dev or boot. • `df_type`: Display value for a certain type. Type values: free, reserved, and used.

Disk Metrics

Every 10 seconds, the collectd service uses the disk plugin to gather the following disk metrics on each node in the cluster.

Name	Description	Additional Tag(s)
disk.disk_await	The average time in milliseconds to complete I/O requests. This includes the time request are waiting in queue and the time spent processing the request. This metric is available as of EEP 3.0.	• `disk_name:` Display values for a specified disk.
disk.disk_avg_requests_size	The average size in kilobytes for I/O requests issued to the disk. This metric is available as of EEP 3.0.	• `disk_name:` Display values for a specified disk.
disk.disk_avg_queue_size	The average number of requests issued to the disk. This metric is available as of EEP 3.0.	• `disk_name:` Display values for a specified disk.

last-updated: Apr 11, 2022

Name	Description	Additional Tag(s)
disk.disk_io_time.io_time	The disk I/O time in milliseconds (ms).	• `disk_name`: Display values for a specified disk.
disk.disk_io_time.weighted_io_time	The aggregate time in milliseconds (ms) spent on I/O operations that are either in progress or have completed	• `disk_name`: Display values for a specified disk.
disk.disk_merged.read	The number of physical read operations.	• `disk_name`: Display values for a specified disk.
disk.disk_merged.write	The number of physical write operations.	• `disk_name`: Display values for a specified disk.
disk.disk_octets.read	The number of bytes read from disk.	• `disk_name`: Display values for a specified disk.
disk.disk_octets.write	The number of bytes written to disk.	• `disk_name`: Display values for a specified disk.
disk.disk_ops.read	The number of completed read operations.	• `disk_name`: Display values for a specified disk.
disk.disk_ops.write	The number of completed write operations.	• `disk_name`: Display values for a specified disk.
disk.disk_time.read	The average time in milliseconds(ms) to read from disk.	• `disk_name`: Display values for a specified disk.
disk.disk_time.write	The average time in milliseconds(ms) to write to disk.	• `disk_name`: Display values for a specific disk.
disk.disk_utilization	The disk utilization percentage. This metric is available as of EEP 3.0.	• `disk_name`: Display values for a specific disk.
disk.pending_operations	The number of pending disk operations.	• `disk_name`: Display values for a specific disk.

Drill Metrics

Every 10 seconds, the collectd service uses the plugin to gather the following Drill metrics on each node in the cluster.

Name	Description
mapr.drill.allocator_root_used	The amount of memory used in bytes by the internal memory allocator.
mapr.drill.allocator_root_peak	The peak amount of memory used in bytes by the internal memory allocator.
mapr.drill.blocked_count	The number of threads that are blocked because they are waiting for a monitor lock.
mapr.drill.count	The number of live threads (including both daemon and non-daemon threads).
mapr.drill.fd_usage	The ratio of used to total file descriptors.

last-updated: Apr 11, 2022

Name	Description
mapr.drill.fragments_running	The number of query fragments currently running in the drillbit.
mapr.drill.heap_used	The amount of heap memory used in bytes by the JVM.
mapr.drill.non_heap_used	The amount of non-heap memory used in bytes by the JVM.
mapr.drill.queries_completed	The number of completed, canceled or failed queries for which this drillbit is the foreman.
mapr.drill.queries_running	The number of running queries for which this drillbit is the foreman.
mapr.drill.runnable_count	The number of threads executing in the JVM.
mapr.drill.waiting_count	The number of threads that are waiting to be executed. This can occur when a thread must wait for another thread to perform an action before proceeding.

Hive JMX Metrics

Every 10 seconds, the `collectd` service uses the Hive plug-in to gather the following Hive JMX metrics on each node in the cluster. Descriptions for the Hive metrics are not currently available.

Metric Collection

Metrics collected in Hive relate specifically to the HiveServer2 and Hive metastore processes. Each process runs in a separate JVM, and the JVMs provide values for the metrics.

Starting in EEP 6.3.2 and EEP 7.0.1, the `hive.exec.submit.local.task.via.child` option (in hive-site.xml) is set to `true`, by default, and enables HiveServer2 to spawn local tasks (typically mapjoin hashtable generation phase) in child JVMs. The system does not collect metrics for the child JVMs. The system only collects metrics for the HiveServer2 and Hive metastore processes.

Hive Metastore Metrics

The following are the JMX metrics provided for the Hive metastore:

- mapr.hivemetastore.hivemetastore_buffers_direct_capacity
- mapr.hivemetastore.hivemetastore_buffers_direct_count
- mapr.hivemetastore.hivemetastore_buffers_direct_used
- mapr.hivemetastore.hivemetastore_buffers_mapped_capacity
- mapr.hivemetastore.hivemetastore_buffers_mapped_count
- mapr.hivemetastore.hivemetastore_buffers_mapped_used
- mapr.hivemetastore.hivemetastore_class_loading_loaded
- mapr.hivemetastore.hivemetastore_class_loading_unloaded
- mapr.hivemetastore.hivemetastore_gc_ps_mark_sweep_count
- mapr.hivemetastore.hivemetastore_gc_ps_mark_sweep_time
- mapr.hivemetastore.hivemetastore_gc_ps_scavenge_count
- mapr.hivemetastore.hivemetastore_gc_ps_scavenge_time

last-updated: Apr 11, 2022

- mapr.hivemetastore.hivemetastore_init_total_count_dbs
- mapr.hivemetastore.hivemetastore_init_total_count_partitions
- mapr.hivemetastore.hivemetastore_init_total_count_tables
- mapr.hivemetastore.hivemetastore_memory_heap_committed
- mapr.hivemetastore.hivemetastore_memory_heap_init
- mapr.hivemetastore.hivemetastore_memory_heap_max
- mapr.hivemetastore.hivemetastore_memory_heap_usage
- mapr.hivemetastore.hivemetastore_memory_heap_used
- mapr.hivemetastore.hivemetastore_memory_non_heap_committed
- mapr.hivemetastore.hivemetastore_memory_non_heap_init
- mapr.hivemetastore.hivemetastore_memory_non_heap_max
- mapr.hivemetastore.hivemetastore_memory_non_heap_usage
- mapr.hivemetastore.hivemetastore_memory_non_heap_used
- mapr.hivemetastore.hivemetastore_memory_pools_code_cache_usage
- mapr.hivemetastore.hivemetastore_memory_pools_compressed_class_space_usage
- mapr.hivemetastore.hivemetastore_memory_pools_metaspace_usage
- mapr.hivemetastore.hivemetastore_memory_pools_ps_eden_space_usage
- mapr.hivemetastore.hivemetastore_memory_pools_ps_old_gen_usage
- mapr.hivemetastore.hivemetastore_memory_pools_ps_survivor_space_usage
- mapr.hivemetastore.hivemetastore_memory_total_committed
- mapr.hivemetastore.hivemetastore_memory_total_init
- mapr.hivemetastore.hivemetastore_memory_total_max
- mapr.hivemetastore.hivemetastore_memory_total_used
- mapr.hivemetastore.hivemetastore_threads_blocked_count
- mapr.hivemetastore.hivemetastore_threads_count
- mapr.hivemetastore.hivemetastore_threads_daemon_count
- mapr.hivemetastore.hivemetastore_threads_deadlock_count
- mapr.hivemetastore.hivemetastore_threads_deadlocks
- mapr.hivemetastore.hivemetastore_threads_new_count
- mapr.hivemetastore.hivemetastore_threads_runnable_count
- mapr.hivemetastore.hivemetastore_threads_terminated_count

- mapr.hivemetastore.hivemetastore_threads_timed_waiting_count
- mapr.hivemetastore.hivemetastore_threads_waiting_count
- mapr.hivemetastore.hivemetastore_active_calls_api_get_all_databases
- mapr.hivemetastore.hivemetastore_active_calls_api_get_all_functions
- mapr.hivemetastore.hivemetastore_active_calls_api_get_all_tables
- mapr.hivemetastore.hivemetastore_active_calls_api_get_database
- mapr.hivemetastore.hivemetastore_active_calls_api_get_multi_table
- mapr.hivemetastore.hivemetastore_active_calls_api_get_table_objects_by_name_req
- mapr.hivemetastore.hivemetastore_active_calls_api_init
- mapr.hivemetastore.hivemetastore_jvm_pause_extra_sleep_time
- mapr.hivemetastore.hivemetastore_open_connections
- mapr.hivemetastore.hivemetastore_api_get_all_databases_count
- mapr.hivemetastore.hivemetastore_api_get_all_databases_max
- mapr.hivemetastore.hivemetastore_api_get_all_databases_mean
- mapr.hivemetastore.hivemetastore_api_get_all_databases_min
- mapr.hivemetastore.hivemetastore_api_get_all_databases_p50
- mapr.hivemetastore.hivemetastore_api_get_all_databases_p75
- mapr.hivemetastore.hivemetastore_api_get_all_databases_p95
- mapr.hivemetastore.hivemetastore_api_get_all_databases_p98
- mapr.hivemetastore.hivemetastore_api_get_all_databases_p99
- mapr.hivemetastore.hivemetastore_api_get_all_databases_p999
- mapr.hivemetastore.hivemetastore_api_get_all_databases_stddev
- mapr.hivemetastore.hivemetastore_api_get_all_databases_m15_rate
- mapr.hivemetastore.hivemetastore_api_get_all_databases_m1_rate
- mapr.hivemetastore.hivemetastore_api_get_all_databases_m5_rate
- mapr.hivemetastore.hivemetastore_api_get_all_databases_mean_rate
- mapr.hivemetastore.hivemetastore_api_get_all_functions_count
- mapr.hivemetastore.hivemetastore_api_get_all_functions_max
- mapr.hivemetastore.hivemetastore_api_get_all_functions_mean
- mapr.hivemetastore.hivemetastore_api_get_all_functions_min
- mapr.hivemetastore.hivemetastore_api_get_all_functions_p50

last-updated: Apr 11, 2022

- mapr.hivemetastore.hivemetastore_api_get_all_functions_p75
- mapr.hivemetastore.hivemetastore_api_get_all_functions_p95
- mapr.hivemetastore.hivemetastore_api_get_all_functions_p98
- mapr.hivemetastore.hivemetastore_api_get_all_functions_p99
- mapr.hivemetastore.hivemetastore_api_get_all_functions_p999
- mapr.hivemetastore.hivemetastore_api_get_all_functions_stddev
- mapr.hivemetastore.hivemetastore_api_get_all_functions_m15_rate
- mapr.hivemetastore.hivemetastore_api_get_all_functions_m1_rate
- mapr.hivemetastore.hivemetastore_api_get_all_functions_m5_rate
- mapr.hivemetastore.hivemetastore_api_get_all_functions_mean_rate
- mapr.hivemetastore.hivemetastore_api_get_all_tables_count
- mapr.hivemetastore.hivemetastore_api_get_all_tables_max
- mapr.hivemetastore.hivemetastore_api_get_all_tables_mean
- mapr.hivemetastore.hivemetastore_api_get_all_tables_min
- mapr.hivemetastore.hivemetastore_api_get_all_tables_p50
- mapr.hivemetastore.hivemetastore_api_get_all_tables_p75
- mapr.hivemetastore.hivemetastore_api_get_all_tables_p95
- mapr.hivemetastore.hivemetastore_api_get_all_tables_p98
- mapr.hivemetastore.hivemetastore_api_get_all_tables_p99
- mapr.hivemetastore.hivemetastore_api_get_all_tables_p999
- mapr.hivemetastore.hivemetastore_api_get_all_tables_stddev
- mapr.hivemetastore.hivemetastore_api_get_all_tables_m15_rate
- mapr.hivemetastore.hivemetastore_api_get_all_tables_m1_rate
- mapr.hivemetastore.hivemetastore_api_get_all_tables_m5_rate
- mapr.hivemetastore.hivemetastore_api_get_all_tables_mean_rate
- mapr.hivemetastore.hivemetastore_api_get_multi_table_count
- mapr.hivemetastore.hivemetastore_api_get_multi_table_max
- mapr.hivemetastore.hivemetastore_api_get_multi_table_mean
- mapr.hivemetastore.hivemetastore_api_get_multi_table_min
- mapr.hivemetastore.hivemetastore_api_get_multi_table_p50
- mapr.hivemetastore.hivemetastore_api_get_multi_table_p75

- mapr.hivemetastore.hivemetastore_api_get_multi_table_p95
- mapr.hivemetastore.hivemetastore_api_get_multi_table_p98
- mapr.hivemetastore.hivemetastore_api_get_multi_table_p99
- mapr.hivemetastore.hivemetastore_api_get_multi_table_p999
- mapr.hivemetastore.hivemetastore_api_get_multi_table_stddev
- mapr.hivemetastore.hivemetastore_api_get_multi_table_m15_rate
- mapr.hivemetastore.hivemetastore_api_get_multi_table_m1_rate
- mapr.hivemetastore.hivemetastore_api_get_multi_table_m5_rate
- mapr.hivemetastore.hivemetastore_api_get_multi_table_mean_rate
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_count
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_max
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_mean
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_min
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_p50
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_p75
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_p95
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_p98
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_p99
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_p999
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_stddev
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_m15_rate
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_m1_rate
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_m5_rate
- mapr.hivemetastore.hivemetastore_api_get_table_objects_by_name_req_mean_rate
- mapr.hivemetastore.hivemetastore_api_init_count
- mapr.hivemetastore.hivemetastore_api_init_max
- mapr.hivemetastore.hivemetastore_api_init_mean
- mapr.hivemetastore.hivemetastore_api_init_min
- mapr.hivemetastore.hivemetastore_api_init_p50
- mapr.hivemetastore.hivemetastore_api_init_p75
- mapr.hivemetastore.hivemetastore_api_init_p95

last-updated: Apr 11, 2022

- mapr.hivemetastore.hivemetastore_api_init_p98
- mapr.hivemetastore.hivemetastore_api_init_p99
- mapr.hivemetastore.hivemetastore_api_init_p999
- mapr.hivemetastore.hivemetastore_api_init_stddev
- mapr.hivemetastore.hivemetastore_api_init_m15_rate
- mapr.hivemetastore.hivemetastore_api_init_m1_rate
- mapr.hivemetastore.hivemetastore_api_init_m5_rate
- mapr.hivemetastore.hivemetastore_api_init_mean_rate

HiveServer2 Metrics

The following are the JMX metrics provided for HiveServer2 (hs2):

- mapr.hiveserver2.hiveserver2_buffers_direct_capacity
- mapr.hiveserver2.hiveserver2_buffers_direct_count
- mapr.hiveserver2.hiveserver2_buffers_direct_used
- mapr.hiveserver2.hiveserver2_buffers_mapped_capacity
- mapr.hiveserver2.hiveserver2_buffers_mapped_count
- mapr.hiveserver2.hiveserver2_buffers_mapped_used
- mapr.hiveserver2.hiveserver2_class_loading_loaded
- mapr.hiveserver2.hiveserver2_class_loading_unloaded
- mapr.hiveserver2.hiveserver2_exec_async_pool_size
- mapr.hiveserver2.hiveserver2_exec_async_queue_size
- mapr.hiveserver2.hiveserver2_gc_ps_mark_sweep_count
- mapr.hiveserver2.hiveserver2_gc_ps_mark_sweep_time
- mapr.hiveserver2.hiveserver2_gc_ps_scavenge_count
- mapr.hiveserver2.hiveserver2_gc_ps_scavenge_time
- mapr.hiveserver2.hiveserver2_active_sessions
- mapr.hiveserver2.hiveserver2_open_sessions
- mapr.hiveserver2.hiveserver2_init_total_count_dbs
- mapr.hiveserver2.hiveserver2_init_total_count_partitions
- mapr.hiveserver2.hiveserver2_init_total_count_tables
- mapr.hiveserver2.hiveserver2_memory_heap_committed
- mapr.hiveserver2.hiveserver2_memory_heap_init

- mapr.hiveserver2.hiveserver2_memory_heap_max
- mapr.hiveserver2.hiveserver2_memory_heap_usage
- mapr.hiveserver2.hiveserver2_memory_heap_used
- mapr.hiveserver2.hiveserver2_memory_non_heap_committed
- mapr.hiveserver2.hiveserver2_memory_non_heap_init
- mapr.hiveserver2.hiveserver2_memory_non_heap_max
- mapr.hiveserver2.hiveserver2_memory_non_heap_usage
- mapr.hiveserver2.hiveserver2_memory_non_heap_used
- mapr.hiveserver2.hiveserver2_memory_pools_code_cache_usage
- mapr.hiveserver2.hiveserver2_memory_pools_compressed_class_space_usage
- mapr.hiveserver2.hiveserver2_memory_pools_metaspace_usage
- mapr.hiveserver2.hiveserver2_memory_pools_ps_eden_space_usage
- mapr.hiveserver2.hiveserver2_memory_pools_ps_old_gen_usage
- mapr.hiveserver2.hiveserver2_memory_pools_ps_survivor_space_usage
- mapr.hiveserver2.hiveserver2_memory_total_committed
- mapr.hiveserver2.hiveserver2_memory_total_init
- mapr.hiveserver2.hiveserver2_memory_total_max
- mapr.hiveserver2.hiveserver2_memory_total_used
- mapr.hiveserver2.hiveserver2_threads_blocked_count
- mapr.hiveserver2.hiveserver2_threads_count
- mapr.hiveserver2.hiveserver2_threads_daemon_count
- mapr.hiveserver2.hiveserver2_threads_deadlock_count
- mapr.hiveserver2.hiveserver2_threads_deadlocks
- mapr.hiveserver2.hiveserver2_threads_new_count
- mapr.hiveserver2.hiveserver2_threads_runnable_count
- mapr.hiveserver2.hiveserver2_threads_terminated_count
- mapr.hiveserver2.hiveserver2_threads_timed_waiting_count
- mapr.hiveserver2.hiveserver2_threads_waiting_count
- mapr.hiveserver2.hiveserver2_active_calls_api_get_all_databases
- mapr.hiveserver2.hiveserver2_active_calls_api_get_all_functions
- mapr.hiveserver2.hiveserver2_active_calls_api_get_all_tables

last-updated: Apr 11, 2022

- mapr.hiveserver2.hiveserver2_active_calls_api_get_database
- mapr.hiveserver2.hiveserver2_active_calls_api_get_multi_table
- mapr.hiveserver2.hiveserver2_active_calls_api_get_table_objects_by_name_req
- mapr.hiveserver2.hiveserver2_active_calls_api_init
- mapr.hiveserver2.hiveserver2_jvm_pause_extra_sleep_time
- mapr.hiveserver2.hiveserver2_open_connections
- mapr.hiveserver2.hiveserver2_api_get_all_databases_count
- mapr.hiveserver2.hiveserver2_api_get_all_databases_max
- mapr.hiveserver2.hiveserver2_api_get_all_databases_mean
- mapr.hiveserver2.hiveserver2_api_get_all_databases_min
- mapr.hiveserver2.hiveserver2_api_get_all_databases_p50
- mapr.hiveserver2.hiveserver2_api_get_all_databases_p75
- mapr.hiveserver2.hiveserver2_api_get_all_databases_p95
- mapr.hiveserver2.hiveserver2_api_get_all_databases_p98
- mapr.hiveserver2.hiveserver2_api_get_all_databases_p99
- mapr.hiveserver2.hiveserver2_api_get_all_databases_p999
- mapr.hiveserver2.hiveserver2_api_get_all_databases_stddev
- mapr.hiveserver2.hiveserver2_api_get_all_databases_m15_rate
- mapr.hiveserver2.hiveserver2_api_get_all_databases_m1_rate
- mapr.hiveserver2.hiveserver2_api_get_all_databases_m5_rate
- mapr.hiveserver2.hiveserver2_api_get_all_databases_mean_rate
- mapr.hiveserver2.hiveserver2_api_get_all_functions_count
- mapr.hiveserver2.hiveserver2_api_get_all_functions_max
- mapr.hiveserver2.hiveserver2_api_get_all_functions_mean
- mapr.hiveserver2.hiveserver2_api_get_all_functions_min
- mapr.hiveserver2.hiveserver2_api_get_all_functions_p50
- mapr.hiveserver2.hiveserver2_api_get_all_functions_p75
- mapr.hiveserver2.hiveserver2_api_get_all_functions_p95
- mapr.hiveserver2.hiveserver2_api_get_all_functions_p98
- mapr.hiveserver2.hiveserver2_api_get_all_functions_p99
- mapr.hiveserver2.hiveserver2_api_get_all_functions_p999

last-updated: Apr 11, 2022

- mapr.hiveserver2.hiveserver2_api_get_all_functions_stddev
- mapr.hiveserver2.hiveserver2_api_get_all_functions_m15_rate
- mapr.hiveserver2.hiveserver2_api_get_all_functions_m1_rate
- mapr.hiveserver2.hiveserver2_api_get_all_functions_m5_rate
- mapr.hiveserver2.hiveserver2_api_get_all_functions_mean_rate
- mapr.hiveserver2.hiveserver2_api_get_all_tables_count
- mapr.hiveserver2.hiveserver2_api_get_all_tables_max
- mapr.hiveserver2.hiveserver2_api_get_all_tables_mean
- mapr.hiveserver2.hiveserver2_api_get_all_tables_min
- mapr.hiveserver2.hiveserver2_api_get_all_tables_p50
- mapr.hiveserver2.hiveserver2_api_get_all_tables_p75
- mapr.hiveserver2.hiveserver2_api_get_all_tables_p95
- mapr.hiveserver2.hiveserver2_api_get_all_tables_p98
- mapr.hiveserver2.hiveserver2_api_get_all_tables_p99
- mapr.hiveserver2.hiveserver2_api_get_all_tables_p999
- mapr.hiveserver2.hiveserver2_api_get_all_tables_stddev
- mapr.hiveserver2.hiveserver2_api_get_all_tables_m15_rate
- mapr.hiveserver2.hiveserver2_api_get_all_tables_m1_rate
- mapr.hiveserver2.hiveserver2_api_get_all_tables_m5_rate
- mapr.hiveserver2.hiveserver2_api_get_all_tables_mean_rate
- mapr.hiveserver2.hiveserver2_api_get_multi_table_count
- mapr.hiveserver2.hiveserver2_api_get_multi_table_max
- mapr.hiveserver2.hiveserver2_api_get_multi_table_mean
- mapr.hiveserver2.hiveserver2_api_get_multi_table_min
- mapr.hiveserver2.hiveserver2_api_get_multi_table_p50
- mapr.hiveserver2.hiveserver2_api_get_multi_table_p75
- mapr.hiveserver2.hiveserver2_api_get_multi_table_p95
- mapr.hiveserver2.hiveserver2_api_get_multi_table_p98
- mapr.hiveserver2.hiveserver2_api_get_multi_table_p99
- mapr.hiveserver2.hiveserver2_api_get_multi_table_p999
- mapr.hiveserver2.hiveserver2_api_get_multi_table_stddev

last-updated: Apr 11, 2022

- mapr.hiveserver2.hiveserver2_api_get_multi_table_m15_rate
- mapr.hiveserver2.hiveserver2_api_get_multi_table_m1_rate
- mapr.hiveserver2.hiveserver2_api_get_multi_table_m5_rate
- mapr.hiveserver2.hiveserver2_api_get_multi_table_mean_rate
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_count
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_max
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_mean
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_min
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_p50
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_p75
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_p95
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_p98
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_p99
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_p999
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_stddev
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_m15_rate
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_m1_rate
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_m5_rate
- mapr.hiveserver2.hiveserver2_api_get_table_objects_by_name_req_mean_rate
- mapr.hiveserver2.hiveserver2_api_init_count
- mapr.hiveserver2.hiveserver2_api_init_max
- mapr.hiveserver2.hiveserver2_api_init_mean
- mapr.hiveserver2.hiveserver2_api_init_min
- mapr.hiveserver2.hiveserver2_api_init_p50
- mapr.hiveserver2.hiveserver2_api_init_p75
- mapr.hiveserver2.hiveserver2_api_init_p95
- mapr.hiveserver2.hiveserver2_api_init_p98
- mapr.hiveserver2.hiveserver2_api_init_p99
- mapr.hiveserver2.hiveserver2_api_init_p999
- mapr.hiveserver2.hiveserver2_api_init_stddev
- mapr.hiveserver2.hiveserver2_api_init_m15_rate

- mapr.hiveserver2.hiveserver2_api_init_m1_rate
- mapr.hiveserver2.hiveserver2_api_init_m5_rate
- mapr.hiveserver2.hiveserver2_api_init_mean_rate

Load Metrics
Every 10 seconds, the collectd service uses the load plugin to gather the following load metrics on each node in the cluster.

Name	Description
load.load.longterm	The number of tasks running on the node every 15 minutes.
load.load.midterm	The number of tasks running on the node every 10 minutes.
load.load.shortterm	The number of tasks running on the node every minute.

MapR Alarm Metrics
Every 10 seconds, the collectd service uses a plugin to gather the cluster alarms.

Name	Description	Additional Tag(s)
mapr.alarms.alarm_raised	The number of alarms raised. The timestamp for each alarm is based on the time that MapR raised the alarm, not the time when the collectd service gathered the alarm data.	• `alarm_name:` Display values for a specified alarm name. • `alarm_entity:.` Display values for a specified volume, node, user, or group.

MapR Cache Metrics
Every 10 seconds, the collectd service uses a plugin to gather the following MapR filesystem cache metrics on each node in the cluster.

Name	Description
mapr.cache.lookups_data	The number of cache lookups in the block cache.
mapr.cache.lookups_dir	The number of cache lookups in the table LRU cache. The table LRU is used for storing internal B-Tree leaf pages.
mapr.cache.lookups_inode	The number of cache lookups in the inode cache.
mapr.cache.lookups_largefile	The number of cache lookups in the large file LRU cache. The large file LRU is used for storing files with size greater than 64K and also MapR Database data pages.
mapr.cache.lookups_meta	The number of cache lookups on the meta LRU cache. The meta LRU is used for storing internal B-Tree pages.
mapr.cache.lookups_smallfile	The number of cache lookups on the small file LRU cache. This LRU is used for storing files with size less than 64K and also MapR Database index pages.
mapr.cache.lookups_table	The number of cache lookups in the table LRU cache. The table LRU is used for storing internal B-Tree leaf pages.
mapr.cache.misses_data	The number of cache misses in the block cache.
mapr.cache.misses_dir	The number of cache misses on the table LRU cache.
mapr.cache.misses_inode	The number of cache misses in the inode cache.

Name	Description
mapr.cache.misses_largefile	The number of cache misses on the large file LRU cache.
mapr.cache.misses_meta	The number of cache misses on the meta LRU cache.
mapr.cache.misses_smallfile	The number of cache misses on the small file LRU cache.
mapr.cache.misses_table	The number of cache misses on the table LRU cache.

MapR CLDB Metrics

Every 10 seconds, the collectd service uses a MapR plugin to gather the following CLDB metrics on the primary CLDB node in the cluster.

Name	Description
mapr.cldb.cluster_cpu_total	The number of physical CPUs in the cluster.
mapr.cldb.cluster_cpubusy_percent	The aggregate percentage of busy CPUs in the cluster.
mapr.cldb.cluster_disk_capacity	The storage capacity for MapR disks in GB.
mapr.cldb.cluster_diskspace_used	The amount of MapR disks used in GB.
mapr.cldb.cluster_memory_capacity	The memory capacity in MB.
mapr.cldb.cluster_memory_used	The amount of used memory in MB.
mapr.cldb.containers	The number of containers currently in the cluster.
mapr.cldb.containers_created	The cumulative number of containers created in the cluster. This value includes containers that have been deleted.
mapr.cldb.containers_unusable	The number of containers that are no longer usable. The CLDB marks a container as unusable when the node that stores the container is offline for 1 hour or more.
mapr.cldb.disk_space_available	The amount of disk space available in GB.
mapr.cldb.nodes_in_cluster	The number of nodes in the cluster.
mapr.cldb.nodes_offline	The number of nodes in the cluster that are offline.
mapr.cldb.rpc_received	The number of RPCs received.
mapr.cldb.rpcs_failed	The number of RPCs failed.
mapr.cldb.storage_pools_cluster	The number of storage pools.
mapr.cldb.storage_pools_offline	The number of offline storage pools.
mapr.cldb.volumes	The number of volumes created, including system volumes.

MapR Database Metrics

Every 10 seconds, the collectd service uses a plugin to gather MapR Database metrics on each node in the cluster. MapR Database provides both node and table metrics.

Node metrics capture data for operations across a MapR node. You can use them to assess the performance of individual nodes in your MapR cluster.

Starting in MapR 6.1, MapR Database supports table metrics. Table metrics provide more granular metrics. They allow you detect and diagnose bottlenecks and performance issues that are specific to individual tables. For example, suppose a node metric shows a spike in a particular RPC. Knowing this, you can use the corresponding table metric to determine if the spike originates from a single table.

Examples of other use cases that benefit from table metrics are the following:

last-updated: Apr 11, 2022

- You want to measure the latency of different RPC operations on a table
- You want to determine which operations on a table are slow
- You want to determine which tables are most frequently accessed

MapR Database Node Metrics
This section describes the available MapR Database node metrics.

The following table lists MapR Database node metrics:

Metric Category	Name	Description
Throughput - RPC counts	`mapr.db.append_rpcs`	The number of MapR Database append RPCs completed
	`mapr.db.checkandput_rpcs` [1]	The number of MapR Database check and put RPCs completed
	`mapr.db.get_currpcs`	The number of MapR Database get RPCs in progress
	`mapr.db.get_rpcrows`	The number of get rows completed. Each get RPC can include multiple get rows.
	`mapr.db.get_rpcs`	The number of MapR Database get RPCs completed
	`mapr.db.increment_rpcs` [1]	The number of MapR Database increment RPCs completed
	`mapr.db.put_currpcs`	The number of MapR Database put RPCs in progress
	`mapr.db.put_rpcs`	The number of MapR Database put RPCs completed
	`mapr.db.scan_currpcs`	The number of MapR Database scan RPCs in progress
	`mapr.db.scan_rpcrows`	The number of scan rows completed. Each scan RPC can include multiple scan rows.
	`mapr.db.scan_rpcs`	The number of MapR Database scan RPCs completed
	`mapr.db.updateandget_rpcs`	The number of MapR Database update and get RPCs completed
Throughput - Row count written	`mapr.db.append_rpcrows` [1]	The number of rows written by append RPCs
	`mapr.db.checkandput_rpcrows` [1]	The number of rows written by check and put RPCs
	`mapr.db.increment_rpcrows` [1]	The number of rows written by increment RPCs
	`mapr.db.put_rpcrows`	The number of rows written by put RPCs. Each MapR Database put RPC can include multiple put rows.
	`mapr.db.updateandget_rpcrows` [1]	The number of rows written by update and get RPCs
Throughput - Rows returned	`mapr.db.get_resprows` [1]	The number of rows returned from get RPCs
	`mapr.db.scan_resprows` [1]	The number of rows returned from scan RPCs

last-updated: Apr 11, 2022

Metric Category	Name	Description
Throughput - Row count read	`mapr.db.get_readrows` [1]	The number of rows read by get RPCs
	`mapr.db.put_readrows` [1]	The number of rows read by put RPCs
	`mapr.db.scan_readrows` [1]	The number of rows read by scan RPCs
Throughput - Bytes written	`mapr.db.append_bytes` [1]	The number of bytes written by append RPCs
	`mapr.db.checkandput_bytes` [1]	The number of bytes written by check and put RPCs
	`mapr.db.put_bytes`	The number of bytes written by put RPCs
	`mapr.db.increment_bytes` [1]	The number of bytes written by increment RPCs
	`mapr.db.updateandget_bytes` [1]	The number of bytes written by update and get RPCs
Throughput - Bytes read	`mapr.db.get_bytes` [1]	The number of bytes read by get RPCs
	`mapr.db.scan_bytes` [1]	The number of bytes read by scan RPCs
Value cache usage	`mapr.db.valuecache_hits`	The number of MapR Database operations that utilized the MapR Database value cache
	`mapr.db.valuecache_lookups`	The number of MapR Database operations that performed a lookup on the MapR Database value cache
	`mapr.db.valuecache_usedSize`	The MapR Database value cache size in MB
Compactions	`mapr.db.fullcompacts` [1]	The number of compactions that combine multiple MapR Database data files containing sorted data (known as spills) into a single spill file. MapR Database creates a spill file each time it flushes files containing unsorted data (known as buckets). Full compactions improve read performance because after compaction, MapR Database needs to read only the single resulting sorted spill file. But they incur I/O costs because the compaction must read, sort, and rewrite all data in the spill files.
	`mapr.db.minicompacts` [1]	The number of compactions that combine multiple small data files containing sorted data (known as spills) into a single spill file. MapR Database creates a spill file each time it flushes files containing unsorted data (known as buckets). After a mini compaction, MapR Database needs to read only two spill files.
	`mapr.db.ttlcompacts` [1]	The number of compactions that result in reclamation of disk space due to removal of stale data. You can configure the TTL for a table if it has only a default column family. See table cf edit on page 1806 for details.

Metric Category	Name	Description
Table flushes	`mapr.db.flushes` [1]	The number of flushes that reorganize data from bucket files (unsorted data) to spill files (sorted data) when the bucket size exceeds a threshold
	`mapr.db.forceflushes` [1]	The number of flushes that reorganize data from bucket files (unsorted data) to spill files (sorted data) when the in-memory bucket file cache fills up
CDC - Data sent	`mapr.db.cdc.sent_bytes` [1]	The number of bytes of CDC data sent
	`mapr.db.cdc.sent_rows` [1]	The number of rows of CDC data sent
Secondary indexes - Data sent	`mapr.db.index.sent_bytes` [1]	The number of bytes sent for secondary index updates
	`mapr.db.index.sent_rows` [1]	The number of rows sent for secondary index updates
Replication - Data sent	`mapr.db.repl.sent_bytes` [1]	The number of bytes sent to replicate data
	`mapr.db.repl.sent_rows` [1]	The number of rows sent to replicate data
CDC - Data pending	`mapr.db.cdc.pending_bytes` [1]	The number of bytes of CDC data remaining to be sent
	`mapr.db.cdc.pending_rows` [1]	The number of rows of CDC data remaining to be sent
Secondary indexes - Data pending	`mapr.db.index.pending_bytes` [1]	The number of bytes of secondary index data remaining to be sent
	`mapr.db.index.pending_rows` [1]	The number of rows of secondary index data remaining to be sent
Replication - Data pending	`mapr.db.repl.pending_bytes` [1]	The number of bytes of replication data remaining to be sent
	`mapr.db.repl.pending_rows` [1]	The number of rows of replication data remaining to be sent

[1] Available starting in MapR 6.1

MapR Database Table Metrics

Starting in MapR 6.1, MapR Database supports table metrics. This section describes the available MapR Database table metrics. It also describes how to configure metrics per table, disable them in your cluster, and how to filter them by operation and table.

Table metrics are collected per node. By default, these metrics are written to OpenTSDB every minute. Pre-existing tables in your MapR cluster inherit this default setting. You can change this default per table using either the table create on page 1788 or table edit on page 1822 command. Secondary indexes inherit their metric configuration from their parent table.

You cannot disable metrics on individual tables. During installation, you can disable table metrics across your entire cluster. If you are using the MapR installer, select the minimum metrics collection configuration option. If you are doing a manual install, see Installing Metering Using Manual Steps on page 284.

The following table lists MapR Database table metrics:

mapr.db.table.latency	The latency of RPC operations on tables, represented as a histogram. Endpoints identify histogram bucket boundaries.

Note: Put latency metrics for indexes reflect only the index update time. They do not include lag time due to asynchronous index updates.

mapr.db.table.read_bytes	The number of bytes read from tables.
mapr.db.table.read_rows	The number of rows read from tables.
mapr.db.table.resp_rows	The number of rows returned from tables.
mapr.db.table.rpcs	The number of RPC calls completed on tables.
mapr.db.table.value_cache_hits	The number of MapR Database operations on tables that utilized the MapR Database value cache.
mapr.db.table.value_cache_lookups	The number of MapR Database operations on tables that performed a lookup on the MapR Database value cache.
mapr.db.table.write_rows	The number of rows written to tables.
mapr.db.table.write_bytes	The number of bytes written to tables.

Tags for Table Metrics

Each table metric collects data across all operations on tables and secondary indexes in a MapR node. To view metrics for a particular RPC operation on a specific table, you must filter the metric by RPC type and table path, as described in the following list:

rpc_type	The name of the RPC: • `append` • `check_and_put` • `get` • `increment` • `put` • `scan` • `update_and_get`
table_path	The path of the MapR Database table. OpenTSDB supports only the following characters: • Alphanumeric characters • Dot (.) • Underscore (_) • Dash (-) • Slash (/) • Space ( )

- Unicode letters

If a table path contains characters that are not in this list, MapR cannot collect those table metrics.

index

The name of the secondary index defined on the table specified in `table_path`, if you want to filter on a specific index. When you specify this tag, you cannot specify the `noindex` tag.

Note: You can use regular expressions when specifying the index name to filter for a group of indexes.

noindex

Set to `"//primary"` if you want to filter metrics for only the primary table and exclude metrics for secondary indexes defined on the table. When you specify this tag, you cannot specify the `index` tag.

The following list shows examples of the table-specific tags to use to filter different table metrics:

mapr.db.table.write_rows

Tag Name: `table_path`

Tag Value: `/var/mapr/mapr.monitoring/tsdb-meta`

Description: The number of rows written to a table in the path `/var/mapr/mapr.monitoring/tsdb-meta`, including writes to its secondary indexes.

mapr.db.table.rpcs

- *Tag Name*: `rpc_type`
  - *Tag Value*: `get`
- *Tag Name*: `table_path`
  - *Tag Value*: `/var/mapr/mapr.monitoring/tsdb-meta`
- *Tag Name*: `noindex`
  - *Tag Value*: `//primary`

Description: The number of completed *get* RPCs for a table in the path `/var/mapr/mapr.monitoring/tsdb-meta`, excluding metrics on the table's secondary indexes.

mapr.db.table.read_rows

- *Tag Name*: `table_path`
  - *Tag Value*: `/var/mapr/mapr.monitoring/tsdb-meta`
- *Tag Name*: `index`
  - *Tag Value*: `tsdbIdx`

Description: The number of rows read from a secondary index named `tsdbIdx`, defined on the table in the path `/var/mapr/mapr.monitoring/tsdb-meta`.

last-updated: Apr 11, 2022

Note: In addition to these table-specific tags, Metric Collection on page 1334 describes other available tags.

MapR Event Store For Apache Kafka Metrics
Every 10 seconds, the collectd service uses a plugin to gather the following Streams metrics on each node in the cluster.

Warning: MapR monitoring uses 2 MB disk space per minute per node when MapR Event Store For Apache Kafka metrics is enabled. This is approximately 3 GB per day on a single node or 7 GB per node per day with a 3X replication. This stream metrics data is automatically deleted every 3 days.

Name	Description
mapr.streams.produce_rpcs	The number of Streams producer RPCs. This metric is available as of EEP 2.0.
mapr.streams.produce_msgs	The number of Streams messages produced. This metric is available as of EEP 2.0.
mapr.streams.produce_bytes	The number of megabytes produced by Streams messages. This metric is available as of EEP 2.0.
mapr.streams.listen_rpcs	The number of Streams consumer RPCs. This metric is available as of EEP 2.0.
mapr.streams.listen_msgs	The number of Streams messages read by the consumer. This metric is available as of EEP 4.0. **Note:** If you upgrade to EEP 4.0, all Streams messages are consumed, however this metric only reports a count on messages that were produced in EEP 4.0. Messages produced prior to EEP 4.0 are consumed, but not counted. In this scenario, the metric reports a partial count or a zero count if all messages were produced prior to EEP 4.0.
mapr.streams.listen_currpcs	The number of concurrent Stream consumer RPCs. This metric is available as of EEP 2.0.
mapr.streams.listen_bytes	The number of megabytes consumed by Streams messages. This metric is available as of EEP 2.0.

MapR File System Metrics
Every 10 seconds, the collectd service uses a MapR plugin to gather the following MapR File System metrics on each node in the cluster.

Name	Description
mapr.fs.bulk_writes	The number of bulk-write operations. Bulk-write operations occur when the primary MapR File System container aggregates multiple small file writes from one or more clients into one big RPC, before replicating the writes.
mapr.fs.bulk_writesbytes	The amount of MB written by bulk-write operations. Bulk-write operations occur when the primary MapR File System container aggregates multiple small file writes from one or more clients into one big RPC, before replicating the writes.
mapr.fs.kvstore_delete	The number of delete operations on key-value store files which are used by the CLDB and MapR Database.

last-updated: Apr 11, 2022

Name	Description
mapr.fs.kvstore_insert	The number of insert operations on key-value store files which are used by the CLDB and MapR Database.
mapr.fs.kvstore_lookup	The number of lookup operations on key-value store files which are used by the CLDB and MapR Database.
mapr.fs.kvstore_scan	The number of scan operations on key-value store files which are used by the CLDB and MapR Database.
mapr.fs.local_readbytes	The amount of MB read by applications that are running on the MapR File System node where the data resides.
mapr.fs.local_reads	The number of file read operations by applications that are running on the MapR File System node where the data resides.
mapr.fs.local_writebytes	The amount of MB written by applications that are running on the MapR File System node where the data resides.
mapr.fs.local_writes	The number of file write operations by applications that are running on the MapR File System node where the data resides.
mapr.fs.read_bytes	The amount of data (in MB) read remotely.
mapr.fs.read_cachehits	The number of cache hits for file reads. This value includes pages that MapR File System populates using the readahead mechanism.
mapr.fs.read_cachemisses	The number of cache misses for file read operations.
mapr.fs.reads	The number of remote reads.
mapr.fs.statstype_create	The number of file create operations.
mapr.fs.statstype_lookup	The number of lookup operations.
mapr.fs.statstype_read	The number of file read operations.
mapr.fs.statstype_write	The number of file write operations.
mapr.fs.write_bytes	The amount of data (in MB) written remotely.
mapr.fs.writes	The number of remote writes.

MapR Process Metrics

Every 10 seconds, the collectd service uses a plugin to gather the following MapR process metrics on each node in the cluster.

Name	Description	Additional Tag(s)
mapr.process.context_switch_involuntary	The number of involuntary context switches for MapR processes. An involuntary context switch occurs when a process consumes more CPU time than what it was allocated by the kernel. When an involuntary context switch occurs, the kernel stops a process to provide resources to another process.	• `process_name`: Display values for a specified process. Process values: hoststats,mfs,warden,.etc.

Name	Description	Additional Tag(s)
mapr.process.context_switch_voluntary	The number of voluntary context switches for MapR processes. A voluntary context switch occurs when a process does not require the entire CPU time it was allocated by the kernel or when a process is suspended. When a voluntary context switch occurs, the kernel provides CPU resources to another process.	• `process_name`: Display values for a specified process. Process values: hoststats,mfs,warden,.etc.
mapr.process.cpu_percent	The percentage of CPU used for MapR processes.	• `process_name`: Display values for a specified process. Process values: hoststats,mfs,warden,.etc.
mapr.process.cpu_time.syst	The amount of time, measured in seconds, that the process has been in kernel mode.	
mapr.process.cpu_time.user	The amount of time, measured in seconds, that the process has been in user mode	
mapr.process.data	The amount memory, in MB, used by the data segments of MapR processes.	• `process_name`: Display values for a specified process. Process values: hoststats,mfs,warden,.etc.
mapr.process.disk_octets.read	The number of bytes read from disk for MapR processes.	
mapr.process.disk_octets.write	The number of bytes written to disk for MapR processes.	
mapr.process.disk_ops.read	The number of read operations for MapR processes.	
mapr.process.disk_ops.write	The number of write operations for MapR processes.	
mapr.process.mem_percent	The percentage of total system memory (not capped by MapR processes) used for MapR processes.	• `process_name`: Display values for a specified process. Process values: hoststats,mfs,warden,.etc.
mapr.process.page_faults.majflt	The number of major MapR process faults that required loading a memory page from disk.	• `process_name`: Display values for a specified process. Process values: hoststats,mfs,warden,.etc.
mapr.process.page_faults.minflt	The number of minor MapR process faults that required loading a memory page from disk.	• `process_name`: Display values for a specified process. Process values: hoststats,mfs,warden,.etc.
mapr.process.rss	The actual amount of memory, in MB, used by MapR processes.	• `process_name`: Display values for a specified process. Process values: hoststats,mfs,warden,.etc.
mapr.process.vm	The amount of virtual memory, in MB, used by MapR processes.	• `process_name`: Display values for a specified process. Process values: hoststats,mfs,warden,.etc.

MapR I/O Metrics
Every 10 seconds, the collectd service uses a plugin to gather the following MapR I/O metrics on each node in the cluster.

Name	Description
mapr.io.read_bytes	The number of MB read from disk.
mapr.io.reads	The number of MapR File System disk read operations.
mapr.io.write_bytes	The number of MB written to disk.
mapr.io.writes	The number of MapR File System disk write operations.

MapR RPC Metric
Every 10 seconds, the collectd service uses a plugin to gather the following MapR RPC metrics on each node in the cluster.

Name	Description
mapr.rpc.bytes_recd	The number of bytes received by the MapR File System over RPC.
mapr.rpc.bytes_sent	The number of bytes sent by the MapR File System over RPC.
mapr.rpc.calls_recd	The number of RPC calls received by the MapR File System.

Spark JMX Metrics
Every 10 seconds, the `collectd` service gathers the following Spark JMX metrics on each node in the cluster.

For detailed information about Spark metrics, see the Apache Spark documentation.

- mapr.spark.driver_block_manager_disk_space_used_mb
- mapr.spark.driver_block_manager_memory_max_mem_mbb
- mapr.spark.driver_block_manager_memory_max_off_heap_mem_mb
- mapr.spark.driver_block_manager_memory_max_on_heap_mem_mb
- mapr.spark.driver_block_manager_memory_mem_used_mb
- mapr.spark.driver_block_manager_memory_off_heap_mem_used_mb
- mapr.spark.driver_block_manager_memory_on_heap_mem_used_mb
- mapr.spark.driver_block_manager_memory_remaining_mem_mb
- mapr.spark.driver_block_manager_memory_remaining_off_heap_mem_mb
- mapr.spark.driver_block_manager_memory_remaining_on_heap_mem_mb
- mapr.spark.driver_dag_scheduler_job_active_jobs
- mapr.spark.driver_dag_scheduler_job_all_jobs
- mapr.spark.driver_dag_scheduler_stage_failed_stages
- mapr.spark.driver_dag_scheduler_stage_running_stages
- mapr.spark.driver_dag_scheduler_stage_waiting_stages

- mapr.spark.driver_live_listener_bus_queue_app_status_size
- mapr.spark.driver_live_listener_bus_queue_event_log_size
- mapr.spark.driver_live_listener_bus_queue_executor_management_size
- mapr.spark.driver_hive_external_catalog_file_cache_hits
- mapr.spark.driver_hive_external_catalog_files_discovered
- mapr.spark.driver_hive_external_catalog_hive_client_calls
- mapr.spark.driver_hive_external_catalog_parallel_listing_job_count
- mapr.spark.driver_hive_external_catalog_partitions_fetched
- mapr.spark.driver_live_listener_bus_num_events_posted
- mapr.spark.driver_live_listener_bus_queue_app_status_num_dropped_events
- mapr.spark.driver_live_listener_bus_queue_event_log_num_dropped_events
- mapr.spark.driver_live_listener_bus_queue_executor_management_num_dropped_events
- mapr.spark.driver_code_generator_compilation_time_count
- mapr.spark.driver_code_generator_compilation_time_max
- mapr.spark.driver_code_generator_compilation_time_mean
- mapr.spark.driver_code_generator_compilation_time_min
- mapr.spark.driver_code_generator_compilation_time_p50
- mapr.spark.driver_code_generator_compilation_time_p75
- mapr.spark.driver_code_generator_compilation_time_p95
- mapr.spark.driver_code_generator_compilation_time_p98
- mapr.spark.driver_code_generator_compilation_time_p99
- mapr.spark.driver_code_generator_compilation_time_p999
- mapr.spark.driver_code_generator_compilation_time_stddev
- mapr.spark.driver_code_generator_generated_class_size_count
- mapr.spark.driver_code_generator_generated_class_size_max
- mapr.spark.driver_code_generator_generated_class_size_mean
- mapr.spark.driver_code_generator_generated_class_size_min
- mapr.spark.driver_code_generator_generated_class_size_p50
- mapr.spark.driver_code_generator_generated_class_size_p75
- mapr.spark.driver_code_generator_generated_class_size_p95
- mapr.spark.driver_code_generator_generated_class_size_p98

- mapr.spark.driver_code_generator_generated_class_size_p99
- mapr.spark.driver_code_generator_generated_class_size_p999
- mapr.spark.driver_code_generator_generated_class_size_stddev
- mapr.spark.driver_code_generator_generated_method_size_count
- mapr.spark.driver_code_generator_generated_method_size_max
- mapr.spark.driver_code_generator_generated_method_size_mean
- mapr.spark.driver_code_generator_generated_method_size_min
- mapr.spark.driver_code_generator_generated_method_size_p50
- mapr.spark.driver_code_generator_generated_method_size_p75
- mapr.spark.driver_code_generator_generated_method_size_p95
- mapr.spark.driver_code_generator_generated_method_size_p98
- mapr.spark.driver_code_generator_generated_method_size_p99
- mapr.spark.driver_code_generator_generated_method_size_p999
- mapr.spark.driver_code_generator_generated_method_size_stddev
- mapr.spark.driver_code_generator_source_code_size_count
- mapr.spark.driver_code_generator_source_code_size_max
- mapr.spark.driver_code_generator_source_code_size_mean
- mapr.spark.driver_code_generator_source_code_size_min
- mapr.spark.driver_code_generator_source_code_size_p50
- mapr.spark.driver_code_generator_source_code_size_p75
- mapr.spark.driver_code_generator_source_code_size_p95
- mapr.spark.driver_code_generator_source_code_size_p98
- mapr.spark.driver_code_generator_source_code_size_p99
- mapr.spark.driver_code_generator_source_code_size_p999
- mapr.spark.driver_code_generator_source_code_size_stddev
- mapr.spark.driver_dag_scheduler_message_processing_time_count
- mapr.spark.driver_dag_scheduler_message_processing_time_max
- mapr.spark.driver_dag_scheduler_message_processing_time_mean
- mapr.spark.driver_dag_scheduler_message_processing_time_min
- mapr.spark.driver_dag_scheduler_message_processing_time_p50
- mapr.spark.driver_dag_scheduler_message_processing_time_p75

last-updated: Apr 11, 2022

- mapr.spark.driver_dag_scheduler_message_processing_time_p95
- mapr.spark.driver_dag_scheduler_message_processing_time_p98
- mapr.spark.driver_dag_scheduler_message_processing_time_p99
- mapr.spark.driver_dag_scheduler_message_processing_time_p999
- mapr.spark.driver_dag_scheduler_message_processing_time_stddev
- mapr.spark.driver_dag_scheduler_message_processing_time_m15_rate
- mapr.spark.driver_dag_scheduler_message_processing_time_m1_rate
- mapr.spark.driver_dag_scheduler_message_processing_time_m5_rate
- mapr.spark.driver_dag_scheduler_message_processing_time_mean_rate
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_count
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_max
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_mean
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_min
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_p50
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_p75
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_p95
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_p98
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_p99
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_p999
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_stddev
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_m15_rate
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_m1_rate
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_m5_rate
- mapr.spark.driver_live_listener_bus_listener_processing_time_heartbeat_receiver_mean_rate
- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_count
- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_max
- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_mean
- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_min
- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_p50
- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_p75
- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_p95

last-updated: Apr 11, 2022

- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_p98
- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_p99
- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_p999
- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_stddev
- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_m15_rate
- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_m1_rate
- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_m5_rate
- mapr.spark.driver_live_listener_bus_listener_processing_time_event_logging_listener_mean_rate
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_count
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_max
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_mean
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_min
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_p50
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_p75
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_p95
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_p98
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_p99
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_p999
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_stddev
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_m15_rate
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_m1_rate
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_m5_rate
- mapr.spark.driver_live_listener_bus_listener_processing_time_app_status_listener_mean_rate
- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_count
- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_max
- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_mean
- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_min
- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_p50
- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_p75
- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_p95
- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_p98

last-updated: Apr 11, 2022

- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_p99
- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_p999
- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_stddev
- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_m15_rat
- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_m1_rate
- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_m5_rate
- mapr.spark.driver_live_listener_bus_queue_app_status_listener_processing_time_mean_rate
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_count
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_max
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_mean
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_min
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_p50
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_p75
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_p95
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_p98
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_p99
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_p999
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_stddev
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_m15_rate
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_m1_rate
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_m5_rate
- mapr.spark.driver_live_listener_bus_queue_event_log_listener_processing_time_mean_rate
- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_count
- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_max
- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_mean
- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_min
- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_p50
- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_p75
- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_p95
- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_p98
- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_p99

- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_p990
- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_stddev
- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_m15_rate
- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_m1_rate
- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_m5_rate
- mapr.spark.driver_live_listener_bus_queue_executor_management_listener_processing_time_mean_rate
- mapr.spark.driver_jvm_ps_mark_sweep_count
- mapr.spark.driver_jvm_ps_mark_sweep_time
- mapr.spark.driver_jvm_ps_scavenge_count
- mapr.spark.driver_jvm_ps_scavenge_time
- mapr.spark.driver_jvm_direct_capacity
- mapr.spark.driver_jvm_direct_count
- mapr.spark.driver_jvm_direct_used
- mapr.spark.driver_jvm_direct_heap_committed
- mapr.spark.driver_jvm_direct_heap_init
- mapr.spark.driver_jvm_direct_heap_max
- mapr.spark.driver_jvm_direct_heap_usage
- mapr.spark.driver_jvm_direct_heap_used
- mapr.spark.driver_jvm_direct_mapped_capacity
- mapr.spark.driver_jvm_direct_mapped_count
- mapr.spark.driver_jvm_direct_mapped_used
- mapr.spark.driver_jvm_direct_non_heap_committed
- mapr.spark.driver_jvm_direct_non_heap_init
- mapr.spark.driver_jvm_direct_non_heap_max
- mapr.spark.driver_jvm_direct_non_heap_usage
- mapr.spark.driver_jvm_direct_non_heap_used
- mapr.spark.driver_jvm_pools_code_cache_committed
- mapr.spark.driver_jvm_pools_code_cache_init
- mapr.spark.driver_jvm_pools_code_cache_max
- mapr.spark.driver_jvm_pools_code_cache_usage

last-updated: Apr 11, 2022

- mapr.spark.driver_jvm_pools_code_cache_used
- mapr.spark.driver_jvm_pools_compressed_class_space_committed
- mapr.spark.driver_jvm_pools_compressed_class_space_init
- mapr.spark.driver_jvm_pools_compressed_class_space_max
- mapr.spark.driver_jvm_pools_compressed_class_space_usage
- mapr.spark.driver_jvm_pools_compressed_class_space_used
- mapr.spark.driver_jvm_pools_metaspace_committed
- mapr.spark.driver_jvm_pools_metaspace_init
- mapr.spark.driver_jvm_pools_metaspace_max
- mapr.spark.driver_jvm_pools_metaspace_usage
- mapr.spark.driver_jvm_pools_metaspace_used
- mapr.spark.driver_jvm_pools_ps_eden_space_committed
- mapr.spark.driver_jvm_pools_ps_eden_space_init
- mapr.spark.driver_jvm_pools_ps_eden_space_max
- mapr.spark.driver_jvm_pools_ps_eden_space_usage
- mapr.spark.driver_jvm_pools_ps_eden_space_used
- mapr.spark.driver_jvm_pools_ps_old_gen_committed
- mapr.spark.driver_jvm_pools_ps_old_gen_init
- mapr.spark.driver_jvm_pools_ps_old_gen_max
- mapr.spark.driver_jvm_pools_ps_old_gen_usage
- mapr.spark.driver_jvm_pools_ps_old_gen_used
- mapr.spark.driver_jvm_pools_ps_survivor_space_committed
- mapr.spark.driver_jvm_pools_ps_survivor_space_init
- mapr.spark.driver_jvm_pools_ps_survivor_space_max
- mapr.spark.driver_jvm_pools_ps_survivor_space_usage
- mapr.spark.driver_jvm_pools_ps_survivor_space_used
- mapr.spark.driver_jvm_total_committed
- mapr.spark.driver_jvm_total_init
- mapr.spark.driver_jvm_total_max
- mapr.spark.driver_jvm_total_used

MapR Topology Metrics

Every 60 seconds, the collectd service uses a plugin to gather the following topology metrics on each node in the cluster. Use these metrics to understand disk utilization across a topology or rack. By default, these metrics include all racks and topologies associated with the cluster. However, you can use tags to specify which rack(s) or topologies(s) to include. **Note:** Racks and topologies can span multiple nodes and one rack can be associated with multiple topologies.

Name	Description	Tag
mapr.topology.disks_total_capacity	The disk capacity in gigabytes. This metric is available as of EEP 3.0.	• `rack_name:` Display values for a specified rack. • `topology_name`: Display values for a specified topology. Provide the full topology path.
mapr.topology.disks_used_capacity	The amount disk space used in gigabytes. This metric is available as of EEP 3.0.	• `rack_name:` Display values for a specified rack. • `topology_name`: Display values for a specified topology. Provide the full topology path.
mapr.topology.utilization	The aggregate percentage of CPU utilization. This metric is available as of EEP 3.0.	• `rack_name:` Display values for a specified rack. • `topology_name`: Display values for a specified topology. Provide the full topology path.

MapR Volume Metrics

Every 10 seconds, the collectd service uses a plugin to gather the following MapR volume metrics on each CLDB node in the cluster.

For volumes prefixed with `mapr.internal*`, the reported volume metrics are not meaningful.

Name	Description	Tag(s)
mapr.volume.logical_used	The number of MBs used for logical volumes before compression is applied to the files.	• `volume_name`: Display values for a specified volume. • `entity_name`: Display values for a specified user or group.
mapr.volume.snapshot_used	The number of MBs used for snapshots.	• `volume_name`: Display values for a specified volume. • `entity_name`: Display values for a specified user or group.
mapr.volume.total_used	The number of MB used for volumes and snapshots.	• `fqdn`: Display values for a specified node. • `volume_name`: Display values for a specified volume. • `entity_name`: Display values for a specified user or group.

last-updated: Apr 11, 2022

Name	Description	Tag(s)
mapr.volume.used	The number of MB used for volumes after compression is applied to the files.	• `fqdn`: Display values for a specified node. • `volume_name`: Display values for a specified volume. • `entity_name`: Display values for a specified user or group.
mapr.volume.quota	The number of megabytes(MB) used for volume quota.	• `fqdn`: Display values for a specified node. • `volume_name`: Display values for a specified volume. • `entity_name`: Display values for a specified user or group.

Every 10 seconds, the collectd service uses a plugin to gather the following MapR volume metrics on each CLDB node in the cluster.

Name	Description	Tag(s)
mapr.volmetrics.read_throughput	The per volume read throughput in KB	• `fqdn`: Display values for a specified node. • `volume_name`: Display values for a specified volume. • `clusterid`: Display values for a cluster specified by ID. • `clustername`: Display values for a cluster specified by name.
mapr.volmetrics.write_throughput	The per volume write throughput in KB	• `fqdn`: Display values for a specified node. • `volume_name`: Display values for a specified volume. • `clusterid`: Display values for a cluster specified by ID. • `clustername`: Display values for a cluster specified by name.
mapr.volmetrics.read_latency	The per volume read latency in milliseconds	• `fqdn`: Display values for a specified node. • `volume_name`: Display values for a specified volume. • `clusterid`: Display values for a cluster specified by ID. • `clustername`: Display values for a cluster specified by name.

last-updated: Apr 11, 2022

Name	Description	Tag(s)
mapr.volmetrics.write_latency	The per volume write latency in milliseconds	• `fqdn`: Display values for a specified node. • `volume_name`: Display values for a specified volume. • `clusterid`: Display values for a cluster specified by ID. • `clustername`: Display values for a cluster specified by name.
mapr.volmetrics.read_ops	A count of the read operations per volume	• `fqdn`: Display values for a specified node. • `volume_name`: Display values for a specified volume. • `clusterid`: Display values for a cluster specified by ID. • `clustername`: Display values for a cluster specified by name.
mapr.volmetrics.write_ops	A count of the write operations per volume	• `fqdn`: Display values for a specified node. • `volume_name`: Display values for a specified volume. • `clusterid`: Display values for a cluster specified by ID. • `clustername`: Display values for a cluster specified by name.

Virtual Memory Metrics

Every 10 seconds, the collectd service uses the vmem plugin to gather the following memory metrics on each node in the cluster.

Name	Description
vmem.vmpage_faults.majflt	The number of major page faults. Major page faults require pages to be accessed from disk.
vmem.vmpage_faults.minflt	The number of minor page faults. Minor page faults can be resolved by sharing pages that are already in memory.

Memory Metrics

Every 10 seconds, the collectd service uses the memory and swap plugins to gather the following memory metrics on each node in the cluster.

Name	Description	Additional Tag(s)
memory.memory	The amount of physical memory in bytes.	• `memory_type`: Display values for a specified memory type. Memory type values: free, used, buffered, etc.

last-updated: Apr 11, 2022

Name	Description	Additional Tag(s)
swap.swap	The amount of swap space in bytes.	• `swap_type`: Display values for a specified swap type. Swap type values: used and free.
swap.swap_io	The amount of swap I/O in bytes.	

Network Metrics
Every 10 seconds, the collectd service uses the interface plugin to gather network metrics on each node in the cluster.

Name	Description	Additional Tag(s)
interface.if_errors.rx	The number of network errors received.	• `interface_name`: Display values for a specified interface. Interface values: eth0, eth1, etc.
interface.if_errors.tx	The number of network errors transmitted.	• `interface_name`: Display values for a specified interface. Interface values: eth0, eth1, etc.
interface.if_octets.rx	The number of bytes received over the network per second.	• `interface_name`: Display values for a specified interface. Interface values: eth0, eth1, etc.
interface.if_octets.tx	The number of bytes transmitted over the network per second.	• `interface_name`: Display values for a specified interface. Interface values: eth0, eth1, etc.
interface.if_packets.rx	The number of packets received over the network per second.	• `interface_name`: Display values for a specified interface. Interface values: eth0, eth1, etc.
interface.if_packets.tx	The number of packets transmitted over the network per second.	• `interface_name`: Display values for a specified interface. Interface values: eth0, eth1, etc.

Node Manager Metrics
Every 10 seconds, the collectd service uses a plugin to gather the following Node Manager metrics on each node in the cluster.

Name	Description
mapr.nm.allocated_GB	The amount of memory allocated to the Node Manager in GB.
mapr.nm.allocated_containers	The number of containers allocated to the Node Manager.
mapr.nm.allocated_vcores	The number of CPUs allocated to the Node Manager.
mapr.nm.available_vcores	The number of CPUs available to the Node Manager.
mapr.nm.available_GB	The amount of memory available to the Node Manager in GB.
mapr.nm.containers_completed	The number of containers that have completed.
mapr.nm.containers_failed	The number of containers that have failed.

Name	Description
mapr.nm.containers_initing	The number of containers that are initializing.
mapr.nm.containers_killed	The number of containers that have been killed by the Node Manager.
mapr.nm.containers_running	The number of containers that are running.
mapr.nm.containers_launched	The number of containers started by the Node Manager.
mapr.nm.jvm.gc_count	The number of garbage collections.
mapr.nm.jvm.gc_count_ps_mark_sweep	The number of parallel scavenge mark sweep collections.
mapr.nm.jvm.gc_count_ps_scavenge	The number of parallel scavenge collections.
mapr.nm.jvm.gc_time_millis	The amount of time spent on garbage collection in milliseconds.
mapr.nm.jvm.gc_time_millis_ps_mark_sweep	The amount of time spent on parallel scavenge mark sweep collection in milliseconds.
mapr.nm.jvm.gc_time_millis_ps_scavenge	The amount of time in milliseconds spent on parallel scavenge collection.
mapr.nm.jvm.log_error	The total number of ERROR logs.
mapr.nm.jvm.log_fatal	The total number of FATAL logs.
mapr.nm.jvm.log_info	The total number of INFO logs
mapr.nm.jvm.log_warn	The total number of WARN logs.
mapr.nm.jvm.mem_heap_committed_m	The amount of heap memory committed to the Node Manager in megabytes.
mapr.nm.jvm.mem_heap_max_m	The maximum amount of heap memory that can be committed to the Node Manager in megabytes.
mapr.nm.jvm.mem_heap_used_m	The amount of heap memory used by the Node Manager in megabytes.
mapr.nm.jvm.mem_max_m	The maximum amount of memory that can be committed to the Node Manager in megabytes.
mapr.nm.jvm.mem_non_heap_committed_m	The amount of non-heap memory committed to the Node Manager in megabytes.
mapr.nm.jvm.mem_non_heap_max_m	The maximum amount of non-heap memory that can be committed to the Node Manager in megabytes.
mapr.nm.jvm.mem_non_heap_used_m	The maximum amount of non-heap memory that can be used by the Node Manager in megabytes.
mapr.nm.jvm.threads_blocked	The number of Node Manager threads in BLOCKED state.
mapr.nm.jvm.threads_new	The number of Node Manager threads in NEW state.
mapr.nm.jvm.threads_runnable	The number of Node Manager threads in RUNNABLE state.
mapr.nm.jvm.threads_terminated	The number of Node Manager threads in TERMINATED state.
mapr.nm.jvm.threads_time_waiting	The number of Node Manager threads in TIMED_WAITING state.
mapr.nm.jvm.threads_waiting	The number of Node Manager threads in WAITING state.
mapr.nm.shuffle.shuffle_connection	The number of Node Manager shuffle connections.

last-updated: Apr 11, 2022

Name	Description
mapr.nm.shuffle.shuffle_output_bytes	The amount of Node Manager shuffle output in bytes.
mapr.nm.shuffle.shuffle_outputs_failed	The number of failed Node Manager shuffle outputs.
mapr.nm.shuffle.shuffle_outputs_ok	The number of completed Node Manager shuffle outputs.
mapr.nm.ugi.get_groups_avg_time	The average amount of time spent by Node Manager on group resolution.
mapr.nm.ugi.get_groups_num_ops	The number of group resolutions completed by the Node Manager.
mapr.nm.ugi.login_failure_avg_time	The average amount of time spent by Node Manager on failed login attempts.
mapr.nm.ugi.login_failure_num_ops	The number of failed login attempts by the Node Manager.
mapr.nm.ugi.login_success_avg_time	The average amount of time spent by Node Manager to successfully login.
mapr.nm.ugi.login_success_num_ops	The number of successful logins by the Node Manager.

Resource Manager Metrics

Every 10 seconds, the `collectd` service uses a MapR plugin to gather Resource Manager metrics on the active Resource Manager. `Collectd` gathers metrics on the Resource Manager JVM process, YARN applications, and nodes that are managed by the Resource Manager. The method used to gather the metrics differs based on the metric type.

YARN Application Metrics

`Collectd` gathers YARN application metrics via JMX and REST API. The application metrics that are collected by JMX have the metric name `mapr.rm.<metric_name>`. Application metrics collected via REST API have the metric name `mapr.rm_queue.<metric_name>`.

Metrics Collected Using JMX

The following metrics are collected using JMX. To filter these metrics by queue using the *rm_queue* tag, see Configure Queue Filters for mapr.rm.<value> Metrics on page 1335.

mapr.rm.active_applications

Additional Tags: `rm_queue`: Display values for a specified queue.

Description: The number of active applications.

mapr.rm.active_users

Additional Tags: `rm_queue`: Display values for a specified queue.

Description: The number of users with active applications.

mapr.rm.aggregate_containers_allocated

Additional Tags: `rm_queue`: Display values for a specified queue.

Description: The number of allocated containers.

mapr.rm.aggregate_containers_released

Additional Tags: `rm_queue`: Display values for a specified queue.

Description: The number of released containers.

mapr.rm.allocated_MB

Additional Tags: `rm_queue`: Display values for a specified queue.

Description: The amount of memory allocated to the Resource Manager in MB.

mapr.rm.allocated_vcores	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of CPUs allocated to the Resource Manager.
mapr.rm.apps_completed	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of completed applications.
mapr.rm.apps_failed	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of failed applications.
mapr.rm.apps_killed	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of killed applications.
mapr.rm.apps_pending	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of pending applications.
mapr.rm.apps_running	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of running applications.
mapr.rm.apps_submitted	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of submitted applications.
mapr.rm.available_MB	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The amount of memory available to the Resource Manager in MB.
mapr.rm.available_disks	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of disks available to the Resource Manager.
mapr.rm.available_vcores	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of CPUs available to the Resource Manager.
mapr.rm.pending_MB	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The amount of memory, in MB, waiting to be allocated by the Resource Manager.
mapr.rm.pending_containers	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of containers waiting to be allocated by the Resource Manager.
mapr.rm.pending_disks	*Additional Tags*: `rm_queue`: Display values for a specified queue.

	Description: The number of disks waiting to be allocated by the Resource Manager.
mapr.rm.pending_vcores	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of CPUs waiting to be allocated by the Resource Manager.
mapr.rm.reserved_MB	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The amount of memory reserved by the Resource Manager in MB.
mapr.rm.reserved_containers	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of containers reserved by the Resource Manager.
mapr.rm.reserved_disks	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of disks reserved by the Resource Manager.
mapr.rm.reserved_vcores	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of CPUs reserved by the Resource Manager.

Metrics Collected Using REST API

The following YARN application metrics are collected using REST API.

mapr.rm_queue.aggregate_containers_allocated	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of containers allocated for applications in the default and custom queues.
mapr.rm_queue.appmaster_used_disks	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: When queue resources are managed by the Capacity Scheduler, this parameter denotes the number of disks used by the Application Master for applications in the default and custom queues.
mapr.rm_queue.appmaster_used_memory	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: When queue resources are managed by the Capacity Scheduler, this parameter denotes the amount of memory, in MB, used by the Application Master for applications in the default and custom queues.
mapr.rm_queue.appmaster_used_vcores	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: When queue resources are managed by the Capacity Scheduler, this parameter denotes the number of CPUs used by the Application Master for applications in the default and custom queues.

mapr.rm_queue.apps_pending	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of pending applications in the default and custom queues.
mapr.rm_queue.apps_running	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of applications running in the default and custom queues.
mapr.rm_queue.fairshare_disks	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: When queue resources are managed by the Fair Scheduler, this parameter is the number of disks allocated to default and custom queues.
mapr.rm_queue.fairshare_memory	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: When queue resources are managed by the Fair Scheduler, this parameter denotes the amount of memory, in MB, allocated to default and custom queues.
mapr.rm_queue.fairshare_vcores	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: When queue resources are managed by the Fair Scheduler, this parameter denotes the number of CPUs used by applications in the default and custom queues.
mapr.rm_queue.used_disks	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of disks used by applications in the default and custom queues.
mapr.rm_queue.used_memory	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The amount of memory, in MB, used by applications in the default and custom queues.
mapr.rm_queue.used_vcores	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: The number of CPUs used by applications in the default and custom queues.
mapr.rm_queue.max_disks	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: When queue resources are managed by the Fair Scheduler, this parameter denotes the maximum number of disks available to default and custom queues.
mapr.rm_queue.max_memory	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: When queue resources are managed by the Fair Scheduler, this parameter denotes the maximum amount of memory, in MB, available to default and custom queues.

last-updated: Apr 11, 2022

mapr.rm_queue.max_vcores	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: When queue resources are managed by the Fair Scheduler, this parameter denotes the maximum number of CPUs available to default and custom queues.
mapr.rm_queue.user_allocated_disks	*Additional Tags*: • `rm_queue`: Display values for a specified queue. • `rm_user`: Display values for a specified user. *Description*: When queue resources are managed by the Capacity Scheduler, this parameter denotes the number of disks allocated to the queues.
mapr.rm_queue.user_allocated_memory	*Additional Tags*: • `rm_queue`: Display values for a specified queue. • `rm_user`: Display values for a specified user. *Description*: When queue resources are managed by the Capacity Scheduler, this parameter denotes the amount of memory, in MB, allocated to the queues.
mapr.rm_queue.user_allocated_vcores	*Additional Tags*: • `rm_queue`: Display values for a specified queue. • `rm_user`: Display values for a specified user. *Description*: When queue resources are managed by the Capacity Scheduler, this parameter denotes the number of CPUs allocated to queues.
mapr.rm_queue.user_appmaster_used_disks	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: When queue resources are managed by the Capacity Scheduler, this parameter denotes the number of disks used by the queues.
mapr.rm_queue.appmaster_used_memory	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: When queue resources are managed by the Capacity Scheduler, this parameter denotes the amount of memory used by the queues.
mapr.rm_queue.appmaster_used_vcores	*Additional Tags*: `rm_queue`: Display values for a specified queue. *Description*: When queue resources are managed by the Capacity Scheduler, this parameter denotes the number of CPUs used by the queues.
mapr.rm_queue.user_apps_pending	*Additional Tags*: • `rm_queue`: Display values for a specified queue. • `rm_user`: Display values for a specified user. *Description*: When queue resources are managed by the Capacity Scheduler, this parameter denotes the number of applications pending in the queues.

last-updated: Apr 11, 2022

mapr.rm_queue.user_apps_running	*Additional Tags*: • `rm_queue`: Display values for a specified queue. • `rm_user`: Display values for a specified user. *Description*: When queue resources are managed by the Capacity Scheduler, this parameter denotes the number of applications running in the queues.
mapr.rm_queue.user_used_disks	*Additional Tags*: • `rm_queue`: Display values for a specified queue. • `rm_user`: Display values for a specified user. *Description*: When queue resources are managed by the Capacity Scheduler, this parameter denotes the number of number of disks used by the queues.
mapr.rm_queue.user_used_memory	*Additional Tags*: • `rm_queue`: Display values for a specified queue. • `rm_user`: Display values for a specified user. *Description*: When queue resources are managed by the Capacity Scheduler, this parameter denotes the amount of memory, in MB, used by the queues.
mapr.rm_queue.user_used_vcores	*Additional Tags*: • `rm_queue`: Display values for a specified queue. • `rm_user`: Display values for a specified user. *Description*: When queue resources are managed by the Capacity Scheduler, this parameter denotes the number of CPUs used by the queues.

Resource Manager Node Metrics

The following are the Node metrics:

mapr.rm_cluster.active_nodes	The number of nodes in the cluster where containers are running.
mapr.rm_cluster.total_nodes	The number of nodes in the cluster.
mapr.rm_cluster.unhealthy_nodes	The number of nodes in the cluster that are unable to accept applications.

Resource Manager JVM Metrics

The following Resource Manager metrics are collected using JMX:

mapr.rm.jvm.gc_count	The number of garbage collections.
mapr.rm.jvm.gc_count_ps_mark_sweep	The number of parallel scavenge mark sweep collections.
mapr.rm.jvm.gc_count_ps_scavenge	The number of parallel scavenge collections.

mapr.rm.jvm.gc_time_millis	The amount of time, in milliseconds, spent on garbage collection.
mapr.rm.jvm.gc_time_millis_ps_mark_sweep	The amount of time, in milliseconds, spent on parallel scavenge mark sweep collection.
mapr.rm.jvm.gc_time_millis_ps_scavenge	The amount of time, in milliseconds, spent on parallel scavenge collection.
mapr.rm.jvm.log_error	The total number of ERROR logs.
mapr.rm.jvm.log_fatal	The total number of FATAL logs.
mapr.rm.jvm.log_info	The total number of INFO logs.
mapr.rm.jvm.log_warn	The total number of WARN logs.
mapr.rm.jvm.mem_heap_committed_m	The amount of heap memory, in megabytes, committed to the Resource Manager.
mapr.rm.jvm.mem_heap_max_m	The maximum amount of heap memory, in megabytes, that can be committed to the Resource Manager.
mapr.rm.jvm.mem_heap_used_m	The amount of heap memory, in megabytes, used by the Resource Manager.
mapr.rm.jvm.mem_max_m	The maximum amount of memory, in megabytes, that can be committed to the Resource Manager.
mapr.rm.jvm.mem_non_heap_committed_m	The amount of non-heap memory, in megabytes, committed to the Resource Manager.
mapr.rm.jvm.mem_non_heap_max_m	The maximum amount of non-heap memory, in megabytes, that can be committed to the Resource Manager.
mapr.rm.jvm.mem_non_heap_used_m	The maximum amount of non-heap memory, in megabytes, that can be used by the Resource Manager.
mapr.rm.jvm.threads_blocked	The number of Resource Manager threads in BLOCKED state.
mapr.rm.jvm.threads_new	The number of Resource Manager threads in NEW state.
mapr.rm.jvm.threads_runnable	The number of Resource Manager threads in RUNNABLE state.
mapr.rm.jvm.threads_terminated	The number of Resource Manager threads in TERMINATED state.
mapr.rm.jvm.threads_time_waiting	The number of Resource Manager threads in TIMED_WAITING state.
mapr.rm.jvm.threads_waiting	The number of Resource Manager threads in WAITING state.

Configure the OpenTSDB Service Heap Size

By default, the OpenTSDB service is configured to use a default heap size of 6 gigabytes. The default heap size can be adjusted by modifying certain configuration files.

last-updated: Apr 11, 2022

In EEPs 6.0.1 and later, the OpenTSDB service is configured to use a default heap size of 6 GB. In earlier EEPs, the default was configured to be 2 GB. If you configure a custom value for the OpenTSDB service heap size and then upgrade to EEPs 6.0.1 or later, you will see the 6-GB default implemented in the `/opt/mapr/conf/conf.d/warden.opentsdb.conf` file.

To change the heap size to a different setting, edit the configuration files on all OpenTSDB nodes as follows. The following steps change the heap size from 2 GB to 6 GB. The default heap size might need further adjustment if your cluster grows to a large number of nodes. If the 6-GB heap size is insufficient, you can use these same steps to adjust it:

1. Edit the `/opt/mapr/conf/conf.d/warden.opentsdb.conf` file to change:

   ```
   service.heapsize.max=2000
   service.heapsize.min=2000
   ```

   to

   ```
   service.heapsize.max=6000
   service.heapsize.min=6000
   ```

2. Edit the `/opt/mapr/opentsdb/opentsdb-*/etc/init.d/opentsdb` file to change:

   Note: Step 2 is not required for EEPs 6.0.1 and later, which contain logic to edit the `opentsdb` file automatically.

   ```
   $
   {JVMXMX:=-Xmx2000m -Xss1m -XX:MaxMetaspaceSize=128m}
   ```

   to

   ```
   $
   {JVMXMX:=-Xmx6000m -Xss1m -XX:MaxMetaspaceSize=128m}
   ```

3. Restart the OpenTSDB service:

   ```
   maprcli node services -name opentsdb -nodes <space-separated list of
   OpenTSDB nodes> -action restart
   ```

Metric Visualization
Use dashboards to visualize metrics across multiple nodes and clusters.

You can use a single dashboard to visualize metrics for multiple nodes in the cluster, for an entire cluster, or for multiple clusters. In a dashboard, you can use metric tags to filter the type of data that you want to see. To learn more about the tags available for each metric, see Metric Collection on page 1334.

Before you start visualizing metrics, review the following notes:

Sample Dashboards

As of EEP 1.1, the Grafana UI includes sample CLDB, node, and volume dashboards. For more information, see Sample Dashboards in Grafana on page 1384.

Grafana Documentation

For information about creating and using dashboards, see the Grafana documentation.

The Embedded Grafana Database

Grafana uses an embedded database to store configuration data such as users, data sources, and dashboards. When used with the MapR Data Platform, Grafana uses SQLite for its embedded database. Other databases, such as MySQL and PostgreSQL, are not supported for use with the MapR implementation of Grafana.

Note also that in a data-fabric cluster, each instance of Grafana has its own embedded SQLite database. Therefore, if you make a change to the Grafana configuration data (for example, if you add a new user) on one node, you must repeat the change on all other nodes where Grafana is installed.

Access the Grafana UI

You can launch the Grafana UI from the Control System or directly from a web browser.

1. Use one of the following methods to launch the Grafana UI:
   - In the Control System, go to the **Services** page and click **Grafana** to launch the Grafana UI in another tab.
   - From a web browser, launch the following URL: `http://<IPaddressOfGrafanaNode>:3000`
2. Provide user credentials. See Logging on to Grafana on page 1382.
3. Click **Log in**.

Logging on to Grafana

This page describes the credentials needed to log on to Grafana for secure and nonsecure clusters for MapR 6.0 and later.

EEP 5.0.0 and MapR Installer 1.9.0 implemented some changes in security that affect the user IDs and passwords you need to log on to Grafana. Beginning with EEP 5.0.0, Grafana no longer uses its own database to authenticate on secure MapR clusters. On secure clusters, when you enter your user name and password into the browser, the Grafana server sends the user name and password to the CLDB, and the CLDB authenticates using the Pluggable Authentication Module (PAM) and HTTPS. Grafana still relies on its database for some user information. However, because authentication relies on the CLDB, any new user that you add to Grafana must also exist on the MapR cluster.

EEP 4.x.x (Grafana 4.4.2) and Earlier

To log on, enter the default Grafana `admin` user and password:

- **User:** `admin`
- **Password:** `admin`

EEP 5.x.x (Grafana 4.6.1) and Later

To log on, specify the user and password as follows:

	Secure Cluster	Nonsecure Cluster
User	Type the MapR cluster admin user ID. **Note:** If the cluster admin user has no password or is not able to log in, see Logging on to Grafana Without Using the Cluster Admin on page 1383.	Type the Grafana `admin` user ID.
Password	Type the MapR cluster admin password that you specified during cluster installation.	Type the password for the Grafana `admin` user that you specified during cluster installation. (During cluster installation,

	Secure Cluster	Nonsecure Cluster
		the MapR Installer web interface asks you to provide this password. You can also provide the password by using `configure.sh` during a manual installation.)

For More Information

To change the Grafana password, see Changing the Password for Grafana on page 1383.

For more information about the Grafana versions supported by each EEP, see EEP Components and OS Support on page 5536.

For information about the EEPs supported by different MapR Core versions, see EEP Support and Lifecycle Status on page 5531.

Logging on to Grafana Without Using the Cluster Admin
In secure Grafana installations where the cluster admin (typically `mapr`) has no password and is not allowed to log in as a user, special steps are required to enable login with a user other than the cluster admin.

Use this procedure only if the cluster admin has no password. After an upgrade or a new installation of Grafana, perform these steps on the Grafana nodes:

1. Remove the old Grafana database. You will be able to access the Grafana database later using the new Grafana admin user:

   ```
   cd /opt/mapr/grafana/grafana-<version>/var/lib/grafana
   mv grafana.db grafana.db.sv
   ```

2. Use the `export` command to specify the new Grafana admin user:

   ```
   export GRAFANA_ADMIN_ID=<username>
   ```

3. Run `configure.sh` with the `-R` option:

   ```
   configure.sh -R
   ```

4. Restart Grafana.

Changing the Password for Grafana
Describes how to change the Grafana password for secure and nonsecure clusters.

Secure Clusters

To change the Grafana password for a secure cluster, you must change the password for the MapR cluster admin user. The security implementation determines how you change the password. For example, if your security implementation is PAM and LDAP, you need to use LDAP to change the cluster admin password.

Note: If the cluster is secure, attempting to change the Grafana password using the Grafana interface has no effect.

Nonsecure Clusters

For a nonsecure cluster:

1. Log in to the interface using the `admin` user ID and the password that you specified for the `admin` user ID when you installed the cluster.
2. Select **Home > Admin > Profile**.
3. On the **User Profile** screen, click **Change Password**, located near the bottom of the screen.
4. Enter and save the new password.

Adding a New Grafana User to a Secure MapR Cluster
In a secure MapR cluster, adding a new Grafana user requires an extra step to ensure that the user can be authenticated through the CLDB.

Beginning with EEP 5.0.0, Grafana no longer uses its own database to authenticate on secure MapR clusters. On secure clusters, when you enter your user name and password into the browser, the Grafana server sends the user name and password to the CLDB, and the CLDB authenticates using the Pluggable Authentication Module (PAM) and HTTPS.

Because authentication relies on the CLDB, any new user that you add to Grafana must also exist on the MapR cluster. Therefore, to add a new Grafana user in a secure MapR cluster:

1. Add the user to the Linux system first. This user must have the same Linux UID and GID on every node in the MapR cluster.
2. Add the same user to Grafana, making sure that you use the user ID you added to the Linux system. In Grafana: **Settings > ServerAdmin > Users > Add new user**.

Sample Dashboards in Grafana
Use the sample dashboards to get familiar with the types of graphs you can create. As of EEP 1.1, sample dashboards are available by default in Grafana.

Displaying the Sample Dashboards

For Grafana 4.x, 5.x, and 6.x, navigate to the **Welcome to Grafana** page, and click the **Home** drop-down menu to display the sample dashboards. Once you select a dashboard, it displays in the **Recently viewed dashboards** list on the Home page.

In Grafana 7.5.x and later, use these steps to display the dashboards:

1. Navigate to the **Welcome to Grafana** page.
2. On the left-side icon menu, click **Dashboard > Manage**. The list of dashboards is displayed.
3. Click a dashboard in the list to load the dashboard. The **General / Node Dashboard** page is displayed.
4. Loading the dashboard adds the dashboard to the **Recently viewed dashboards** list on the **General / Home** page.

Sample Dashboard Descriptions

CLDB Dashboard	The CLDB dashboard provides a high-level view of the MapR cluster. It displays the following information about the cluster: number of nodes, status of nodes, number of volumes, container information, disk capacity, and the utilization of CPU, memory, and disks across the cluster.
Node Dashboard	The Node dashboard provides node-specific information. It displays the following information for the selected node: CPU, memory, network I/O, MapR File System I/O, MapR Database operations, and Node Manager metrics. All the metrics are tagged with node

	hostname and the `fqdn` drop-down menu on the top left can be used to switch between nodes.
Volume Dashboard	The Volume dashboard provides volume-specific information. It displays the following information for the selected volume: raw data size, snapshot size, total size (including the snapshot size), volume utilization trends, read/write latency, number of reads/writes, and read/write throughput. All the metrics are tagged with volume name. Use the `VolumeName` drop-down menu on the top left to switch between volumes.

Troubleshooting Sample Dashboards

The sample dashboards should display metrics automatically. However, with certain EEPs, some manual configuration may be required to view sample dashboard metrics.

Configure the ClusterID in the CLDB dashboard	When the `ClusterID` drop-down menu on the CLDB dashboard is set to `None`, you must manually enter the ClusterID. You can determine the ClusterID from the Manage Licenses page on the Control System. ClusterID is usually an eighteen digit number.
Configure the Hostname in the Node dashboards	When the `fqdn` drop-down menu on the Node dashboard is set to `None`, you must manually enter the hostname for the node that you want to view metrics for.
Configure the Volume in the Volume dashboard	When the `VolumeName` drop-down menu on the Volume dashboard is set to `None`, you must manually enter the volume name for the volume that you want to view metrics for. The Volumes page on the Control System lists the volume names in the format required by the field. For example, you can enter `mapr.cluster.root` in the `VolumeName` drop-down menu.

Note: You must apply the manual configuration each time you view a dashboard.

Update the OpenTSDB Data Source For Grafana

Grafana connects to a single OpenTSDB node to read metrics. If Grafana cannot read the metrics because an OpenTSDB node has failed, you must configure Grafana to connect to a different OpenTSDB node.

Note: The OpenTSDB node that Grafana connects to by default is determined by the first OpenTSDB node that was specified when the cluster was configured to use MapR monitoring.

1. Use one of the following methods to launch the Grafana user interface:
   - From the Control System, select the **Grafana** view. After you select the **Grafana** view, you might also need to select the **Pop-out page into a tab** option.
   - From a web browser, launch the following URL: `http://<IPaddressOfGrafanaNode>:3000`
2. Click the Grafana icon in the upper left corner to toggle the side-menu bar.
3. Select **Data Sources** from the menu.
4. Click the **MapRMonitoringOpenTSDB** data source.
5. In the **Http setting** section, update the **Url** field to point to an active OpenTSDB node.
6. Click **Save & Test**.

last-updated: Apr 11, 2022

Log Collection

Fluentd collects log events from each node in the cluster and stores them in a centralized location so that administrators can search the logs when troubleshooting issues in the cluster. The process that fluentd uses to parse and send log events to Elasticsearch differs based on the formatting of log events in each log file.

Fluentd uses one or both of the following mechanisms to parse logs:

multi-line matching	Using the log time stamp as a delimiter, multi-line matching uses the `tail` plugin to read logs and determine the end of a log event. Each log event is sent to Elasticsearch when the next log event is written to the log file. This mechanism is often used when each log event starts with a timestamp and then includes a stack trace.
multi-pattern matching	Multi-pattern matching uses the `grok` plugin to parse logs events using complex expressions. This mechanism is often used to parse logs events that have non-uniform log formatting.

Before Fluentd sends the log entries to Elasticsearch, Fluentd assigns the following columns to each log event:

Tag	Description
level	The message level of the log entry. For example, info, warning, or error.
class	Java or C++ process name associated with the log entry.
message	The log message.
event_time	The time, with millisecond precision, when the log entry was written to the log file.
service_name	The name of the service that generated the log entry.
@timestamp	The time, with second precision, when fluentd read the message.
fqdn	The node on which the log entry was written.
clusterid	The clusterid of the cluster on which the log was written.

Note: The log event contents differs based on the service that logs it and the type of log. Therefore, the log events sent to Elasticsearch may include empty columns.

For more information about Elasticsearch, see the Elasticsearch website.

Configure Logs to Index

Edit the *fluentd.conf* file (`/opt/mapr/fluentd/fluentd-<version>/etc/fluentd/fluentd.conf`) to enable or disable the indexing of a specific log.

The *fluentd.conf* file includes a source parameter for each log file that it indexes.

1. To disable the indexing of a log, comment all lines for the associated source parameter.
2. To enable the indexing of a log, for example syslogs, uncomment the lines for the associated source parameter.

last-updated: Apr 11, 2022

3. Restart `fluentd` on each node in the cluster which is impacted by changes to the index configuration. For example, if you disable the indexing of Kibana logs, restart `fluentd` on the node that runs Kibana.

```
maprcli node services -name fluentd -nodes <space separated list of
hostname/IPaddresses> \
 -action restart
```

For example, in this excerpt of the *fluentd.conf* file, NodeManager error logs are disabled and ResourceManager logs are enabled:

```
# yarn nodemanager log
# <source>
#  @type tail
#  @id yarn_nodemanager_input
#  format multiline
#  format_firstline /\d{4}-\d{1,2}-\d{1,2}/
#  format1 /^(?<my_event_time>[^ ]* [^ ]*) (?<level>[^ ]*) (?<class>[^:]*):
(?<message>.*)$/
#  time_key my_event_time
#  keep_time_key true
#  path /opt/mapr/hadoop/hadoop-*/logs/yarn-*-nodemanager-*.log
#  tag nodemanager
#  pos_file /opt/mapr/fluentd/fluentd-0.14.00/var/log/fluentd/tmp/
nodemanager.pos
# </source>

# yarn resourcemanager log
<source>
  @type tail
  @id yarn_resourcemanager_input
  format multiline
  format_firstline /\d{4}-\d{1,2}-\d{1,2}/
  format1 /^(?<my_event_time>[^ ]* [^ ]*) (?<level>[^ ]*) (?<class>[^:]*):
(?<message>.*)$/
  time_key my_event_time
  keep_time_key true
  path /opt/mapr/hadoop/hadoop-*/logs/yarn-*-resourcemanager-*.log
  tag resourcemanager
  pos_file /opt/mapr/fluentd/fluentd-0.14.00/var/log/fluentd/tmp/
resourcemanager.pos
</source>
```

Forward Logs to Syslog Server

You can configure `fluentd` to send logs to a syslog server in addition to Elasticsearch. This topic provides instructions for configuring `fluentd` to send logs to syslog compatible collectors. However, it only provides guidelines for the syslog configuration, as syslog parameters differ by version. Knowledge of how to configure a syslog compatible collector is required to complete this configuration.

Complete the following steps:

1. Configure `fluentd` to send logs to the syslog server.
2. Configure `syslog` server to accept logs from `fluentd`.

Step 1: Configure fluentd to send logs to the syslog server

Complete the following steps on each `fluentd` node.

1. Open the `fluentd.conf` file (*/opt/mapr/fluentd/fluentd-<version>/etc/fluentd/fluentd.conf*).

last-updated: Apr 11, 2022

2. Remove the # to uncomment the following store section:

```
# <store>
# @type remote_syslog
# host 10.10.100.92
# port 51400
# severity debug
# tag fluentd
# </store>
```

3. Update the `host` parameter to the hostname/IP address of the receiving `syslog` server.
4. Update the `port` parameter to match the port that the receiving `syslog` server is expecting remote logging information on.
5. Restart the `fluentd` service:

```
maprcli node services -name fluentd -nodes <space separated list of
hostname/IPaddresses> -action restart
```

Note: You can run this command after completing the steps on a node or run this command with a list of nodes once you have configured each `fluentd` node.

Step 2: Configure syslog to accept logs from fluentd

In general, you need to perform the following steps on the `syslog` collection server:

- Configure `syslogd` to listen for logs outside of the `syslog` node.
- Set up rules for how `syslog` handles the logs once it receives it.

1. In */etc/rsyslog.d/listen.conf*, comment out the following parameter:

```
$SystemLogSocketName /run/systemd/journal/syslog
```

2. In */etc/rsyslog.conf*, uncomment the following properties:

```
#$ModLoad imudp
#$UDPServerRun 514
```

3. In */etc/rsyslog.conf*, update the UDPServerRun to a value above 1000 that matches the port you configured in *fluentd.conf*. For example: `Set UDPServerRun` to 51400
4. In */etc/rsyslog.conf*, configure rules for handling logs. For example, add the following before the RULES section to route messages from the `fluentd` node to a log file named *qa-node91.log*.

```
if $fromhost-ip == '10.10.100.91' then /var/log/qa-node91.log
& ~
```

Note: In this example, the IP address must match the IP address of the `fluentd` node.

MapR Core Logs

The `fluentd` component reads and parses the following MapR Core log files on each node in the cluster.

Service Name	Parsing Method	Description
adminuiapp	Multi-line	Control System logs from `/opt/mapr/apiserver/logs/apiserver.log`.
cldb	Multi-line	CLDB server logs from `$MAPR_HOME/logs/cldb.log`.
cldbfssummary	Multi-line	Summary of the CLDB server logs from `$MAPR_HOME/logs/cldbfssummary.log`
mfs_maprdb	Multi-line	MapR Database logs from `$MAPR_HOME/logs/mfs.log-5`. **Note:** If nodes in the cluster run two filesystem instances, Fluentd only reads and parses logs from primary fileserver instance. Therefore, logs from the secondary fileserver instance will not be indexed by Elasticsearch.
mfs	Multi-line	MapR filesystem logs from `$MAPR_HOME/logs/mfs.log-3`. **Note:** If nodes in the cluster run two filesystem instances, Fluentd only reads and parses logs from primary fileserver instance. Therefore, logs from the secondary fileserver instance will not be indexed by Elasticsearch.
nfsserver	Multi-line and Multi-pattern	NFS server log from `$MAPR_HOME/logs/nfsserver.log`
nodemanager	Multi-line	Node Manager logs from `$MAPR_HOME/hadoop/hadoop-*/logs/yarn-*-nodemanager-*.log`.
resourcemanager	Multi-line	ResourceManager logs from `$MAPR_HOME/hadoop/hadoop-*/logs/yarn-*-resourcemanager-*.log`
warden	Multi-line and Multipattern	Warden logs from `$MAPR_HOME/logs/warden.log`.
zookeeper	Multi-line	Zookeeper logs from `$MAPR_HOME/zookeeper/zookeeper-*/logs/zookeeper.log`

MapR Ecosystem Logs

The `fluentd` component reads and parses the following MapR ecosystem component logs on each node in the cluster.

Service Name	Parsing Method	Description
drill	Multi-line	Drill logs from `$MARP_HOME/drill-*/logs/drillbit.log`.

last-updated: Apr 11, 2022

Service Name	Parsing Method	Description
drillbitsSqlline	Multi-line	Drill SQL query logs from `$MAPR_HOME/drill/drill-*/logs/sqlline.log`.
hbaserest	Multi-line and Multipattern	HBase REST Server logs from `$MAPR_HOME/hbase/hbase-*/logs/hbase-*-rest-*.log`.
hbasethriftserver	Multi-line	Hbase Thrift Server logs from `$MAPR_HOME/hbase/hbase-*/logs/hbase-*-thrift-*.log`.
hive	Multi-line	HiveServer logs from `$MAPR_HOME/hive/hive-*/logs/root/hive.log`.
httpfs	Multi-line	HttpFS logs from `$MAPR_HOME/httpfs/httpfs-*/logs/httpfs.log`.
hue	Multi-line	Hue logs from `$MAPR_HOME/hue/hue-*/logs/hue-mapr-runcpserver-*.out`.
oozie	Multi-line	Oozie logs from `$MAPR_HOME/oozie/oozie-*/logs/oozie.log` .
oozieOps	Multi-line	Oozie operation logs from `$MAPR_HOME/oozie/oozie-*/logs/oozie-ops.log`.
oozieCatalina	Multi-line and Multipattern	Oozie Catalina logs from `$MAPR_HOME/oozie/oozie-*/logs/catalina.out`.
sparkhistory	Multi-line	Spark HistoryServer logs from `$MAPR_HOME/spark/spark-*/logs/spark-mapr-org.apache.spark.deploy.history.HistoryServer-1-*.out`.

System Logs

The `fluentd` component does not collect the following system logs by default because they require the configuration of additional permissions for the *$MAPR_USER*.

Service name	Parsing Method	Description
kernlog	Multi-line	Kernel logs from `/var/log/kern.log`.
syslog	Multi-line	System logs from `/var/log/syslog` and `/var/log/messages`.
mysql_errors	Multi-line	MySQL errors from `/var/log/mysql/error.log`.

last-updated: Apr 11, 2022

Note: To enable `fluentd` to read and parse these logs, see Configure Logs to Index on page 1386 and also perform the following:

- On Ubuntu and RHEL/CentOS, add *$MAPR_USER* to the `admin` group.
- On RHEL/CentOS, change the ownership of the log file so that it is owned by both the `root` user and the `admin` group.

MapR Monitoring Logs
The `fluentd` component reads and parses the following MapR Monitoring component logs on each node in the cluster.

Service Name	Parsing Method	Description
collectd	Multi-line	The collectd component logs from `$MAPR_HOME/collectd/collectd-*/var/log/collectd/collectd_daemon.log`
fluentd	Multi-line	The fluentd component logs from `$MAPR_HOME/fluentd/fluentd-*/var/log/fluentd/fluentd.log`,
grafana	Multi-line	Grafana logs from `$MAPR_HOME/grafana/grafana-*/var/log/grafana/grafana.log`.
kibana	Multi-line	Kibana logs from `$MAPR_HOME/kibana/kibana-*/var/log/kibana/kibana.log`.

Log Aggregation and Storage
`Fluentd` uses a round-robin approach when writing logs to Elasticsearch nodes. If an Elasticsearch node in unavailable, `Fluentd` can fail over log storage to another Elasticsearch node.

Each `Fluentd` service connects to each Elasticsearch node that you configure to aggregate and store logs. The Elasticsearch nodes are set when you configure MapR Monitoring with the MapR Installer or when you run configure.sh with the `-ES` parameter.

The Elasticsearch index directory is shared among all the Elasticsearch nodes in the cluster. When you use the MapR Installer to install Elasticsearch, each Elasticsearch node writes index data to `/opt/mapr/es_db`, unless you specified a different location during the installation. When you manually install Elasticsearch, each Elasticsearch node writes index data to `/opt/mapr/elasticsearch/elasticsearch-<version>/var/lib/MaprMonitoring/`, unless you specified a different location using the `configure.sh -ESDB` option. For a cluster with one Elasticsearch node, the index directory is allocated 5 shards. For clusters with 2 or more Elasticsearch nodes, the index directory is allocated a number of shards equal to 3 times the number of Elasticsearch nodes in the cluster.

`Fluentd` does not require additional configuration to enable automatic failover to an available Elasticsearch node. However, it is important that at least three Elasticsearch nodes are configured to aggregate and store logs so that failure of one node does not prevent logs from being used for monitoring purposes. Based on your environment, more Elasticsearch nodes may be required. Service Layout Guidelines for Large Clusters on page 115.

For more information about Elasticsearch, see the Elastic website.

Configure Log Retention
By default, Elasticsearch indexes 2 days of logs. Based on your requirements, you can configure a different retention period for Elasticsearch.

The following cron job runs each day to purge logs based on the retention period.

```
$min $hour * * * $ES_HOME/bin/curator --config $ES_HOME/etc/elasticsearch/
curator.yml \
 $ES_HOME/etc/elasticsearch/curator_actions/delete_indices.yml >>
$ES_HOME/var/log/elasticsearch/purgeData.log 2>&1 "
```

Log Retention for Elasticsearch

Complete the following steps to edit the log retention period for Elasticsearch:

1. Open the `/opt/mapr/elasticsearch/elasticsearch-<version>/etc/elasticsearch/curator_actions/delete_indices.yml` file.
2. Update the `unit` and `unit_count` to the new retention period.

unit	The unit of measure for the retention period. Valid parameter values: `days` and `weeks`.
unit_count	The number of days or weeks.

For example, this version of the *delete_indices.yml* file retains logs for 2 days.

```
actions:
  1:
    action: delete_indices
    description: >-
      Delete indices older than 2 days (based on index name), for
mapr_monitoring-
      prefixed indices. Ignore the error if the filter does not result
in an
      actionable list of indices (ignore_empty_list) and exit cleanly.
    options:
      ignore_empty_list: True
      timeout_override:
      continue_if_exception: False
      disable_action: False
    filters:
    - filtertype: pattern
      kind: prefix
      value: mapr_monitoring-
      exclude:
    - filtertype: age
      source: name
      direction: older
      timestring: '%Y.%m.%d'
      unit: days
      unit_count: 2
      exclude:
```

Log Retention for Kibana

Each time you start Kibana, it logs data to its log file. You cannot delete the log file while Kibana is running.

To purge the log files:

1. Restart Kibana so that a new log file is created.

```
maprcli node services -name kibana -nodes <kibana hostname/
IPaddress> -action restart
```

2. Delete all the old log files (`kibana.*.<#>`) from the following location: `/opt/mapr/kibana/kibana-<version>/var/log/kibana/`.

Configure Purge Duration
Based on your requirements, you can configure a purge duration for Elasticsearch.

To update the purge duration on node with installed Elasticsearch 6.5.3, run the following command:

```
/opt/mapr/elasticsearch/elasticsearch-<Version>/usr/share/elasticsearch/bin/es_cluster_mgmt.sh --purgeAge <newValue>
```

This command will automatically update the `delete_indices.yml` file. This change updates only the `unit_count` value to `newValue`, without changing the unit value.

Configure Log Rotation Policies for MapR Monitoring Services
New log files are created based on the log rotation policy. By default, each MapR Monitoring service has a log rotation policy. In most cases, you can change the policy based on your requirements.

OpenTSDB Log Rotation Policy

By default, OpenTSDB creates a new log file when each log file reaches the maximum file size of 128MB. After 4 log files are generated, it deletes the oldest log file.

To change the log rotation policy, edit the following file: `/opt/mapr/opentsdb/opentsdb-<version>/etc/opentsdb/logback.xml`. For more information, see the OpenTSDB Logging documentation.

Fluentd and CollectD Log Rotation Policy

By default, `Fluentd` and `Collectd` create a new log file each day and they both retain 30 log files. Log rotation for `Fluentd` and `Collectd` logs is managed by `logrotate`.

To change the log rotation policies, edit the following files: `/etc/logrotate.d/fluentd` and `etc/logrotate.d/collectd`. For details on how to update the log rotation policy, see the logrotate documentation.

Elasticsearch Log Rotation Policy

By default, Elasticsearch creates a new log file each day and it retains 7 days of logs.

To change the log rotation policy, edit the following file `/opt/mapr/elasticsearch/elasticsearch-<version>/etc/elasticsearch/logging.yml`. For details on how to update logging.yml, see the Elasticsearch documentation.

Grafana Log Rotation Policy

By default, Grafana creates a new log file whenever the current log file exceeds the 256MB. It retains log files that were generated in the last 7 days.

To change the log rotation policy, edit the `[log]` section of the following file: `/opt/mapr/grafana/grafana-<version>/etc/grafana/grafana.ini`. For details on how to update the grafana.ini, see the Grafana documentation.

Kibana Log Rotation Policy

Each time you start Kibana, it logs data to its log file and it does not automatically delete old log files. A new log file is created when you restart Kibana. To purge the log files, see Configure Log Retention on page 1391.

Configure the Elasticsearch Service Heap Size

The Elasticsearch service is memory-intensive. By default, the Elasticsearch service is configured to use a minimum and maximum heap size of 2 GB. You can override these default values by making changes in the Elasticsearch Warden configuration file and the `jvm.options` file. Restart Elasticsearch after you modify the settings.

Configuring Memory in the Warden Configuration File

You can enable memory settings in the ElasticSearch Warden configuration file, located in the `/opt/mapr/conf/conf.d/warden.elasticsearch.conf` directory. Modify the `service.heapsize.min` and `service.heapsize.max` values set in `warden.elasticsearch.conf`, as shown:

```
service.heapsize.min=2000
service.heapsize.max=2000
```

The `service.heapsize.min` and `service.heapsize.max` values are set in megabytes as an integer. For older EEPs, you must make sure that the `-Xms` and `-Xmx` values in the jvm.options file match the settings in the Warden configuration file.

About the jvm.options File

The `jvm.options` file centralizes arguments to the Java Virtual Machine to simplify the management of the JVM options. You can no longer set the JVM options through the ES_MIN_MEM, ES_MAX_MEM, ES_HEAP_SIZE, ES_HEAP_NEWSIZE, ES_DIRECT_SIZE, ES_USE_IPV4,ES_GC_OPTS, ES_GC_LOG_FILE, and JAVA_OPTS environment variables.

If you installed the Elasticsearch service from the TAR or ZIP distributions, you can locate the `jvm.options` file in `config/jvm.options`. If you installed ElasticSearch from the Debian or RPM packages, you can locate the `jvm.options` file in the `$ES_HOME/etc/elasticsearch/jvm.options` directory, for example:

```
/opt/mapr/elasticsearch/elasticsearch-<version>/etc/elasticsearch
```

To specify an alternative location, set the ES_JVM_OPTIONS environment variable to the file path.

Configuring Memory in the jvm.options File

Note: If you configured Elasticsearch memory in the Warden configuration file, configuring memory in the `jvm.options` file is not required for EEPs 6.2.0, 6.1.1, 6.0.2 and later, which contain logic to edit the `jvm.options` file automatically. For other EEPs, you must ensure that memory-configuration changes are made *both* in the Warden configuration file and in the`jvm.options` file.

The `-Xms` and `-Xmx` values in the `jvm.options` file set the Elasticsearch heap size, as shown:

```
-Xms2g
-Xmx2g
```

The `-Xms` parameter sets the minimum heap size in gigabytes. The `-Xmx` parameter sets the maximum heap size in gigabytes. Elasticsearch recommends that both parameters have the same value.

Restarting the Elasticsearch Service

After you modify memory settings, issue the following command to restart the Elasticsearch service:

```
maprcli node services -name elasticsearch -nodes <space separated list of
Elasticsearch nodes> -action restart
```

Tip:

- On a production cluster, you can lock Elasticsearch memory to improve performance. To lock Elasticsearch memory, set the `bootstrap.mlockall: true` option in `$ES_HOME/etc/elasticsearch/elasticsearch.yml`.
- If Elasticsearch uses more than 75% of the configured heap size, you may want to increase the maximum heapsize value.

  For more information, see the Elasticsearch documentation.

Configure Fluentd Services to Write to Elasticsearch Nodes on the Same Rack
On clusters with high-density racks, ensure you have at least one Elasticsearch server per rack and configure each Fluentd service to write to Elasticsearch nodes that run on the same rack as the Fluentd service. This configuration minimizes the impact of log aggregation on other processes that run on the cluster and in particular, minimizes the amount of backbone bandwidth used by the log aggregation.

Complete the following steps on each node that runs the Fluentd service.

1. Open the `/opt/mapr/fluentd/fluentd-<version>/etc/fluentd/es_config.conf` file.
2. Edit the `hosts` property to only include Elasticsearch nodes that are on the same rack as the Fluentd service.
   Example:

   ```
   hosts qa-node90:9200,qa-node91.qa.lab:9200,qa-node92.qa.lab:9200
   ```

3. Restart Fluentd.

   ```
   maprcli node services -name fluentd -nodes <space separated list of
   fluentd nodes> -action restart
   ```

Warning: Changes to the es_config.conf files are overridden by configure.sh. Therefore, you will need reconfigure the hosts property in the es_config.conf file after configure.sh is run on Fluentd nodes.

60-mapr_elasticsearch.conf

The `/etc/sysctl.d/60-mapr_elasticsearch.conf` file is created when you install Elasticsearch. This file specifies a Docker host setting for Elasticsearch.

The `vm.max_map_count` setting is a tuning parameter that limits the number of discrete mapped memory areas for the Linux VM:

Example

```
vm.max_map_count=262144
```

For more information about the `max_map_count` parameter, see the Linux Kernel documentation.

60-mapr_fluentd.conf

The `/etc/sysctl.d/60-mapr_fluentd.conf` file is created when you install Fluentd. This file contains Linux kernel tuning parameters.

The `tcp_tw_reuse` parameter allows you to reuse sockets in the TIME_WAIT state for new connections.

The `ip_local_port_range` parameter defines the local port range that is used by TCP and UDP to choose the local port.

last-updated: Apr 11, 2022

Example

```
net.ipv4.tcp_tw_reuse = 1
net.ipv4.ip_local_port_range = 10240    65535
```

Log Visualization
Use dashboards to visualize the logs across multiple nodes and clusters.

For information about creating and using dashboards, see the Kibana documentation.

Access the Kibana UI
You can launch the Kibana UI from the Control System or directly from a web browser.

1. Use one of the following method to launch the Kibana UI:
   - In the Control System, go to **Services** and click **Kibana** to launch the Kibana UI in another tab.
   - From a web browser, launch the following URL: `http://<IPaddressOfKibanaNode>:5601`
2. Log on as needed. See Logging on to Kibana on page 1396.

Logging on to Kibana
This page describes the credentials needed to log on to Kibana for secure and nonsecure clusters for MapR 6.0 and later.

Elasticsearch has its own user database, which also serves Kibana. EEP 5.0.0 and MapR Installer 1.9.0 implemented some changes in security that affect the user IDs and passwords you need to log on to Kibana.

EEP 4.x.x

To log on, specify the user and password as follows:

	Secure Cluster	Nonsecure Cluster
User	Type the Kibana `admin` user.	No logon required.
Password	Type `admin`.	No logon required.

EEP 5.0.0 and Later

To log on, specify the user and password as follows:

	Secure Cluster	Nonsecure Cluster
User	Type the Kibana `admin` user ID.	No logon required.
Password	Type the ElasticSearch/Kibana password that you specified during cluster installation.	No logon required.

For More Information

To change the Elasticsearch/Kibana password, see Changing the Password for Elasticsearch and Kibana on page 1397.

For more information about the Grafana versions supported by each EEP, see EEP Components and OS Support on page 5536.

For information about the EEPs supported by different MapR Core versions, see EEP Support and Lifecycle Status on page 5531.

Changing the Password for Elasticsearch and Kibana
Describes how to change the password for Elasticsearch / Kibana.

Kibana gets its password from Elasticseach. To change the password for the `admin` user for Elasticsearch and Kibana:

1. On one of the Elasticsearch nodes, run these commands:

```
ESHOME=/opt/mapr/elasticsearch/elasticsearch-<es_version>
cd $ESHOME/usr/share/elasticsearch/usr/share/elasticsearch/plugins/
search-guard-6
bash tools/hash.sh -p "NewPasswordYouWantForAdmin"
  $2a$12$6ASxMQEBKYPyGUc10RyleOhz3c8RrvPGb7oqLC9xGGwPxJFwOLJtq
```

   The es_version depends on the EEP that you have installed. To determine the `es_version`, use the following table, or refer to Component Versions for Released EEPs on page 5586:

Core Version	EEP Version	<es_version>
6.2.0	8.0.0	6.8.8
6.2.0	7.1.1	6.8.8
6.2.0	7.1.0	6.8.8
6.2.0	7.0.1	6.8.8
6.2.0	7.0.0	6.8.8
6.1.1	6.3.5	6.8.8
6.1.1 or 6.1.0	6.3.4	6.8.8
6.1.1 or 6.1.0	6.3.3	6.8.8
6.1.0	6.3.2	6.8.8
6.1.0	6.3.1	6.8.8
6.1.0	6.3.0	6.5.3
6.1.0	6.2.0	6.5.3
6.1.0	6.1.1	6.5.3
6.1.0	6.1.0	6.5.3
6.1.0	6.0.2	6.2.3
6.1.0	6.0.1	6.2.3
6.1.0	6.0.0	6.2.3

2. Using the hash generated in step 1, edit the `sgconfig/sg_internal_users.yml` file. Change this:

```
admin:
hash: $2a$12$VcCDgh2NDk07JGN0rjGbM.Ad41qVR/YFJcgHp0UGns5JDymv..TOG
#password is: <PasswordSpecifiedAtClusterInstallation>
```

   to this:

```
admin:
hash: $2a$12$6ASxMQEBKYPyGUc10RyleOhz3c8RrvPGb7oqLC9xGGwPxJFwOLJtq
#password is: <NewPasswordYouWantForAdmin>
#hash: $2a$12$VcCDgh2NDk07JGN0rjGbM.Ad41qVR/YFJcgHp0UGns5JDymv..TOG
#password is: <PasswordSpecifiedAtClusterInstallation>
```

3. Save the file.

4. Load the new users database into Elasticsearch:

```
KSPW=$(cat $ESHOME/etc/elasticsearch/.keystore_password)
./tools/sgadmin.sh -h <esHostname> -f sgconfig/sg_internal_users.yml -t
internalusers -ks
../../../../../etc/elasticsearch/keystores/
admin-usr-keystore.jks -kspass $KSPW -ts
../../../../../etc/elasticsearch/keystores/truststore.jks -tspass
$KSPW -cn MaprMonitoring
```

Related concepts

Checking the EEP Version on page 5413
Some MapR Installer operations require you to know the version of the currently installed MapR Ecosystem Pack (EEP). You can check the EEP version easily from within the MapR Installer user interface or derive the EEP version from your repository information.

Related tasks

Checking the MapR Installer Version on page 5412
Some MapR Installer features require you to use the latest version of the Installer. You can check the MapR Installer version easily from within the user interface.

Display Logs Chronologically

To display logs chronologically in Kibana, sort the log events by the event_time column.

@timestamp indicates the time with second precision and is not as precise as event_time which indicates the time with millisecond precision. Therefore, if you want to display logs chronologically in Kibana, sort the log events by the event_time column, not the @timestamp column.

Update the Elasticsearch URL for Kibana

Kibana connects to a single Elasticsearch node to read logs. In the event that Kibana is unable to read logs due to the failure of an Elasticsearch node, configure Kibana to connect to an available Elasticsearch node.

Note: The Elasticsearch node that Kibana connects to by default is determined by the first Elasticsearch node that was specified when the cluster was configured to use MapR Monitoring

1. Open `/opt/mapr/kibana/kibana-<version>/config/kibana.yml`.

2. Update the `elasticsearch.url` parameter to point to an available Elasticsearch node.

   Tip: If you want to configure Kibana to work even if the Elasticsearch node is unavailable, see the Kibana documentation for the steps to configure Kibana to load balance across multiple Elasticsearch nodes.

MapR Monitoring Tips and Troubleshooting

Lists the nuances of monitoring clusters.

Monitoring a Secure Cluster

After regenerating the MapR user ticket, service failures occur for `collectd` and OpenTSDB

If you delete or regenerate the MapR user ticket, the running `collectd` and OpenTSDB services will fail. After updating the MapR user ticket, restart `collectd` and OpenTSDB services.

Monitoring Logs

I notice a sudden increase in `fluentd` logs. What can I do?

A sudden increase in the log file for `fluentd` could mean that a feedback loop is occurring where `fluentd` logs an error in the log file for a `fluentd` issue and that log entry causes yet another error when `fluentd` tries to parse it. In this case, consider disabling the index of `fluentd` logs. See Configure Logs to Index on page 1386.

I see "400 - Rejected by Elasticsearch" messages in the fluentd logs. What can I do?

Messages such as the following can accumulate in the `fluentd` log when a process does not produce logs with valid UTF-8 output:

```
2019-04-25 17:00:11 -0700 [warn]: #0
dump an error event:
error_class=Fluent::Plugin::Elasticsea
rch
ErrorHandler::ElasticsearchError
error="400 - Rejected by
Elasticsearch" location=nil
after setting this option in
es_config.conf
```

In a message such as the following, you might see invalid characters represented as a diamond with a question mark:  . The `"service_name":"collectd"` part of the message indicates that `collectd` is generating the invalid UTF-8 output:

```
[2019-04-30T19:06:29,495][DEBUG]
[o.e.a.b.TransportShardBulkAction]
[mfs73] [mapr_monitoring-2019.05.01]
[4] failed
to execute bulk item (index) index
{[mapr_monitoring-2019.05.01]
[mapr_monitoringv1]
[taQkcWoBCeW3tMAsn1cW],
source[{"my_event_time":"2019-04-30
18:36:39","level":"info","message":"wr
ite_maprstreams plugin: Produced:
Offset: 1247132; Size: 152;
[{\"metric\":\"mapr.streams.produce_ms
gs\",\"value\":448,\"tags\":
{\"fqdn\":\
"qa-node91.qa.lab\",\"clusterid\":\"63
78079583755418855\",\"clustername\":\"
my.cluster.com\"}}]
\n","@timestamp":"2019-04-30T18:36:39.
```

```
000000000-07:00","service_name":"colle
ctd"}]}
org.elasticsearch.index.mapper.MapperP
arsingException: failed to parse
field [message] of type [text]
Caused by:
com.fasterxml.jackson.core.JsonParseEx
ception: Invalid UTF-8 middle byte
0x5c
```

One workaround is to comment out the log producing the invalid character. You can do this in the `fluentd.conf` file. For more information, see Configure Logs to Index on page 1386.

Another workaround is to fix the application that produces the error message. If the log file is produced by an application that you control, change the output of the log producing the invalid character.

Monitoring Metrics

Where should I store the Elasticsearch index?

Elasticsearch requires a lot of disk space. Also, when you upgrade Elasticsearch, the default index directory is removed along with the package update. Therefore, it is recommended to configure a separate filesystem for the index data. It is not recommended to store index data under the `/` or the `/var` filesystem.

Note: If you store the Elasticsearch index on a filesystem that is locally hosted, you will be able to access logs in the event that the MapR cluster is not available.

For more information about the Elasticsearch index and the default index directory, see Log Aggregation and Storage on page 1391.

I see a "Bad Request" error message for my MapR Database metrics? What can I do?

If you have more than 1000 active tables in MapR Database and the MapR monitoring request size to OpenTSDB is more than 4 KB, you may see the following error message:

```
"Sorry but your request was rejected
as being invalid.
The reason provided was: Chunked
request not supported."
```

You can increase the maximum request size of OpenTSDB to up to 64 KB by setting the following parameters in the `opentsdb.conf` file:

```
tsd.http.request.enable_chunked=true
tsd.http.request.max_chunk=65536
```

For more information, see the OpenTSDB configuration guide.

Installation and Configuration Errors

See Troubleshoot MapR Monitoring Installation Errors on page 172

Reconfiguring MapR Monitoring

Changes to an existing cluster, such as the addition of services, may require additional steps to enable the collection of metrics and logs.

Configure Monitoring for Additional Services

When you add services to a cluster where MapR Monitoring is already configured, you must restart collectd and Fluentd services to enable the collection of logs and metrics for the newly added services.

1. Restart the collectd service on each node that runs the service that was added to the cluster.

```
maprcli node services -name collectd -nodes <space separated list of
hostname/IPaddresses> -action restart
```

2. Restart the Fluentd service on each node that runs the service that was added to the cluster.

```
maprcli node services -name fluentd -nodes <space separated list of
hostname/IPaddresses> -action restart
```

Update the Monitoring Storage Nodes

You must reconfigure MapR Monitoring when you add additional OpenTSDB or Elasticsearch nodes, or when you change the OpenTSDB or Elasticsearch node locations.

1. Run configure.sh on each node in the MapR cluster with the `-R`, `-ES`, and `-OT` parameters. Optionally, you can include the `-ESDB` parameter.

```
/opt/mapr/server/configure.sh -R -ES <comma-separate list of
Elasticsearch nodes> \
 -OT <comma-separate list of OpenTSDB nodes> [-ESDB <filepath>]
```

   For the entire list of available `configure.sh` parameters, see configure.sh

   If you encounter any errors after running `configure.sh`, see Troubleshoot MapR Monitoring Installation Errors on page 172

2. If you updated the list of Elasticsearch nodes, restart the all the Fluentd, Elasticsearch, and Kibana services.

```
maprcli node services -name fluentd -nodes <space separated list of
Fluentd nodes> -action restart
```

```
maprcli node services -name elasticsearch -nodes <space separated list
of Elasticsearch nodes> -action restart
```

```
maprcli node services -name kibana -nodes <space separated list of
Kibana nodes> -action restart
```


last-updated: Apr 11, 2022

3. If you updated the list of OpenTSDB nodes, restart the collectd, OpenTSDB, and Grafana services.

```
maprcli node services -name opentsdb -nodes <space separated list of
OpenTSDB nodes> -action restart
```

```
maprcli node services -name collectd -nodes <space separated list of
collectd nodes> -action restart
```

```
maprcli node services -name grafana -nodes <space separated list of
Grafana nodes> -action restart
```

Configuring Security

Describes how to configure security and manage secure clusters.

Configuring Data-Fabric Security

Provides usage information for frequently used security functionality, including Access Control Lists (ACLs), Access Control Expressions (ACEs), file permissions, and subnet whitelisting.

Note: Release 6.1 makes it easier to secure new data-fabric installations. See Using the Enable MapR Secure Cluster Option on page 5427 in the MapR Installer on page 5395.

Wired encryption and authentication (including impersonation) for the data-fabric platform and for all supported ecosystem products are enabled on all new installations through MapR Installer on page 5395 and through manual installation by running the `configure.sh on page 2053` command with the `-secure` option.

You can enable security features at any time, but additional configuration is required for the individual components to work with security enabled. This section discusses initial configuration of a secure cluster as well as other forms of security.

The following access control elements are available irrespective of whether you have enabled security features for your cluster. Additionally, once security features are enabled, these elements benefit from encrypted traffic within the cluster and strong authentication to the cluster.

- Access Control Lists (ACLs) for the cluster, the volumes in the cluster, and the MapReduce application queue
- *ACE*s control user permissions for directories, files, and MapR Database tables that are stored natively
- File permissions for objects in the MapR File System layer
- Subnet whitelisting restricts access to the cluster's FileServer service

On clusters with security features enabled, ecosystem components may require additional configuration. For example, Hive functionality has different security requirements depending on the interaction between the HiveServer2 component, the Hive command-line interface, and the Hive metastore.

See the MapR Security Support Matrix on page 5602 for more information about supported security options for Ecosystem components. See the specific Ecosystem component in Ecosystem Components on page 3174 for information on security configuration.

See Security Vulnerabilities on page 6569 for a list of known vulnerabilities.

Verifying if Files Needed for Security are Present

When you run `configure.sh` with the `-secure` option, the following files are automatically created in the `/opt/mapr/conf` directory. To ensure that security is properly configured, navigate to the `/opt/mapr/conf` directory and verify that the files are present.

Master value controlling the cluster secure or non-secure state	*File or command:* `/opt/mapr/conf/mapr-clusters.conf` `maprcli dashboard info -cluster <clusterName> -json | grep secure` *Default secure setting:* `secure=true` *Alternate possible values/notes:* `secure=false` disables security on restarting the cluster.
Data-fabric service account	*File or command:* `sudo passwd -S mapr` *Default secure setting:* Site Specific Password *Alternate possible values/notes:* No password. Use `su` to access.
CLDB key file	*File or command:* `/opt/mapr/conf/cldb.key` *Default secure setting:* Created at install, do not change *Alternate possible values/notes:* Must exist on all CLDB nodes and be identical.
Server ticket	*File or command:* `/opt/mapr/conf/maprserverticket` *Default secure setting:* Created at install, do not change *Alternate possible values/notes:* Must exist on all cluster nodes and be identical.
User ticket	*File or command:* `/opt/mapr/conf/maprusertickets` *Default secure setting:* Created at install, do not change *Alternate possible values/notes:* Must exist on all cluster nodes and be identical. This ticket is owned and used by the service account as needed.
SSL keys	*File or command:* `/opt/mapr/conf/ssl_truststore` `/opt/mapr/conf/ssl_keystore` *Default secure setting:* Created at install, and should rarely change. These keys are used by web and REST HTTPS interfaces. *Alternate possible values/notes:* Add site specific certificates with the keytool utility.
Java (JAAS) authentication service settings	*File or command:* `/opt/mapr/conf/mapr.login.conf` *Default secure setting:* Created at install, do not change *Alternate possible values/notes:* Must exist on all cluster nodes and be identical.

last-updated: Apr 11, 2022

Roles for use with ACEs	*File or command:* `/opt/mapr/conf/m7_permissions_roles_refimpl.conf` *Default secure setting:* Specific roles defined using automation *Alternate possible values/notes:* Use should be deprecated. Linux groups are a much better method, centralized and consistent with enterprise standards.
Default security settings for some data-fabric services	*File or command:* `/opt/mapr/conf/env.sh` *Default secure setting:* Created at install, do not change *Alternate possible values/notes:* Must exist on all cluster nodes and be identical. You can view the list of settings by using this command: `grep -i secure env.sh`
ZooKeeper security setting	*File or command:* `/opt/mapr/zookeeper/zookeeper-$zkver/conf/zoo.cfg` *Default secure setting:* `authMech=MAPR-SECURITYauthProvider.1=org.apache.zookeeper.server.auth.SASLAuthenticationProvide` *Alternate possible values/notes:* `authMech=SIMPLE-SECURITY`
JMX remote access (debug and metrics monitoring)	*File or command:* `/opt/mapr/conf/jmxremote.{access,password}` *Default secure setting:* `read-only` and with the password limited to the data-fabric service account. *Alternate possible values/notes:* `read-write` but is not recommended

Determining if Wire-Level Security is Enabled Using the CLI

When you run `configure.sh` with the `-secure` option, wire-level security is automatically enabled at the cluster level. You can, optionally, disable wire-level security at the individual volume-level. To determine if wire-level security is enabled for a volume, run the following command:

```
/opt/mapr/bin/maprcli volume list -json |grep wire
```

This command returns the value of `wireSecurity` as `1` if wire-level security is enabled for the volume; `0` otherwise.

Enabling Cluster Wide Data Access Auditing

To enable auditing data access operations at a cluster level, run:

```
/opt/mapr/bin/maprcli audit data -enabled
```

Determining if per Volume Data Access Auditing is Enabled

To determine if auditing data access operations is enabled for a volume, run:

```
/opt/mapr/bin/maprcli volume info -name <volume_name> -json | grep -i
'audited\|coalesce'
```

This command returns the value of `audited` as `1` if data access auditing is enabled for the volume; `0` otherwise.

Getting Started with MapR Security
Describes quick implementation of security.

MapR 6.1 introduced enhanced security settings that simplify the process of creating secure clusters. For a brief introduction, see Security. To learn how to secure a cluster, see this course.

To set up a secure cluster:

1. Enable cluster security, authentication, and wire-level encryption by running configure.sh if you performed a manual installation.

   See Enabling Security on page 1405 for more information.

   Note: If you selected the Using the Enable MapR Secure Cluster Option on page 5427 when installing with the MapR Installer on page 5395, you can proceed to the next step.

2. Generate MapR user tickets to authenticate with your username and password.

   See Generating a MapR User Ticket on page 1426 for more information.

3. Configure each Ecosystem component, where necessary, for security.

   See Security and Ecosystem Components on page 703 for more information.

4. (Optional) Enable encryption of data at rest at the cluster level and selectively for volumes as well.

   See Enabling Encryption of Data at Rest on page 1413 for more information.

5. (Optional) Turn on auditing for the cluster and for directories that contain sensitive data.

   See Enabling and Disabling Auditing of Cluster Administration on page 758 and Enabling and Disabling Auditing of Data Access Operations on page 758 for more information.

6. (Optional) Enable authorization using ACEs for files, tables, streams, or volumes and ACLs for administrative activities that can be performed on the cluster.

   See Managing Access Control Expressions on page 1448 and Managing Access Control Lists on page 1445 for more information.

Tip: After you enable security, review the System Behavior Changes After Enabling Security on page 1410.

Enabling Security
Describes how to enable security for the cluster, platform, ecosystem components, and network-based connections.

The following steps enable:

- Security for the cluster nodes
- Wire-level encryption for the platform and ecosystem components
- Authentication for all network-based connections
- (Optional) Data-at-rest encryption on the cluster

When you set up a cluster, you must run the `configure.sh on page 2053` script on each node that you want to add to the cluster. After you enable security, review the System Behavior Changes After Enabling Security on page 1410.

last-updated: Apr 11, 2022

Basic Procedure

To enable security for the cluster, follow these steps in order:

1. If the cluster is running, shut it down.
2. Delete the `ssl_keystore`, `ssl_keystore.p12`, `ssl_keystore.pem`, `ssl_truststore`, `ssl_truststore.p12`, and `ssl_truststore.pem` files on a machine where wire-level security is not enabled, as the `configure.sh` script fails if you already have these files in the directory.

   Note: The `ssl_keystore`, `ssl_keystore.p12`, `ssl_keystore.pem`, `ssl_truststore`, `ssl_truststore.p12`, and `ssl_truststore.pem` files are generated during installation of the Web server even if you did not enable security.

   For example, to delete the files, run the following commands:

   ```
   cd /opt/mapr/conf
   /bin/rm -f ssl_keystore ssl_keystore.p12 ssl_keystore.pem ssl_truststore
   ssl_truststore.p12 ssl_truststore.pem
   ```

   If you are re-running the script due to an invocation error from a previous run, the `cldb.key` and `maprserverticket` files may have already been created. Delete these files, as the script fails if you already have these files in the directory. For example, run the following command to delete these files:

   ```
   cd /opt/mapr/conf
   /bin/rm -f cldb.key maprserverticket ssl_keystore ssl_keystore.p12
   ssl_keystore.pem ssl_truststore ssl_truststore.p12 ssl_truststore.pem
   ```

3. Run the `configure.sh` script with the `-secure -genkeys -dare` options on the first CLDB node in your cluster:

   ```
   /opt/mapr/server/configure.sh -secure -dare -genkeys -Z
   <Zookeeper_node_list> -C <CLDB_node_list> -N <cluster_name>
   ```

   where both `<Zookeeper_node_list>` and `<CLDB_node_list>` have the form `hostname[:port_no][,hostname[:port_no]...]` and `-N <cluster_name>` specifies the cluster name. For the hostname, specify a fully-qualified domain name (FQDN) as described in Connectivity on page 133. Do not specify an alias or IP address. The `-dare` option is required only if you wish to enable data at rest encryption at the cluster-level.

   Important: You must run `configure.sh -dare -genkeys` only *once* on one CLDB node, since the resulting files must be copied to other nodes.

   This command generates the following files in the `/opt/mapr/conf` directory:

   - `cldb.key`
   - `dare.master.key`
   - `maprserverticket`
   - `ssl_keystore`
   - `ssl_keystore.p12`
   - `ssl_keystore.pem`
   - `ssl_truststore`

last-updated: Apr 11, 2022

- `ssl_truststore.p12`
- `ssl_truststore.pem`

Note: The `dare.master.key` file is generated only if data at rest encryption is enabled on the cluster.

Tip: A comprehensive listing of the Trust and Key Store files is at: Understanding the Key Store and Trust Store Files on page 1408.

4. Copy the `cldb.key` to any node with the CLDB or Zookeeper service installed, and copy the `dare.master.key` to any node with the CLDB service installed.

   Note: Copy the `dare.master.key` file only if you are enabling data at rest encryption on the cluster.

5. Verify that the files `ssl_keystore`, `ssl_keystore.p12`, `ssl_keystore.pem`, `ssl_truststore`, `ssl_truststore.p12`, and `ssl_truststore.pem` are owned by the user that runs cluster services. This user is `mapr` by default. Also, the `maprserverticket`, `ssl_keystore`, `ssl_keystore.p12`, and `ssl_keystore.pem` files must have their UNIX permission-mode bits set to `400`, and the `ssl_truststore`, `ssl_truststore.p12`, and `ssl_truststore.pem` files must be readable to all users.

6. Copy the `maprserverticket`, `ssl_keystore`, `ssl_keystore.p12`, `ssl_keystore.pem`, `ssl_truststore`, `ssl_truststore.p12`, and `ssl_truststore.pem` files to the `/opt/mapr/conf` directory of *every* node in the cluster.

7. Run `configure.sh` on each existing node in the cluster using the same arguments as in Step 3 but without the `-genkeys` option.

   ```
   /opt/mapr/server/configure.sh -secure -dare -Z <Zookeeper_node_list> -C
   <CLDB_node_list> -N <cluster_name>
   ```

   The `-secure` option indicates that security must be enabled on the node where the command is run and the `-dare` option indicates that data at rest encryption must be enabled on the node and must be specified only if it was specified in Step 3.

   Important: You must also do this on any nodes that you add to the cluster in the future.

8. Copy the `ssl_truststore`, `ssl_truststore.p12`,and `ssl_truststore.pem` files to any client nodes outside the cluster.

   Important: If you run `configure.sh -secure` on a node *before* you copy the necessary files to that node, the command fails.

9. Optionally, enable encrypted quorum ZooKeeper communication. See zoo.cfg on page 2220 for more information.

10. Log in as the `mapr` superuser using the maprlogin command: `maprlogin password` (in this command, `password` is literal text).

11. If clients will connect to multiple secure clusters, merge the `ssl_truststore` files with the `/opt/mapr/server/manageSSLKeys.sh` tool. See Setting Up the Client for more information on MapR clients.

last-updated: Apr 11, 2022

Advanced Procedure

In certain situations, you may opt for variant of the basic procedure. Such situations include, but are not limited to the following:

- You are running the script on a host that is configured without a domain name.
- You have a cluster where all the machines do not have the same domain name.
- You wish to import your own custom certificates instead of the self-signed certificates generated by the `configure.sh` utility.

Running on Hosts with no Configured Domain Name

When used without the `-certdomain` argument, the `configure.sh` script discovers the domain name of the node on which it is being executed using the `hostname -d` command and then creates a 100-year self-signed certificate using the PKCS#1 v1.5 with SHA-512 hash function (SHA512withRSA) with a wildcard certificate with the common name `(CN) *.<domain>`. For example, if `hostname -d` returns the domain name `mycompany.com`, then the CN of the certificate is `*.mycompany.com`. This certificate works for all machines within the `mycompany.com` domain and can therefore be copied to all cluster nodes as specified in Step 5 in the Basic Procedure.

Certificate generation fails if the host that you are running the script from is configured without a domain name. To fix this, modify your machine configuration so that `hostname -d` returns a non-empty string and then run the `configure.sh` script.

Alternatively, re-run the script with the `-certdomain` option as shown in Step 3 of the Basic Procedure:

```
/opt/mapr/server/configure.sh -secure -genkeys -certdomain <domain_name> -Z
<Zookeeper_node_list> -C <CLDB_node_list> [ -N <cluster_name> ]
```

Securing Clusters with Multiple Domain Names

Generally, all machines within a cluster should belong to the same domain. In the unusual case where you have a cluster with different machines belonging to different domains, applications that perform hostname verification can fail if you run the `configure.sh` script (as described in Step 3 of the Basic Procedure on page 1406) to generate a single-domain wildcard certificate. In this case, you must provide your own multi-domain wildcard certificate and import the custom certificate into the keystore as described in Importing a Certificate Authority Signed (CA Signed) SSL Certificate Into a MapR Cluster.

Using Custom Certificates

To import your own custom certificates into the keystore instead of using the self-signed certificates, see Importing a Certificate Authority Signed (CA Signed) SSL Certificate Into a MapR Cluster.

Related reference

zoo.cfg on page 2220
Lists the ZooKeeper configuration file.

Understanding the Key Store and Trust Store Files
Provides a comprehensive listing of the key store and trust store files.

Key Stores and Trust Stores in Release 6.1.0

The following files are generated by running `configure.sh -dare -genkeys` on a CLDB node. Alternatively, you can generate them by running the manageSSLKeys.sh on page 2120 script. The `ssl_keystore`, `ssl_keystore.p12`, `ssl_keystore.pem`, `ssl_truststore`, `ssl_truststore.p12`, and `ssl_truststore.pem` files are also generated during installation of the Web server, even if you did not enable security. For more information, see Enabling Security on page 1405.

cldb.key *Location:* `/opt/mapr/conf`

	Description: The CLDB key file. This file must exist on all CLDB nodes and be identical.
dare.master.key	*Location:* `/opt/mapr/conf`
	Description: The key file that enables data-at-rest encryption. The `dare.master.key` file is generated only if data at rest encryption is enabled on the cluster. This file must be copied to all the nodes with the CLDB service installed.
maprserverticket	*Location:* `/opt/mapr/conf`
	Description: The server ticket. This file must exist on all cluster nodes and be identical.
ssl-client.xml	*Location (symlink):* `/opt/mapr/conf`
	Location (file): `${MAPR_HOME}/hadoop/hadoop-<version>/etc/hadoop/ssl-client.xml`
	Description: Contains the SSL configuration for the client in XML format.
ssl_keystore	*Location:* `/opt/mapr/conf`
	Description: This file is needed on all nodes where the webserver is running.
ssl_keystore.p12	*Location:* `/opt/mapr/conf`
	Description: When upgrading from MapR Core 5.2.2 or Core 6.0.x to MapR 6.1 or later, create the `ssl_keystore.p12` and `ssl_truststore.p12` files, and copy them to the `/opt/mapr/conf` directory on all nodes in the cluster.The `.p12` files are required to generate the `.pem` files needed by Grafana and the Data Access Gateway. This step is necessary only for manual upgrades.
ssl_keystore.pem	*Location:* `/opt/mapr/conf`
	Description: When upgrading from MapR Core 5.2.2 or Core 6.0.x to MapR 6.1 or later, create the `ssl_truststore.pem` and `ssl_keystore.pem` files, and copy them to the `/opt/mapr/conf` directory on all nodes in the cluster. The Data Access Gateway, Grafana, and Hue components use these files. This step is necessary only for manual upgrades.
ssl-server.xml	*Location (symlink):* `/opt/mapr/conf`
	Location (file): `${MAPR_HOME}/hadoop/hadoop-<version>/etc/hadoop/ssl-server.xml`
	Description: Contains the SSL configuration for the server in XML format.
ssl_truststore	*Location:* `/opt/mapr/conf`
	Description: contains the certificates required by nodes initiating communication over TLS.
ssl_truststore.p12	*Location:* `/opt/mapr/conf`
	Description: When upgrading from MapR Core 5.2.2 or Core 6.0.x to MapR 6.1 or later, create the `ssl_keystore.p12` and `ssl_truststore.p12` files, and copy them to the `/opt/mapr/conf` directory on all nodes in the cluster.The `.p12` files are required to generate the `.pem` files needed by Grafana

last-updated: Apr 11, 2022

	and the Data Access Gateway. This step is necessary only for manual upgrades.
ssl_truststore.pem	*Location:* `/opt/mapr/conf` *Description:* When upgrading from MapR Core 5.2.2 or Core 6.0.x to MapR 6.1 or later, create the `ssl_truststore.pem` and `ssl_keystore.pem` files, and copy them to the `/opt/mapr/conf` directory on all nodes in the cluster. The Data Access Gateway, Grafana, and Hue components use these files. This step is necessary only for manual upgrades.

Disabling Security

To disable security features for your cluster:

1. If the cluster is running, shut it down.
2. On all nodes, run the `configure.sh` on page 2053 script with the `-unsecure` option and specify the CLDB and ZooKeeper nodes.

   ```
   configure.sh -unsecure -C <CLDB_Node> -Z <ZK_Node> -N <cluster_name>
   ```

3. Start the cluster.

System Behavior Changes After Enabling Security

After enabling security features for your cluster, the following behaviors change:

- Users must authenticate with the `maprlogin` utility.
- Components that have web UIs, such as the Control System, Hive, and Oozie, require authentication.

  Warning: Note that you must also complete the PAM configuration to set up user authentication for the Control System logins.
- Several components that communicate over HTTP use HTTPS instead.
- Encryption is used for network traffic.
- Access to a cluster using URIs that use the CLDB node's name or IP address, instead of the cluster name, is no longer supported, as in the following examples. The following URIs no longer work after enabling security:

  ```
  http://cldb1.cluster.com:7222/f1
  ```

  ```
  http://10.10.20.10:7221/f1
  ```

  The following URIs work after enabling security:

  ```
  http:///f1 <access f1 in default cluster>
  ```

  ```
  http://my.cluster.com/f1
  ```

Managing Encryption for MapR Core

Provides information that allows you to use encryption across the MapR platform.

This section describes how to enable security for data at rest and on the wire as well as general security components and system changes.

last-updated: Apr 11, 2022

Enabling Wire-level Security

Wire-level security encrypts data transmission between the nodes in your cluster.

You can enable encryption for data on the wire at the volume level only if security is enabled at the cluster level. If necessary, refer to Determining if a Cluster is Secure Using the CLI and REST API on page 1421 to determine if the cluster is secure before enabling wire-level encryption on a volume. If your cluster is enabled for security, wire-level security is enabled by default on all new volumes and no additional steps are required. This section describes how to enable wire-level security on new and existing volumes (if the volume is already not enabled for wire-level security).

Enabling Wire-level Security for a Volume Using the Control System

1. Log in to the Control System and click **Data** > **Volumes**.
2. Click **Create Volume** to display the **Create New Volume** page or go to the Edit Volume page.
3. Set the value for the **Data on Wire Encryption** property to **Yes** (to enable).
   See Creating a Volume on page 864 or Modifying a Volume on page 892 for more information.
4. Complete the steps to create or modify the volume.
   See Creating a Volume on page 864 or Modifying a Volume on page 892 for more information.

Enabling Wire-Level Security for a Volume Using the CLI and REST API

CLI

Set the value for the `wiresecurityenabled` parameter to `true` when you:

- Create the volume. For example:

```
maprcli volume create -name
<volName> -path
<volMountPath> -wiresecurityenabled
 true
```

- Modify the volume. For example:

```
maprcli volume modify -name
<volName> -wiresecurityenabled true
```

REST

Send a request of type POST and set the value for the `wiresecurityenabled` parameter to `true` when you:

- Create the volume. For example:

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=<volName>&path=<volMountPath>&
wiresecurityenabled=true' --user
mapr:mapr
```

- Modify the volume. For example:

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/modify?
name=<volName>&wiresecurityenabled=
true' --user mapr:mapr
```


last-updated: Apr 11, 2022

See `volume create` on page 1931 and `volume modify` on page 2005 for more information.

Disabling Wire-level Security

You can disable wire encryption for a volume using the Control System, the CLI, and REST API.

Disabling Wire-level Security for a Volume Using the Control System

1. Log in to the Control System and click **Data** > **Volumes**.
2. Click **Create Volume** to display the **Create New Volume** page or go to the Edit Volume page.
3. Set the value for the **Data on Wire Encryption** property to **No** (to disable).
   See Creating a Volume on page 864 or Modifying a Volume on page 892 for more information.
4. Complete the steps to create or modify the volume.
   See Creating a Volume on page 864 or Modifying a Volume on page 892 for more information.

Disabling Wire-Level Security for a Volume Using the CLI and REST API

You can disable encryption of data on wire at the volume level.

CLI

Set the value for the `wiresecurityenabled` parameter to `false` when you:

- Create the volume. For example:

```
maprcli volume create -name
<volName> -path
<volMountPath> -wiresecurityenabled
 false
```

- Modify the volume. For example:

```
maprcli volume modify -name
<volName> -wiresecurityenabled
false
```

REST

Send a request of type POST and set the value for the `wiresecurityenabled` parameter to `false` when you:

- Create the volume. For example:

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=<volName>&path=<volMountPath>&
wiresecurityenabled=false' --user
mapr:mapr
```

- Modify the volume. For example:

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/modify?
name=<volName>&wiresecurityenabled=
false' --user mapr:mapr
```

See `volume create` on page 1931 and `volume modify` on page 2005 for more information.

Enabling Encryption of Data at Rest

You can enable or disable data at rest encryption at the volume level using the Control System, CLI, and REST API if encryption of data at rest is enabled at the cluster level. If you installed using the MapR Installer on page 5395 and selected the **Enable DARE** option, the cluster is automatically enabled for data at rest encryption during installation. If you are upgrading or if you did not enable data at rest encryption at the time of installation, you can enable encryption of data at rest at the cluster level if the cluster is a secure cluster. See Enabling Data at Rest Encryption at the Cluster Level from the Command-Line on page 1413 below for more information.

If encryption is enabled at the cluster level, data at rest encryption is also enabled at the volume level by default through the `mapr.volume.dare.default` configuration parameter. If you do not wish to encrypt data at rest in a volume, you can disable encryption when you create a volume; you cannot modify data at rest encryption setting on a volume after the volume is created. See Enabling or Disabling Data at Rest Encryption at the Volume Level Using the Control System on page 1414 or Enabling or Disabling Data at Rest Encryption at the Volume Level Using the CLI and REST API on page 1415 below for more information.

Standard volumes inherit the data at rest encryption setting from a volume by default if the `inherit` property is specified. If you create a mirror volume for a (source) volume enabled for data at rest encryption, the mirror volume:

- Inherits the data at rest encryption setting from the source volume if the mirror volume is in the same cluster as the source volume or if the mirror volume is on a remote cluster enabled for encryption of data at rest.
- Does not inherit the data at rest encryption setting from the source volume if the mirror volume is on an unsecure cluster or if the mirror volume is on secure cluster that is not enabled for encryption of data at rest.

Note: If you want to create a mirror volume enabled for data at rest encryption for a source volume not enabled for data at rest encryption, set the value to `true` for `dare` property when creating the mirror volume.

This section describes how to enable data at rest encryption at the cluster and volume levels.

Enabling Data at Rest Encryption at the Cluster Level from the Command-Line

To enable encryption at the cluster level:

1. Run `configure.sh` on page 2053 with the `-dare` and `-genkeys` option on a CLDB node.

   For example:

   ```
   /opt/mapr/server/configure.sh -N
   <cluster-name> -secure -dare -genkeys -C <CLDB-node-list> -Z
   <ZK-node-list>
   ```

   The Master Key is generated and stored in a master key file at `/opt/mapr/conf/dare.master.key`. When CLDB comes up, it sets the value for the following CLDB configuration properties to `1` to allow encryption of data at the volume level:

   - `mfs.enable.dare`
   - `mapr.volume.dare.default`

   You can verify that these parameter values are correctly set by running the `config load` on page 1585 command. For example:

   ```
   # maprcli config load -json
   {
   ```


last-updated: Apr 11, 2022

```
    "timestamp":1524669009979,
    "timeofday":"2018-04-25 08:10:09.979 GMT-0700 AM",
    "status":"OK",
    "total":1,
    "data":[
        {
            "bulk.container.create.support":"1",
            "cldb.accept.unknown.replica.delay.mins":"5",
            ...,
            "mapr.volume.dare.default":"1",
            ...,
            "mfs.enforce.dare":"1",
            ....,
            "pernode.numcntrs.alarm.thr":"50000"
        }
    ]
}
```

Note: An informational alarm, reminding you to create a copy of the master key, is raised on the CLDB node. This is because, the loss of the master key can be catastrophic and irreversible. Loss of this key might result in loss of data. Before clearing the alarm, it is strongly recommended to create a copy of this file in other location(s) for backup purposes.

2. Copy the master key file (generated in step 1 above) to `/opt/mapr/conf` directory on all the other CLDB nodes in the cluster.
3. Run `configure.sh` on page 2053 on all the nodes, except the node on which the command was run to generate the master key (in step 1), with the `-dare` option.

   For example:

```
/opt/mapr/server/configure.sh -N <cluster-name> -secure -dare -C
<CLDB-node-list> -Z <ZK-node-list>
```

If this is a new installation, no additional steps are needed. You have enabled data at rest encryption at the cluster level and, by default, all new volumes are enabled for data at rest encryption. You can, however, still create volumes that are not enabled for encryption of data at rest. See Enabling or Disabling Data at Rest Encryption at the Volume Level Using the Control System on page 1414 or Enabling or Disabling Data at Rest Encryption at the Volume Level Using the CLI and REST API on page 1415 for more information.

Enabling or Disabling Data at Rest Encryption at the Volume Level Using the Control System

You can enable data at rest encryption at the volume level only if data at rest encryption is enabled at the cluster level. If necessary, refer to Determining if a Secure Cluster is Enabled for Encryption Using the Control System on page 1421 to determine if the cluster is enabled for encryption of data at rest before enabling data at rest encryption on a volume.

Note: If you do not wish to encrypt data-at-rest in a volume, you can disable encryption when you create a volume; you cannot modify data-at-rest encryption setting on a volume after the volume is created.

To enable or disable data-at-rest encryption for a new volume using the Control System:

1. Log in to the Control System and click **Data** > **Volumes**.
2. Click **Create Volume** to display the **Create New Volume** page.
3. Select volume type, specify values for required and optional properties, and set the value for the **Data at Rest Encryption** property to **Yes** (to enable) or **No** (to disable).

   See Creating a Volume on page 864 for more information.

4. Click **Create Volume** to create a volume enabled for encryption of data at rest.

Enabling or Disabling Data at Rest Encryption at the Volume Level Using the CLI and REST API

You can enable DARE at the volume level only if data at rest encryption is enabled at the cluster level. If necessary, refer to Determining if a Secure Cluster is Enabled for Encryption of Data at Rest Using the CLI and REST API on page 1422 to determine if the cluster is enabled for encryption of data at rest before enabling a volume for data at rest encryption.

Note: If you do not wish to encrypt data-at-rest in a volume, you can disable encryption when you create a volume; you cannot modify data-at-rest encryption setting on a volume after the volume is created.

CLI

Set the value for the `dare` parameter to one of the following when you create the volume:

- `true` to enable data-at-rest encryption.

  Note: This is the default value.

  For example:

```
maprcli volume create -name
<volName> -path <volMountPath>
[-dare true]
```

- `false` to disable data-at-rest encryption.

  For example:

```
maprcli volume
create -name <volName> -path
<volMountPath> -dare false
```

REST

Send a request of type POST and set the value for the `dare` parameter to one of the following when you create the volume:

- `true` to enable data-at-rest encryption.

  Note: This is the default value.

  For example:

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=<volName>&path=<volMountPath>[
&dare=true]' --user mapr:mapr
```

- `false` to disable data-at-rest encryption.

  For example:

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=<volName>&path=<volMountPath>&
dare=false' --user mapr:mapr
```

See `volume create` on page 1931 for more information.

Converting to Cluster Enabled for Data at Rest Encryption

Enable Data-At-Rest Encryption for a Cluster.

last-updated: Apr 11, 2022

MapR's data-at-rest encryption allows you to protect the data in the event a disk is compromised. Using the MapR Installer on page 5395, you can select the **Enable MapR DARE** option during an incremental install after upgrading. You can also convert a cluster not enabled for encryption at rest to a cluster enabled for encryption at rest from the command-line during a Manual Rolling Upgrade Description on page 319 or after an Offline and Manual Upgrade Procedure on page 316.

Note: Encryption of data at rest can only be enabled on a secure cluster.

To convert:

1. Perform the following steps on a CLDB node to generate the master key for encryption of data at rest:
   a) Stop Warden on a CLDB node by running the following command:
   ```
   /bin/systemctl stop mapr-warden
   ```
   b) Install MapR v6.1 packages if this is a rolling upgrade.
   See Upgrading MapR Core Without the MapR Installer on page 309. You can skip this step if you have already installed the MapR v6.1 packages.
   c) Run the `configure.sh on page 2053` command as follows:
   ```
   /opt/mapr/server/configure.sh -genkeys -nocerts -dare  -R
   ```
   When you run the `configure.sh on page 2053` command with the `genkeys` and `dare` options, a MasterKey file is generated and stored in `/opt/mapr/conf/dare.master.key`.
   Important: You must create a copy of this file in other location(s) for backup purposes. Loss of this key will result in loss of cluster.
   d) Start Warden by running the following command:
   ```
   /bin/systemctl start mapr-warden
   ```
2. Copy the data at rest encryption master key file (generated above) to the `/opt/mapr/conf` directory on all the other CLDB nodes on the cluster.
3. Perform the following steps on all the nodes, one node at a time if you are doing a rolling upgrade, in the cluster:
   a) Stop Warden by running the following command:
   ```
   /bin/systemctl stop mapr-warden
   ```
   b) Install MapR v6.1 packages if this is a rolling upgrade.
   See Upgrading MapR Core Without the MapR Installer on page 309. You can skip this step if you have already installed the MapR v6.1 packages.
   c) Run the `configure.sh on page 2053` command as follows:
   ```
   /opt/mapr/server/configure.sh -dare -R
   ```
   d) Start Warden by running the following command:
   ```
   /bin/systemctl start mapr-warden
   ```

4. Enable encryption of data at rest through the `mfs.feature.dare` property, and optionally enable other features, if they are not yet enabled.

   Run `maprcli cluster feature enable -name mfs.feature.dare` to enable encryption of data at rest.

   See Step 4: Enable New Features on page 330 for information on enabling other features.

5. Specify whether (`1`) or not (`0`) to convert all existing volumes not enabled for encryption of data at rest to volumes enabled for encryption of data at rest by setting the value for the `cldb.enforce.old.volumes.dare` property using the `config save` on page 1586 command.

   By default, all existing volumes are converted to volumes enabled for data at rest encryption because the default value for `cldb.enforce.old.volumes.dare` property is `1`. To not convert all existing volumes to volumes enabled for data at rest encryption, run the following command:

   ```
   maprcli config save -values '{"cldb.enforce.old.volumes.dare":"0"}'
   ```

6. Format the storage pools (SPs) on the nodes for data at rest encryption, one node and one SP at a time. That is:

   a) Decommission the node.

      See Decommissioning a Node for Enabling Encryption of Data at Rest on page 1417 for more information.

   b) Format the SPs on the node.

      See Formatting a Storage Pool for Encryption of Data at Rest on page 1418 for more information.

   c) Move the node back to the original topology.

      See Changing the Topology of one or more Nodes on page 805 for more information.

      CAUTION: Wait for few minutes and make sure that the Under Replicated Alarm is cleared for all the Volumes on a SP, before formatting the next SP.

7. Enable encryption of data at rest at the cluster-level by running the following command:

   ```
   maprcli config save -values '{"mfs.enforce.dare":"1"}'
   ```

8. Verify that encryption of data at rest is enabled at the cluster-level by running the following command:

   ```
   maprcli config load -json | grep dare
   ```

   Your output should look similar to the following:

   ```
   "cldb.enforce.old.volumes.dare":"1",
   "mapr.default.dare.alarm.pending":"0",
   "mapr.volume.dare.default":"1",
   "mfs.enforce.dare":"1",
   "mfs.feature.dare":"1",
   ```

Encryption of data at rest is enabled by default for all new volumes on the cluster. You can disable data at rest encryption for volumes that do not require encryption of data at rest. See Enabling or Disabling Data at Rest Encryption at the Volume Level Using the Control System on page 1414 or Enabling or Disabling Data at Rest Encryption at the Volume Level Using the CLI and REST API on page 1415.

Decommissioning a Node for Enabling Encryption of Data at Rest

Decommission Node to Enable DARE.

last-updated: Apr 11, 2022

Use the following procedure to decommission a node for preparing the node for data at rest encryption.

1. Drain the node of data by moving the node to the `/decommissioned` physical topology.
   See Changing the Topology of one or more Nodes on page 805 for more information. By default, all the data on a node in the `/decommissioned` topology is migrated to volumes and nodes in the `/data` topology.
2. Run the following command to check if volumes are present on the node:
   ```
   maprcli dump volumenodes -volumename <volume> -json | grep <ip:port>
   ```
   Note: Run this command for each non-local volume in your cluster to verify that the node being decommissioned is not storing any volume data.
3. Stop Warden on the node as the root user or use `sudo`.
   For example, run the following command to stop Warden using sudo: `sudo /bin/systemctl stop mapr-warden`
4. Run `configure.sh` on page 2053 utility with the `-C` and `-Z` options on all the other nodes if the node you moved is a CLDB or ZooKeeper node.
5. Remove the node.
   See Removing One or More Nodes on page 810 for more information.
6. Verify that the SPs associated with the node (that you removed in the previous step) are not displayed in the list of storage pools in the CLDB service information page.
   See Viewing CLDB Information on page 1169 for more information.

Formatting a Storage Pool for Encryption of Data at Rest

This section describes how to reformat a storage pool for data at rest encryption and determine if a storage pool is enabled for data at rest encryption.

Formatting a Storage Pool on a Node for Encryption of Data at Rest

1. Determine the number of SPs on the node by running the following command:
   ```
   /opt/mapr/server/mrconfig sp list
   ```
2. Select the SP to reformat and run the following command:
   ```
   mrconfig sp offline <sp path>
   ```
3. Wait for the Data Under-Replicated on page 2241 alarm to clear and ensure that there are no Data Unavailable on page 2240 alarms.
4. Remove the disk from MapR File System.
   See Removing Disks from the File System on page 838 for more information.
5. Format the disk on the node.
   See Formatting Disks on a Node From the Command-line on page 840 for more information.
6. Add disk to MapR File System.
   See Adding Disks to MapR File System on page 837 for more information.

Determining if a Storage Pool is Enabled for Data at Rest Encryption

- Run the following command to determine if an SP is enabled for data-at-rest encryption:

```
/opt/mapr/server/mrconfig sp list -v
ListSPs resp: status 0:2
No. of SPs (2), totalsize 3518339 MB, totalfree 691937 MB

SP 0: name SP1, Online, size 1761217 MB, free 377127 MB, path /dev/
sdb, log 200 MB, port 5660, guid 9dd586829e179476005b0ce23f0dae3c,
clusterUuid -7600986066553737256-4524271553806028052, disks /dev/sdb /dev/
sdd, dare 1
SP 1: name SP2, Online, size 1757121 MB, free 314809 MB, path /dev/
sde, log 200 MB, port 5660, guid daa5916af8909118005b0ce2430d6d54,
clusterUuid -7600986066553737256-4524271553806028052, disks /dev/sde /dev/
sdf, dare 1
```

  If enabled, the value for `dare` in the output is `1` (as shown in bold in the above sample command output) and if disabled, the value is `0`.

Managing SSL Certificates

This section decribes how to manage certificates and keystores such as when encryption is initially not enabled or when a custom certificate is used.

Re-running configure.sh after Configuration

If the `configure.sh` script is initially run without the `-genkeys` option, the script generates a `ssl_keystore` file for use by the web server for the Control System.

Then if the `configure.sh` script is re-run with the `-genkeys` option, the system detects the existing `ssl_*` files and exits with an error to prevent inadvertent deletion or reuse of the `ssl_keystore` file.

Note: For general information on certificates, see SSL Certificates on page 689.

To re-run `configure.sh` on clusters without security features enabled:

1. Manually delete the `ssl_keystore` file on each node.
2. Run the `configure.sh -genkeys -R` command.

Note: The contents of the `ssl_keystore` file are unique to each node.

To re-run `configure.sh` on clusters where the contents of the `ssl_keystore` file are customized:

1. Run the `configure.sh -genkeys -nocerts -R` command to preserve your customizations.

SSL Keys Error Message

The error message will look similar to the following example:

```
/opt/mapr/server/configure.sh
        -secure -genkeys -C $CLDB_GRP -Z $ZK_GRP -RM $RM -HS
        $HISTORYSERVER
        <hostname1>: Configuring Hadoop-2.x at
        /opt/mapr/hadoop/hadoop-2.x
        <hostname1>: Done configuring Hadoop
        <hostname1>: CLDB node list:
        <hostname1>:7222,<hostname2>:7222,<hostname3>:7222

        <hostname1>: Zookeeper node
        list: <hostname1>:5181,<hostname2>:5181,<hostname3>:5181

        <hostname1>: Node setup configuration: cldb fileserver
```

```
        historyserver nfs nodemanager resourcemanager webserver
        zookeeper
        <hostname1>: Log can be found at:
        /opt/mapr/logs/configure.log
        <hostname1>: /opt/mapr/conf/ssl_keystore already exists
        <hostname1>: ERROR: could not generate ssl keys. See log file
        for more details
        clush: <hostname1>: exited with exit code 1
```

General Security for Ecosystem Components

Ecosystem components in the MapR Converged Data Platform use the Java Authentication and Authorization Service (JAAS) for security configuration.

- **`/opt/mapr/conf/mapr.login.conf`** file-defines JAAS configurations
- **`MAPR_ECOSYSTEM_LOGIN_OPTS`** environment variable in the `/opt/mapr/conf/env.sh` file-specifies the JAAS configuration used by installed Ecosystem components

Note: See the Ecosystem Guide for component-specific security configuration information.

When security is enabled, the value of the `MAPR_ECOSYSTEM_LOGIN_OPTS` is modified to include the `hybrid` JVM option for `hadoop.login`. This is equivalent to setting the `-Dhadoop.login=hybrid` flag at the command line. This setting specifies a mixed security environment using Kerberos and MapR tickets.

The `mapr.login.conf` file has two stanzas for hybrid security:

```
 /**
 * authenticate using hybrid of kerberos and MapR
 * maprticket must already exist on filesystem as MapR login module
 * cannot get kerberos identity from subject for implicit login.
 */

hadoop_hybrid {
  org.apache.hadoop.security.login.KerberosBugWorkAroundLoginModule optional
      useTicketCache=true
      renewTGT=true
      doNotPrompt=true;
  com.mapr.security.maprsasl.MaprSecurityLoginModule required
      checkUGI=false;
  org.apache.hadoop.security.login.GenericOSLoginModule required;
  org.apache.hadoop.security.login.HadoopLoginModule required
      principalPriority=com.mapr.security.MapRPrincipal;
};

hadoop_hybrid_keytab {
  org.apache.hadoop.security.login.KerberosBugWorkAroundLoginModule optional
      refreshKrb5Config=true
      doNotPrompt=true
      useKeyTab=true
      storeKey=true;
  com.mapr.security.maprsasl.MaprSecurityLoginModule required
      checkUGI=false
      useServerKey=true;
  org.apache.hadoop.security.login.GenericOSLoginModule required;
  org.apache.hadoop.security.login.HadoopLoginModule required
      principalPriority=com.mapr.security.MapRPrincipal;
};
```

Determining if a Cluster is Secure and Enabled for Encryption
Explains how to use the Control System, the CLI, and the REST API to determine whether or not a cluster is secure and whether or not on-wire encryption and data at rest encryption are enabled at the cluster and volume levels.

Determining if a Cluster is Secure Using the Control System

- Log in to the Control System and click to display the **Security** page.
  The **Security** page contains information for determining whether or not the cluster is secure and enabled for on-wire encryption and/or data-at-rest encryption.

Determining if a Secure Cluster is Enabled for Encryption Using the Control System

- Log in to the Control System on a secure cluster and click to display the **Security** page.
  The page displays the following:
  - **Cluster-level Settings** — whether or not on-wire encryption and authentication, and data-at-rest encryption is enabled at the cluster-level. The pane shows:
    - — if enabled
    - — if disabled
  - **Volume Settings** — the number of volumes that are not enabled for:
    - Data on Wire Encryption
    - Data at Rest Encryption

    You can click the number associated with Data on Wire Encryption or Data at Rest Encryption to display the list of volumes filtered by Data on Wire Encryption or Data at Rest Encryption respectively.

Determining if a Cluster is Secure Using the CLI and REST API

CLI

Run the following command to determine if a cluster is secure or unsecure:

```
/opt/mapr/bin/maprcli dashboard
info -cluster <clusterName> -json |
grep secure
```

The value for `secure` is `true` if secure and `false` if unsecure in the command output.

REST

Send a request of type GET. For example:

```
curl -k -X GET 'https://
10.10.82.24:8443/rest/dashboard/
info' --user mapr:mapr
{"timestamp":1525198793701,"timeofday"
:"2018-05-01 11:19:53.701 GMT-0700
AM","status":"OK","total":1,"data":
[{"version":"6.1.0.20180501072815.GA",
"cluster":
{"name":"ksTest","secure":true,"dare":
true,"ip":"10.10.82.24","id":"60002141
79272613712","nodesUsed":1,"totalNodes
Allowed":-1},"volumes":{"mounted":
{"total":17,"size":0},"unmounted":
{"total":1,"size":1}},"utilization":
{"cpu":
```

```
{"util":1,"total":8,"active":0},"memor
y":
{"total":15886,"active":10268},"disk_s
pace":
{"total":273,"active":0},"compression"
:
{"compressed":0,"uncompressed":0},"tie
ring":
{"logicalUsed":0,"replicatedLogicalUse
d":0,"replicatedTotalUsed":0,"ecTotalU
sed":0,"cvTotalUsed":0,"offloaded":0,"
recalled":0}},"clusterReplication":
{"bytesReceived":0,"bytesSend":0},"str
eamThroughput":
{"bytesProduced":0,"bytesConsumed":0},
"services":{"fileserver":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"resourcemanager":
{"active":1,"standby":0,"stopped":0,"f
ailed":0,"total":1},"cldb":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"nfs4":
{"active":0,"stopped":0,"failed":0,"to
tal":1},"mastgateway":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"nodemanager":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"gateway":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"hoststats":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"apiserver":
{"active":1,"stopped":0,"failed":0,"to
tal":1}},"yarn":
{"running_applications":0,"queued_appl
ications":0,"num_node_managers":1,"tot
al_memory_mb":5120,"total_vcores":4,"t
otal_disks":3,"used_memory_mb":0,"used
_vcores":0,"used_disks":0}}]}
```


The value for `secure` is `true` if secure and `false` if unsecure.

If the value for `secure` is `true`, your cluster is enabled for on-wire encryption. See `dashboard info` on `page 1587` for more information.

Determining if a Secure Cluster is Enabled for Encryption of Data at Rest Using the CLI and REST API

CLI

Run the following command to determine if a cluster is enabled or disabled for data at rest encryption:

```
/opt/mapr/bin/maprcli dashboard
info -name <clusterName> -json | grep
dare
```

The value for `dare` is `true` if enabled and `false` if disabled in the command output.

REST

Send a request of type GET. For example:

```
curl -k -X GET 'https://
10.10.82.24:8443/rest/dashboard/
info' --user mapr:mapr
{"timestamp":1525198793701,"timeofday"
:"2018-05-01 11:19:53.701 GMT-0700
AM","status":"OK","total":1,"data":
[{"version":"6.1.0.20180501072815.GA",
"cluster":
{"name":"ksTest","secure":true,"dare":
true,"ip":"10.10.82.24","id":"60002141
79272613712","nodesUsed":1,"totalNodes
Allowed":-1},"volumes":{"mounted":
{"total":17,"size":0},"unmounted":
{"total":1,"size":1}},"utilization":
{"cpu":
{"util":1,"total":8,"active":0},"memor
y":
{"total":15886,"active":10268},"disk_s
pace":
{"total":273,"active":0},"compression"
:
{"compressed":0,"uncompressed":0},"tie
ring":
{"logicalUsed":0,"replicatedLogicalUse
d":0,"replicatedTotalUsed":0,"ecTotalU
sed":0,"cvTotalUsed":0,"offloaded":0,"
recalled":0}},"clusterReplication":
{"bytesReceived":0,"bytesSend":0},"str
eamThroughput":
{"bytesProduced":0,"bytesConsumed":0},
"services":{"fileserver":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"resourcemanager":
{"active":1,"standby":0,"stopped":0,"f
ailed":0,"total":1},"cldb":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"nfs4":
{"active":0,"stopped":0,"failed":0,"to
tal":1},"mastgateway":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"nodemanager":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"gateway":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"hoststats":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"apiserver":
{"active":1,"stopped":0,"failed":0,"to
tal":1}},"yarn":
{"running_applications":0,"queued_appl
ications":0,"num_node_managers":1,"tot
al_memory_mb":5120,"total_vcores":4,"t
otal_disks":3,"used_memory_mb":0,"used
_vcores":0,"used_disks":0}}]}
```

The value for `dare` is `true` if enabled and `false` if disabled.

See `dashboard info` on `page 1587` for more information.

Configuring Authentication
Provides information about MapR ticket, Kerberos, Pluggable Authentication Module (PAM) authentication.

Robust authentication prevents third parties from representing themselves as legitimate users. The core component of user authentication in MapR is the *ticket*. A ticket is an object that contains specific information about a user, an expiration time, and a key. Tickets uniquely identify a user and are encrypted to protect their contents. Tickets are used to establish sessions between a user and the cluster.

MapR supports two methods of authenticating a user and generating a ticket:

- Kerberos
- Username/password pairing with PAM

Both of these methods are mediated by the maprlogin on page 2130 utility. When you authenticate with a username/password pair, the system verifies credentials using Pluggable Authentication Modules (PAM). You can configure the cluster to use any registry that has a PAM module.

Managing Tickets
Introduces authentication using tickets for users and MapR servers.

MapR implements authentication with tickets. *Tickets* contain keys, and are used to authenticate users and MapR servers. In addition, *certificates* are used to implement server authentication. Every user who wants to access a cluster must have a MapR user ticket (`maprticket_<uid>`) and every node in the cluster must have a MapR server ticket (`maprserverticket`).

A ticket is an object that contains specific information about a user and a key. A ticket authenticates a user to the cluster. Tickets are encrypted to protect their contents. The following table describes the tickets used by MapR for internal cluster operations, the user who can generate the ticket, and the command used to generate the ticket. This type of ticket should only be placed on cluster nodes.

Ticket Type	Description	Permissions/Command to Generate Ticket
maprserver	For (internal) cluster operations. This type of ticket can be long lasting.	user root using the `configure.sh` on page 2053 utility
crosscluster	For (internal) cross-cluster operations such as mirroring and replication. This type of ticket can be long lasting.	user mapr using the `maprlogin` on page 2130 utility. The UID of the ticket (mapr) is always used as the identity of the entity using this ticket.

The following table describes the type of tickets supported by MapR for users and services and whether the ticket can be used to impersonate another user. All these tickets, except the user ticket, can only be generated by the cluster administrator using the `maprlogin` on page 2130 utility; the user ticket can be generated by any valid user using the `maprlogin` on page 2130 utility. These type of tickets can be placed on both cluster and client nodes and support (FUSE-based and loopbacknfs) POSIX clients and HDFS APIs.

Ticket Type	Description	Impersonation support	Notes
user	For granting access to individual users. This type of ticket has a short duration.	N/A*	The UID of the ticket (implicit or explicit value of the `-user` parameter to `maprlogin` command) is used as the identity of the entity using this ticket, except for the exceptions noted here for user `root` and user `mapr`.

Ticket Type	Description	Impersonation support	Notes
service	For accessing services running on client nodes. This type of ticket can have long duration.	N/A*	The UID of the ticket (explicit value of the `-user` parameter to `maprlogin` command) is used as the identity of the entity using this ticket, except for the exceptions noted here for user `root` and user `mapr`.
servicewithimpersonation (not scoped)	For accessing services running on client nodes to run jobs on behalf of any user (except user `mapr`). This type of ticket can have long duration.	Yes	The ticket cannot be used to impersonate user `root` or user `mapr`.
servicewithimpersonation (scoped)	For accessing services running on client nodes to run jobs on behalf of the users (except user `root` and user `mapr`) specified in the ticket. This type of ticket can have long duration.	Yes	At ticket generation time, you cannot specify UID/GID of user `root` or user `mapr` to impersonate user root or user mapr respectively. **Note:** In release 6.0.1, scoped impersonation works with FUSE-based POSIX clients; scoped impersonation cannot be used with NFS and loopbacknfs POSIX clients. To use scoped impersonation in release 6.0.1, obtain the 6.0.1 EBF patch for RPM or DEB-based distributions from MapR Support, and install the patch.
tenant	For tenant user(s) to access tenant volume(s) in a multi-tenant environment. This type of ticket can have long duration.	Yes	The ticket can be used to impersonate user `root` but cannot be used to impersonate user `mapr`.

* Exceptions:

- User `mapr` can impersonate other users (including user `root`)
- User `root` can impersonate other users (excluding user `mapr`)

Important: The identity of the user that authenticates with the `maprlogin` utility is independent from the identity of the user of the client OS.

MapR tickets contain the following information:

- UID (generated from the UNIX user ID)
- GIDs (group IDs for each group the user belongs to)

last-updated: Apr 11, 2022

- ticket creation time
- ticket expiration time (initial duration of the ticket)
- renewal expiration time (maximum lifetime of the ticket)
- Whether user can (true) or cannot (false) impersonate another user

Since a ticket contains the GIDs for a user at the time the ticket is generated, a user must re-generate their ticket after changing group memberships.

Syntax and Examples of Creating and Managing User Tickets

For complete syntax, see The maprlogin Utility. For examples of creating and managing user tickets, see maprlogin Command Examples on page 2134

How Tickets Work
Explains the concept of tickets and their working.

When an authenticated user runs a client, the client uses that user's ticket to communicate securely with the server. After Enabling Wire-level Security on page 1411, supported communications channels between client and server are encrypted.

Nodes use tickets to identify themselves to one another in order to prevent *spoofing*, a condition where an untrusted machine presents itself as a trusted machine to gain access to the cluster.

User Blocking

System administrators can use the command line interface to *block* a user. The command to block invalidates all of a user's tickets. Once a block command is received by the CLDB, the name of the blocked user is sent to all FileServer nodes, which reject any request sent by that user that has a ticket older than the block time stamp. Due to the nature of this check, there is no explicit removal of a blocked user. Issuing a new ticket with a time stamp more recent than the block time stamp implicitly permits the user. To permanently prevent a user from logging in again, revoke the user's credentials in the PAM registry.

What Blocking Affects

A blocked user cannot access the MapR filesystem or the CLDB, but since blocking only revokes a user's *existing valid tickets*, be aware of the following interactions:

- Blocking has no effect on Oozie's cached credentials in `~/.oozie-auth-token`, and has no effect on Oozie in general, even after a restart.
- Blocking does not affect a new authentication with user ID and password or with existing Kerberos credentials.
- Since NFS does not use MapR tickets, blocking does not affect NFS access.
- Blocked users can still be impersonated as impersonation does not check whether a user is blocked or not.
- Blocking has no effect on ZooKeeper. Blocked users can still connect to the ZooKeeper server and execute commands. The workaround to resolve this issue is to delete the ticket file.

Generating a MapR User Ticket
Describes what a user ticket is, and how to generate a user ticket.

A user ticket file is stored in `/tmp` and can only be read by that user. To generate a MapR user ticket, run the following command:

```
maprlogin password
```

This command prompts for the user's password, then generates a MapR user ticket associated with the UNIX user ID. By default, tickets on Linux systems are generated in the `/tmp` directory and are named in the form `maprticket_<UID>` . Tickets on Windows systems are generated in the `%TEMP%/` directory and are named in the form `maprticket_<username>` . To change the default location, change the value of the `MAPR_TICKETFILE_LOCATION` environment variable.

Note: The `mapr` user can impersonate any user, including user `root`.

Note: There are no notifications to indicate that a ticket is about to expire. Use the `maprlogin print` command, with the ticket file, to see when the ticket expires. You can renew the ticket until the renewal date mentioned.

To illustrate a typical work flow, suppose a user wants to access two clusters, cluster1 and cluster2. During this session, a user logs in as `root` to cluster1, gets a MapR user ticket, and displays the ticket contents with the `maprlogin print` command.

```
root@qa-node113:~/SecurityInstall# maprlogin password
[Password for user 'root' at cluster 'cluster1': ]
MapR credentials of user 'root' for cluster 'cluster1' are written to '/tmp/
maprticket_0'
root@qa-node113:~/SecurityInstall#
```

First Ticket for Cluster 1

```
root@qa-node113:~/SecurityInstall# maprlogin print
Opening keyfile /tmp/maprticket_0
qasecurity1: user = root, created = 'Wed Sep 11 14:19:02 PDT 2013', expires
= 'Wed Sep 25 14:19:02 PDT 2013', RenewalTill = 'Fri Oct 11 14:19:02 PDT
2013', uid = 0, gids = 0, 42
root@qa-node113:~/SecurityInstall#
```

Now the `root` user logs in to cluster2. The `maprlogin` command returns a ticket for cluster2. This ticket is stored in the common ticket file. Commands now have access to both tickets.

```
root@qa-node113:/opt/mapr/conf# maprlogin password -cluster cluster2
[Password for user 'root' at cluster 'cluster2': ]
MapR credentials of user 'root' for cluster 'cluster2' are written to '/tmp/
maprticket_0'
```

Showing Tickets for both Clusters

```
root@qa-node113:/opt/mapr/conf# maprlogin print
Opening keyfile /tmp/maprticket_0
qasecurity1: user = root, created = 'Thu Sep 12 11:07:54 PDT 2013', expires
= 'Thu Sep 26 11:07:54 PDT 2013', RenewalTill = 'Sat Oct 12 11:07:54 PDT
2013', uid = 0, gids = 0, 42
qasecurity2: user = root, created = 'Thu Sep 12 15:20:49 PDT 2013', expires
= 'Thu Sep 26 15:20:49 PDT 2013', RenewalTill = 'Sat Oct 12 15:20:49 PDT
2013', uid = 0, gids = 0, 500
root@qa-node113:/opt/mapr/conf#
```

Generating a Service Ticket

Applications may have service processes that run outside the MapR cluster but need to access the cluster to run MapR commands. For security reasons, you may not want to run these services as the mapr user. Instead, you can use the maprlogin utility to generate a "service ticket" that can be used to access the cluster for the user account that runs the service. The maprlogin utility uses the current user's ticket (the user running the maprlogin command) to send an authenticated request for a newly generated service ticket.

This type of ticket has a specified duration (expiration), a renewal period (maximum lifetime), and a designated location where the ticket is safely stored. The service process that uses the ticket can access it based on the definition of the `MAPR_TICKETFILE_LOCATION` environment variable, which points to the location of the ticket and should be set for the service process when it starts. Short duration and renewal values may be used for security reasons, but much longer lifetimes are supported for ease of administration.

For example:

```
# maprlogin generateticket -type service -out /tmp/
longlived_ticket -duration 30:0:0 -renewal 90:0:0 -user mapr
MapR credentials of user 'mapr' for cluster 'xxxx' are written to '/tmp/
longlived_ticket'
```

This command generates a service ticket that expires after 30 days and is stored in `/tmp/longlived_ticket`. The ticket may be renewed at any time before the 30 days pass, extending its lifetime to a maximum of 90 days. The ticket must be renewed explicitly before its expiration date; it does not renew automatically when it expires.

Note: This type of ticket can only be generated by a user with full control on a cluster's Access Control List (ACL).

Generating a Service with Impersonation Ticket

Impersonation allows a user to access data and submit jobs on behalf of another user. You may want to allow users, other than the mapr user, to impersonate other users. You can use the `maprlogin` utility to generate a "servicewithimpersonation ticket" that can be used to access a secure cluster impersonating another user. That is, the servicewithimpersonation ticket provides the user the ability to impersonate other users (except the mapr user) in addition to the ability to access a secure cluster. This type of ticket can only be generated by a user with full control on a cluster's Access Control List (ACL).

If this type of ticket is generated and saved in the location specified with the `-out` option, after generating the ticket, do the following:

1. Reset the permissions on the ticket to grant the user, for whom the ticket was generated, read permissions on the ticket.
2. Set the `MAPR_TICKETFILE_LOCATION` environmental variable to point to the ticket file location if the path specified for the `-out` option was not `/tmp/maprticket_<uid>`.

This type of ticket, similar to a service ticket, has a specified duration (expiration), a renewal period (maximum lifetime), and a location where the ticket is safely stored. It grants the specified user the ability to impersonate other users, except the mapr user.

The default duration for this type of ticket is LIFETIME and the duration is not bounded by the CLDB duration properties. Short duration and renewal values may be used for security reasons, but much longer lifetimes are supported for ease of administration.

last-updated: Apr 11, 2022

For example:

```
# maprlogin generateticket -type servicewithimpersonation -user
mapruser1 -out /var/tmp/impersonation_ticket -duration 30:0:0 -renewal
90:0:0
```

The above command generates a service with impersonation ticket that expires after 30 days and is stored in `/var/tmp/impersonation_ticket`. The ticket may be renewed at any time before the 30 days and can be extended up to a maximum of 90 days. The ticket must be renewed explicitly before its expiration date; it does not renew automatically when it expires. The ticket allows the user to impersonate all users on the cluster.

To allow a user to impersonate only specific users and/or groups, use the `impersonateduids` and/or `impersonatedgids` options with the `maprlogin` command. For example:

```
# maprlogin generateticket -type servicewithimpersonation -user
mapruser1 -out /var/tmp/impersonation_ticket -duration
30:0:0 -impersonateduids 1002,1003 -impersonatedgids 1005,1006 -renewal
90:0:0
```

The above command generates a service with impersonation ticket. The ticket holder can impersonate users whose UIDs are 1002 and 1003 and users in the groups with GIDs 1005 and 1006. The ticket expires after 30 days and is stored in `/var/tmp/impersonation_ticket`. The ticket may be renewed at any time before the 30 days and can be extended up to a maximum of 90 days. The ticket must be renewed explicitly before its expiration date; it does not renew automatically when it expires.

Generating a Ticket for a Tenant
Explains what tenant tickets are and how to generate a tenant ticket.

Tenant tickets allow tenant users to access the tenant volume on the cluster (when you have a multi-tenant environment on MapR File System). Generate the tenant ticket on the cluster and copy it to the tenant host(s) to grant tenant users access to the provisioned storage.

- To generate a tenant ticket, run one of the following commands on the cluster:

  ```
  maprlogin generateticket -type tenant -cluster <cluster_name> -user
  <tenant_admin_user> \
  -duration <seconds> -out <ticket_file_path>.txt
  ```

  Note: For more information, see the `maprlogin` command.

  By default:

  - Tenant ticket is stored in `/tmp` and can only be read by that user. To change the default location, specify the path to the desired location with the `out` parameter.
  - Tenant ticket has no expiration. To change the expiration time, specify `duration` for the ticket with the command.

  In the tenant tickets, the value for `CanImpersonate` and `tenant` will always be `true`. For example, when you run the `maprlogin print` command, your output should look similar to the following:

  ```
  Opening keyfile /user/clstrAdmin/tenant_user_ticket.txt
  tenantHost: user = tenant_user, created = 'Mon Jul 11 07:14:53 UTC 2016',
  expires = 'Mon Jul 11 07:14:53 UTC 12016', RenewalTill = 'Mon Jul 11
  07:14:53 UTC 12016',
  uid = 500, gids = 500, 42, CanImpersonate = true, tenant = true
  ```

  To grant access to tenant users, the tenant ticket must be copied over to the tenant hosts.

After generating the ticket, do the following:

1. Reset the permissions on the ticket to grant the tenant admin read permissions on the ticket.
2. Move the ticket out of the default `/tmp` directory to a secure location on the tenant host(s).

Generating a MapRTicket from a Kerberos Ticket

On clusters that use Kerberos for authentication, a MapR ticket is implicitly obtained for a user that runs a MapR command without first using the `maprlogin` utility.

If you want to use a Kerberos ticket to generate a `maprticket`, follow these steps:

1. Obtain a Kerberos identity by running `kinit` or by another mechanism.
2. Run `maprlogin kerberos` to indicate that you have an existing Kerberos ticket. You can also specify these options: `[ -cluster ]` The name of the cluster. `[ -duration ]` The ticket duration in seconds.

Configuring MapR PAM Authenticator

The MapR Converged Data Platform supports Pluggable Authentication Modules (PAM) in the UNIX authentication stack. MapRMapR Data Platform provides a PAM Authenticator module that generates MapRMapR Data Platform tickets in conjunction with the `maprlogin` utility. After you install the MapR Converged Data Platform, the PAM Authenticator module is located at *`$INSTALL_DIR`*`/lib/libmapr_pam.so`. Configuration files for PAM are located in the `/etc/pam.d` directory, and each UNIX operation, such as `su`, `login`, or `ssh`, has a specific PAM configuration file in that directory.

Configure the PAM Authenticator on Ubuntu or SLES

To configure the MapR PAM Authenticator, append the following line to the end of the `/etc/pam.d/common-auth` file:

```
auth optional /opt/mapr/lib/libmapr_pam.so # MapR PAM module
```

Warning: An absolute path to the location of the `libmapr_pam.so` file is required. By default, this location is `$MAPR_HOME/lib/libmapr_pam.so`.

Configure the PAM Authenticator on Red Hat or CentOS

1. Insert the following line in the `/etc/pam.d/system-auth` file immediately before the first module that uses the `auth sufficient` configuration:

```
auth optional libmapr_pam.so # MapR PAM module
```

2. Append the string `try_first_pass` to the end of the module that uses `auth sufficient`, as in this example:

   Before modification:

   ```
   auth required pam_env.so
   auth sufficient pam_unix.so nullok
   auth requisite pam_succeed_if.so uid >= 500 quiet
   auth required pam_deny.so
   ```

   After modification, changes in **bold**:

   ```
   auth required pam_env.so
   auth optional libmapr_pam.so # MapR PAM module
   auth sufficient pam_unix.so nullok try_first_pass
   auth requisite pam_succeed_if.so uid >= 500 quiet
   auth required pam_deny.so
   ```

Enable Debugging for PAM

To enable debugging for the client traffic used by the `maprlogin` utility, update the `/opt/mapr/conf/log4j.properties` file with the following line:

`log4j.logger.com.mapr.login=DEBUG`

After updating the `log4j.properties` file, trace the `com.mapr.login` package at the DEBUG level.

Be sure to update the correct instance of the `log4j.properties` file. Traffic specific to MapRMapR Data Platform, such as `maprlogin` and Control System traffic, uses the instance in the `/opt/mapr/conf` directory. Hadoop applications use the `log4j.properties` file in the `/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop` directory.

To perform the same tracing activity on the server side, modify the appropriate instance of the `log4j.properties` file on the server, or specify the page `com.mapr.login` in the Control System UI's tracing/logger settings. To trace PAM activity from the server, add the following line to the server's `log4j.properties` instance:

```
log4j.logger.net.sf.jpam=DEBUG
```

After modifying this setting, the server log will contain a message similar to the following:

`2013-07-23 16:05:25,200 DEBUG Pam [1068409264@qtp-874242484-3]: Debug mode active.`

Detailed information about PAM activity is written to the `/opt/mapr/logs/pam.log` file.

Other Packages

The following packages are not directly related to PAM, but can provide useful insights for subtler errors.

- `org.apache.hadoop.security` - This package contains Apache security code, including MapRMapR Data Platform enhancements. Tracing this package can provide information about what login configuration is in use.
- `com.mapr.fs.cldb.http.login` - This package contains code that the CLDB uses to validate `maprlogin` calls.

Common Issues

The Linux Documentation Project's HOWTO on LDAP Implementation has a section on PAM and NSS that may prove helpful.

If a user's credentials appear valid, for example in a case where the `su` and `ssh` commands work normally, but PAM does not correctly authenticate, the issue may relate specifically to MapRMapR Data Platform's use of PAM as a normal user, compared to the usual case where PAM consumers run as the root user, causing permissions issues. The two most common issues relating to this condition are:

- The `/etc/shadow` directory is not readable to the `mapr` user. This directory is made readable to the `mapr` user during install, but some secure environments and configuration management tools undo these changes.
- A Kerberos PAM module is attempting to create and change the ownership of a kerberos ticket file. This attempt fails, since these changes require root privileges. Different Kerberos PAM modules can report errors differently, leading to difficulty tracking down root causes of errors. To address permissions problems with Kerberos PAM modules, configure the Kerberos PAM module to skip creating a ticket file, using the KDC only to validate the password. PAM configuration information is located in the `/etc/pam.d` directory. MapRMapR Data Platform can use a custom PAM configuration specified in the `web.conf` file.

Configuring Kerberos
Describes how Kerberos works with MapR tickets.

MapR does not directly support Kerberos. However, Kerberos is indirectly supported through the MapR login utility, which is used to generate MapR tickets. This topic describes how Kerberos works with MapR tickets.

Kerberos Compatibility with RHEL 8

When you install Kerberos out of the box with RHEL 8, it uses the new default Kerberos Cache Manager (KCM) credentials cache type, which fails to work with the `maprlogin kerberos` command. To resolve this issue, disable KCM.

Open the file `/etc/krb5.conf.d/kcm_default_ccache` and comment out the following lines:

```
[libdefaults]
        default_ccache_name = KCM:
```

Alternatively, remove this file.

Configuring Kerberos for Authentication Using MapR Tickets

To use Kerberos to generate MapR tickets for users, enable Kerberos on CLDB by creating a Kerberos identity on the Kerberos server used by the cluster and distributing that identity to the other CLDB nodes in the cluster.

Note: You must enable wire-level security on your clusters before using Kerberos. See Enabling Wire-level Security.

MapR clusters do not provide Kerberos infrastructure. This section assumes you have a functioning Kerberos realm and your systems have the Kerberos client installed. The tips in this section assume a Linux-based Kerberos environment, and the specific commands for your environment may vary. Please consult with your Kerberos administrator for assistance.

Important: If you are using strong encryption with Kerberos with the Oracle JDK, you will require a new Java Cryptography Extension (JCE) policy file.

Creating a Kerberos Identity for the CLDB

The CLDB requires a Kerberos server identity, but no other nodes do. By default, this identity takes the form `mapr/<cluster name>`. You can use `configure.sh` on page 2053 or edit the

last-updated: Apr 11, 2022

`mapr-clusters.conf` file to change this default. Use the following commands in a Linux-based Kerberos environment to set up the identity:

```
kadmin
        : addprinc -randkey mapr/my.cluster.com
        : ktadd -k /opt/mapr/conf/mapr.keytab mapr/my.cluster.com
```

Copy the resulting `mapr.keytab` file to the same location on every CLDB node. The `mapr.keytab` file must be owned and readable only by the `mapr` user. You can specify the location of the `mapr.keytab` file in the `conf/mapr.login.conf` file. The default location for `mapr.keytab` is `/opt/mapr/conf`.

Updating the keytab File

You can use the `kadmin` tool to update the server keys that are stored in the keytab file. Because the server tickets used to authenticate to the CLDB use the new keys immediately, you must copy the new keytab file to all the CLDB servers in the cluster immediately after updating the server keys.

To update the keytab file with a new key, run the following command:

```
kadmin
        : ktadd -k /opt/mapr/conf/mapr.keytab mapr/my.cluster.com
```

The CLDB automatically detects changes to the keytab file on systems that use Java 7 or later. Systems that use Java 6 require a CLDB restart to detect changes to the keytab file.

Note: Starting with the 4.0.1 release of the MapR software, Java 6 is deprecated in favor of Java 7 and Java 8.

Running configure.sh

After a Kerberos principal is created for the CLDB, that principal is added to the `mapr.keytab` file, and the `mapr.keytab` file is copied to all the CLDB servers, Kerberos user authentication is fully enabled for the MapR cluster.

Two `configure.sh` parameters are important for Kerberos:

- `-K|-kerberosEnable` — lets the rest of the cluster know that Kerberos is enabled, so that clients can auto detect Kerberos tickets and use them to get MapR tickets.
- `-P "<cldbPrincipal>"` — specifies the Kerberos instance which is used to form the CLDB Kerberos principal in the form of `mapr/<instance-name>@<realm-name>`. Enclose this value in quotes (").

Run `configure.sh` on each MapR cluster node, and on each MapR client node that will communicate with one or more clusters. For more information, see `configure.sh` on page 2053.

```
configure.sh -K -P "<cldbPrincipal>"
```

Running `configure.sh` on each node enters the Kerberos information into the local `clusters.conf` file, so that the following command is all that is required for the client to access the cluster:

```
hadoop fs -ls
```

If you do not run `configure.sh` on each node, the following two commands are required from the client:

```
maprlogin kerberos
hadoop fs -ls
```


last-updated: Apr 11, 2022

Kerberos Command Summary

- **kinit**: Creates a Kerberos ticket. Prompts the user for userid and password. After validating, Kerberos creates a ticket file in `/tmp` that is owned by the user. Use the `-R` option to renew an existing ticket. Kerberos credentials expire in 8-10 hours. Expired credentials must be renewed or replaced. By default, tickets can be renewed for up to 24 hours.
- **klist**: Lists the contents of the user's ticket file.
- **kdestroy**: Destroys the contents of the user's ticket file. The user is no longer authenticated.
- **kadmin**: Used to administer Kerberos. The login for this command is implicitly `<userid>/admin`, since administrator IDs typically end in `/admin`.
- **ktutil**: Kerberos keytab maintenance utility. Used to combine, or alter Kerberos keytabs.

Disabling Replay Detection for Kerberos Authentication

You can set an option in `mapr-clusters.conf` file to disable replay detection for Kerberos runtime authentication.

```
disableReplayDetection=true
```

By default, this parameter is set to `false`, meaning that MapR clients enable Kerberos replay detection. Typically, replay detection is enabled to prevent potential attacks such as the replay of Kerberos packets or multiple login attempts with the same user ID. Set this parameter to `true` only if you want MapR clients not to enforce this detection.

This parameter applies when users attempt an implicit or explicit `maprlogin`, such as by using the `maprlogin kerberos` command or by submitting jobs and other operations with `kerberosEnable=true` set in the `mapr-clusters.conf` file.

This parameter is used when applications connect to the cluster using Kerberos; `mapr-clusters.conf` only needs to be updated when it is used by such applications. If all Kerberos access to the cluster is from clients outside the cluster, only the `mapr-clusters.conf` file on those client machines has to be updated. If Kerberos is used from applications running on the cluster, `mapr-clusters.conf` file should be updated there as well.

Configuring YARN with Kerberos
Lists the process to use YARN with Kerberos.

Make sure that the following tasks are already completed, as directed in earlier sections of this guide:

- Enabling Wire-level Security on page 1411 by running `configure.sh` with the security options.
- Configuring Kerberos on page 1432 by creating a Kerberos principle and keytab file.

Note: To enable YARN REST SPNEGO, see Configuring SPNEGO on MapR on page 1436.

Now complete the following tasks.

Configure the yarn-site.xml File

Add the following properties to the `yarn-site.xml` file on every node in the cluster.

```
/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/yarn-site.xml
```

Note: You need to use `/opt/mapr/conf/mapr.keytab` for the `keytab` property and `mapr` instead of `yarn` for the `principal` property.

```
<!-- ResourceManager security configs -->
<property>
 <name>yarn.resourcemanager.keytab</name>
 <value>/opt/mapr/conf/mapr.keytab</value>    <!-- path to the YARN
keytab -->

</property>
<property>
 <name>yarn.resourcemanager.principal</name>
 <value>mapr/clustername@YOUR-REALM.COM</value>
</property>

<!-- NodeManager security configs -->
<property>
 <name>yarn.nodemanager.keytab</name>
<value>/opt/mapr/conf/mapr.keytab</value>    <!-- path to the YARN
keytab -->

</property>
<property>
 <name>yarn.nodemanager.principal</name>
 <value>mapr/clustername@YOUR-REALM.COM</value>
</property>
<property>
 <name>yarn.nodemanager.container-executor.class</name>
 <value>org.apache.hadoop.yarn.server.nodemanager.LinuxContainerExecutor</
value>
</property>
<property>
 <name>yarn.nodemanager.linux-container-executor.group</name>
 <value>mapr</value>
</property>
```

Configure the mapred-site.xml File

Add the following properties to the `mapred-site.xml` file on every node in the cluster.

```
/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/mapred-site.xml
```

Note that you need to use `/opt/mapr/conf/mapr.keytab` for the `keytab` property and `mapr` instead of `yarn` for the `principal` property.

```
<!-- MapReduce Job History Server security configs -->
<property>
  <name>mapreduce.jobhistory.address</name>
  <value>host:port</value> <!-- Host and port of the MapReduce Job History
Server; default port is 10020  -->
</property>
<property>
  <name>mapreduce.jobhistory.keytab</name>
  <value>/opt/mapr/conf/mapr.keytab</value><!-- path to the YARN
keytab -->
</property>
<property>
  <name>mapreduce.jobhistory.principal</name>
  <value>mapr/clustername@YOUR-REALM.COM</value>
</property>
```


last-updated: Apr 11, 2022

Modifying the env_override.sh File

Either the `/opt/mapr/conf/env.sh` file or the `/opt/mapr/conf/env_override.sh` file contains a setting for MapR login option that defaults to the value `maprsasl`. Change this value to `hybrid`, which includes Kerberos and other security protocols. For more information about the `env_override.sh` file, see About env_override.sh on page 2290.

The new line (after the change) should be as follows:

```
MAPR_LOGIN_OPTS="-Dhadoop.login=hybrid ${MAPR_JAAS_CONFIG_OPTS} $
{MAPR_ZOOKEEPER_OPTS}"
```

Restart ResourceManager, NodeManager, and JobHistoryServer

Restart the NodeManager, ResourceManager, and JobHistoryServer services, using either the `maprcli node services` command (with the `name` option) or the Control System. After restarting the services, make sure you can run simple Hadoop jobs by running:

```
hadoop jar /opt/mapr/hadoop/hadoop-<version>/share/hadoop/mapreduce/
hadoop-mapreduce-examples-<version>.jar pi
```

Configuring SPNEGO on MapR

MapR uses the Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO) to secure several Web UIs in a secure cluster, as well as the REST calls to the Control System.

Configuring SPNEGO for the Web Server Nodes on Secure Clusters

The following procedure configures SPNEGO support for the apiserver nodes on your secure cluster.

1. Generate a Kerberos principal with the user name HTTP, of the form `HTTP/<webserver name>` on each node in the secure cluster that will receive inbound SPNEGO traffic.

   Use the fully qualified domain name (FQDN) as the name in the principal. Although you could also use a short name or the IP address for the principal name, using the FQDN keeps the name consistent with principal names that `configure.sh` generates and includes in the `mapr.login.conf` file.

   Whatever you use as the principal name is what users will have to match exactly in a browser to access the web pages that are protected.

   Note: Several services and components in a MapR cluster handle SPNEGO traffic, including the Control System. You can name the keytab file `mapr.keytab` if that file does not already exist. If the `mapr.keytab` file already exists, generate the new principal to a different file name and merge it to the `mapr.keytab` file using the `ktutil` tool. For example:

   ```
   kadmin
    : addprinc -randkey HTTP/<webserver name>
    : ktadd -k /opt/mapr/conf/mapr.keytab HTTP/<webserver name>
   ```

2. Verify that the `/opt/mapr/conf/mapr.login.conf` file lists the correct principal in the `MAPR_WEBSERVER_KERBEROS` section.

   To enable SPNEGO for the Control System Web UI or for the Control System REST calls, on all nodes with the webserver role, add the following line to the `/opt/mapr/apiserver/conf/properties.cfg` file. For example:

   ```
   mapr.rest.auth.methods=kerberos,basic
   ```

   Important: The `mapr.rest.auth.methods=kerberos,basic` option, which is shown above, is valid only on a secure cluster. If a cluster is not secure, only basic authentication (`WWW-Authenticate: Basic`) is available to clients.

3. Restart the Control System for the changes to take effect.

Testing SPNEGO With curl

This example tests that the Control System is using GSS for REST calls made with `curl`.

Use the following command to verify that your version of `curl` supports SPNEGO. Under the **Features** header, output of the command should show either **GSS-Negotiate** or SPNEGO. For example:

```
# curl --versioncurl 7.22.0 (x86_64-pc-linux-gnu) libcurl/7.22.0
OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23 librtmp/2.3Protocols:
dict file ftp ftps gopher http https imap imaps ldap pop3 pop3s rtmp rtsp
smtp smtps telnet tftp
Features: GSS-Negotiate IDN IPv6 Largefile NTLM NTLM_WB SSL libz TLS-SRP
```

Verify that you have a valid Kerberos ticket-granting-ticket (TGT) with the `kinit -p <user>` command,. Then, test `curl` with the following command:

```
curl --negotiate -u : -b ~/cookiejar.txt -c ~/cookiejar.txt
https://<web server node>:8443/rest/<API call> -k -v
```

This command returns HTTP/1.1 200 OK when `curl` is working correctly with SPNEGO.

Configuring Browsers for SPNEGO

Use the following processes to configure browsers for SPNEGO connections.

Firefox

The process below configures your Firefox browser for SPNEGO connections.

Note: These instructions are specific for Firefox version 40.0.3xj. The details may differ slightly if you are using a different version.

1. Open the Firefox configuration page by navigating to the address `about:config`.
2. In the **Search** text field, enter `network.negotiate-auth.trusted-uris` to bring up that property.
3. Right-click on `network.negotiate-auth.trusted-uris` and select **Modify** to edit the property and enter the hostnames of the web server nodes in your cluster as a comma-separated list.
4. Click **OK**.

Chromium on Ubuntu

last-updated: Apr 11, 2022

To configure the Chromium browser on Ubuntu for SPNEGO, edit the `/etc/chromium-browser/default` file and add the following property:

```
CHROMIUM_FLAGS="--user-data-dir --auth-server-whitelist=<web server host
names>"
```

The `--user-data-dir` flag enables the root user to launch the browser.
The `--auth-server-whitelist` flag specifies the web servers that support SPNEGO authentication.

Troubleshooting Kerberos

Java errors from Kerberos problems can be obscure and difficult to interpret. To see the Kerberos error messages, enable Kerberos debugging by adding these settings to your JVM:

```
-Dsun.security.krb5.debug=true -Dsun.security.spnego.debug=true -Djavax.net.
debug=all
```

You can also enable Kerberos debugging for the MapR-provided `maprcli` and Hadoop clients by adding the following line to the `/opt/mapr/conf/env_override.sh` shell script:

```
#MAPRLOGIN_OPTS="$
{MAPRLOGIN_OPTS} -Dsun.security.krb5.debug=true -Dsun.security.spnego.debug=
true -Djavax.net.debug=all"
```

The `env.sh` script reads this file as part of its execution. For more information, see About env_override.sh on page 2290.

Capture the Kerberos error to research the issue.

The following sections list common Kerberos error conditions:

Incorrect JVM

Nodes often have multiple Java Virtual Machines (JVM) installed. The MapR `env.sh` script automatically configures a JVM to use. To change the automatically configured JVM, set the value of the `JAVA_HOME` environment variable in the `/opt/mapr/conf/env_override.sh` file. The `env.sh` script reads this file as part of its execution. For more information, see About env_override.sh on page 2290.

Incorrect Server Name

The following error message is caused by an incorrect CLDB server name in the `mapr.login.conf` file. The error message mentions passwords, but the error condition is unrelated to password authentication.

```
2018-04-25 16:46:02,324 ERROR MapRLoginServlet [185087767@qtp-648535353-2]:
Failed to obtain kerberos identity, continuing anyway...
        javax.security.auth.login.LoginException: Unable to obtain password
from user

        at
com.sun.security.auth.module.Krb5LoginModule.promptForPass(Krb5LoginModule.j
ava:789)
        at
com.sun.security.auth.module.Krb5LoginModule.attemptAuthentication(Krb5Login
Module.java:654)
        at
com.sun.security.auth.module.Krb5LoginModule.login(Krb5LoginModule.java:542)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39
)
        at
```

```
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl
.java:25)
        at java.lang.reflect.Method.invoke(Method.java:597)
        at
javax.security.auth.login.LoginContext.invoke(LoginContext.java:769)
        at
javax.security.auth.login.LoginContext.access$000(LoginContext.java:186)
        at
javax.security.auth.login.LoginContext$4.run(LoginContext.java:683)
        at java.security.AccessController.doPrivileged(Native Method)
        at
javax.security.auth.login.LoginContext.invokePriv(LoginContext.java:680)
        at
javax.security.auth.login.LoginContext.login(LoginContext.java:579)
        at
com.mapr.fs.cldb.http.login.MapRLoginServlet.init(MapRLoginServlet.java:73)
        at
org.mortbay.jetty.servlet.ServletHolder.initServlet(ServletHolder.java:440)
        at
org.mortbay.jetty.servlet.ServletHolder.getServlet(ServletHolder.java:339)
        at
org.mortbay.jetty.servlet.ServletHolder.handle(ServletHolder.java:487)
        at
org.mortbay.jetty.servlet.ServletHandler.handle(ServletHandler.java:401)
        at
org.mortbay.jetty.security.SecurityHandler.handle(SecurityHandler.java:216)
        at
org.mortbay.jetty.servlet.SessionHandler.handle(SessionHandler.java:182)
        at
org.mortbay.jetty.handler.ContextHandler.handle(ContextHandler.java:766)
        at
org.mortbay.jetty.webapp.WebAppContext.handle(WebAppContext.java:450)
        at
org.mortbay.jetty.handler.ContextHandlerCollection.handle(ContextHandlerColl
ection.java:230)
        at
org.mortbay.jetty.handler.HandlerWrapper.handle(HandlerWrapper.java:152)
        at org.mortbay.jetty.Server.handle(Server.java:326)
        at
org.mortbay.jetty.HttpConnection.handleRequest(HttpConnection.java:542)
        at
org.mortbay.jetty.HttpConnection$RequestHandler.content(HttpConnection.java:
945)
        at org.mortbay.jetty.HttpParser.parseNext(HttpParser.java:756)
        at org.mortbay.jetty.HttpParser.parseAvailable(HttpParser.java:212)
        at org.mortbay.jetty.HttpConnection.handle(HttpConnection.java:404)
        at
org.mortbay.jetty.bio.SocketConnector$Connection.run(SocketConnector.java:22
8)
        at
org.mortbay.jetty.security.SslSocketConnector$SslConnection.run(SslSocketCon
nector.java:713)
        at
org.mortbay.thread.QueuedThreadPool$PoolThread.run(QueuedThreadPool.java:582
)
```

Invalid or missing keytab file

The `keytab` file must be consistent with the key versions of the Kerberos principal. The following example shows an inconsistent keytab file:

```
kadmin:  getprinc mapr/realm1
        Principal: mapr/realm1@mapr
        Expiration date: [never]
        Last password change: Thu May 23 15:36:01 PDT 2013
        Password expiration date: [none]
        Maximum ticket life: 0 days 10:00:00
        Maximum renewable life: 7 days 00:00:00
        Last modified: Thu May 23 15:36:01 PDT 2013 (mapr/admin@mapr)
        Last successful authentication: Thu May 23 19:31:59 PDT 2013
        Last failed authentication: Thu May 23 15:35:41 PDT 2013
        Failed password attempts: 0
        Number of keys: 8
        Key: vno 15, aes256-cts-hmac-sha1-96, no salt
        Key: vno 15, arcfour-hmac, no salt
        Key: vno 15, des3-cbc-sha1, no salt
        Key: vno 15, des-cbc-crc, no salt
        Key: vno 15, des-cbc-md5, Version 4
        Key: vno 15, des-cbc-md5, Version 5 - No Realm
        Key: vno 15, des-cbc-md5, Version 5 - Realm Only
        Key: vno 15, des-cbc-md5, AFS version 3
        MKey: vno 1

        ktutil:  rkt mapr.keytab
        ktutil:  l
        slot KVNO Principal

 ---- ---- ---------------------------------------------------------------------

        1   14            mapr/realm1@mapr
        2   14            mapr/realm1@mapr
        3   14            mapr/realm1@mapr
        4   14            mapr/realm1@mapr
        ktutil:  q
```

Note that the key versions in the Kerberos principal `/realm1` are 15, and the versions in the `keytab` file are 14. This mismatch can result in errors about missing keys or mismatched encryption.

Note:

This error state can also be caused by the `/opt/mapr/conf/mapr.keytab` file not being owned by the user `mapr` or not being present. The `keytab` file is owned by `root` at generation. Be sure to use the `chown` command to set the `mapr` user as the owner:

```
$ chown mapr:mapr /opt/mapr/conf/mapr.keytab
```

Incompatible encryption on Java runtime

Incompatible cryptography between the KDC and the JDK results in failed handshakes, leading to errors similar to the following:

```
Caused by: javax.security.auth.login.LoginException: Unable to obtain
Principal Name for authentication
```

With debugging active, the following message is displayed:

```
>>>DEBUG <CCacheInputStream>  client principal is username@hostname
        >>>DEBUG <CCacheInputStream> server principal is X-CACHECONF:/
krb5_ccache_conf_data/fast_avail/krbtgt/user@hostname
        >>>DEBUG <CCacheInputStream> key type: 0
        >>>DEBUG <CCacheInputStream> auth time: Wed Dec 31 16:00:00 PST 1969
        >>>DEBUG <CCacheInputStream> start time: null
        >>>DEBUG <CCacheInputStream> end time: Wed Dec 31 16:00:00 PST 1969
        >>>DEBUG <CCacheInputStream> renew_till time: null
        >>> CCacheInputStream: readFlags()
        >>> unsupported key type found the default TGT: 18
        Negotiate support not initiated, will fallback to other scheme if
allowed. Reason:
```

This debug message indicates that the problem is an unsupported key type.

Incompatible encryption errors can occur due to a `keytab` file that is not present or contains outdated keys.

Be sure to update the Java jurisdiction policy file. Jurisdiction policy files are available from Oracle.

A persistent encryption incompatibility problem may require you to edit the `krb5.conf` file to ensure compatible algorithms between Java and Kerberos.

Bugs in Java

The following error occurs in Java version 1.6.0_25. Upgrade to 1.6.0_45 to resolve the error.

```
java.io.IOException: extra data given to DerValue constructor
        at sun.security.util.DerValue.init(DerValue.java:368)
        at sun.security.util.DerValue.<init>(DerValue.java:277)
        at sun.security.krb5.internal.Ticket.<init>(Ticket.java:81)
        at
sun.security.krb5.internal.ccache.CCacheInputStream.readData(CCacheInputStre
am.java:250)
        at
sun.security.krb5.internal.ccache.CCacheInputStream.readCred(CCacheInputStre
am.java:357)
        at
sun.security.krb5.internal.ccache.FileCredentialsCache.load(FileCredentialsC
ache.java:225)
        at
sun.security.krb5.internal.ccache.FileCredentialsCache.acquireInstance(FileC
redentialsCache.java:104)
        at
sun.security.krb5.internal.ccache.CredentialsCache.getInstance(CredentialsCa
che.java:75)
        at
sun.security.krb5.Credentials.acquireTGTFromCache(Credentials.java:304)
        at
com.sun.security.auth.module.Krb5LoginModule.attemptAuthentication(Krb5Login
Module.java:589)
        at
com.sun.security.auth.module.Krb5LoginModule.login(Krb5LoginModule.java:542)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39
)
        at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl
.java:25)
        at java.lang.reflect.Method.invoke(Method.java:597)
```


last-updated: Apr 11, 2022

```
        at
javax.security.auth.login.LoginContext.invoke(LoginContext.java:769)
        at
javax.security.auth.login.LoginContext.access$000(LoginContext.java:186)
        at
javax.security.auth.login.LoginContext$5.run(LoginContext.java:706)
        at java.security.AccessController.doPrivileged(Native Method)
        at
javax.security.auth.login.LoginContext.invokeCreatorPriv(LoginContext.java:7
03)
        at
javax.security.auth.login.LoginContext.login(LoginContext.java:575)
        at
org.apache.hadoop.security.UserGroupInformation.getLoginUser(UserGroupInform
ation.java:554)
        at
org.apache.hadoop.security.UserGroupInformation.getCurrentUser(UserGroupInfo
rmation.java:528)
        at
org.apache.hadoop.fs.FileSystem$Cache$Key.<init>(FileSystem.java:1656)
        at
org.apache.hadoop.fs.FileSystem$Cache$Key.<init>(FileSystem.java:1649)
        at org.apache.hadoop.fs.FileSystem$Cache.get(FileSystem.java:1517)
        at org.apache.hadoop.fs.FileSystem.get(FileSystem.java:235)
        at org.apache.hadoop.fs.FileSystem.get(FileSystem.java:115)
        at org.apache.hadoop.fs.FsShell.init(FsShell.java:87)
        at org.apache.hadoop.fs.FsShell.run(FsShell.java:1808)
        at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:65)
        at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:79)
        at org.apache.hadoop.fs.FsShell.main(FsShell.java:1967)
        13/05/10 15:24:00 DEBUG security.SaslRpcClient: Creating SASL
GSSAPI client. Server's Kerberos principal name is hdfs/
qa-node24@dev-maprtech
        13/05/10 15:24:00 WARN ipc.Client: Exception encountered while
connecting to the server : javax.security.sasl.SaslException: GSS initiate
failed [Caused by GSSException: No valid credentials provided (Mechanism
level: Failed to find any Kerberos tgt)]
```

Note:

Starting with the 4.0.1 release of the MapR software, Java 6 is deprecated in favor of Java 7 and Java 8.

Kerberos and PAM validation

Standard Kerberos implementations are predicated on access to elevated user privileges that are not present on secure MapR clusters. In a MapR cluster, the Control System console and other components call PAM as an ordinary user process. This discrepancy in expected and actual privileges can cause a variety of obscure file permisson errors. Since different Kerberos PAM modules are available, error reports can vary.

To diagnose this issue, attempt starting the Control System as the root user, or clear out the `/tmp` folder. If there are no problems when starting the Control System as root, or if clearing out the `/tmp` folder enables a single login before errors appear again, the problem may lie in the Kerberos PAM configuration.

To resolve this condition, prevent Kerberos from creating a ticket file. MapR security does not use Kerberos tickets. The Kerberos KDC is used to validate passwords. Typically the configuration file for PAM is in the `/etc/pam.d` directory. See the documentation for your specific Kerberos PAM module for more information.

Configuring PAM
Describes how PAM works with MapR.

MapR uses Pluggable Authentication Modules (PAM) for password verification in a variety of places. Make sure PAM is installed and configured on the node running the `mapr-apiserver`, or other components that will use PAM to verify passwords.

There are typically several PAM modules (profiles), configurable via configuration files in the `/etc/pam.d/` directory. Any component verifying user passwords tries the following three profiles in order:

1. sudo (`/etc/pam.d/sudo`)
2. sshd (`/etc/pam.d/sshd`)
3. mapr-admin (If you have created the `/etc/pam.d/mapr-admin` profile and the component checks beyond the first two profiles.)

The profile configuration file (for example, `/etc/pam.d/sudo`) should contain an entry corresponding to the authentication scheme used by your system. For example, if you are using the simplest form of local OS authentication, check for an entry similar to the following - consult with your Unix system administrator if you are uncertain:

```
auth    sufficient      pam_unix.so  # For local OS Auth
```

Component-specific PAM Configurations

Some ecosystem components have unique requirements that require setup of a component-specific PAM configuration. See the Ecosystem Guide for the specific Ecosystem component.

Configuring PAM for the Control System and the REST API
Describes how to create a custom PAM profile and use a specific PAM file for authentication.

Starting in MapRMapR Data Platform v6.0, no additional configuration is needed to use PAM files for authentication. The apiserver supports PAM and automatically loads the following PAM files, if they exist, in the following order for authentication:

```
/etc/pam.d/mapr-admin
/etc/pam.d/sudo
/etc/pam.d/sshd
/etc/pam.d/chkpasswd
/etc/pam.d/passwd
```

However, you can create a custom PAM profile and set the admin server property to point to a specific PAM file to use for authentication:

1. Open the `/opt/mapr/apiserver/conf/properties.cfg` file and set the PAM file as the value for the `authentication.pam.service` property.

   For example, to set `mapr-admin` as the file to use for authentication, your entry in the file should look similar to the following:

   ```
   ojai.cache.size=64
   mapr.webui.https.port=8443
   doc.url=https://docs.datafabric.hpe.com/home
   proxy.zkservices=elasticsearch,opentsdb
   authentication.pam.service=mapr-admin
   ```

2. Save and close the file.

Configuring PAM to use LDAP

For instructions, refer to your operating system vendor documentation.

last-updated: Apr 11, 2022

Configuring PAM to use Kerberos

To configure PAM with Kerberos:

1. Install the `krb5` packages and configure the Kerberos client as per the configuration for your environment.
2. Install the appropriate PAM packages:
   - On Redhat/Centos, `sudo yum install pam_krb5`
   - On Ubuntu, `sudo apt-get install -krb5`

Creating a Custom PAM Profile

If you wish to ensure that MapR uses a MapR-unique PAM configuration, you can:

- Leave the `/etc/pam.d/sudo` file as is - MapR strongly recommends against manually editing the `/etc/pam.d/sudo` file.
- Create your own PAM profile in `/etc/pam.d`, naming it `mapr-admin`.
- Manually edit `mapr.login.conf` and other ecosystem component configuration files to use `mapr-admin` only.

Example /etc/pam.d/mapr-admin File

Below are some simple examples of what might work in the PAM profile by editing `mapr-admin` or a different PAM profile.

Note: Be sure to consult with your Linux administrator before modifying PAM profiles.

```
account     required      pam_unix.so
account     sufficient    pam_succeed_if.so uid < 1000 quiet
account     [default=bad success=ok user_unknown=ignore] pam_ldap.so
account     required      pam_permit.so

auth    sufficient      pam_unix.so nullok_secure
auth    requisite       pam_succeed_if.so uid >= 1000 quiet
auth    sufficient      pam_ldap.so use_first_pass
auth    required        pam_deny.so

password    sufficient    pam_unix.so md5 obscure min=4 max=8 nullok
try_first_pass
password    sufficient    pam_ldap.so
password    required      pam_deny.so

session     required      pam_limits.so
session     required      pam_unix.so
session     optional      pam_ldap.so
```

Note: The file `/etc/pam.d/sudo` should be modified only with care and only when absolutely necessary.

last-updated: Apr 11, 2022

Example for Hue

- In Hue, you can set which PAM profiles to use by modifying `pam_service` option in the `<HUE_HOME>/desktop/conf/hue.ini` file:

```
[desktop]
  ...
# Configuration options for user authentication into the web application
# ------------------------------------------------------------------------
 [[auth]]
# Authentication backend...
backend=desktop.auth.backend.PamBackend
  ...
# The service to use when querying PAM.
## pam_service=sudo sshd login
```

Note: The `mapr-admin` profile is not used in the default Hue configuration.

Note: Hue respects only `auth` section from the PAM profiles.

Example for Livy

- With Livy you can authenticate users with PAM only by using MapR MultiMechs authentication, so it uses the configuration from `/opt/mapr/conf/mapr.login.conf`.

Managing Access Controls
Describes how to create, enable, and use ACLs and ACEs.

ACLs specify users or system processes that can perform specific actions on an object. ACEs are Boolean expressions that defines a combination of users, groups, or roles that have access to an object.

Managing Access Control Lists
Defines and describes how to create ACLs.

An access control list (ACL) specifies users or system processes that can perform specific actions on an object.

Creating Cluster-level ACLs

A cluster-level Access Control List (ACL) determines who has access to a cluster and which actions they are allowed to perform. ACLs on a secure MapR cluster are predicated on a locally managed OS registry.

Important: Before you create an ACL that applies to a particular group, you must create that group and assign users to it.

For example, the Red Hat Linux commands for creating a group called `developers` and adding a user named `jsmith` on a locally managed OS registry are:

```
groupadd developers
useradd -g developers jsmith
```

Once users and groups have been defined, an administrator can create a cluster-level ACL using the Control System and the CLI.

Creating an ACL from the Control System

1. Click **Admin** > **User Settings** > **Permissions**.
2. Follow steps for Adding Cluster Permissions on page 755.

   Each allowed action has a permission code associated with it. The codes are explained below.

last-updated: Apr 11, 2022

Permission Code	Allowed Action
login	Log in to the MapR Control System, use the API and command-line interface, read access on cluster and volumes. **Note:** Read access allows you to only view the MapR objects that already exist. You cannot create volumes, policies, schedules, snapshots, nor any other MapR objects.
ss	Start/stop services
cv	Create volumes
cp	Create security policies
a	Administrative access to cluster ACLs. Grants no other permissions.
fc	Full control over the cluster. This enables all cluster-related administrative options with the exception of changing the cluster ACLs.

Creating an ACL from the Command Line

To create an ACL at the command line, use the `acl set` command. Include spaces between multiple entries, such as a list of usernames and their associated permission levels (or *actions*).

The syntax is:

```
maprcli acl set -type volume -name <volume name>
                [-group <groupname>:<action> -user <username>:<action>]
```

Note: The `acl set` command *removes* previously set permissions if they are not explicitly called out in the command line.

Other ACL commands include:

- acl edit - to modify permissions in an ACL (use this command instead of `acl set` to change some permissions while leaving others intact)
- acl show - to display permissions in an ACL

Example

To create an ACL for a cluster named `my.cluster.com` that allows administration of cluster ACLs to user `root` and control over all other aspects of the cluster to all users in the `developers` group, enter this command:

```
maprcli acl set -type cluster -cluster my.cluster.com -user root:a -group
developers:fc
```

Now suppose you want to change the `developers` group permissions so they can only log in and start or stop services. Use the `acl edit` command as shown:

```
maprcli acl edit -type cluster -cluster my.cluster.com -group
developers:login,ss
```

Note that only the `developers` group's permissions change, while the user named `root` retains control over the cluster's ACL settings.

last-updated: Apr 11, 2022

Creating Job Queue ACLs

A job queue ACL controls who can submit jobs to a queue, kill jobs, or modify their priority. The default behavior is that any user can submit a job, and jobs can only be seen and killed by the administrator or the user that submitted those jobs.

To create a job queue ACL, specify the following parameters in the `mapred-queue-acls.xml` file:

Parameter	What it does
`mapred.queue.names` *`queue1,queue2,...`*	Names the queues to which jobs can be submitted.
`mapred.acls.enabled=true`	Indicates that ACLs will be checked whenever a user or group submits a job, tries to kill a job, or tries to change its priority. This parameter is set to true by default when security features for your cluster are enabled.
`mapred.queue.<queue-name>.acl-submit-job` *`user1,user2,... group1,group2,...`*	Identifies the users and groups that can submit jobs to the specified *queue-name*.
`mapred.queue.<queue-name>.acl-administer-job` *`user1,user2,... group1,group2,...`*	Identifies the users and groups that can change the priority or kill jobs submitted to the specified *queue-name*. Note that the job owner can always kill his own job or change its priority.

For information on configuring queue properties, see Configuring Properties for Queues. You can also set job initialization parameters for a queue.

Creating Volume-level ACLs

MapR provides volumes as a way to organize data and manage cluster performance. For example, you might want to create a volume for each user, department, or project. You can then create a volume-level ACL that controls which users and groups have access to that volume, and what actions they may perform.

You can create volume-level ACLs from the Control System or from the command line.

Creating Volume-level ACLs from the Control System

- For:
  - New volumes, see Creating a Volume on page 864 to set volume-level ACLs.
  - Existing volumes, see Modifying a Volume on page 892 to modify volume-level ACLs.

Creating Volume-level ACLs from the Command Line

To create an ACL at the command line, use the `acl set` command to specify a list of authorized users (or groups) and the actions they are allowed to perform.

The syntax is:

```
maprcli acl set -type volume -name <volume name> [-user
<username>:<action> -group <groupname>:<action>]
```

Include spaces between multiple entries, such as a list of usernames and their associated permission levels (or *actions*). Each allowed action has a permission code associated with it. The codes are explained below.

Permission Code	Allowed Action
dump	Dump or back up the volume
restore	Restore or mirror the volume
m	Modify the volume's properties
d	Delete the volume

last-updated: Apr 11, 2022

Permission Code	Allowed Action
a	Administrator (can edit and view ACLs, but cannot perform volume operations)
fc	Full control over the volume (this enables all volume-related administrative options with the exception of changing the volume ACLs)

Example

This example shows how to create an ACL for a volume named `test-volume` that allows full control over volume ACLs for user `rjones`. In addition, all users in the `developers` group are given permission to dump, restore, and modify volume properties.

```
maprcli acl set -type volume -name test-volume -user rjones:fc
-group developers:dump,restore,m
```

Managing Access Control Expressions
Defines and describes how to create, enable, and use ACEs.

Access Control Expressions (ACEs) are defined by a combination of user, group, or role definitions.

ACE Syntax
Describes how to construct access control expressions (ACEs).

An *ACE* is defined by a combination of user, group, or *role* definitions. You can combine these definitions using the following syntax:

Operator	Description
u	Username or user ID, as they appear in `/etc/passwd`, of a specific user. Usage: `u:<username or user ID>`
g	Group name or group ID, as they appear in /etc/group, of a specific group. Usage: `g:<group name or group ID>`
r	Name of a specific role. Usage: `r:<role name>`.
p	Public. Specifies that this operation is available to the public without restriction. Cannot be combined with any other operator. API request or CLI command to save such settings will return an error.
!	Negation operator. Usage: `!<operator>`.
&	AND operation.
|	OR operation
()	Delimiters for subexpressions.
""	The empty string indicates that no user has the specified permission.

An example definition is `u:1001 | r:engineering`, which restricts access to the user with ID 1001 or to any user with the role `engineering`.

In this next example, members of the group `admin` are given access, and so are members of the group `qa`:

```
g:admin | g:qa
```

For another example, suppose that you have this list of groups to which you want to give read permissions:

- The `admin` group as a whole, but not the admins for a particular cluster (which is named `cl3`).

last-updated: Apr 11, 2022

- Members of the `qa` group who are responsible for testing the two applications (named `app2` and `app3`).
- The business analysts (group `ba`) in department 7A (group `dept_7a`)
- All of the data scientists (group `ds`) in the company.

To grant the read permission, you construct this boolean expression:

```
u:cfkane | (g:admin & !g:cl3) | (g:qa & (g:app2 | g:app3)) | (g:ba &
g:dept_7a) | g:ds
```

This expression is made up of five subexpressions which are separated by OR operators.

- • The first subexpression `u:cfkane` grants the read permission to the username `cfkane`.
  - The subexpression `(g:admin & !g:cl3)` grants the read permission to the admins for all clusters except cluster `cl3`. The operator `g` is the group operator, the value `admin` is the name of the group of all admins. The `&` operator limits the number of administrators who have read permission because only those administrators who meet the additional condition will have it.

    The condition `!g:cl3` is a limiting condition. The operator `!` is the NOT operator. Combined with the group operator, this operator means that this group is excluded and does not receive the read permission.

    Warning: Be careful when using the NOT operator. You might exclude fewer people than you intended. For example, suppose that you do not want anyone in the group `group_a` to have access. You therefore define this *ACE*: `!g:group_a` You might think that the data is now protected because members of `group_a` do not have access to it. However, you have not restricted access for anyone else except the members of `group_a`. The rest of the world can access the data. You should not define *ACE*s through exclusion by using the NOT operator. You should define them by inclusion and use the NOT operator to limit further the access of the groups or roles that you have included.

    In the subexpression `(g:admin & !g:cl3)`, the NOT operator limits the number of members within the admin group who have access. The `admin` group is included, and all users who are also part of the `cl3` group are excluded.

- The subexpression `(g:qa & (g:app2 | g:app3))` demonstrates that you can use a subexpression within a subexpression. The larger subexpression means that only members of group `qa` who are also members of group `app2` or `app3` have read access to the data. The smaller subexpression limits the number of people who have this permission in the `qa` group.
- • The next two subexpressions -- `(g:ba & g:dept_7a)` and `g:ds` -- grant the read permission to the members of group `ba` who are also in the group `dept_7a`. It also grants permission to the members of the group `ds`.

Creating User Roles for ACEs
Describes how to create and use roles for access control.

Note: MapR recommends that you use Unix Groups over Roles whenever possible, for centralized maintenance. Use Roles only when you are unable to modify LDAP or AD groups easily.

A role is a label attached to a set of users, which defines a common task, or set of behaviors for those users. Roles enable you to use functionality similar to Unix groups for your users, without requiring you to alter the existing group hierarchy of your system. Role names can be up to 64 characters long, and cannot use the `:`, `&`, `|`, or `!` characters.

Standard Reference Implementation
User Information

The standard reference implementation is a library called `libmapr_roles_refimpl.so`. This library is located at `/opt/mapr/server/permissions`. This library opens a configuration file named `m7_permissions_roles_refimpl.conf`, which should contain a list of all the roles, and the users associated with those given roles. This configuration file is located at `/opt/mapr/conf`, and should be identical across all clusters.

The structure of the configuration file is as follows. Roles end with `:`, while user names are written on each subsequent line. For example:

```
Role_1:
            user_a
            user_b
```

```
Role_2:
            user_b
            user_c
            user_d
            #comment
```

This example file states that there are two roles to choose from when assigning permissions - `Role_1` and `Role_2`. The users located under `Role_1` are `user_a`and `user_b`. `Role_2` contains `user_b`, `user_c`, and `user_d`. Blank lines, and lines beginning with # are ignored.

Assume a table has permissions `r:Role_2`. `user_b` has access to this table, while `user_a` does not have access.

After adding a new role to the `m7_permissions_roles_refimpl.conf` file, you must issue the following command to enable the MapR File System layer to pick up the new role: `$ /opt/mapr/server/mrconfig dbrolescache invalidate`

Run this command on all the nodes, whenever there is a change in the roles configuration file.

Developer Information

The functions that the `libmapr_roles_refimpl.so` exposes, are found in the extensibility implementation. When the library is called initially through `GetSecurityMembership`, it parses the `m7_permissions_roles_refimpl.conf` file, and loads it into memory. All user names are read, and parsed into user IDs (uid_t). If a user ID is not found, the ID is skipped.

The library uses a HashTable. The roles are the keys. The values are a Binary Tree of user IDs.

Each call checks the given user ID and role. The HashTable keys off the role, and then searches the Binary Tree for the user ID. If the HashTable finds a user ID, it sets the boolean value of that role to `true`. If the HashTable does not find a user ID, or if any errors occur, such as `Role not found`, it sets the boolean value to `false`.

There is also a cleanup method which frees the memory allocated to the HashTable, along with all of its children. If the `GetSecurityMembership` method is called again, the library reloads the configuration file, and loads all the values into memory.

Extensible Implementation

If users decide not to use the reference implementation, they can replace the shared library with their own. In the `mfs.conf` file, add a parameter that specifies the name of the file. If the name of the file is changed, then MFS searches `/opt/mapr/server/permissions` for the new file. If the file is found, it is loaded into memory. If not, then all roles evaluate to `false`.

The user's shared library should contain two functions specified under the `mapr::fs` namespace:

```
extern "C"
            void GetSecurityMembership(uid_t uid, const char *roles[], int
numRoles, bool truthValue[]) {
            }
```

```
extern "C"
            void cleanup() {
            }
```

`GetSecurityMembership` takes the given user ID along with a list of all the roles, the amount of roles in the array, and an array of all the results, as booleans.

Users must code their own implementation of populating the `truthValue` array with either `true` or `false`. The `truthValue` array has the same length as `numRoles`, and is initialized. Do not modify any other variables.

Use the `cleanup` method to reset the shared library to an initialized state. This method resets all values, and frees memory, since the shared library is not closed, till the class calling it, gets destructed.

Invoke Shared Library from MFS

The `TablePermissions` class handles opening, and closing the shared library. During class initialization, the name of the shared library that is read from the `mfs.conf`, file, is passed to the constructor. The constructor loads the shared library into memory, using the `LoadSO` method from `filterutils.cc`. The constructor also loads the `GetSecurityMembership` method, along with the `cleanup` method, as variables that can be called.

`TablePermissions` contains two methods that can be called, to access the shared library:

- The `GetSecurityMembership` method takes 3 arguments - the user ID, the array of roles, and the amount of roles in the array. This method returns a `RolePermission` structure, which contains all the same data, as well as the boolean of the successful roles for that given user ID. To evaluate the user roles, pass this `RolePermission` structure to the `TablePermission::checkTablePermissions` method.
- The `cleanup` method calls the `cleanup` method in the shared library. This method takes no arguments.

The entity that allocates the `RolePermission` structure into memory, also needs to deallocate this structure.

Deallocating the `TablePermissions` class, calls the `cleanup` method, and closes the shared library.

Shared Library Security

The `/opt/mapr/server/permissions` folder is initialized with `755` permissions. This implies that only the user who installed MapR has access to writing to that folder. These permissions prevent a user from replacing a shared library with a malicious file.

The `m7_permissions_roles_refimpl.conf` file has `755` permission, which means that only an administrator can make changes to this file.

The Roles Library Shared Object and Access Control Expressions

When you access an object that is secured by an *ACE* , the MapR File System layer calls the roles library shared object, and checks the permissions of the entity requesting access against the contents of the roles file. The roles library shared object reads the roles file every 600 seconds. You can specify your own roles

last-updated: Apr 11, 2022

library shared object, and specify the location of that object, using the parameter `mfs.dbroles.sopath` in the `/opt/mapr/conf/mfs.conf` file.

Enabling Volume, Directory, and File Authorizations with ACEs
Describes how to set access control expressions for volumes, directories and files.

*ACE*s allow you to define whitelists (to grant access) and blacklists (to deny access) for a combination of users, roles, and groups. You can grant different permissions to multiple users, groups, and roles for MapR File System files, directories, and whole volume data using boolean expressions and subexpressions.

ACEs for Files, Directories, and Whole Volume

An *ACE* is defined by a combination of user, group, and/or role definitions. You can combine these definitions using the supported syntax. For more information, see Syntax of Access Control Expressions.

The examples in the following table demonstrate how *ACE*s can be used to create whitelists to grant access, and blacklists to deny access.

This Access Control Expression...	Grants access to...	Denies access to...
`(u:u1&g:g1)`	only user 'u1', if user 'u1' is a member of group 'g1'	users who are not 'u1' and members of group 'g1'
`(g:g1&g:g2)|r:r1`	only users who are in both the groups 'g1' and 'g2', or users who are assigned role 'r1'	users who are not in both the groups 'g1' and 'g2', and users who are not assigned role 'r1'
`(g:g1&!g:g2)`	only users who are in group 'g1' and not in group 'g2'	users who are in group 'g2', even if they are in group 'g1', and all other users
`(g:g1|g:g2)`	users who are in groups 'g1' or 'g2' only	users who are not in groups 'g1' or 'g2'
`(g:g1|g:g2)&!r:r1`	only users in groups 'g1' or 'g2' and who are not assigned role 'r1'	users who are not members of groups 'g1' or 'g2', users who are assigned role 'r1', even if they are in group 'g1' or 'g2', and all other users
`(p)`	everyone	none
`(!g:g1&!g:g2&!g:g3)`	users who are not in groups 'g1', 'g2', and 'g3'	only users who are in groups 'g1', 'g2', or 'g3'
`((u:u1|u:u2|u:u3)&g:g1&g:g2)&!r:r1`	only users 'u1', 'u2', or 'u3', who are also members in groups 'g1' and 'g2', but not assigned role 'r1'	users who are not 'u1', 'u2', or 'u3' and members of groups 'g1' and 'g2', and users who are assigned role 'r1'
`(u:u1|u:u2|u:u3)&g:g1|g:g2`	only users who are 'u1', 'u2', or 'u3' and who are members in groups 'g1' or 'g2'	users who are not 'u1', 'u2', or 'u3' and members of groups 'g1' or 'g2'

Note: The entities — user, group, role, and public — must be the same for MapR File System and MapR Database *ACE*s.

Managing File and Directory ACEs
Describes the implications of setting access control expressions on files and directories.

File *ACE* allows you to define access (whitelist and blacklist) to files and directories for a combination of users, groups, and roles. If *ACE*s are not set, POSIX mode bits for the file or directory are used to grant or deny access to the file or directory.

When you set *ACE*s, MapR sets or resets the corresponding POSIX mode bits to match the permissions granted through *ACE*s. For more information, see Setting/Modifying File and Directory ACEs.

last-updated: Apr 11, 2022

- If both *ACE*s and POSIX mode bits are set, access is granted if access is allowed through *ACE*s or POSIX mode bits.
- If *ACE*s are not set, POSIX mode bits are used to grant access.
- If neither *ACE*s nor POSIX mode bits are set, access is denied.

The owner of the file or directory (and mapr and root users) can set, modify, and remove *ACE*s for that file or directory using `hadoop mfs` commands.

File ACEs

You can set and modify permissions to read, write, and execute files using the `hadoop mfs` command or the FileACE Java APIs on page 1457 and FileACE C APIs on page 1457. Specifically, the following access types are supported.

Access Type		Description
Command-Line	**Java API (Enum)**	
`-readfile`	`READFILE`	Read a file.
`-writefile`	`WRITEFILE`	Write to a file.
`-executefile`	`EXECUTEFILE`	Execute a file.

For more information, see `hadoop mfs`, FileACE Java APIs on page 1457, and FileACE C APIs on page 1457.

Directory ACEs

You can set the same *ACE*s on directories as for files. In addition, directory *ACE*s support permissions to list, add child, delete child, and lookup directories using `hadoop mfs` command. Specifically, the following access types are supported.

Access Type		Description
Command-Line	**Java API (Enum)**	
`-readfile`	`READFILE`	Read a file.
`-writefile`	`WRITEFILE`	Write to a file.
`-executefile`	`EXECUTEFILE`	Execute a file.
`-readdir`	`READDIR`	List the contents of a directory. This access is required to write and/or execute files in the directory.
`-lookupdir`	`LOOKUPDIR`	Lookup a file in a directory. This access is required to find, read, write, and/or execute files in the directory.
`-addchild`	`ADDCHILD`	Add a file or subdirectory.
`-deletechild`	`DELETECHILD`	Delete a file or subdirectory.

Although you can set both file and directory *ACE*s on directories, only the directory *ACE*s are used for determining access to the directory. The file *ACE* on the directory is used as the default *ACE* setting for new files under that directory.

By default, when you set *ACE*s on a parent directory:

- Permissions for existing files and subdirectories under that parent remain unchanged.

last-updated: Apr 11, 2022

- New files under that parent inherit the file *ACE*s and corresponding POSIX mode bits of the parent directory, if available. Otherwise, new files get the default *ACE*, the empty string (""), which indicates that no one has permissions to read, write, or execute the file; POSIX mode bits are set on the file in the traditional way.
- New subdirectories under the parent inherit both the directory and file *ACE*s and corresponding POSIX mode bits from the parent directory.

Note: When accessing files and directories, the *ACE*s on files have no effect on accessing the parent directory.

Workaround for Execute Operation when ACES are set on an executable file

When *ACE*s are set on any file, mode bits are cleared. However, for a binary to execute, the kernel checks whether the execute bit is set or not, and restricts execution if it is not set. To run an executable file with *ACE*s set on it, use one of the following workarounds:

1. Set owner mode exec bit on binaries/shell scripts.
2. Set group mode exec bit on binaries/shell scripts.
3. Change owning group for the files to the group used in MapRAces, and set the executable group mode bit.

Setting File and Directory ACEs

Describes how to set access control expressions (ACEs) for files and directories.

For files and directories, run the `hadoop mfs` command to set *ACE*s. When *ACE*s are set, by default, the corresponding POSIX mode bits are also set. POSIX mode bits for owner and owning group are deduced by evaluating the corresponding *ACE*s. POSIX mode bits for others is set only if `"p"` is given as the value for an *ACE*.

The following table lists the POSIX mode bits that correspond to the access types.

	ACE	POSIX Mode Bits
File	`readfile`	r
	`writefile`	w
	`executefile`	x
Directory	`readdir`	r
	`addchild`	w
	`deletechild`	w
	`lookupdir`	x

The POSIX mode bit granting write (`w`) access to directory is set only if user, role, or group is granted permission for both (`addchild` and `deletechild`) access types.

The `hadoop` command, by default, sets the POSIX mode bits corresponding to the given *ACE*s, and:

- Overwrites existing *ACE* values with new values, if specified, for access types that were previously set.
- Sets *ACE* values for access types that have not yet been set, if specified.
- Does not modify access types that are not specified with the command, whether or not they were previously set.

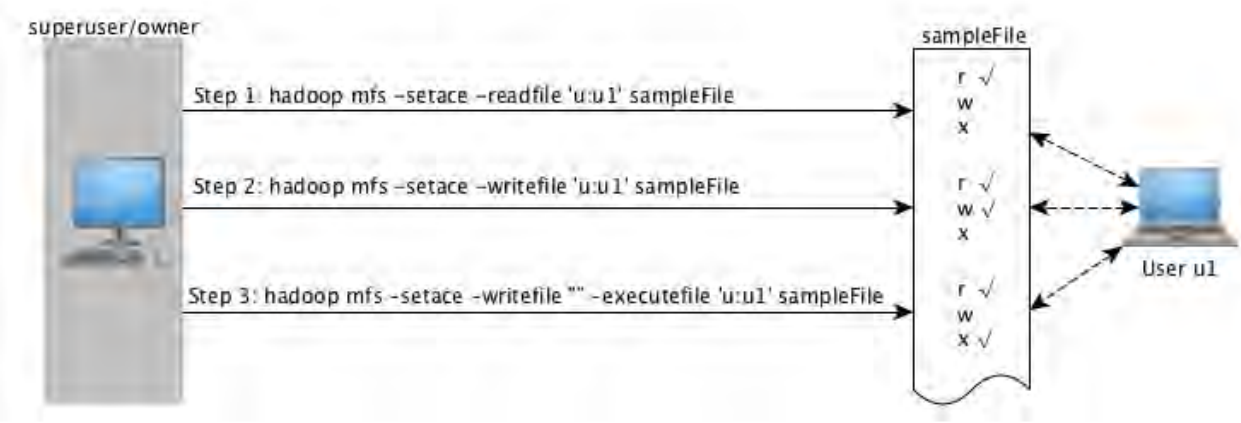
last-updated: Apr 11, 2022

Warning: Changing the POSIX mode bits using `chmod` does not change the corresponding *ACE* setting and may result in different, conflicting permissions to files and directories.

File ACE Example

Illustrates setting access control expressions for files.

Suppose the following sequence of file *ACE* settings (and corresponding POSIX mode bits) are set for user u1.



As shown in the preceding illustration, in:

Step 1:

User u1 is granted permissions to read a file, `sampleFile`.

After the command runs, user u1 has permissions to (only) read the file. The POSIX mode bit for reading the file is set to u1 for owner/users.

There is no change in *ACE*s or POSIX mode bits for all other (write and execute) access types.

Step 2:

User u1 is granted permissions to write to the same file.

After the command runs, user u1 has permissions to write to the file. The POSIX mode bit for writing to the file is set to u1 for owner/users.

There is no change in *ACE*s or POSIX mode bits for all other (read and execute) access types.

Step 3:

User u1's permissions are modified to remove write permission (using the empty string) and to grant access to execute file.

After the command runs, user u1 has permissions to execute the file, but user u1 can no longer write to the file. The POSIX mode bit for:

- Writing to the file is set to 0 for owner/users, groups, and others.
- Executing the file is set to u1 for owner/users.

Note: When the empty string ("") is used to deny a specific type of file access, that type of file access is denied to all users, groups, and roles. To deny access to specific users only, use the negation operator (`!`).

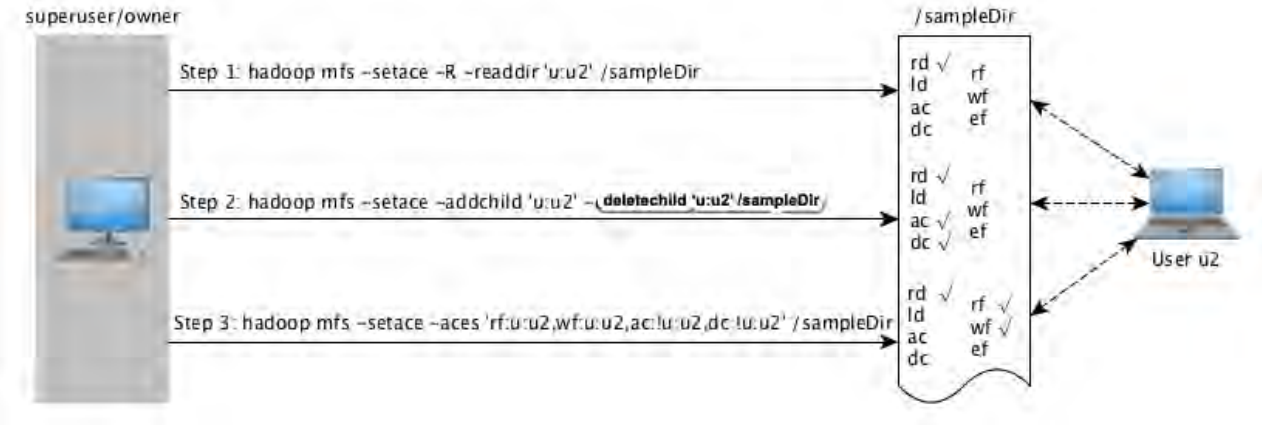
last-updated: Apr 11, 2022

There is no change in *ACE*s or POSIX mode bits for all other (read) access types.

Directory ACEs Example

Explains how to set access control expressions for directories.

For example, suppose the following diagram depicts the (command-line) sequence of directory *ACE* settings for user u2:



As shown in the preceding illustration, in:

Step 1:

User u2 is granted access to read directory, and sampleDir, while all other directory/file *ACE*s are not specified.

After the command runs, user u2 has permissions to list the contents of the directory. The POSIX mode bits for listing the contents of the directory (`r`) is set to u2 for owner/users.

There is no change in *ACE*s or POSIX mode bits for all other (file- and directory-level) access types.

Step 2:

User u2 is granted permission only to add and delete child directories, while all other directory/file *ACE*s are not specified.

After the command runs, user u2 has permissions to create and delete child directories. The POSIX mode bit for writing (`w`) to the directory for owner/user is set to u2 because user u2 is granted access for both (`addchild` and `deletechild`) access types.

If user u2 creates child directories, the child directories, by default, inherit the *ACE* settings of the parent directory.

There is no change in *ACE*s or POSIX mode bits for all other (file- and directory-level) access types.

Step 3:

User u2's permissions are modified to grant access to read and write to files in the directory, user u2's permissions for adding and deleting child directories are removed (using the negation operator), and all other directory/file *ACE*s are not specified.

After the command runs, user u2 can read and write to files in the directory, but user u2 can no longer add

last-updated: Apr 11, 2022

and delete child directories. The POSIX mode bits for directory write access (`w`) is set to 0 for owner/user.

Although, at the directory level, user u2 has permissions to read and write to files in the directory, for existing files, the file level *ACE*s or the POSIX mode bits for the file are used to determine access. However, by default, user u2 gets read and write permissions to all new files created under the directory. If user u2 creates new files under the directory, the files inherit the file *ACE*s from the parent directory by default and the POSIX mode bits for read (`r`) and write (`w`) access are set to u2 for owner/user.

There is no change in *ACE*s or POSIX mode bits for all other (`lookupdir` and `executefile`) access types.

Deleting File and Directory ACEs
: Describes how to delete file and directory ACEs using the CLI.

  You can remove all *ACE* associated with a file or directory using the `hadoop mfs -delace` command. When you delete all the *ACE*s, the system sets the *ACE* for the file or directory to the default value, which is the empty string (""). The POSIX mode bits are not reset; if necessary, run the `chmod` command to reset POSIX mode bits.

  You cannot remove specific access types that have been set; instead, use the empty string to deny specific types of access. When the empty string ("") is used to deny a specific type of access, that type of access is denied to all users, groups, and roles. To deny access to specific users only, use the negation operator (`!`). If you use the empty string ("") or the negation operation (`!`) to deny a specific type of access, the corresponding POSIX mode bit are also reset to match the *ACE* setting.

FileACE Java APIs
: Contains the path to the Java FileACE APIs.

  The Java FileACE APIs are located at File ACE APIs.

FileACE C APIs
: Describes the C FileACE APIs.

  The FileACE C APIs are defined in the header file `hdfs.h` and are as follows:

  hdfsSetAces

  Sets the ACEs for a file or directory.

  Syntax

```
int hdfsSetAces(hdfsFS fs, const char *path, hdfsFileAces *faces, int
isSet, int isRecursive);
```

  Parameters

  - fs: The configured filesystem handle
  - path: The path to the file or directory for which the ACEs need to be set.
  - faces: The ACEs to set
  - isSet: Set to 0 to merge with any existing ACEs, or set to 1 to replace all existing ACEs.
  - isRecursive: Set to 1 to apply ACEs recursively when setting on a directory.

  Return Value

0 on success, else an error code.

hdfsGetAces

Gets the ACEs from a file or directory.

Syntax

```
int hdfsGetAces(hdfsFS fs, const char *path, void *aceBuf, int bufLen,
hdfsFileAces *faces);
```

Parameters

- fs: The configured filesystem handle
- path: The path to the file or directory from which the ACEs need to be fetched.
- aceBuf: The buffer to hold the ACEs.
- bufLen: Length of the ACE buffer (*aceBuf*)
- faces: The structure that contains the returned ACEs.

Return Value

0 on success, else error ERANGE, which indicates that the buffer is too small to hold the ACE entries.

hdfsDeleteAces

Deletes all ACEs from a file or directory.

Syntax

```
int hdfsDeleteAces(hdfsFS fs, const char *path);
```

Parameters

- fs: The configured filesystem handle
- path: The path to the file or directory from which the ACEs need to be deleted.

Return Value

0 on success, else an appropriate error.

Managing Whole Volume ACEs

Describes how to grant permissions to users, groups, and roles for the volume data using whole volume access control expressions (ACEs).

Whole volume *ACE*s allow you to define whitelists to grant access, and blacklists to deny access, for files and tables within a volume.

Volume administrators and mapr user can set and modify whole volume *ACE*s. By default, *ACE*s grant everyone access to read and write to files and tables in the volume at the volume-level; however, inside the volume, to determine access for:

- Files, the file *ACE*s or POSIX mode bits are used.
- Tables, the table *ACE*s are used.

Supported Access Types

At the volume level, the following access types are supported:

Access Type	Description
`-readAce`	Read files, MapR Database binary tables, MapR Database JSON tables, and MapR streams in the volume. By default, this is set to p to grant all users this permission.
`-writeAce`	Write to files, MapR Database binary tables, MapR Database JSON tables, and MapR streams in the volume. By default, this is set to p to grant all users this permission.

ACE Behavior on Snapshots and Mirrors

Volume Snapshots

Volume snapshots reflect the *ACE*s of the volume at that point in time. Changes in volume *ACE*s:

- Are carried over to a new snapshot of the volume.
- Do not propagate to older snapshots of the volume.

Volume Mirrors

*ACE*s of a volume are propagated to mirror volumes. After each mirroring operation, mirror volumes reflect the current *ACE* setting of their source volume. After a mirror volume is promoted to a read-write volume, you can modify the *ACE*s on the mirror volume from the command line. *ACE*s on the promoted mirror volume can be different from the source volume.

Setting Whole Volume ACEs

Describes how to set ACE expressions when creating or modifying volumes.

You can set *ACE*s at the time of volume creation using the `volume create` on page 1931 command and modify them at a later time using the `volume modify` on page 2005 command. When you run the command to set or modify *ACE*s, the command does the following:

- Overwrites existing values with new values, if specified, for access types that were previously set.
- Sets values for access types that have not yet been set, if specified.
- Does not modify access types that were not specified with the command, whether they were previously set or are unset.

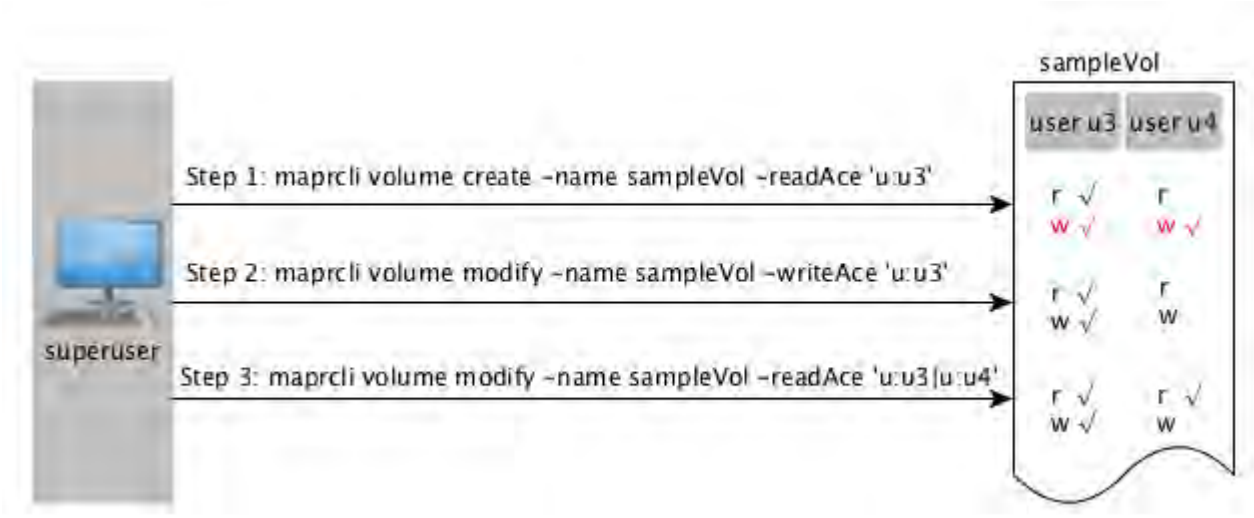
When you set whole volume *ACE*s, permissions on files and tables under that volume remain unchanged. Also, new files and tables in the volume do not inherit the whole volume *ACE*s of that volume. Instead, whole volume *ACE*s, if set, are used to determine volume level access to tables and files within the volume. To gain access to volume data, the user must have access at both the volume and file/table levels.

Whole Volume *ACE* Example

For example, suppose the following sequence of whole volume *ACE* settings for users u3 and u4 is as follows.

Note: In the following illustration, default *ACE* values are shown in red.

last-updated: Apr 11, 2022



As shown in the illustration above, in:

Step 1:

User u3 is granted permissions to read.

User u3: User u3 has permissions to read files and tables at the volume level and by default, user u3 has write permission (shown in red) at the volume level. However, for:

- Files in the volume, file *ACE* or POSIX mode bits are used to determine read and write access for user u3.
- Tables in the volume, table *ACE*s are used to determine read and write access for user u3.

User u4: User u4 cannot read files and tables within the volume because the *ACE* for the volume does not explicitly grant access to user u4. Although user u4 has write permission by default, user u4 cannot write to files/tables in the volume because user u4 does not have read permission.

Step 2:

User u3 is granted permissions to write.

User u3: User u3's read access remains unchanged and although user u3 has permissions to write to files and tables, for:

- Files in the volume, file *ACE* or POSIX mode bits are used to determine write access for user u3.
- Tables in the volume, table *ACE*s are used to determine write access for user u3.

User u4: User u4 cannot write to files/tables in the volume.

Step 3

User u4 is granted read access.

User u3: User u3's read and write access remains unchanged.

User u4: User u4 has permissions to read files and tables at the volume-level; however, for:

last-updated: Apr 11, 2022

- Files in the volume, file *ACE* or POSIX mode bits are used to determine read access for user u4.
- Tables in the volume, table *ACE*s are used to determine read access for user u4.

Deleting Whole Volume ACEs

You cannot remove ACEs that have been set; instead, if necessary, use the empty string (`""`) to deny specific types of access. If the empty string is used to deny a specific type of access, that type of access is denied to all users. To deny access to specific users only, use the negation operator (`!`).

Enabling Table and Stream Authorizations with ACEs

Permissions for MapR tables, column families, and columns are defined by Access Control Expressions (ACEs). You can set permissions for tables when you create or edit tables. You can set default permissions for column families when you create or edit tables, and you can override these defaults when you create column families.

For the syntax to use when creating Access Control Expressions, see ACE Syntax on page 1448.

When a user, group, or role requests to read data from, write data to, or append data to a column, MapR Database checks whether that user, group, or role has read or write permission for the column family AND read or write permission for the column. By default, columns allow read and write access to all users; in such cases, only the read or write permission for the column family matters.

However, suppose that a table contains columns `col1` and `col2` in column family `cf1`, and these columns grant read and write permission only to the table creator. A different user tries to write data to these columns. MapR Database checks whether this user has write permission on `cf1` AND `col1` AND `col2`. If the user does not have all three permissions, MapR Database returns an error that says access for the write is denied.

If this user were to try to read from the same two columns, MapR Database would simply not return the data. If the user tried to read from those two columns and additional columns on which he had read permissions, the results would contain the data for those additional columns but exclude the data for `col1` and `col2`.

Note: Table Permissions for Older Releases: Because MapR tables are stored at the file-system level, you can also set permissions for MapR Database tables directly in the file system, if your version of MapR does not support Access Control Expressions (ACEs). Support for ACEs was introduced in version 3.1.

To set permissions directly in the filesystem, see Performing File System Operations on MapR Database Tables on page 1041.

Setting Table ACEs Using the CLI

You can set ACEs with the following commands:

- `table create` on page 1788 — Creates a new MapR table.
- `table edit` on page 1822 — Edits a MapR table.
- `table cf create` on page 1799 — Creates a column family for a MapR table.
- `table cf edit` on page 1806 — Edits a column-family definition.
- `table cf colperm set` on page 1795 — Set Access Control Expressions (ACEs) for a specified column.

Setting Stream ACEs Using the CLI

You can set ACEs with the following commands:

- `stream create` on page 1758 — Creates a new MapR stream.
- `stream edit` on page 1765 — Edits a MapR stream.

Permission Types for Fields and Column Families in JSON Tables

By using access-control expressions (ACEs), you can grant or deny access to fields and column families that are in JSON tables.

For an explanation of the syntax of ACEs, see ACE Syntax.

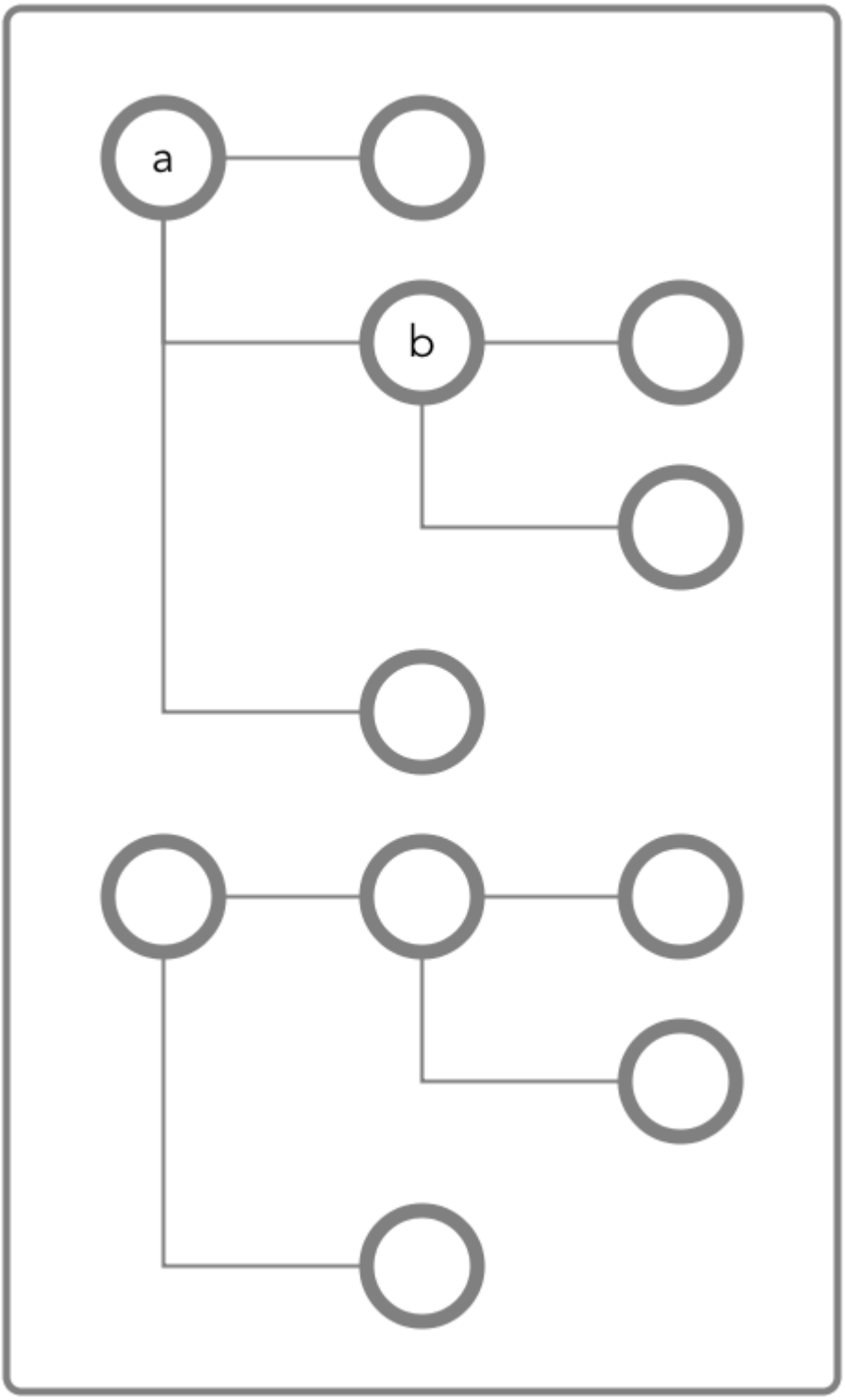
There are three types of permission:

- Traverse (`traverseperm`)
- Read (`readperm`)
- Write (`writeperm`)

Traverse (`traverseperm`)

This permission allows the grantee to descend a hierarchy of fields to access the fields that the grantee has write or read permission on.

For example, suppose that a user has read and write access only on field b in this document.

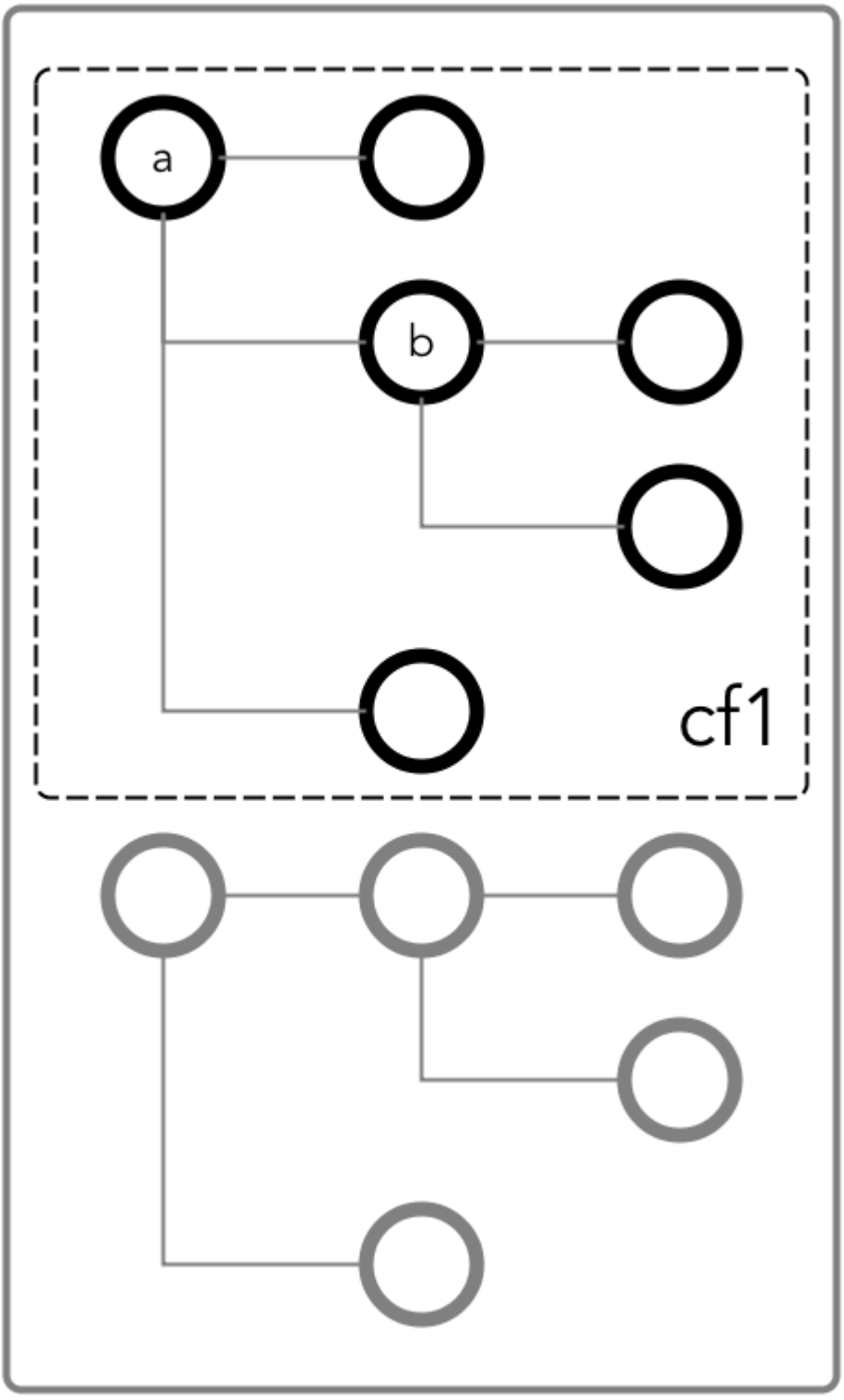


To access field b, the user would need to be able to traverse (pass through) field a. In this case, because the entire document is in the default column family, the user could be granted traverse permission on the default column family. Field a would inherit the traverse permission.

If traverse permission on the default column family were denied the user, it would not be possible for the user to access field b. Granting traverse permission on field a in this case would have no effect.

last-updated: Apr 11, 2022

In the next example, field a is a column family named `cf1`.



To be able to read and write at field b, the user could be granted the traverse permission on the column family.

Read (`readperm`)

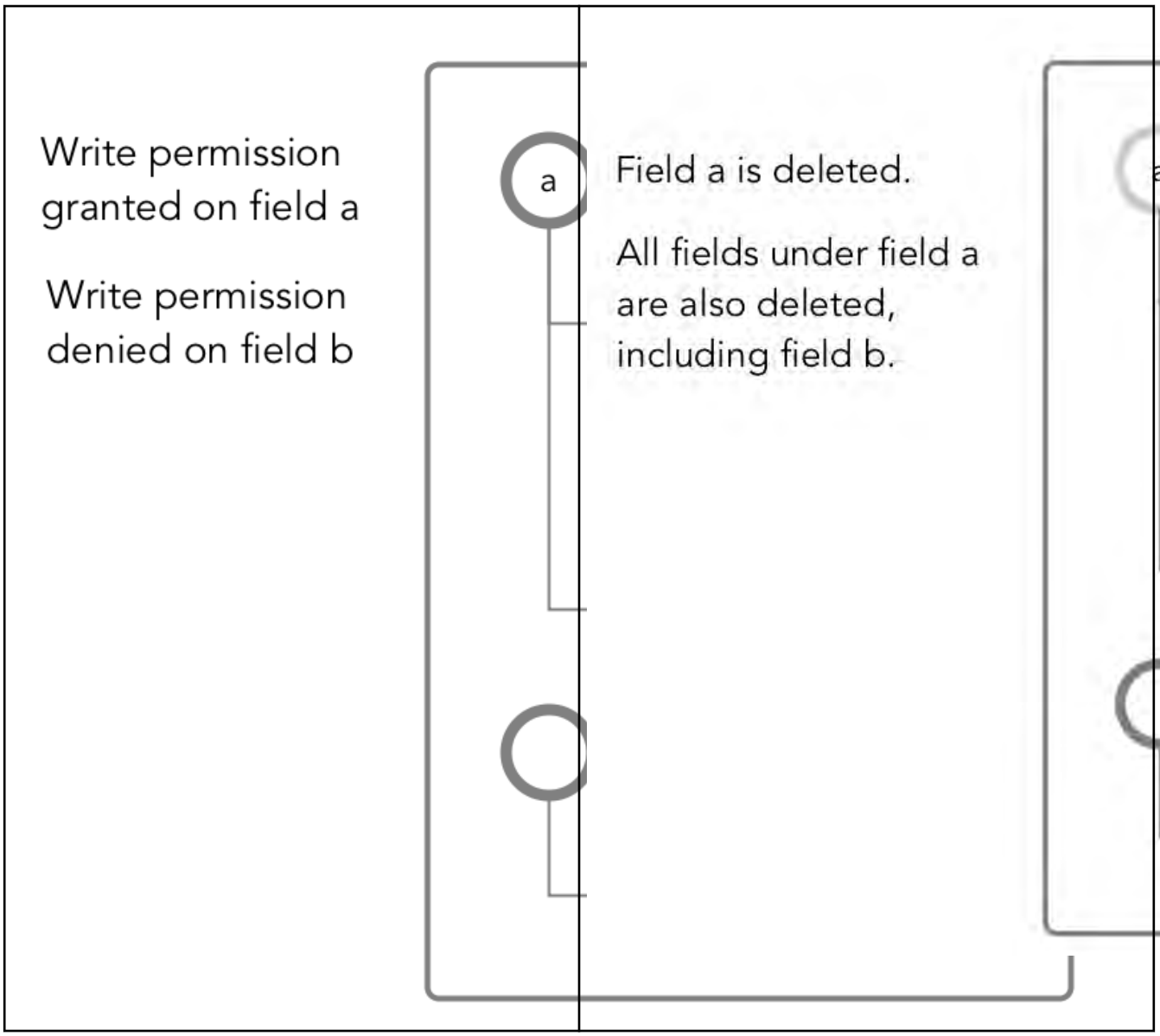
This permission allows the grantee to read from a field.

This permission extends to fields that are nested below the field that the permission was granted on. However, grantees can be explicitly denied the permission on any of the nested fields.

Write (`writeperm`)

This permission allows the grantee to delete a field, insert a value into a field, or overwrite a field's value.

As illustrated in the following two diagrams, deleting a field also deletes all fields that are nested within that field, even those fields on which the write permission is explicitly denied.



Obtaining `readperm` and `writeperm` on Fields

In this scenario, you want to perform an operation on a field, and the operation requires that you have readperm and writeperm permissions on that field. How you obtain these permissions depends on whether the field is in the default column family or a non-default column family.

last-updated: Apr 11, 2022

When the field is in the default column family

In the document below, you want to perform an operation on field c, which is in the default column family. The operation requires you to have `readperm` and `writeperm` on field c.

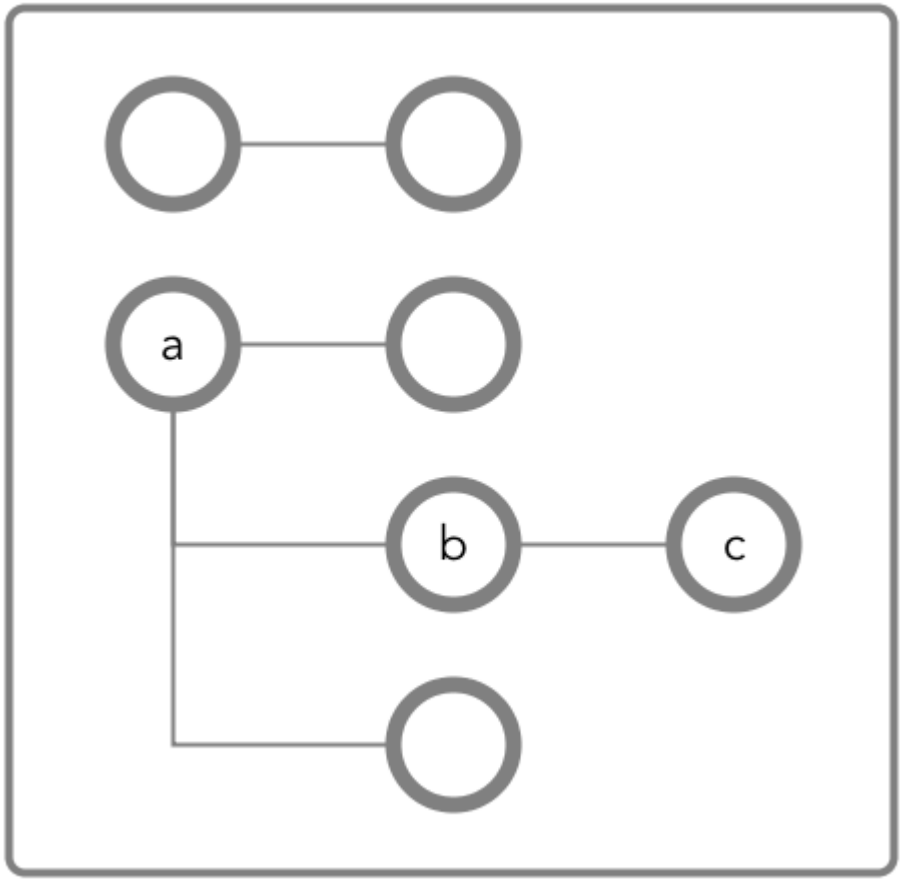


Figure 19: Schematic diagram of an JSON document in which all fields are in the default column family

Case 1: You have `readperm` and `writeperm` on the default column family

In this case, field c inherits these permissions, assuming that the permissions were not denied on field a or b.

If you do not have `readperm` and `writeperm` on field a or b, you need `traverseperm` on the field that denied you those permissions. You also need `readperm` and `writeperm` explicitly granted to you on field c. You could be granted these permissions with the `maprcli table cf colperm set` command, as in these examples:

```
maprcli table cf colperm set -path
<path to JSON table>
-cfname default -name
a.b -traverseperm u:<user ID> |
<existing ACE for this field>
maprcli table cf colperm set -path
<path to JSON table> -cfname default
```

```
-name a.b.c -readperm u:<user ID>
| <existing ACE for this
field> -writeperm
u:<user ID> | <existing ACE for this
field>
```

Case 2: You do not have `readperm` and `writeperm` on the default column family

In this case, you need the `traverseperm` permission on the default column family. Fields a and b inherit this permission. You also need `readperm` and `writeperm` on field c.

You could be granted these permissions with commands similar to these:

```
maprcli table cf edit -path
<path to JSON table> -cfname
default -traverseperm
u:<user ID> | <existing ACE for this
field>
maprcli table cf colperm set -path
<path to JSON table> -cfname
default -name a.b.c
-readperm u:<user ID> | <existing ACE
for this field> -writeperm u:<user
ID> |
<existing ACE for this field>
```

When the field is in a non-default column family

Note: Non-default column families are an advanced feature of MapR Database's native JSON support. For information about them, see Column Families in JSON table.

In the following document, you want to perform an operation on field c, which is in the column family `cf1` that is defined at field b with the path `a.b`.

last-updated: Apr 11, 2022

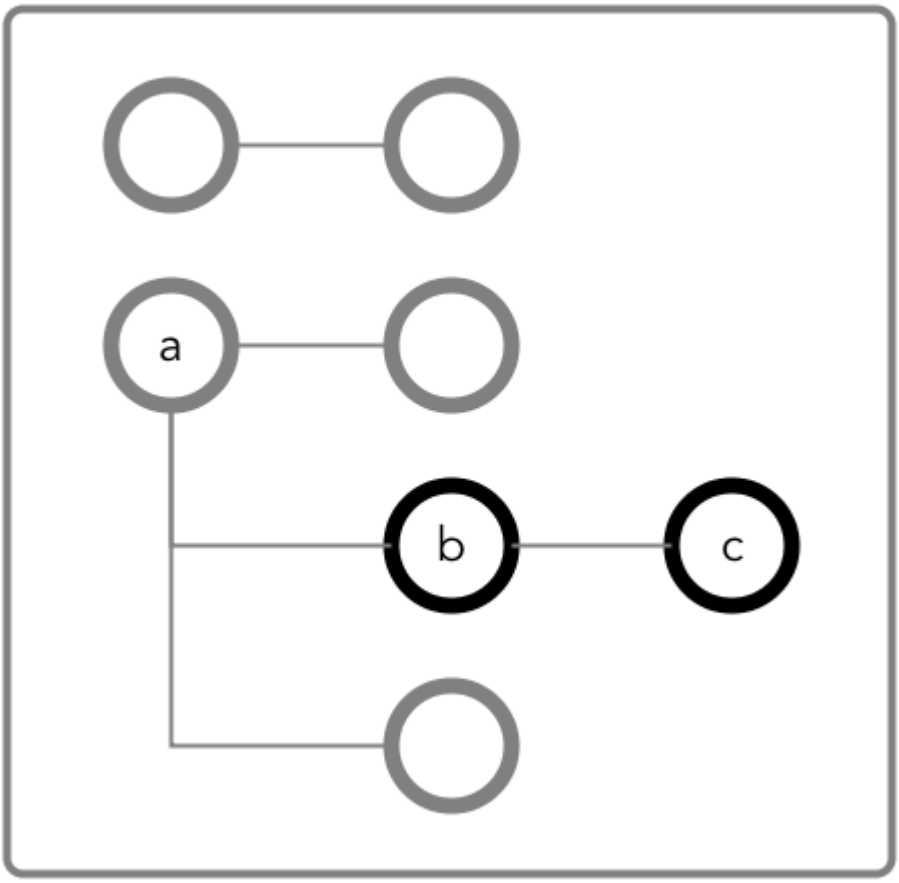


Figure 20: Schematic diagram of an JSON document in which fields b and c are in a column family that has the path `a.b`

Case 1: You do not have `readperm` and `writeperm` on field `b`	You need `traverseperm` on field b and both `readperm` and `writeperm` on field c. You can be granted these permissions with commands similar to these:
	`/opt/mapr/bin/maprcli table cf edit -path <path to JSON table> -cfname cf1 -traverseperm u:<user ID> | <existing ACE for this field>`
	`maprcli table cf colperm set -path <path to JSON table> -cfname cf1 -name a.b.c -readperm u:<user ID> | <existing ACE for this field> -writeperm u:<user ID> | <existing ACE for this field>`

Case 2: You do have `readperm` and `writeperm` on field `b`	You do not need any further permissions. Field c inherits your `readperm` and `writeperm` permissions from field b.

Obtaining `readperm` or `writeperm` on Fields

In this scenario, you want to perform an operation on a field, and the operation requires that you have readperm or writeperm permissions on that field. How you obtain either permission depends on whether the field is in the default column family or a non-default column family.

When the field is in the default column family

In the following document, you want to perform an operation on field `c`, which is in the default column family. The operation requires you to have `readperm` or `writeperm` on field `c`.

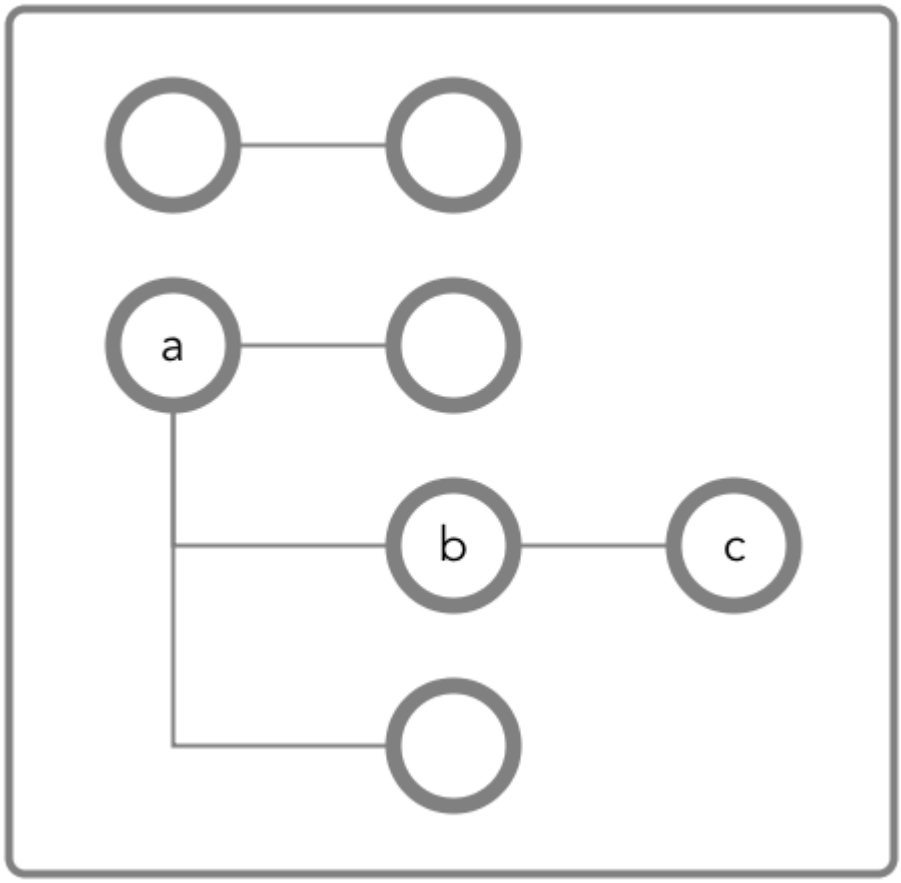


Figure 21: Schematic diagram of an JSON document in which all fields are in the default column family

Case 1: You have the same permission (`readperm` or `writeperm`) on the default column family

In this case, field `c` inherits the permission, assuming that the permission was not denied on field `a` or `b`.

If you do not have `readperm` or `writeperm` on field `a` or `b`, you need `traverseperm` on the field that denied you the permission that you need. You also need `readperm` or `writeperm` explicitly granted to you on field `c`.

Example commands to grant these permissions:

```
/opt/mapr/bin/maprcli table cf
colperm set -path <path to JSON
table> -cfname
default -name a.b -traverseperm
```

```
u:<user ID> | <existing ACE for this
field>
```

The next example command grants `readperm`:

```
/opt/mapr/bin/maprcli table cf
colperm set -path <path to JSON
table> -cfname
default -name a.b.c -readperm u:<user
ID> | <existing ACE for this field>
```

Case 2: You do not have the same permission (`readperm` or `writeperm`) on the default column family

In this case, you need the `traverseperm` permission on the default column family. You also need `readperm` or `writeperm` explicitly granted to you on field `c`.

Example commands to grant these permissions:

```
/opt/mapr/bin/maprcli table cf
edit -path <path to JSON
table> -cfname cf1
-traverseperm u:<user ID> | <existing
ACE for this field>
```

This next example command grants `readperm`:

```
/opt/mapr/bin/maprcli table cf
colperm set -path <path to JSON
table> -cfname cf1
-name a.b.c -readperm u:<user ID> |
<existing ACE for this field>
```

When the field is in a non-default column family

Note: Non-default column families are an advanced feature of MapR Database's native JSON support. For information about them, see Column Families in JSON Tables.

In the following document, you want to perform an operation on field `c`, which is in the column family that is defined at field `b` with the path `a.b`. The operation requires you to have `readperm` or `writeperm` on field `c`.

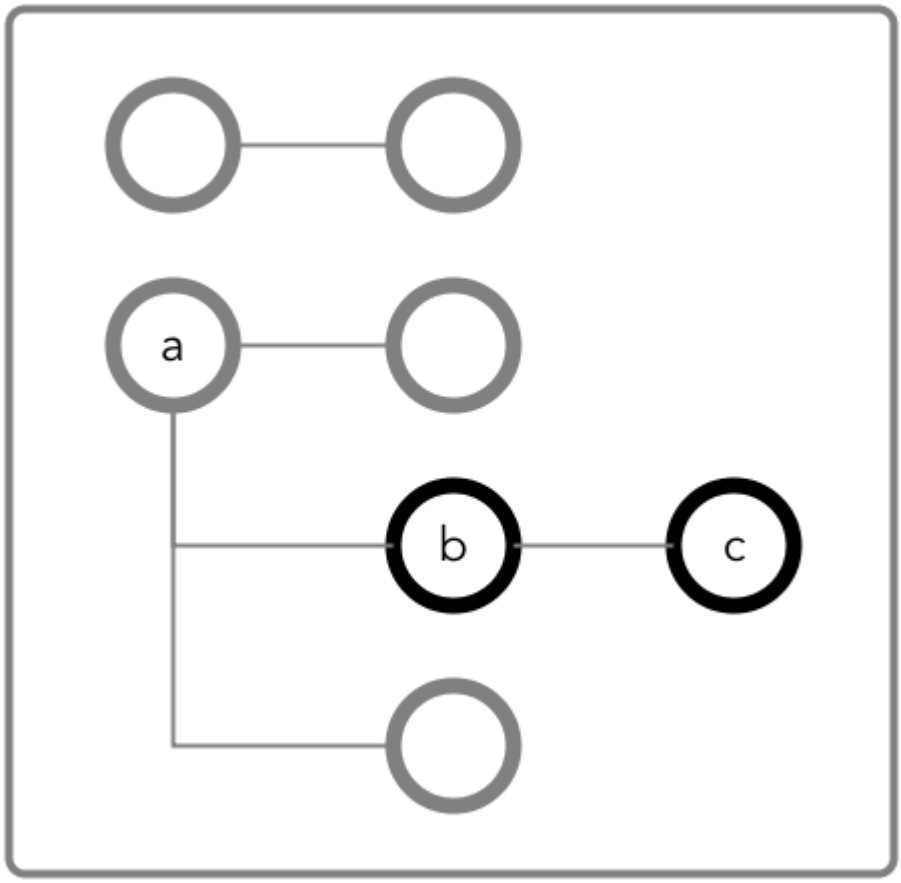


Figure 22: Schematic diagram of an JSON document in which fields b and c are in a column family that has the path `a.b`

Case 1: You do not have the permission you need (`readperm` or `writeperm`) on field `b`	You need `traverseperm` on field b, and you need `readperm` or `writeperm` granted to you explicitly on field c.
	Example commands to grant these permissions:
	`/opt/mapr/bin/maprcli table cf edit -path <path to JSON table> -cfname cf1 -traverseperm u:<user ID> | <existing ACE for this field>`
	`maprcli table cf colperm set -path <path to JSON table> -cfname cf1 -name a.b.c -readperm u:<user ID> | <existing ACE for this field>`

Case 2: You do have the permission you need (`readperm` or `writeperm`) on field `b`	You do not need any further permissions. Field c inherits your `readperm` and `writeperm` permissions from field b.

Setting Permissions on Arrays

When granting permissions on a field, if the field contains array data, you must grant the permission on the array field. This grants access not only to array data in the field, but also nested documents and scalar data. It is also possible to set permissions on subfields within nested documents that are stored in an array.

Note: This topic describes the behavior of permissions in MapR Database version 6.1 and later, regardless of the MapR version you used to grant the permissions. To understand how permissions on arrays behave in earlier releases, see Operational Changes (MapR 6.1.0) - Permissions on Arrays in MapR Database JSON.

Granting Permissions on Array Elements

Suppose you have the following documents where `person` is:

- An array of nested documents in document `id001`
- A single nested document in document `id002`
- A scalar value in document `id003`

```
{
    "_id" : "id001",
    "person" : [
        {"name" : {"last" : "Smith", "first" : "John"}},
        {"name" : {"last" : "Subramanium", "first" : "Ananya"}}
    ]
}
{
    "_id" : "id002",
    "person" : {"name" : {"last" : "Doe", "first" : "Jane"}}
}
{
    "_id" : "id003",
    "person" : "Unknown"
}
```

If you grant a user read permission on the array `person[]`, that user can read every field in every nested document within the array in document `id001`. The permission also enables the user to read the `person` field in documents `id002` and `id003`.

If you receive an error when trying to grant permission on `person[]` because you previously granted permission on `person`, then you (or an administrator with the appropriate permissions) must first remove the existing permission on `person`. If you expect the schema of the `person` field to evolve to include non-array and array data, then you should grant the permission on `person[]` rather than `person`, to avoid having to remove the conflicting `person` permission.

You cannot grant permissions on individual elements in an array; for example: `person[1]`. Granting permission on an array enables access to the entire array.

Granting Permissions on Nested Document Fields in an Array

If you want to restrict read access to only specific fields in `person`, whether the field is an array of nested documents or a single nested document, perform the following steps:

1. Deny the user read permission on the array `person[]`.
2. Grant the user traverse permission on the array `person[]`.
3. Grant the user read permission on the specific fields.

For example, to grant the user read permission on only the first names in the nested documents, for the third step, grant read permission on `person[].name.first`. The permission enables the user to read the field in all nested documents in documents `id001` and `id002`.

last-updated: Apr 11, 2022

If permissions already exist on `person.name.first`, then all attempts to define permissions on `person[].name.first` fails. You (or an administrator with the appropriate permissions) must first remove the existing permission on `person.name.first`. Similar to the scenario described in the previous section, if you expect the schema of the `person` field to evolve to include individual nested documents as well as arrays of nested documents, then you should grant the permission on `person[].name.first` to avoid having to remove the conflicting permission.

If you already have permissions on `person[].name.first`, then attempting to define permissions on `person.name.first` fails. There is no need to add this permission.

Granting Permissions on JSON Tables

This page summarizes the default access-control expressions for the supported ways of setting read, traverse, and write permissions.

The default permissions for column families are determined when tables are created. The default permissions for fields are inherited from the column family where the fields are located.

Action	Method	Permissions	Default Access-Control Expressions
Set default permissions on new column families when creating a JSON table.	Java API	`-defaultreadperm -defaulttraversepe rm -defaultwriteperm`	`u:<ID of the process>`
	`maprcli table create`		`u:<user ID of table creator>`
	mapr dbshell		
	Control System		
Set default permissions on new column families when editing a JSON table.	`maprcli table edit`		Current ACEs
	Control System		
Set permissions on a column family when creating the column family.	`maprcli table cf create`	`-readperm -traverseperm -writeperm -indexperm`	ACEs for `-defaultreadperm`, `-defaulttraverseperm`, and `-defaultwriteperm`
	Control System		
Set permissions on a column family when editing the column family.	`maprcli table cf edit`		Current ACEs
	Control System		
Set permissions on individual fields.	`maprcli table cf colperm set`		Inherited from column family or parent field
	Control System		

Defining ACEs

Indicates how to build access control expressions (ACEs) using the Expression Builder.

To define access control expressions using the **Access Control Expression** builder in the MapR Control System:

1. Choose **All** or **Any** (from the drop-down menu) of the settings to match for access.

   Here:

All	AND (&) operation	Indicates that all of the conditions must be met for public or user, group, and/or role to access the volume.
Any	OR (|) operation	Indicates that any one of the conditions must be met for public or user, group, and/or role to access the volume.

last-updated: Apr 11, 2022

2. Click:

+	To add an expression.
( )	To add a subexpression.
x	To remove an expression or subexpression.

3. Select Public or User, Group, or Role from the drop-down menu and:
   a) Choose **Is** to grant or **Is not** to block access to the user, group, or role.
   b) Enter name of the user, group, or role.
4. Click **Save Changes** to create an Access Control Expression.

Setting Whole Volume ACEs Using the CLI

See Setting Whole Volume ACEs on page 1459.

Setting Table ACEs Using the CLI

See Enabling Table and Stream Authorizations with ACEs on page 1461.

Setting Stream ACEs Using the CLI

See Enabling Table and Stream Authorizations with ACEs on page 1461.

Creating Subnet Whitelists

Provides the procedure necessary to restrict access to cluster data.

To provide additional cluster security, you can limit cluster data access to a whitelist of trusted subnets. The `mfs.subnets.whitelist` parameter in `mfs.conf` accepts a comma-separated list of subnets in CIDR notation. When this parameter is set, the FileServer service only accepts requests from the specified subnets.

1. Edit `/opt/mapr/conf/mfs.conf` and modify the `mfs.subnets.whitelist` parameter.
2. Add a comma-separated list of subnets in CIDR notation.
3. Restart the FileServer.

Customizing Security in MapR

Describes the `.customSecure` file and how MapR 6.x handles custom security settings.

This topic contains the following subsections:

Note: Implementing custom security is not recommended unless your installation demands it. Using the custom security option means that MapR software does not ensure that your system is secure by default, and that you need to manually perform all security configuration.

In MapR 6.x, the configure.sh script detects that a cluster is in one of three security states:

Secure	The cluster is configured with the default MapR security settings.
Unsecure	No security settings are configured for the cluster.
Custom secure	The cluster has a mixture of MapR security settings and custom settings.

Understanding how `configure.sh` handles custom security settings is important when you upgrade a cluster, add services, add nodes, or change security settings.

What is Custom Security?

Any change to the default MapR configuration for authentication, authorization, or encryption represents a "custom security" change. Users who make such changes are encouraged to create a `.customSecure` file to ensure that `configure.sh` does not remove these changes. Custom security changes include any change to the keystore or truststore passwords or the number of keys in those files or the names of the keys.

Other examples of custom security changes include:

- Implementing Kerberos security
- Changing the Hive authorization model
- Changing the Oozie authorization model

Identifying the Current Security State of the Cluster

If the current security state of the cluster (secure, unsecure, or custom secure) is unknown, you can use one of these checks to identify which state the cluster is in:

- Check the security value in the `/opt/mapr/conf/mapr-clusters.conf` file. For example:

  ```
  <clustername1> secure=true <CLDB> <CLDB> … <CLDB>
  ```

  For more information, see mapr-clusters.conf on page 2200.

- Check for the presence of the `.customSecure` file:

  ```
  /opt/mapr/conf/.customSecure
  ```

  If the file is present, `configure.sh` treats the cluster as custom secure.

About the .customSecure File

If you customized the security settings for your cluster and you want to ensure that `configure.sh` does not change any of the settings, you can create a `.customSecure` file. Create the file in the following location on every node:

```
/opt/mapr/conf/.customSecure
```

The `.customSecure` file does not contain any information. The presence of the file tells `configure.sh` that the cluster has security settings that must not be changed by `configure.sh`.

Typically, you create the `.customSecure` file manually. However, in some cases, `configure.sh` creates or removes the `.customSecure` file for you. For example, if `configure.sh` detects that it is being run after an upgrade from a MapR 5.x secure cluster, it creates the `.customSecure` file automatically. If you use the `-forceSecurityDefaults` option and `-secure` or `-unsecure` with `configure.sh`, the script removes the `.customSecure` file because you are forcing the removal of custom security settings.

last-updated: Apr 11, 2022

Forcing a Change to the Security Configuration

If your MapR 6.x cluster has custom security settings (the `.customSecure` file is present), and you want to change to the default MapR secure or non-secure settings, you can use the `-forceSecurityDefaults` option of `configure.sh` to make the change. Note these considerations:

- Using the `-forceSecurityDefaults` option removes the `.customSecure` file. You must specify the `-secure` or `-unsecure` option with `-forceSecurityDefaults`. Otherwise, the command will have no effect.
- The `-forceSecurityDefaults` option might not remove all of your custom settings. Some manual editing might be necessary to return the cluster to a usable state.
- When forcing a custom-secured cluster to be MapR secure, you still need to include other `configure.sh` options that are required for security. And you need to perform any steps required to add security. For example, see Enabling Wire-level Security on page 1411.

Custom Security and the MapR Installer

Using the MapR Installer or MapR Installer Stanzas is not supported on clusters with custom security or customized configurations.

Adding a Node to a Cluster with Custom Security

Adding a node to a cluster with custom security is similar to adding a node to a cluster with MapR security, but there are some additional steps:

1. Add the node with default MapR security as described in Adding Nodes to a Cluster on page 807.
2. To support your custom security mode, copy any custom resources or settings as needed from existing nodes to the added node.
3. Create the `/opt/mapr/conf/.customSecure` file on the added node:

   ```
   /usr/bin/touch /opt/mapr/conf/.customSecure
   ```

Adding a Service to a Cluster with Custom Security

If you add a new service (ecosystem component) to a secure or custom-secure cluster, `configure.sh` configures the service for MapR security automatically. If the cluster is custom secure, you need to change the security settings for the service to be compatible with the current cluster settings and restart the service. Any subsequent use of `configure.sh -R` will leave the customization in place.

Managing Impersonation

Provides instructions for enabling and using MapR impersonation features.

Impersonation, also known as identity assertion, is one user accessing data and submitting jobs on behalf of another user. Impersonation in MapR allows centralized control of access to resources in the MapR File System and MapR Database.

Example: Access Control and Impersonation

As an example of impersonation, consider user Bob and a generic Service X:

1. Bob launches a client for the service and may or may not provide credentials.
2. Service X authenticates Bob and establishes a connection for him to use.

3. Bob issues a command to the service that will produce a query.
4. The service uses any user's servicewithimpersonation ticket to authenticate with the datastore - MapR File System/MapR Database.
5. The datastore authenticates the user with the impersonation ticket - the service can now proceed.
6. The service sends the datastore a query, as user Bob.
7. The datastore checks permissions for Bob on the assets that the query will access.
8. If Bob has permissions, the datastore returns the query results to the service, which relays the results to the client, and the query succeeds.
9. If Bob does not have permissions, the datastore sends an access error to the service, which relays the error to the client, and the query fails.

When you use impersonation in MapR:

- The datastore permissions are authoritative.
- The process has end-to-end security.
- Users can do nothing more and nothing less than what they are authorized to do.
- This control is independent of remote authentication and security mechanisms that control user access to application features.
- Any permissions set up within applications, or within the UNIX filesystem permissions on servers where MapR components reside, have no effect on user access to MapR resources.
- The mapr superuser is allowed to impersonate any MapR user in any group, connecting from any host. Other users with impersonation capability can impersonate any MapR user in any group, except the mapr superuser and the root user.

Using Impersonation without Security

Although it is possible to enable impersonation in a non-secure MapR installation, MapR strongly recommends against doing this. The implementation rules are different, and setting up the MapR environment with impersonation operating under those rules makes it very difficult to enable security at a later date. Disabling security in a secure MapR installation is easy, if the need arises.

If you choose to implement impersonation in a non-secure MapR cluster, see Configuring Impersonation when Cluster Security is not Enabled.

Using Impersonation with Security

In general, this documentation assumes that security is enabled in your MapR installation. See Enabling Wire-level Security on page 1411 and Enabling Encryption of Data at Rest on page 1413.

You can use the `maprlogin` utility to generate a **servicewithimpersonation** ticket that can be used to access a secure cluster impersonating another user. That is, the **servicewithimpersonation** ticket provides the user the ability to impersonate other users (except the mapr user) in addition to the ability to access a secure cluster. The servicewithimpersonation ticket generated with the list of `impersonatedgids` and `impersonateduids` cannot be used to impersonate user root or user mapr. If the user is other than root or mapr, CLDB resolves the username to UID locally and then checks if the resolved UID can be impersonated (i.e., if it is a part of the ticket's `impersonateduids`) or at least one of the GIDs of the resolved UID can be impersonated (i.e., if at least one of the GIDs should be part of the ticket's `impersonatedgids`). The servicewithimpersonation ticket can only be generated by a user with full control on a cluster's Access Control List (ACL).

If you are setting up user impersonation in a secure cluster, you need to generation an impersonation ticket. See the *Generating and Printing Service with Impersonation Ticket* section in the maprlogin Command Examples on page 2134 topic and Generating a Service with Impersonation Ticket on page 1428 for information on generating an impersonation ticket.

After generating the ticket:

1. Ensure that mapruser1 has read permissions on the ticket.
2. If you moved the ticketfile to a different location, set the `$MAPR_TICKETFILE_LOCATION` environment variable.

How Impersonation Works
Introduces impersonation functionality, limitations, and core requirements.

When a user attempts to impersonate another user to the MapR File System or MapR Database systems and the configuration parameters for resolving the UID and GIDs on the server (see Resolving Username with UID and GIDs During Impersonation) are disabled:

1. The MapR client looks for that user name in the local operating system registry.
2. If the user name is:
   - Found, MapR sends the user's UID and GID to the server for impersonation.
   - Not found in the local operating system registry, the user action is not processed.

When a user attempts to impersonate another user to the MapR File System or MapR Database systems and if the configuration parameters for resolving the UID and GIDs on the server (see Resolving Username with UID and GIDs During Impersonation) are enabled:

1. The MapR client asks CLDB to look for that user name and resolve the UID and GIDs for that user on the server.
2. If the user name is:
   - Found on the server, the server allows the user to proceed with the impersonation.
   - Not found, the user action is not processed.

Note: If the configuration property for resolving the username is set on the client and the configuration property for resolving the username is not set on CLDB, the operation will fail with an error.

Limitations on Impersonation
Service with impersonation tickets cannot be used to impersonate user mapr or user root. A scoped service with impersonation ticket cannot contain the UID of user root or user mapr (in the impersonated UIDs) and the GID of user root or user mapr (in the impersonated GIDs). User mapr can impersonate any user, including user root.

Core Requirements for Impersonation

The mapr superuser is allowed to access to the MapR File System and MapR Database systems. The following conditions must be met in order for the mapr superuser to be able to impersonate another MapR user:

1. The `hadoop.proxyuser.mapr.groups` and `hadoop.proxyuser.mapr.hosts` parameters must be set correctly in the `core-site.xml` file.

   See Enabling Impersonation for the mapr Superuser.

   These settings are not always required. The hadoop proxy user functionality is only applicable to ecosystem components included in the MapR distribution for Apache Hadoop. If the MapR client accesses an ecosystem component, such as HiveServer2, these settings may be required. These settings are never needed when the MapR client accesses MapR File System or MapR Database directly. Enabling impersonation here ensures that the correct settings are in place if they are needed.

2. The name of the MapR user that you want the mapr superuser to be able to impersonate must appear in the local operating system registry where the MapR client is running if server-side resolution of UID and GIDs is not enabled.

3. The UID and GUID of the user name under which the MapR client is running must match exactly the UID and GUID for that user name on the server.

Note: The user mapr can impersonate any user, including user root.

For all other users with access to the MapR File System and MapR Database systems, the following conditions must be met for the user to impersonate another user.

1. A valid servicewithimpersonation ticket must be present for the user who intends to impersonate on the system.

2. The name of the MapR user to impersonate must appear in the local operating system registry where the MapR client is running if server-side resolution of UID and GIDs is not enabled.

3. The UID and GUID of the user name under which the MapR client is running must match exactly the UID and GUID for that user name on the server.

Note: If a user is not authorized to impersonate, then the operations will proceed as the user, not the target user, and some operations will succeed, and some will not even if the user has all the permissions for these operations. Also, if a user with full access and impersonating capability tries to impersonate another user, the operations will succeed only if the target user has permissions on the directory.

Component Requirements for Impersonation

Some MapR ecosystem components have additional requirements to enable impersonation.

The following components must have settings that support impersonation in the configuration files indicated, on each node where the component resides:

- **Drill**: Edit the `drill-env.sh` file. See Configuring User Impersonation in the Apache Drill documentation.
- **Flume**: Edit the `flume.conf` file. See Configure User Impersonation for Flume.
- **HBase**: Edit the `hbasesite.xml` file. See Impersonation via the HBase REST Gateway.
- **HiveServer2**: Edit the `hive-site.xml` file. See Hive User Impersonation.
- **Hue**: Edit the `hue.ini` file. See Configure Hue with Impala.
- **Impala**: Does not support impersonation.
- **Oozie**: Edit the `oozie-site.xml` file. See User Impersonation for Oozie.

- **Spark**: No special settings are required for Spark in MapReduce 2 (YARN) mode, since Spark automatically inherits the correct behavior from YARN. When running standalone, Spark cannot perform impersonation and should not be used if security is important.

Application Development Requirements

You can set up impersonation in an application programmatically.

- **C/C++**: Use `hb_connection_create_as_user()`. See the “C API for impersonation” section of Creating C Apps - Binary Tables on page 2453 and Impersonation Example on page 2456..
- **Java**: Use `UserGroupInformation.doAs()`. See Class UserGroupInformation in the Hadoop documentation.

Enabling Impersonation for the MapR Superuser

Provides a procedure necessary to implement superuser impersonation.

To enable impersonation in your MapR installation:

1. Open the following file in a text editor:

   ```
   /opt/mapr/hadoop/hadoop-<version>/etc/hadoop/core-site.xml
   ```

2. Add the following `hadoop.proxyuser` properties:

   ```
   <property>
           <name>hadoop.proxyuser.mapr.hosts</name>
           <value>*</value>
   </property><property>
           <name>hadoop.proxyuser.mapr.groups</name>
           <value>*</value>
   </property>
   ```

   The `hosts` setting (`*`) allows the `mapr` superuser to connect from any host to impersonate a user.

   The `groups` setting (`*`) allows the `mapr` superuser to impersonate any user in any group.

   Note: Do not use anything other than a single asterisk here. Other parts of MapR ignore the values here and treat them as if each is set to a single asterisk.

3. Close the file, saving any changes that you made.

Enabling Impersonation for any User

Provides a procedure necessary to implement impersonation for any MapR user.

To enable impersonation for any MapR user:

1. Log in to the system as root, mapr user, or any user with full control.

last-updated: Apr 11, 2022

2. Generate a servicewithimpersonation ticket for the MapR user.

   For example:

   ```
   $ maprlogin generateticket -type servicewithimpersonation -user
   mapruser1 -out /var/tmp/sample_ticket
   ```

   Warning: The `mapr` user ticket can be used to impersonate any user, including user root.

   You can generate a scoped servicewithimpersonation ticket for the user. The scoped impersonation tickets allows the user using the ticket to impersonate only the UIDs and or GIDs specified in the ticket. For example:

   ```
   $ maprlogin generateticket -type servicewithimpersonation -user
   mapruser1 -impersonateduids 550 -impersonatedgids 500 -out /var/tmp/
   sample_ticket
   ```

   Note: When generating a scoped impersonation ticket, the impersonated UIDs cannot contain the UID of user `root` or user `mapr`, and the impersonated GIDs cannot contain the GID of user `root` or user `mapr`.

   For more information, see `maprlogin` on page 2130.

3. Move the ticket to a secure location and share the ticket with the user (for whom this ticket was generated).

4. (Optional) Copy the file to a permanent directory.

Configuring Impersonation without Cluster Security
Describes how to use impersonation on a non-secure cluster.

To configure impersonation without enabling cluster security, perform the following steps on the client:

1. Enable impersonation for each ecosystem component you will use that supports impersonation. See *Component Requirements for Impersonation* in How Impersonation Works.

2. Enable impersonation for the data-fabric core components. See Enabling Impersonation for the mapr Superuser.

3. On each client system from which you wish to be able to use impersonation:
   a) Set a `MAPR_IMPERSONATION_ENABLED` environment variable with the value `true`. This value must be set in the environment of any process you start that does impersonation.

   b) Create a file in `/opt/mapr/conf/proxy/` that has the name of the data-fabric superuser or any other user. For the data-fabric superuser, the default file name would be `mapr`. For all other users, you can use their username for the proxy file.

   If the data-fabric superuser has a different name in your cluster, use that name for the proxy file. To verify the data-fabric superuser name, check the `mapr.daemon.user=` line in the `/opt/mapr/conf/daemon.conf` file on a data-fabric cluster server.

Resolving Username with UID and GIDs During Impersonation
Lists parameters for configuring impersonation.

To resolve username with UID and GIDs on the server (and not the local operating system registry) during impersonation, you can set the following configuration parameters on the client and CLDB:

last-updated: Apr 11, 2022

Parameter	Description
`fs.mapr.server.resolve.user`	Must be set in `core-site.xml` file on the client machine. Value can be one of the following: • `true` - enable • `false` - disable By default, this is disabled. If enabled, client will request CLDB to resolve user with UID/GIDs. For example, to enable this property, your entry in `core-site.xml` file should be as shown below: `<configuration> <property> <name>fs.mapr.server.resolve.user</ name> <value>true</value> </property> </configuration>`
`cldb.security.resolve.user`	Must be set using the `config` command. Value can be one of the following: • `0` - disable • `1` - enable By default, this is disabled. If enabled, CLDB will resolve the user with UID/GIDs for all incoming client requests. For example, to enable this property, run the following command: `maprcli config save -values {cldb.security.resolve.user:1}`

Note: Both the configuration parameters must be set to enable support for UID/GID resolution on the server. If the configuration parameter is set on the client to resolve on the server, and if the configuration parameter is not set on CLDB, the operation will fail with an error.

Managing Secure Clusters

Provides procedures that will enable you to use MapR clusters securely.

Administrative topics such as configuring cross-cluster security, managing mirror volumes in secure clusters, running commands on remote secure clusters are discussed. In addition, access scenarios for secure and non-secure MapR clusters and HDFS clusters are described.

Setting Up Cross-Cluster Security

Provides an overview of the `configure-crosscluster.sh` utility that is used to set up security between two clusters.

When all local and remote CLDB nodes are reachable from the local node, you can run the `configure-crosscluster.sh` on page 2065 utility on any CLDB node to automatically set up a trust relationship between clusters.

For two or more MapRMapR Data Platform clusters to communicate with one another, a secure trust relationship must exist between the clusters. A secure trust relationship between clusters is required for

running commands remotely, creating remote replicas and mirror copies of volumes, and accessing data using NFS on the other cluster. The following sections describe the quick way to configure both the clusters for mirroring, replication, and remote access, and the advanced manual way to configure the clusters for mirroring, replication, remote access, and/or NFS server access.

Quick Configuration

You can run the `configure-crosscluster.sh` on page 2065 utility on any CLDB node in a cluster to automatically set up a trust relationship between the cluster and another cluster. To automatically configure two clusters for remote access, mirroring, and replication in both directions:

1. Log in to the CLDB node on a cluster.
2. Run the `configure-crosscluster.sh` on page 2065 utility with the `all` parameter.
   For example:

   ```
   # /opt/mapr/server/configure-crosscluster.sh create all -remoteip
   <remote_node_IP>
   ```

   When the utility runs, it performs the following actions on all the clusters:

   a. Updates the `/opt/mapr/conf/mapr-clusters.conf` file to include the first entry from the `/opt/mapr/conf/mapr-clusters.conf` file on the other cluster.

   b. Imports the certificate of the other cluster in the `/opt/mapr/conf/ssl_truststore` file, and copies the updated `/opt/mapr/conf/ssl_truststore` file to all the other nodes on the cluster.

   c. Generates a cross-cluster ticket for the other cluster, copies the ticket to the CLDB node on the other cluster, merges the ticket with the `/opt/mapr/conf/maprserverticket` file on the node in the other cluster, and copies the updated `/opt/mapr/conf/maprserverticket` file to all other CLDB nodes on the other cluster.

   For more information on the arguments, syntax, and options, see the `configure-crosscluster.sh` on page 2065 utility.
3. Verify access to the remote cluster by:
   - Running remote commands on a node in either cluster.
   - Creating mirror volumes on any node in the destination cluster.
   - Setting up table and stream replication on tables and streams in the source cluster.

   To configure access over NFS, see Configuring Secure Clusters for Cross-Cluster NFS Access on page 1490.

Advanced Configuration

Using the `configure-crosscluster.sh` on page 2065 utility with the default configuration works only when all local and remote CLDB nodes are reachable from the local node. It does not work, for example, if you set up multi-homed clusters as documented in the MAPR_SUBNETS section in Designating NICs for MapR on page 844, because the `configure-crosscluster.sh` on page 2065 utility cannot traverse between local and remote IPs (for example, from the external IP 23.21.203.95 to internal IP 10.10.100.100). In such environments, run the `configure-crosscluster.sh` on page 2065 utility with the `-remotehosts` parameter.

You can configure the clusters manually for unidirectional or bidirectional remote access, mirroring, or replication only. The following sections describe the manual steps for:

Configuring Secure Clusters for Running Commands Remotely
Describes how to configure secure clusters to access them all from a single cluster and run commands remotely on them.

You can configure a number of secure clusters to access them all from one cluster. You need not log into each secure cluster separately and run `maprcli` commands locally on them.

For example, suppose you need to manage two secure clusters, clusterA and clusterB. One method is to log into each cluster separately and run commands locally on each. However, it is possible to log into clusterA only and manage both clusters from clusterA, running commands locally for clusterA and remotely for clusterB. When you type the `maprcli` commands, you must use the `-cluster` parameter in those commands to specify the cluster on which you want the commands to run.

You can configure the secure clusters for remote access manually (as described in the following section) or automatically by running the `configure-crosscluster.sh` utility. If you run the `configure-crosscluster.sh` utility, the utility configures the clusters for running commands remotely in both directions. See `configure-crosscluster.sh` on page 2065 for more information.

Prerequisite

Ensure that you have the relevant ports open for secure cluster communication.

Setting Up Secure Clusters Manually for Cross-Cluster Access

To manually configure two secure clusters for remote access:

1. Log in to the secure cluster from which you want to run commands.
   In the rest of this procedure, this cluster is referred to as clusterA and the remote cluster is referred to as clusterB.

2. Configure clusterA for communicating with the other clusters by editing `mapr-clusters.conf` file on each node clusterA to specify the hostname or IP address of the CLDB nodes on the other clusters.

   For example, suppose:

   - clusterA's `/opt/mapr/conf/mapr-clusters.conf` file contains the following:

     ```
     clusterA.cluster.com secure=true perfnode50.lab:7222
     ```

   - clusterB's `/opt/mapr/conf/mapr-clusters.conf` file contains the following:

     ```
     clusterB.cluster.com secure=true perfnode100.lab:7222
     ```

   Perform the following steps to configure the nodes on the clusters:

   a) On any node in clusterA, append the first entry from clusterB's `mapr-clusters.conf` file, entry which is prefixed with the cluster name, to the end of clusterA's `mapr-clusters.conf` file.
      Note that clusterA's entry must be the first line of the `mapr-clusters.conf` file:

      ```
      clusterA.cluster.com secure=true perfnode50.lab:7222
      clusterB.cluster.com secure=true perfnode100.lab:7222
      ```

      The clusterA's `mapr-clusters.conf` file now contains two entries.

   b) Copy the updated `/opt/mapr/conf/mapr-clusters.conf` file to all the other nodes in clusterA.

c) On any node in clusterB, append the first entry from clusterA's `mapr-clusters.conf` file, entry which is prefixed with the cluster name, to the end of the remote cluster's `mapr-clusters.conf` file.

Note that clusterB's entry must be the first line of mapr-clusters.conf file:

```
clusterB.cluster.com secure=true perfnode100.lab:7222
clusterA.cluster.com secure=true perfnode50.lab:7222
```

The clusterB's `mapr-clusters.conf` file now contains two entries.

d) Copy the updated `/opt/mapr/conf/mapr-clusters.conf` file to all the nodes in clusterB.

See mapr-clusters.conf on page 2200.

3. Perform the following steps on clusterA to ensure that the `ssl_truststore` file has signers for all the clusters:

a) Copy the `ssl_truststore` from the `/opt/mapr/conf` directory of clusterB into a temporary directory on clusterA.

For example:

```
# scp mapr@<remote-ip>:/opt/mapr/conf/ssl_truststore /tmp/
clusterB_ssl_truststore
```

b) Merge the `ssl_truststore` of clusterB with the `ssl_truststore` of clusterA using the `/opt/mapr/server/manageSSLKeys.sh` utility.

For example, if you copied the `ssl_truststore` file of clusterB as `/tmp/clusterB_ssl_truststore`, run the following command to merge the files:

```
/opt/mapr/server/manageSSLKeys.sh merge /tmp/
clusterB_ssl_truststore /opt/mapr/conf/ssl_truststore
```

c) Copy the merged `ssl_truststore` file to every node on clusterA.

4. Perform the following steps on clusterB *only* if you want to set up access to clusterA from clusterB:

a) Copy the `ssl_truststore` from the `/opt/mapr/conf` directory of clusterA into a temporary directory on clusterB.

For example:

```
# scp mapr@<remote-ip>:/opt/mapr/conf/ssl_truststore /tmp/
clusterA_ssl_truststore
```

b) Merge the `ssl_truststore` of clusterB with the `ssl_truststore` of clusterA using the `/opt/mapr/server/manageSSLKeys.sh` utility.

For example, if you copied the `ssl_truststore` file of clusterA as `/tmp/clusterA_ssl_truststore`, run the following command to merge the files:

```
/opt/mapr/server/manageSSLKeys.sh merge /tmp/
clusterA_ssl_truststore /opt/mapr/conf/ssl_truststore
```

c) Copy the merged `ssl_truststore` file to every node on clusterB.

5. For crossclusters to work using the Control System, place the mapruserticket of the remote cluster into the local cluster.
   a) Generate a mapruserticket for the remote cluster as `mapr` user:

```
maprlogin password -cluster demo
[Password for user 'mapr' at cluster 'demo': ]
MapR credentials of user 'mapr' for cluster 'demo' are written to
'/tmp/maprticket_5000'
```

   b) Merge the generated maprticket:

```
cat /tmp/maprticket_5000 >>/opt/mapr/conf/
maprusertickets
```

6. Verify access by running remote commands on clusterA.
   See Verifying Access to run Remote Commands on page 1486.

Verifying Access to run Remote Commands

1. Log in to any node on clusterA and run the `maprlogin` on page 2130 utility from clusterA to obtain user ticket for accessing the remote cluster.
   For example, to obtain tickets for managing the remote cluster from clusterA, run the following command::

```
# /opt/mapr/bin/maprlogin password -cluster clusterB.cluster.com
```

2. Verify access by running remote commands on clusterA.
   For example, the following command, executed from a node in clusterA, lists the volumes on clusterB:

```
# /opt/mapr/bin/maprcli volume list -cluster clusterB.cluster.com
```

Configuring Secure Clusters for Cross-Cluster Mirroring and Replication
Describes configuring clusters for cross-cluster operations such as mirroring and replication.

Cross-cluster tickets are required on secure clusters that need to pull data from another secure cluster and on secure clusters that need to push data to another secure cluster. For example:

- Volume mirroring is a pull operation. The volume data is pulled by the destination cluster from the source cluster. Since the destination cluster performs the operation, the destination cluster receives a ticket that is generated on the source cluster.
- Table and streams replication is a push operation. The table or stream data is pushed from the source cluster to the destination cluster. Since the source cluster performs the operation, the source cluster receives a ticket that is generated on the destination cluster.

You can configure secure clusters for cross-cluster mirroring and replication manually (as described in Setting up Secure Clusters Manually for Cross-Cluster Mirroring on page 1487 and Setting up Secure Clusters Manually for Cross-Cluster Replication on page 1489). You can configure secure clusters automatically, by running the `configure-crosscluster.sh` utility. This utility configures the clusters for both mirroring and replication in both directions. For more information, see `configure-crosscluster.sh` on page 2065.

Setting up Secure Clusters Manually for Cross-Cluster Mirroring

To set up secure clusters for cross-cluster mirroring:

1. Verify that the user for whom you are configuring access, exists in the registry on both the clusters and has the following permissions:
   - Permissions to create volumes on the source cluster.
   - Permissions to mirror volumes on the destination cluster.

   You can set up access for the *mapr* user, who already has permissions to create volumes and mirror volumes.
2. Configure clusterA for communicating with the other clusters by editing `mapr-clusters.conf` file on each node clusterA to specify the hostname or IP address of the CLDB nodes on the other clusters.

   For example, suppose:
   - clusterA's `/opt/mapr/conf/mapr-clusters.conf` file contains the following:

     ```
     clusterA.cluster.com secure=true perfnode50.lab:7222
     ```
   - clusterB's `/opt/mapr/conf/mapr-clusters.conf` file contains the following:

     ```
     clusterB.cluster.com secure=true perfnode100.lab:7222
     ```

   Perform the following steps to configure the nodes on the clusters:

   a) On any node in clusterA, append the first entry from clusterB's `mapr-clusters.conf` file, entry which is prefixed with the cluster name, to the end of clusterA's `mapr-clusters.conf` file.

   Note that clusterA's entry must be the first line of the `mapr-clusters.conf` file:

   ```
   clusterA.cluster.com secure=true perfnode50.lab:7222
   clusterB.cluster.com secure=true perfnode100.lab:7222
   ```

   The clusterA's `mapr-clusters.conf` file now contains two entries.

   b) Copy the updated `/opt/mapr/conf/mapr-clusters.conf` file to all the other nodes in clusterA.

   c) On any node in clusterB, append the first entry from clusterA's `mapr-clusters.conf` file, entry which is prefixed with the cluster name, to the end of the remote cluster's `mapr-clusters.conf` file.

   Note that clusterB's entry must be the first line of mapr-clusters.conf file:

   ```
   clusterB.cluster.com secure=true perfnode100.lab:7222
   clusterA.cluster.com secure=true perfnode50.lab:7222
   ```

   The clusterB's `mapr-clusters.conf` file now contains two entries.

   d) Copy the updated `/opt/mapr/conf/mapr-clusters.conf` file to all the nodes in clusterB.

   See `mapr-clusters.conf` on page 2200.
3. Log in to any node on the source cluster (ClusterA) and perform the following steps:

last-updated: Apr 11, 2022

a) Generate a cross-cluster ticket for the destination cluster for this user.

For example, to generate a cross-cluster for destination cluster, run the following command on the source cluster:

```
source$ /opt/mapr/bin/maprlogin generateticket -type
crosscluster -out /tmp/crossclusterticket -user destinationclusteruser
```

b) Copy the cross-cluster ticket file to any node on the destination cluster (clusterB).

For example:

```
source$ scp /tmp/crossclusterticket mapr@<dest-ip>:/tmp/
sourceClusterTicketFile
```

4. Log in to the node on the destination cluster (clusterB) where the cross-cluster ticket was copied, and perform the following steps:

a) Merge the cross-cluster ticket file with the `/opt/mapr/conf/maprserverticket` file on the node.

For example, to merge, run the following command:

```
dest$ cat /tmp/sourceClusterTicketFile >> /opt/mapr/conf/
maprserverticket
```

b) Copy the `/opt/mapr/conf/maprserverticket` file to the CLDB nodes on the destination cluster.

5. Perform the following steps on clusterB to ensure that the `ssl_truststore` file has signers for all the clusters:

a) Copy the `ssl_truststore` from the `/opt/mapr/conf` directory of clusterA into a temporary directory on clusterB.

For example:

```
# scp mapr@<remote-ip>:/opt/mapr/conf/ssl_truststore /tmp/
clusterA_ssl_truststore
```

b) Merge the `ssl_truststore` of clusterA with the `ssl_truststore` of clusterB using the `/opt/mapr/server/manageSSLKeys.sh` utility.

For example, if you copied the `ssl_truststore` file of clusterA as `/tmp/clusterA_ssl_truststore`, run the following command to merge the files:

```
/opt/mapr/server/manageSSLKeys.sh merge /tmp/
clusterA_ssl_truststore /opt/mapr/conf/ssl_truststore
```

c) Copy the merged `ssl_truststore` file to every node on clusterB.

6.

7. Perform the steps to verify configuration for mirroring.

You can now create mirror volumes on the destination cluster and set up a schedule to pull data from the volumes on the source cluster. However, you cannot create volumes on the source cluster that pull data from volumes in the destination cluster, because the setup described above is unidirectional. To configure

the clusters for bidirectional mirroring, repeat the steps above, by switching the source and destination clusters.

For example, suppose there are two clusters, clusterA and clusterB, and you performed the steps above for clusterA as the source cluster and clusterB as the destination cluster. After you complete the steps above, your destination cluster, clusterB can pull data from volumes on clusterA. For clusterA to mirror data on clusterB, perform the steps above with clusterB as the source cluster and clusterA as the destination cluster.

Setting up Secure Clusters Manually for Cross-Cluster Replication

To set up secure clusters for cross-cluster replication:

1. Verify that the user, for whom you are configuring access, exists in the registry on the destination cluster.
2. Log in to any node on the destination cluster and perform the following steps:
   a) Generate a cross-cluster ticket for the source cluster.
      For example, to generate a cross-cluster for the source cluster, run the following command on the destination cluster:
      ```
      dest$ /opt/mapr/bin/maprlogin generateticket -type
      crosscluster -out /tmp/crossclusterticket -user destinationclusteruser
      ```
   b) Copy the cross-cluster ticket file to any node on the source cluster.
      For example:
      ```
      dest$ scp /tmp/crossclusterticket mapr@<source-ip>:/tmp/
      sourceClusterTicketFile
      ```
3. Log in to the node in the source cluster where the cross-cluster ticket was copied, and perform the following steps:
   a) Merge the cross-cluster ticket file with the `/opt/mapr/conf/maprserverticket` file on the node.
      For example, to merge, run the following command:
      ```
      cat /tmp/destinationClusterTicketFile >> /opt/mapr/conf/
      maprserverticket
      ```
   b) Copy the `/opt/mapr/conf/maprserverticket` file to all the nodes on the source cluster.
4. Configure the Gateway for table and streams replication.
   See Configuring Gateways for Table and Stream Replication on page 1152 for more information.
5. Perform the steps to verify configuration for replication.

You can now set up volumes on the source cluster to push data to replicas on the destination cluster. However, you cannot create replicas on the source cluster that get data from volumes in the destination cluster because the setup described above is unidirectional. To configure the clusters for bidirectional replication, repeat the steps above by switching the source and destination clusters.

For example, suppose there are two clusters, clusterA and clusterB, and you performed the steps above for clusterA as the source cluster and clusterB as the destination cluster. After you complete the steps above, your source cluster, clusterA can push data to replicas on clusterB. For clusterB to replicate data on clusterA, perform the steps above with clusterB as the source cluster, and clusterA as the destination cluster.

Verifying Cross-Cluster Configuration for Mirroring and Replication

You can verify the cross-cluster configuration for:

1. Mirroring by logging in to a node on the destination cluster as the user for whom access was configured, and creating a mirror volume on the destination cluster for a volume on the source cluster.

   You can create mirror volumes using the Control System and/or the CLI.

2. Replication by logging in to a node on the source cluster as the user for whom access was configured and creating a replica in the destination cluster for a volume, table, and stream on the source cluster.

   You can create replicas using the Control System and the CLI. To set up replication on secure clusters for:

   - Tables, refer to the documentation for the Control System and/or the CLI.
   - Streams, refer to the documentation for the Control System and/or the CLI.

Configuring Secure Clusters for Cross-Cluster NFS Access

Describes how to manually set up cross-cluster NFS access.

MapR-NFS offers many usability and interoperability advantages to the customer, and makes big data radically easier and less expensive to use. In a secure environment, however, you must configure NFS carefully because the NFS protocol is inherently insecure. Running the NFS server on any cluster node might expose the filesystem to be world readable and writeable to any machine that knows the IP address of the cluster node running the NFS server and has access to the network, regardless of the permissions, passwords and other security mechanisms. At the minimum, you should configure iptables firewall rules for all the cluster nodes where the NFS server is running, to restrict incoming NFS traffic to authorized client IP addresses.

Configuring cross-cluster NFS access might expose the entire filesystem of the other cluster to be world readable and writeable as well. Therefore, automated configuration for cross-cluster NFS access is not available with the `configure-crosscluster.sh` on page 2065 utility. You should manually configure cross-cluster NFS access only if you are fully aware of the security risks, and taken appropriate steps to mitigate the risks by securing both your NFS gateway, and incoming client traffic.

This section describes the manual configuration process on a secure cluster for accessing another secure cluster using NFS. There are two methods by which an NFS client can access file systems from multiple clusters:

1. Run the NFS server on one cluster.

   For this method, configure cross-cluster NFS security for the NFS gateway on one cluster, so that the NFS client can mount the mapr filesystem once from the NFS gateway, and then access the file systems for both clusters.

2. Run the NFS server on both clusters.

   For this method, cross-cluster NFS configuration is not needed. The NFS client can mount the MapR filesystem individually for each cluster. This method requires that the NFS gateway to be run on each cluster, and the client performs one NFS mount for each NFS filesystem to be accessed.

The following procedure describes how to setup NFS for the first method:

1. Log in to any node on the secure cluster where the NFS server is running.

   In the rest of this procedure, this cluster is referred to as clusterA.cluster.com and the remote cluster is referred to as clusterB.cluster.com.

2. Set up the `/opt/mapr/conf/maprserverticket` file on clusterA.cluster.com to include the server ticket from clusterB.cluster.com. To set up:

   a) Copy the `/opt/mapr/conf/maprserverticket` file from any node on clusterB.cluster.com to any directory on the node you are logged into on clusterA.cluster.com.

last-updated: Apr 11, 2022

b) Append maprserverticket entry in the `maprserverticket` file from clusterB.cluster.com to the `/opt/mapr/conf/maprserverticket` file on the node you are logged into on clusterA.cluster.com.

Note: If you configured cross-cluster security either automatically using the `configure-crosscluster.sh` on page 2065 utility or manually before, there can be multiple entries in the `maprserverticket` file; copy the first entry with the alias matching the remote cluster name.

For example, to add maprserverticket of clusterB.cluster.com into the `/opt/mapr/conf/maprserverticket` file of clusterA.cluster.com, run the following command:

```
cat /tmp/remoteclusterticketfile | grep B.cluster.com |
head --lines=+1 >> /opt/mapr/conf/maprserverticket
```

c) Copy the `/opt/mapr/conf/maprserverticket` file (on the node you are logged into in clusterA.cluster.com) to all the other nodes running NFS server on clusterA.cluster.com.

3. Verify data access on both clusters using NFS.

   Users with access to the NFS servers must be able to access data in both clusters by providing the correct path. For example, users with NFS server access can verify access by running commands similar to the following:

```
# ls /mapr
clusterA.cluster.com  clusterB.cluster.com
# ls /mapr/clusterB.cluster.com/
apps  file  CLUSTERB hbase  opt  tmp  user  var
```

Accessing External HDFS Clusters

Outlines how to use protocols to connect to other clusters.

MapR clusters can access an external HDFS cluster with the `webhdfs://` protocols.

Prerequisites

Before you begin, verify the following:

- The MapR node accessing the HDFS cluster must have the `mapr-core` or `mapr-client` package installed.
- The HDFS cluster is installed and configured according to the vendor's specifications.
- To use the `hdfs://` protocol, edit the `fs.hdfs.impl` property in the `$HADOOP_HOME/conf/core-site.xml` file to include the value `org.apache.hadoop.hdfs.DistributedFileSystem`.

The following cases provide information about MapR and accessing HDFS clusters.

Configuring Access Between Non-Secure MapR and HDFS Clusters

If the MapR and HDFS clusters are both non-secure, verify that the `fs.hdfs.impl` property in the `$HADOOP_HOME/conf/core-site.xml` file has the following value:

```
org.apache.hadoop.hdfs.DistributedFileSystem
```

No additional configuration is required.

last-updated: Apr 11, 2022

Verifying access to HDFS cluster

Use the following commands to verify access to the remote HDFS cluster from the MapR cluster.

CDH3 Only

```
hadoop fs -ls hdfs://<namenode_host:port>/
```

Other HDFS Versions

```
hadoop fs -ls webhdfs://<namenode_host_running_webhdfs_service>/
```

Using Java Applications with Secure Clusters

Describes ramifications associated with using Java applications in a MapR secure environment.

A secure computing environment places additional requirements on the Java Virtual Machine (JVM) properties of Java clients. The JVMs launched by MapR with scripts, such as those used by the `maprcli`, `hadoop`, or `hbase` commands, have those properties automatically set by the MapR software. The MapR software attempts to set useful values for these properties.

When a JVM is used or launched directly, such as when you write a stand-alone Java program, any Java code that sets values for the following properties may cause issues on your cluster.

Property	Default Value	Description
java.security.auth.login.config	`/opt/mapr/conf/mapr.login.conf`	Path to the file that specifies JAAS configurations used by MapR.
javax.net.ssl.trustStore	`/opt/mapr/conf/ssl_truststore`	Controls the truststore used by MapR clients for HTTPS connections.
http.auth.preference	`basic`	The default setting disables JVM's default handling of SPNEGO, enabling MapR's Hadoop code to handle SPNEGO authentication.
zookeeper.saslprovider	`com.mapr.security.maprsasl.MaprSaslProvider`	Enables ZooKeeper security.
hadoop.login	`hadoop_default`	Controls the JAAS configuration used by MapR security.

Administering the MapR Data Access Gateway

The MapR Data Access Gateway is a service that acts as a proxy and gateway for translating requests between lightweight client applications and the MapR cluster. This section describes considerations when upgrading the service, how to modify configuration settings, and how to administer and manage the service.

Installing the Data Access Gateway Service

The MapR Data Access Gateway is installed when you install the MapR Database using the MapR Installer. To manually install the service, see Installing MapR Data Access Gateway on page 203. For conceptual information, see Understanding the MapR Data Access Gateway on page 750.

Shutting Down and Upgrading the Data Access Gateway Service

When the Data Access Gateway receives a shutdown request, it stops accepting new requests and returns an error to the client. Any in-progress requests are allowed to complete before shutting down the service. This allows you to perform rolling upgrades.

Modifying Configuration Settings for the Data Access Gateway Service

Logging Properties

The MapR Data Access Gateway uses standard Log4J configuration to control its logging. The log4j properties are in the `/opt/mapr/data-access-gateway/conf/log4j2.xml` file on nodes where you have installed the service. After modifying any properties on a node, restart the service. For details, see Administering the Data Access Gateway Service on page 1495.

Log data is stored in the `/opt/mapr/data-access-gateway/logs/data-access-gateway.log` file.

Application Properties

To configure MapR Data Access Gateway properties, modify `/opt/mapr/data-access-gateway/conf/properties.cfg` on nodes where you have installed the service.

The following table lists the properties you can configure:

auth.token.expiration	*Default*: 1800 seconds *Description*: Expiration time (in seconds) for the authentication token.
grpc.service.max-message-size	*Default*: 32MB *Description*: The maximum message size that the gRPC service accepts. The default is set to 32MB, as this is the default maximum document size for MapR Database JSON tables. This property is available in Data Access Gateway 2.0.202104302209 and later or 3.0.0.0.202104302219 and later.
grpc.service.ojai.query.result-limit	*Default*: 5000 *Description*: Limit on the number of documents returned in retrieval requests using the Node.js and Python OJAI clients.
grpc.service.port	*Default*: 5678 *Description*: Port number gRPC clients use to connect to the Data Access Gateway. The Node.js and Python OJAI clients are gRPC clients.
grpc.service.ssl.enabled	*Default*: cluster *Description*: Controls whether TLS is enabled for the gRPC Service. Values: `cluster`|`true`|`false` If set to `cluster`: • TLS is enabled if your MapR cluster is secure. • TLS is disabled if your MapR cluster is not secure. When TLS is enabled, the SSL provider is OpenSSL.
rest.https.port	*Default*:8243 *Description*: Port number used to connect to the Data Access Gateway using HTTPS.

last-updated: Apr 11, 2022

rest.result.limit	*Default*:5000 *Description*: Limit the number of documents returned in retrieval requests using the MapR Database JSON REST API.

There is also a configuration file `/opt/mapr/data-access-gateway/conf/ojai-config.json` for parameters used by Data Access Gateway clients:

- MapR Database JSON REST API
- Node.js OJAI
- Python OJAI
- C# OJAI
- Go OJAI
- Java OJAI Thin Client

A parameter you can modify is the client sort limit:

```
{
    "ojai": {
        "mapr": {
            "query": {
                "max-client-sort-limit": 6000
            }
        }
    }
}
```

To understand why you might want to modify this parameter, see Comparisons and Sorts in OJAI Queries on page 2586.

After modifying any parameters on a node, restart the service as described in Administering the Data Access Gateway Service on page 1495.

Warden Configuration

The Warden configuration for the MapR Data Access Gateway is in the `/opt/mapr/data-access-gateway/conf/warden.data-access-gateway.conf` file on nodes where you have installed the Data Access Gateway. To control the amount of memory allocated to the service, modify the following settings:

service.heapsize.max	*Default*:3000 *Description*: Defines the maximum heap size (in MB) for the service.
service.heapsize.min	*Default*:3000 *Description*: Defines the minimum heap size (in MB) for the service.

After modifying the warden configuration file on a node, run `configure.sh -R`, and restart the service:

```
/opt/mapr/server/configure.sh -R
maprcli node services -nodes <node name> -name data-access-gateway -action
restart
```

Administering the Data Access Gateway Service

The MapR Data Access Gateway is a service that you administer in the same manner as other MapR services. The name of the service is `data-access-gateway`.

To restart the service through the CLI, run the following command:

```
maprcli node services -nodes <node name> -name data-access-gateway -action
restart
```

For details about other operations you can perform on the service, see Managing Services on page 827.

Related concepts

Understanding the MapR Data Access Gateway on page 750
The MapR Data Access Gateway is a service that acts as a proxy and gateway for translating requests between lightweight client applications and the MapR cluster.

Using the MapR Database JSON REST API on page 2696
Starting in the EEP 5.0 release, you can use a REST API to access MapR Database JSON tables. The REST API allows you to use HTTP calls to perform basic operations on MapR Database JSON tables.

Using the Node.js OJAI Client on page 2673
Starting with EEP 6.0, you can use the Node.js OJAI client to write MapR Database JSON applications. The client provides you with a lightweight library that supports the OJAI API. You can connect to MapR Database JSON from middleware components, and add, update, and query documents in a MapR Database JSON table.

Using the Python OJAI Client on page 2678
Starting with EEP 6.0, you can use the Python OJAI client to write MapR Database JSON applications. The client provides you with a lightweight library that supports the OJAI API. You can connect to MapR Database JSON, and add, update, and query documents in a MapR Database JSON table.

Using the C# OJAI Client on page 2688
Starting with EEP 6.1.0, you can use the C# OJAI client to write MapR Database JSON applications. The client provides you with a lightweight library that supports the OJAI API. You can connect to MapR Database JSON, and add, update, and query documents in a MapR Database JSON table.

Using the Go OJAI Client on page 2692
Starting with EEP 6.0.0, you can use the Go OJAI client to write MapR Database JSON applications. The client provides you with a lightweight library that supports the OJAI API. You can connect to MapR Database JSON, and add, update, and query documents in a MapR Database JSON table.

Using the Java OJAI Thin Client on page 2670
Starting with EEP 6.3.0, you can use the Java OJAI Thin Client to write MapR Database JSON applications. The Java OJAI Thin Client provides a lightweight library that supports the OJAI API. You can connect to MapR Database JSON, and add, update, and query documents in a MapR Database JSON table.

L3/L4 Load Balancing with the MapR Data Access Gateway

You can use `haproxy` for L3/L4 load balancing of clients that use the MapR Data Access Gateway. This topic describes how to install, configure, and run `haproxy`, and how to set your client connection string to connect to the load balancing service.

Determine the server where you want to run the load balancing service. The server must be reachable by the clients using the Data Access Gateway. It also must be able to connect to the Data Access Gateway.

1. Install the `haproxy` service on the server you have identified:

CentOS	`sudo yum install haproxy`

last-updated: Apr 11, 2022

Ubuntu	`sudo add-apt-repository ppa:vbernat/haproxy-1.7`
	`sudo apt update`
	`sudo apt install haproxy`

SLES	`sudo zypper install haproxy`

2. Configure the `haproxy` service by setting the following parameters in the configuration file at `/etc/haproxy/haproxy.cfg`:
   a) Create a `frontend` section with the following parameters:

```
frontend <section_name>
   mode tcp
   bind *:<port_to_use_in_the_client_connection_string>
   default_backend <backend_section_name>
```

   b) Create a `backend` section with one `server` entry for each Data Access Gateway server:

```
backend <backend_section_name>
   mode tcp
   server <DAG_server_name1> <DAG_server_host1>:<DAG_server_port1>
   server <DAG_server_name2> <DAG_server_host2>:<DAG_server_port2>
   ...
   server <DAG_server_nameN> <DAG_server_hostN>:<DAG_server_portN>
```

   The <*backend_section_name*> is the parameter you specified in Step 2a.

3. Restart the `haproxy` service:

```
sudo service haproxy restart
```

Setting Your Client Connection String

Assume you have the following `haproxy` configuration settings and you have installed `haproxy` on `node1.cluster.com`:

```
frontend connection_input
  mode tcp
  bind *:8553
  default_backend maprdb_servers

backend maprdb_servers
  mode tcp
  server srv01 node1.cluster.com:5678
  server srv02 node2.cluster.com:5678
```

You can use the following client connection string with this sample configuration:

DAG with HTTPS | TLS	`node1.cluster.com:8553?auth=basic;user=mapr;password=mapr;ssl=true;sslCA=/opt/mapr/conf/ssl_truststore.pem;sslTargetNameOverride=node1.cluster.com`

DAG with HTTP

```
node1.cluster.com:8553?
auth=basic;user=mapr;password=mapr;ssl
=false
```

L7 Load Balancing with the Data Access Gateway

You can use `nginx` for L7 load balancing of clients that use the Data Access Gateway. This topic describes how to install, configure, and run `nginx`, and how to set your client connection string to connect to the load balancing service.

Determine the server where you want to run the load balancing service. The server must be reachable by the clients using the Data Access Gateway. It also must be able to connect to the Data Access Gateway.

1. Install the `nginx` service on the server you have identified:

   CentOS

   ```
   sudo yum install nginx
   ```

   Ubuntu

   ```
   sudo apt install nginx
   ```

   SLES

   ```
   sudo zypper install nginx
   ```

2. Configure the `nginx` service by setting the following parameters in the configuration file at `/etc/nginx/nginx.conf`:
   a) In the `http` section, create an `upstream` block with one `server` entry for each Data Access Gateway server:

   ```
   upstream <upstream_name> {
       server <DAG_server_host1>:<DAG_server_port1>;
       server <DAG_server_host2>:<DAG_server_port2>;
       ...
       server <DAG_server_hostN>:<DAG_server_portN>
       }
   ```

   b) Create (or modify) the `server` block, depending on whether your cluster is secure or nonsecure:

   Secure Cluster

   For a secure cluster, you must specify the following SSL parameters:

   - Listen port and protocol
   - Path to the SSL certificate
   - Path to the SSL key
   - Path to the file containing the SSL password

   ```
   server {
       listen 80 ssl http2;
       listen [::]:80;

       ssl_certificate
   <path_to_certificate>;
       ssl_certificate_key
   <path_to_key>;
       ssl_password_file
   ```


last-updated: Apr 11, 2022

```
<path_to_password_file>;

    access_log logs/access.log
main;

    location / {
            grpc_pass grpcs://
<upstream_name>;
    }
}
```

Nonsecure Cluster

```
server {
    listen 80 http2;
    listen [::]:80;

    access_log logs/access.log
main;

    location / {
        grpc_pass grpc://
<upstream_name>;
    }
}
```

The <*upstream_name*> is the parameter you specified in Step 2a.

3. Restart the `nginx` service:

```
sudo service nginx restart
```

Setting Your Client Connection String

Assume you have the following `nginx` configuration settings and you have installed `nginx` on `node1.cluster.com`:

Secure Cluster

```
user mapr;
worker_processes 1;
error_log /var/log/nginx/error.log
warn;
pid /var/run/nginx.pid;
events {
    worker_connections  1024;
}
http {
    log_format  main  '$remote_addr -
$remote_user [$time_local] "$request"'
                      '$status
$body_bytes_sent "$http_referer"'

'"$http_user_agent"';
    upstream servers {
        server node1.cluster.com:5678;
        server node2.cluster.com:5678;
    }
    server {
        listen 80 ssl http2;
        listen [::]:80;
```

```
        ssl_certificate /opt/mapr/
conf/ssl_keystore.pem;
        ssl_certificate_key /opt/mapr/
conf/ssl_keystore.pem;
        ssl_password_file /root/
passwd;

        access_log logs/access.log
main;

        location / {
            grpc_pass grpcs://servers;
        }
    }
}
```

You can use the following client connection string with this sample configuration:

```
node1.cluster.com:80?
auth=basic;user=mapr;password=mapr;ssl
=true;sslCA=/opt/mapr/conf/
ssl_truststore.pem;sslTargetNameOverri
de=node1.cluster.com
```

Nonsecure Cluster

```
user mapr;
worker_processes 1;
error_log /var/log/nginx/error.log
warn;
pid /var/run/nginx.pid;
events {
    worker_connections  1024;
}
http {
    log_format  main  '$remote_addr -
$remote_user [$time_local] "$request"'
                      '$status
$body_bytes_sent "$http_referer"'

'"$http_user_agent"';
    upstream servers {
        server node1.cluster.com:5678;
        server node2.cluster.com:5678;
    }
    server {
        listen 80 http2;
        listen [::]:80;

        access_log logs/access.log
main;

        location / {
            grpc_pass grpc://servers;
        }
    }
}
```

You can use the following client connection string with this sample configuration:

```
node1.cluster.com:80?
auth=basic;user=mapr;password=mapr;ssl
=false
```

Planning for High Availability

Configuring a cluster for HA (High Availability) involves running redundant instances of specific services, and configuring NFS properly. When properly licensed and configured for HA, the MapR cluster provides *automatic failover* for continuity throughout the stack.

The following table provides the minimum number of instances of each core service required for HA:

Service	Minimum Number of Instances	Comments
CLDB	2	
ZooKeeper	3	At least 3 are needed to maintain a quorum in case one instance fails.
NFS	2	NFS can be configured for HA using virtual IP addresses (VIPs).
ResourceManager	2	

In HA clusters, it is appropriate to run more than one instance of the WebServer with a load balancer to provide failover. NFS can be configured for HA using VIPs.

The following sections provide information about HA planning:

CLDB Failover

Explains the concept of CLDB failover, and its advantages.

The CLDB service automatically replicates its data to other nodes in the cluster, preserving at least two (and generally three) copies of the CLDB data. If the CLDB process dies, it is automatically restarted on the node. All jobs and processes wait for the CLDB to return, and resume from where they left off, with no data or job loss.

If the node itself fails, the CLDB data is still safe, and the cluster can continue normally as soon as the CLDB is started on another node. In an Enterprise Edition-licensed cluster, a failed CLDB node automatically fails over to another CLDB node without user intervention, and without data loss. It is possible to recover from a failed CLDB node on a Community Edition cluster, but the procedure is different.

Complete the following steps to recover from a failed CLDB node on a community edition cluster:

1. Restore ZooKeeper

If the CLDB node that failed was also running ZooKeeper, install ZooKeeper on another node to maintain the minimum required number of ZooKeeper nodes. Before installing ZooKeeper on another node, ensure that the ZooKeeper role is deleted on the failed node. See Removing ZooKeeper Role for more information.

2. Locate the CLDB Data

After restoring the ZooKeeper service on the MapR cluster, use the `maprcli dump zkinfo` command to identify the latest epoch of the CLDB, identify the nodes where replicates of the CLDB are stored, and select one of those nodes to serve the new CLDB node.

Secure cluster must first be converted to non-secure cluster before running the `maprcli dump zkinfo` command. Perform the following steps as root or use sudo:

last-updated: Apr 11, 2022

Note: For non-secure clusters, skip to step 4.

1. On the ZooKeeper nodes, stop Warden and ZooKeeper by running the following commands:

```
service mapr-warden stop
service mapr-zookeeper stop
```

2. Convert the secure cluster to non-secure cluster by running the following command on the ZooKeeper nodes:

   Note: The script `configure.sh` takes comma-separated lists of cluster names and ZooKeeper host names (and optionally ports) or IP addresses.

```
/opt/mapr/server/configure.sh -C <host>[:<port>][,<host>:
[<port>]...]|<IP>[,<IP>...] -Z <host>[:<port>][,<host>:[<port>]...]|
<IP>[,<IP>...] -unsecure -R
```

3. Restart ZooKeeper:

```
service mapr-zookeeper restart
```

4. Issue the `maprcli dump zkinfo` command using the `-json` flag.

```
maprcli dump zkinfo -zkconnect localhost:5181 -json | grep -i "Container
ID"
```

   The output displays the ZooKeeper znodes. For example:

```
maprcli dump zkinfo -zkconnect localhost:5181 -json |grep -i "Container
ID" | more
   "/datacenter/controlnodes/cldb/epoch/1/KvStoreContainerInfo":"
Container ID:1
   VolumeId:1 Master:10.10.104.34:5660-10.10.105.34:5660--9-VALID
Servers:
   10.10.104.34:5660-10.10.105.34:5660--9-VALID
   10.10.104.33:5660-10.10.105.33:5660--9-VALID
   10.10.104.32:5660-10.10.105.32:5660--9-VALID
   Inactive Servers:  Unused Servers:  Latest epoch:9"
```

   In the above example output, the latest epoch is *9*.

5. In the `/datacenter/controlnodes/cldb/epoch/1` directory, locate the CLDB with the latest epoch.

   The Latest Epoch field identifies the current epoch of the CLDB data.

6. Select a CLDB from among the copies at the latest epoch. For example, `10.10.105.32:5660--9-VALID` indicates that the node has a copy at epoch 9 (the latest epoch).

You can now install a new CLDB on the selected node.

To convert the non-secure cluster to a secure cluster, run the the following command:

last-updated: Apr 11, 2022

Note: The script `configure.sh` takes comma-separated lists of cluster names and ZooKeeper host names (and optionally ports) or IP addresses.

```
/opt/mapr/server/configure.sh -C <host>[:<port>][,<host>:
[<port>]...]|<IP>[,<IP>...] -Z <host>[:<port>][,<host>:[<port>]...]|
<IP>[,<IP>...] -secure -R
```

3. Stop the Selected Node

Perform the following steps on the node you have selected for installation of the CLDB:

1. Change to the root user (or use sudo for the following commands).
2. Stop the Warden:

```
service mapr-warden stop
```

4. Remove the CLDB Role on the Failed Node

To remove the CLDB role on the failed node, perform the following steps:

1. Stop Warden on the node.

```
service mapr-warden stop
```

2. Purge the CLDB package mapr-cldb with the `apt-get`, `yum`, or `zypper` commands, depending on your operating system.

5. Install the CLDB on the Selected Node

Perform the following steps on the node you have selected for installation of the CLDB:

1. Login as `root` or use `sudo` for the following commands.
2. Install the CLDB service on the node:
   - RHEL/CentOS: `yum install mapr-cldb`
   - Ubuntu: `apt-get install mapr-cldb`

6. Configure the Selected Node

The script configure.sh configures a node to be part of a MapR cluster, or modifies services running on an existing node in the cluster. The script creates (or updates) configuration files related to the cluster and the services running on the node.

Before you run configure.sh, make sure you have a list of the hostname of the ZooKeeper nodes. You can optionally specify the ports for the CLDB and ZooKeeper nodes as well. The default ports are:

Service	Default Port #
CLDB	7222
ZooKeeper	5181

The script configure.sh takes an optional cluster name and log file, the CLDB hostname, and comma-separated list of ZooKeeper host names or IP addresses (and optionally ports), using the following syntax:

```
/opt/mapr/server/configure.sh -C <host>[:<port>] -Z <host>[:<port>]
[,<host>[:<port>]...] \
 [-L <logfile>][-N <cluster name>]
```

Note: Each time you specify the `-Z <host>[:<port>]` option, you must use the *same order* for the ZooKeeper node list. If you change the order for any node, the ZooKeeper leader election process will fail.

Example

```
/opt/mapr/server/configure.sh -C r1n1.sj.us:7222 \
-Z
r1n1.sj.us:5181,r2n1.sj.us:5181,r3n1.sj.us:5181,r4n1.sj.us:5181,r5n1.sj.us:5
181 -N MyCluster
```

7. Start the Nodes

Perform the following steps on the node you have selected for installation of the CLDB:

Start the Warden:

```
service mapr-warden start
```

After the CLDB restarts, there is a 15-minute delay before replication resumes, in order to allow all nodes to register and heartbeat. This delay can be configured using the `config save` command to set the `cldb.replication.manager.start.mins` parameter.

8. Restart All Nodes

To restart all nodes in the cluster, stop each node, configure the node with the new CLDB and ZooKeeper addresses, and start the node.

Complete the following steps on each node in the cluster:

1. Stop the node.
   a. Change to the root user (or use sudo for the following commands).
   b. Stop the Warden:

      ```
      service mapr-warden stop
      ```

2. Configure all the nodes with the new CLDB and ZooKeeper addresses.

   The script `configure.sh` configures a node to be part of a MapR cluster, or modifies services running on an existing node in the cluster. You must run this script to configure a node. The script creates (or updates) configuration files related to the cluster and the services running on the node.

   Before you run `configure.sh`, make sure you have the hostname of the CLDB node and the hostnames of the ZooKeeper nodes. You can, optionally, specify the ports for the CLDB and ZooKeeper nodes as well. The default ports are:

Service	Default Port #
CLDB	7222

last-updated: Apr 11, 2022

Service	Default Port #
ZooKeeper	5181

The script `configure.sh` takes an optional cluster name and log file, the CLDB hostname, and comma-separated list of ZooKeeper host names or IP addresses (and optionally ports), using the following syntax:

```
/opt/mapr/server/configure.sh -C <host>[:<port>] -Z <host>[:<port>]
[,<host>[:<port>]...] [-L <logfile>][-N <cluster name>]
```

Note: Each time you specify the `-Z <host>[:<port>]` option, you must use the same order for the ZooKeeper node list. If you change the order for any node, the ZooKeeper leader election process will fail.

Example:

```
/opt/mapr/server/configure.sh -C r1n1.sj.us:7222 -Z
r1n1.sj.us:5181,r2n1.sj.us:5181,r3n1.sj.us:5181,r4n1.sj.us:5181,r5n1.sj.u
s:5181 -N MyCluster
```

3. Start Warden.

```
service mapr-warden start
```

Best Practices for Running a Highly Available Cluster

Lists high availability cluster replication types, and the best practices for running such a cluster.

MapR runs a wide variety of concurrent applications in a highly available fashion. Node failures do not have cluster-wide impact, and activities on other nodes in the cluster can continue normally. In parallel, MapR components detect failures and automatically recover from them. During the recovery process, clients may experience latency, the duration of which depends on the nature of the failure.

Node Shutdown Instances

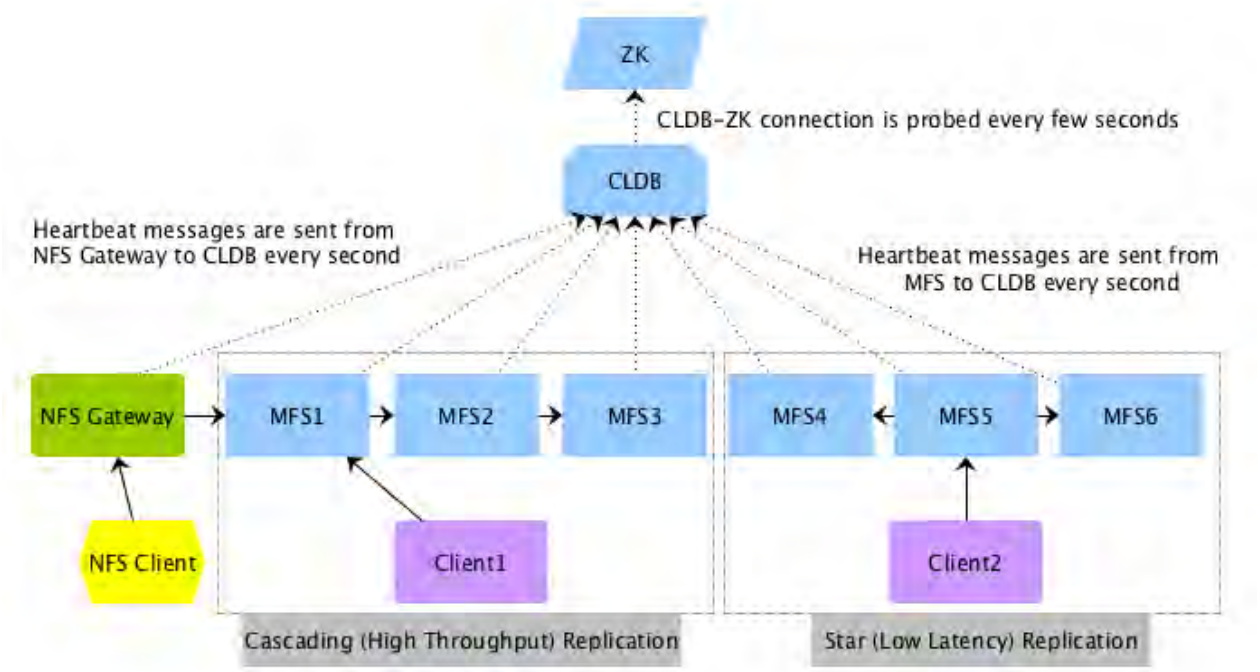
The cause of a service failure can be one of the following:

Planned shutdown	A planned/controlled failure. In this case, MapR is informed that a file server will be stopped. MapR services use this information to improve recovery behavior.
Unexpected shutdown	MapR services (MapR File System, NFS server etc.,) are stopped. However, the host operating system continues to run and failure detection is fast.
Hard unplanned shutdown	A power off, network down, or some other kind of unplanned stop. A node is stopped in a way that it is no longer reachable. Packets sent to this node do not get an error response and failure is detected through network layer's timeout mechanism. This results in longer failure detection times.

In all of these instances, the recovery process typically involves detecting that a node is unreachable, and contacting another available node for the same piece of information (either for reads, writes, or administrative operations).

How MapR File System and Associated Services Work

Let's review how MapR File System and associated services typically work, using the following illustration.



High-throughput or Cascading Replication Type: As shown in the illustration, the client, Client1, writes to a MapR filesystem, MFS1, which in turn talks to MFS2, which in turn talks to MFS3 for cascading (high throughput) replication. The replication is inline and synchronous, which means MFS1 replies to the client only after it receives a response from MFS2. MFS2, in turn, only responds to MFS1 after MFS3 has replied to it. Client1 can read from any MFS, but write only to MFS1.

Low-latency or Star Replication Type: As shown in the illustration, the client, Client2, writes to MFS5. This illustration shows an example of star (low latency) replication where MFS5 replicates to both MFS4 and MFS6 in parallel. Again, the replication is inline and synchronous, which means that MFS5 responds to Client2 only after it has received responses from both MFS4 and MFS6.

Recommended Settings for Running a Cluster with Low Latency and Fast Failover Characteristics

A well designed cluster provides automatic failover for continuity throughout the stack. For an example of a large, high-availability Enterprise Edition cluster, see Example Cluster Designs on page 118. On a large cluster designed for high availability, services should be assigned according to the service layout guidelines. For more information, see Service Layout Guidelines for Large Clusters on page 115. In general, services, specifically CLDB and ZooKeeper, should be installed on separate nodes to prevent the failure of multiple services at the same time and to enable the cluster to recover quickly.

Recommended Settings to Recover from Unplanned Shutdown

Latencies as a result of unplanned or unexpected failures/shutdowns can be improved by performing the following:

Enabling Fast Failover of Services

Describes the Fast Failover feature that allows a cluster to rapidly detect and recover from network failures.

For running a cluster with Fast Failover characteristics, enable the Fast Failover feature:

```
/opt/mapr/bin/maprcli config save -values {mfs.feature.fastfailover:1}
```


last-updated: Apr 11, 2022

If you have enabled the fast failover feature, when the MapR File System detects a failed node, it very quickly declares the node as being down. Clients experience a short latency period while the failure is being detected. Once MapR detects the failure, MapR redirects clients of the failed node to an alternate location (a replica container) for the data. If you have not enabled the fast failover feature, the MapR File System repeatedly contacts the failed node.

This feature is enabled on all new installations. For upgrade installations, this feature is not enabled by default. You need to evaluate whether this feature works well with your existing infrastructure, before enabling it. You cannot turn this feature off after turning it on.

Tuning the TCP for Fast Failure Detection
Describes how to tune the TCP stack to detect node or network failures rapidly.

An unplanned failure chiefly takes the form of a node failure or a network failure. In both instances, the network layer retries to connect to the failed node. The number of retry attempts is dictated by the TCP parameter `/proc/sys/net/ipv4/tcp_syn_retries`. The default value of that parameter is 5 (in Linux), resulting in a latency of more than a minute to detect the node failure. The problem is compounded when the same failed node is contacted repeatedly in the context of a long operation, such as when a client accesses multiple data objects present on that node.

The MapR stack solves the problem by remembering (caching) the information about a node's failure, and by not contacting that node for subsequent operations on data objects present on that node. Since all form of data is replicated, MapR services find alternative locations for a data object. This feature is in-built into the current software and does not have to be enabled explicitly. Hence, the communication between a client and a recently failed node incurs a one-time long-duration latency. As mentioned before, that latency is governed by the number of retries at the TCP level. Hence, to further improve the one-time longer latency of an operation between a pair of nodes, it is recommended that the number of TCP retries be decreased from 5 to 4, resulting in a latency of about 30 seconds.

Setting the Timeout for TCP Connections

To set the TCP retry count, set the value of `tcp_syn_retries` to 4 in the `/proc/sys/net/ipv4/` directory (for IPv4 connections). For example:

```
echo 4 > /proc/sys/net/ipv4/tcp_syn_retries
```

Similarly for IPv6 connections, set:

```
echo 4 > /proc/sys/net/ipv6/tcp_syn_retries
```

This TCP setting of 4 ensures that the TCP stack takes about 30 seconds to detect failure of a remote node. To ensure that this setting is persistent across system reboots, set this value in the `/etc/sysctl.conf` file.

Warning: This setting impacts all TCP connections to and from a node. Hence, caution must be exercised when lowering this further. Also, in some instances, reducing this further may result in a node being incorrectly flagged as unavailable.

Reducing Failure Detection Time for File Clients
Describes how to set the time for Hadoop and POSIX clients to detect node failures.

To reduce the amount of time it takes (Hadoop and FUSE-based POSIX) clients to detect (CLDB and data node) failure, define the property, `fs.mapr.connect.timeout`, in the `core-site.xml` file. The value for this property can be set in 100 milliseconds and will be rounded up to the nearest 100 milliseconds. The minimum value for this property is 100 milliseconds, which can be incremented only by units of 100 milliseconds. Suppose a value of 260 milliseconds is specified, by default, the value will automatically be rounded up to 300 milliseconds. The default value for this property is `0`, which means that the Linux TCP timeout setting will be used for connections if this property is not set.

Your entry in `core-site.xml` file should look similar to the following:

```
<property>
  <name>fs.mapr.connect.timeout</name>
  <value>200</value>
  <description>file client wait time of 200 milliseconds</description>
</property>
```

This setting (for hadoop and FUSE-based POSIX clients) ensures that the clients wait only for the specified amount of time to establish a connection. That is, it is used only for the first request sent to CLDB or a data node before or after a failure. For subsequent requests, the default system connection timeout value is used. In the event of a failure after a connection has been established, the client will wait for the connection to timeout (based on the system timeout value) before it contacts the next (CLDB or data) node to process the request.

For example, suppose the value for this parameter is 100 milliseconds and the Linux TCP connection timeout value is 30 seconds. When a hadoop or FUSE-based POSIX client contacts CLDB or a data node for the first time to establish a connection, the client will wait for 100 milliseconds before trying the next CLDB or data node. After a connection is established, for subsequent requests, the client will wait for 30 seconds for a response. If the node goes down after a connection has been established, the client will wait for 30 seconds before trying the next node. If the client contacts a recovered node for the first time, it will wait for 100 milliseconds to establish the connection.

Note: MapR filesystem does not use this property internally; it is used by Hadoop and FUSE-based POSIX clients only. This setting is not applicable to NFS gateway and loopbacknfs POSIX clients.

Detecting CLDB failures

When a connection with CLDB is established, CLDB returns the list of reachable and unreachable CLDB nodes on the cluster.

Populating the cache

The client stores information about the unreachable CLDB nodes in `/tmp/cldbinfo/unreachableCldbs` file on the client host. The format of this file is the same as the `mapr-clusters.conf` file (i.e., "`clustername ip:port`"). For example:

```
cat /tmp/cldbinfo/unreachableCldbs
  object_pools 10.10.104.33:7222
10.10.104.34:7222
```

The client reads the `mapr-clusters.conf` file and the `unreachableCldbs` file to determine the CLDB to connect to. It then tries to reach the available CLDB nodes first; it tries the unreachable CLDB nodes only if the available CLDB is unable to service its request.

Invalidating the cache

If the available CLDB is unable to service the client request, the client tries the unreachable CLDB. If an unreachable CLDB becomes reachable again, it is removed from the `/tmp/cldbinfo/unreachableCldbs` file, making it reachable for all subsequent IOs and if a reachable CLDB becomes unreachable, it is added to the `/tmp/cldbinfo/unreachableCldbs` file.

Recommended Settings for Planned Shutdown

Explains the modalities of a planned shutdown.

last-updated: Apr 11, 2022

The MapR stack improves the latencies for planned shutdowns by implementing a fast failover mechanism where different services respond to the intimation of a failure.

Notifying CLDB to Allow Fast Failover

When planning to shutdown a node, notifying CLDB of an impending shutdown allows CLDB to update the replication chain such that primary and intermediate containers, if any, are not on the node and re-assign VIPs on the node when the node actually goes down. This, in turn, allows clients to continue activities on available nodes.

MapR (v5.1) includes an argument, `node failover`, to the `maprcli` command that notifies CLDB of impending node shutdown so that CLDB can ensure that the specified node does not have any primary containers and intermediate containers (in a cascaded chain), and VIPs are re-assigned.

Shutting Down a Node

To notify CLDB of a planned shutdown of a node:

1. Enable the fast failover behavior.

   Refer to Enabling Fast Failover for more information.

2. Reset the value of tcp_syn_retries parameter.

   Refer to Tuning TCP for more information.

3. (Optional) Get the hostname of the node to put in maintenance mode by running the following command:

   ```
   /opt/mapr/bin/maprcli node list -columns hostname
   ```

4. Run the `failover` command for that node.

   For example:

   ```
   /opt/mapr/bin/maprcli node failover -nodes <node-hostname>
   ```

   Wait for few minutes (to allow containers to failover) before proceeding to the next step.

5. Stop warden on that node by running the following command:

   ```
   service mapr-warden stop
   ```

6. Notify MapR that the node is in maintenance mode and when the maintenance task is complete, remove the node from maintenance mode.

   See Performing Maintenance on a Node for the commands to run to put and take a node out of maintenance mode.

Warning: Shut down only one node at a time. Do not take down multiple nodes for maintenance at the same time.

ResourceManager High Availability

Provides an overview of how high availability for Resource Manager works.

The ResourceManager service tracks a cluster's resources and schedules YARN applications. Configure high availability for the ResourceManager so that the failure of the ResourceManager service is not a single point of failure for the cluster. The high availability of ResourceManager is based on the cluster configuration of the restart, recovery, and failover features.

last-updated: Apr 11, 2022

Restart

By default, the Warden attempts to restart a failed service three times. You can configure the frequency that Warden attempts to restart failed services before initializing failover in the warden.conf file. For more information, see warden.conf.

Recovery

When a ResourceManager restarts or fails over, the active ResourceManager can recover the state of the previously running ResourceManager. By default, ResourceManager recovery is enabled and it uses the `FileSystemRMStateStore` implementation to store the ResourceManager state in the MapR File System. You can configure the ResourceManager to have no recovery or you can enable the recovery. You can also configure the state store implementation that you want to use. For more information, see Recovery for the ResourceManager.

Failover

When a ResourceManager fails, the cluster can fail over the ResourceManager process to another node. To configure failover, the cluster must have one or more nodes with the ResourceManager role.

Note:

Starting in Version 4.0.2, zero configuration failover provides automatic failover without requiring that you specify the ResourceManager nodes when you run configure.sh. It also does not require any further configuration to yarn-site.xml.

Upgrade any client nodes to the 4.0.2 client to ensure proper communication with the ResourceManager service. Earlier versions of the MapR client do not support the zero configuration feature.

You can select one of the following failover implementations when you use the configure.sh utility to configure each node:

- **Zero Configuration Failover.** With zero configuration failover, the ResourceManager process only runs on one node in the cluster. When the active ResourceManager fails, one of the standby ResourceManager nodes automatically loads the working state from the state store and continues providing services to the cluster. Zero configuration failover is the default, recommended setting for the following reasons:
  - **Only one ResourceManager process consumes cluster resources.** With the manual or automatic failover option, the active and standby ResourceManagers consume cluster resources.
  - **Warden initiates failover automatically.** With the manual failover, you need to manually run the yarn rmadmin command for failover to occur.
  - **Simplified clients connectivity.** Clients identify the active ResourceManager with a single request to the Zookeeper. With the manual or automatic failover option, ResourceManager clients connect to each ResourceManager in a round-robin fashion until they locate the active ResourceManager; this results in delays when launching or querying jobs.
  - **Consistent Configuration.** All cluster nodes and clients can use the same yarn-site.xml configuration file. With manual or automatic failover, you must maintain a customized yarn-site.xml file for each node that runs the ResourceManager.

  For more information, see Zero Configuration Failover for the ResourceManager.

last-updated: Apr 11, 2022

- **Manual or Automatic Failover.** With manual failover or automatic failover, one active ResourceManager and one or more standby Resource Managers run in the cluster. The standby ResourceManager nodes run the ResourceManager process without loading the working state. When the active ResourceManager fails, one of the standby ResourceManager nodes can load the working state from the state store and continue providing services to the cluster. For more information, see Manual or Automatic Failover for the ResourceManager.

You can perform the following procedures to manage ResourceManager:

Manual or Automatic Failover for the ResourceManager

With manual or automatic failover, an active ResourceManager and two standby ResourceManager processes run in the cluster. The standby ResourceManager nodes run the ResourceManager process without loading the working state. When the active ResourceManager fails, one of the standby ResourceManager nodes can load the working state from the ZooKeeper and continue providing services to the cluster.

ResourceManager clients (MapR client nodes, ApplicationMaster processes, and NodeManager nodes) attempt connections to the ResourceManager nodes in a round-robin fashion until they hit an active ResourceManager node. If the active ResourceManager node is down, ResourceManager clients resume round-robin polling until an active ResourceManager node is detected.

For web requests, including REST API requests, standby ResourceManager nodes automatically redirect web requests to the active ResourceManager node.

The difference between manual and automatic failover is how the transition from standby to active occurs for the ResourceManager process.

- With manual failover, you manually invoke the transition of the ResourceManager from standby to active with the yarn rmadmin command.
- With automatic failover, the ResourceManager processes have an embedded ZooKeeper-based ActiveStandbyElector, which chooses the active ResourceManager. This ActiveStandbyElector also detects failures in the currently active ResourceManager and automatically transitions one of the standby ResourceManagers to an active state.

If you specify multiple ResourceManagers when you run configure.sh, automatic failover is configured. However, you can edit the yarn-site.xml file to enable manual failover instead.

Automatic Failover Administration

The Zookeeper-based ActiveStandbyElector on each ResourceManager node detects failures in the currently active ResourceManager and automatically transitions one of the standby ResourceManagers to an active state. Therefore, `rmadmin -transitionToStandby` and `-transitionToActive` are disabled.

Configuring Automatic Failover for the ResourceManager

To use automatic failover, specify multiple ResourceManagers when you run configure.sh on each node in the cluster.

The following configure.sh script syntax configures three ResourceManager nodes (one active and two standby) and one HistoryServer node:

```
/opt/mapr/server/configure.sh -C <CLDB node list> -Z <ZK node list> -RM
<hostname1,hostname2,hostname3> -HS <hostname1> [additional parameters]
```

Note: After you run configure.sh, each ResourceManager node contains a different value for the `yarn.resourcemanager.ha.id` property in the yarn-site.xml.

last-updated: Apr 11, 2022

Example yarn-site.xml file

The following configure.sh syntax specifies three ResourceManager nodes (nodeA, nodeB, and nodeC) and a HistoryServer node (nodeA):

```
/opt/mapr/server/configure.sh -C node1,node2,node3 -Z node1,node2,node3 -RM
nodeA,nodeB,nodeC -HS nodeA [additional parameters]
```

```
<?xml version="1.0"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at
    http://www.apache.org/licenses/LICENSE-2.0
  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->
<configuration>
  <!-- Resource Manager HA Configs -->
  <property>
    <name>yarn.resourcemanager.ha.enabled</name>
    <value>true</value>
  </property>
  <property>
    <name>yarn.resourcemanager.ha.automatic-failover.enabled</name>
    <value>true</value>
  </property>
  <property>
    <name>yarn.resourcemanager.ha.automatic-failover.embedded</name>
    <value>true</value>
  </property>
  <property>
    <name>yarn.resourcemanager.recovery.enabled</name>
    <value>true</value>
  </property>
  <property>
    <name>yarn.resourcemanager.cluster-id</name>
    <value>yarn-my.cluster.com</value>
  </property>
  <property>
    <name>yarn.resourcemanager.ha.rm-ids</name>
    <value>rm1,rm2,rm3</value>
  </property>
  <property>
    <name>yarn.resourcemanager.ha.id</name>
    <value>rm1</value>
  </property>
  <property>
    <name>yarn.resourcemanager.zk-address</name>
    <value>node1:5181,node2:5181,node3:5181</value>
  </property>
  <!-- Configuration for rm1 -->
  <property>
    <name>yarn.resourcemanager.scheduler.address.rm1</name>
    <value>nodeA:8030</value>
  </property>
  <property>
    <name>yarn.resourcemanager.resource-tracker.address.rm1</name>
    <value>nodeA:8031</value>
  </property>
```

```
  <property>
    <name>yarn.resourcemanager.address.rm1</name>
    <value>nodeA:8032</value>
  </property>
  <property>
    <name>yarn.resourcemanager.admin.address.rm1</name>
    <value>nodeA:8033</value>
  </property>
  <property>
    <name>yarn.resourcemanager.webapp.address.rm1</name>
    <value>nodeA:8088</value>
  </property>
  <property>
    <name>yarn.resourcemanager.webapp.https.address.rm1</name>
    <value>nodeA:8090</value>
  </property>
  <!-- Configuration for rm2 -->
  <property>
    <name>yarn.resourcemanager.scheduler.address.rm2</name>
    <value>nodeB:8030</value>
  </property>
  <property>
    <name>yarn.resourcemanager.resource-tracker.address.rm2</name>
    <value>nodeB:8031</value>
  </property>
  <property>
    <name>yarn.resourcemanager.address.rm2</name>
    <value>nodeB:8032</value>
  </property>
  <property>
    <name>yarn.resourcemanager.admin.address.rm2</name>
    <value>nodeB:8033</value>
  </property>
  <property>
    <name>yarn.resourcemanager.webapp.address.rm2</name>
    <value>nodeB:8088</value>
  </property>
  <property>
    <name>yarn.resourcemanager.webapp.https.address.rm2</name>
    <value>nodeB:8090</value>
  </property>
<!-- Configuration for rm3 -->
  <property>
    <name>yarn.resourcemanager.scheduler.address.rm3</name>
    <value>nodeC:8030</value>
  </property>
  <property>
    <name>yarn.resourcemanager.resource-tracker.address.rm3</name>
    <value>nodeC:8031</value>
  </property>
  <property>
    <name>yarn.resourcemanager.address.rm3</name>
    <value>nodeC:8032</value>
  </property>
  <property>
    <name>yarn.resourcemanager.admin.address.rm3</name>
    <value>nodeC:8033</value>
  </property>
  <property>
    <name>yarn.resourcemanager.webapp.address.rm3</name>
    <value>nodec:8088</value>
  </property>
  <property>
    <name>yarn.resourcemanager.webapp.https.address.rm3</name>
```


last-updated: Apr 11, 2022

```
    <value>nodeC:8090</value>
  </property>
  <!-- :::CAUTION::: DO NOT EDIT ANYTHING ON OR ABOVE THIS LINE -->
  <property>
    <name>yarn.resourcemanager.am.max-attempts</name>
    <value>4</value>
  </property>
</configuration>
```

Manual Failover Administration

Configure manual failover for the ResourceManager if you want to manually transition the state of ResourceManagers in the cluster. In the event of a ResourceManager failure, you use `rmadmin` commands to check the status of each ResourceManager and then transition a standby ResourceManager to the active state.

Configuring Manual Failover for the ResourceManager

To configure manual failover, specify multiple ResourceManagers when you run `configure.sh` on each node in the cluster and then edit `yarn-site.xml` to disable automatic failover.

1. Specify multiple ResourceManagers when you run configure.sh on each cluster and client node. The following the `configure.sh` script syntax configures three ResourceManager nodes (one active and two standby):

   ```
   /opt/mapr/server/configure.sh -C <CLDB node list> -Z <ZK node list> -RM
   <hostname1,hostname2,hostname3> -HS <hostname1> [additional parameters]
   ```

   Note: After you run `configure.sh`, each ResourceManager node contains a different value for the `yarn.resourcemanager.ha.id` property in the `yarn-site.xml`.

2. Disable the following automatic failover properties in the `yarn-site.xml` on each node with the ResourceManager role:
   - `yarn.resourcemanager.ha.automatic-failover.enabled`
   - `yarn.resourcemanager.ha.automatic-failover.embedded`
3. Restart the ResourceManager service. For more information, see Restarting the Services on page 831.

Transitioning a Standby ResourceManager to Active

The `yarn rmadmin` command includes options to manage high availability for the ResourceManager, including transitioning a ResourceManager node between active and standby modes. These commands take the ResourceManager service ID as an argument and can be run on any node in the cluster. The *serviceID* of a ResourceManager is set in the `yarn.resourcemanager.ha.rm-ids` property of the *yarn-site.xml* file.

Transition a standby ResourceManager to the active state when the active ResourceManager process has failed or the node that runs the process is no longer accessible.

1. Determine if an active ResourceManager is running in the cluster. See Checking the ResourceManager State.
2. Run the following command to set the current active ResourceManager to standby:

   ```
   yarn rmadmin -transitionToStandby <serviceID>
   ```

3. Run the following command to transition the standby ResourceManager to the active state:

```
yarn rmadmin -transitionToActive <serviceID>
```

Checking the ResourceManager State

When you configure manual or automatic failover, the ResourceManager is either in active or standby state. Each ResourceManager has a serviceID that identifies the service.

- To check the state of a ResourceManager, run the following command with the serviceID:

```
yarn rmadmin -getServiceState <serviceID>
```

  The command returns `active` or `standby` based on the state of the ResourceManager associated with the serviceID that you provide.

  Note: Tip To determine the serviceIDs associated with the ResourceManagers in the cluster, run `hadoop conf | grep yarn.resourcemanager.ha.rm-ids`

Using Central Configuration with Manual and Automatic Failover

When you configure manual or automatic failover for the ResourceManager, the contents of the *yarn-site.xml* configuration file are slightly different on each ResourceManager node as the value of the `yarn.resourcemanager.ha.id` property is distinct for each ResourceManager node. If your cluster is using the central configuration feature, configure central configuration overrides for each ResourceManager node:

1. Keep a central copy of the file at */var/mapr/configuration/default/hadoop/hadoop-2.x/etc/hadoop/yarn-site.xml*.

2. Configure central configuration overrides in the following manner:
   `/var/mapr/configuration/nodes/<HOSTNAME FOR RM 1>/hadoop/hadoop-2.x/etc/hadoop/yarn-site.xml` `/var/mapr/configuration/nodes/<HOSTNAME FOR RM 2>/hadoop/hadoop-2.x/etc/hadoop/yarn-site.xml` `/var/mapr/configuration/nodes/<HOSTNAME FOR RM 3>/hadoop/hadoop-2.x/etc/hadoop/yarn-site.xml`

Zero Configuration Failover for the ResourceManager

As of MapR 4.0.2, you can use zero configuration failover. With zero configuration failover, the ResourceManager role is installed on two or more nodes but the ResourceManager process only runs on one node in the cluster.

If the node running the ResourceManager process fails and the Warden on that node is unable to restart it, the Warden on each node and Zookeeper work together to start a ResourceManager process on the cluster. ResourceManager clients connect to the Zookeeper to determine which ResourceManager node is active. Therefore, when failover occurs, the Resource Manager clients are not affected as they automatically connect to the active ResourceManager.

Note: When you run `maprcli service list` command, the state of the active ResourceManager process displays as 2 (running) but the other ResourceManagers displays as 5 (stand by).

Enabling Zero Configuration Failover for the ResourceManager

To enable zero configuration failover, do not specify the -RM parameter when you run configure.sh on each node in the cluster. However, for failover to occur, at least two nodes in the cluster must have the ResourceManager role.

last-updated: Apr 11, 2022

For example, if the cluster includes multiple nodes with the ResourceManager role, you can run the following configure.sh command on each cluster node and no further configuration is required:

```
/opt/mapr/server/configure.sh -N mycluster -C centos21 -Z centos21 -HS
centos22 -F /tmp/disks.txt -disk-opts F
```

configure.sh automatically populates yarn-site.xml with the following configuration:

```
<configuration>
<!-- Resource Manager MapR HA Configs -->
<property>
        <name>yarn.resourcemanager.ha.custom-ha-enabled</name>
        <value>true</value>
        <description>MapR Zookeeper based RM Reconnect Enabled.
If this is true, set the failover proxy to be the class
MapRZKBasedRMFailoverProxyProvider</description>
</property>
<property>
        <name>yarn.client.failover-proxy-provider</name>

<value>org.apache.hadoop.yarn.client.MapRZKBasedRMFailoverProxyProvider</
value>
        <description>Zookeeper based reconnect proxy provider. Should
be set if and only if mapr-ha-enabled property is true.</description>
</property>
<property>
        <name>yarn.resourcemanager.recovery.enabled</name>
        <value>true</value>
        <description>RM Recovery Enabled</description>
</property>
<!-- :::CAUTION::: DO NOT EDIT ANYTHING ON OR ABOVE THIS LINE -->
</configuration>
```

For more information about the ResourceManager properties in yarn-site.xml, see ResourceManager Configuration Properties.

Updating ResourceManager Ports

To simplify the failover configurations in the *yarn-site.xml* file, Warden maintains the list of ResourceManager ports in the *warden.resourcemanager.conf* file. For a list of the default port numbers, see Ports Used by MapR Software on page 2290. If you want to edit the default ResourceManager ports, edit the *warden.resourcemanager.conf* file and the *yarn-site.xml* file on each ResourceManager node.

Note: If each node requires different ResourceManager ports, you must maintain a separate *yarn-site.xml* file for each node. Therefore, to you use Central Configuration, you must create a customized configuration file for each ResourceManager node in the cluster.

To update the port numbers, edit the values in the *warden.resourcemanager.conf* file and add the values in the *yarn-site.xml* file.

1. Open the warden.resourcemanager.conf file (`/opt/mapr/conf/conf.d/warden.resourcemanager.conf`).

last-updated: Apr 11, 2022

2. Edit the port numbers, which are listed using the following format: `service.extinfo.<port>=<port number>`

Port Name	Property Name in warden.resourcemanager.conf
ResourceManager Scheduler RPC (for ApplicationMasters)	service.extinfo.SCHEDULER_PORT
ResourceManager Resource Tracker RPC (for NodeManagers)	service.extinfo.RESOURCETRACKER_PORT
ResourceManager Client RPC	service.port
ResourceManager Admin RPC	service.extinfo.ADMIN_PORT
ResourceManager Web UI (HTTP)	service.extinfo.WEBAPP_PORT
ResourceManager Web UI (HTTPS)	service.extinfo.WEBAPP_HTTPS_PORT

3. Open the *yarn-site.xml* file (`/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop/yarn-site.xml`).

4. For each port that you edited, add the associated property to the *yarn-site.xml* file:

Port Name	Property Name in yarn-site.xml
ResourceManager Scheduler RPC (for ApplicationMasters)	yarn.resourcemanager.scheduler.address
ResourceManager Resource Tracker RPC (for NodeManagers)	yarn.resourcemanager.resource-tracker.address
ResourceManager Client RPC	yarn.resourcemanager.address
ResourceManager Admin RPC	yarn.resourcemanager.admin.address
ResourceManager Web UI (HTTP)	yarn.resourcemanager.webapp.address
ResourceManager Web UI (HTTPS)	yarn.resourcemanager.webapp.https.address

   For example, to update the port number for the ADMIN_PORT to 9000 on each node, enter the following in the *yarn-site.xml* file on each node:

```
<property>
   <name>yarn.resourcemanager.adminaddress</name>
   <value>10.10.30.140:9000</value>
</property>
```

5. Restart the Warden and the ResourceManager services.

Switching from Zero Configuration to Manual or Automatic Failover

You can change your ResourceManager failover implementation from zero configuration to manual or automatic failover by re-configuring all the cluster and client nodes.

For more information, see Configuring Manual Failover for the Resource Manager or Configuring Automatic Failover for the Resource Manager.

Recovery for the ResourceManager

After a restart or failover, the active ResourceManager recovers the ResourceManager state based on the checkpoints provided in the ResourceManager state store. During recovery, the ResourceManager resumes applications and tasks that were running prior to the failover but were not completed.

Two implementations of the ResourceManager state store are available:

- **FileSystemRMStateStore.** Enables implicit write access to a single ResourceManager node. MapR File System provides fencing implicitly and its state store implementation provides better scalability and failover performance than the ZKRMStateStore. The state store is also naturally protected by MapR File System replication. By default, FileSystemRMStateStore is the state store implementation for the ResourceManager and the ResourceManager state store is maintained in the following MapR filesystem volume: `/var/mapr/cluster/yarn/rm/system`.
- **ZKRMStateStore.** Enables implicit write access to a single ResourceManager node. This is usually recommended for HA implementations where YARN is running on HDFS. However, FileSystemRMStateStore is recommended in a MapR cluster.

Note: For recovery to occur,all ResourceManager nodes must have access to the ResourceManager state store.

ResourceManager Recovery Administration

To change the default behavior, update the ResourceManager configuration in the yarn-site.xml files and restart the ResourceManager(s). The yarn-site.xml is located in the following directory: `/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop/`

You may want to perform the following tasks:

Disabling the restart of applications after failover

You can configure the ResourceManager to not recover its state after a restart or failover occurs.

- Set the value of `yarn.resourcemanager.recovery.enabled` to `false` in `yarn-site.xml` on each ResourceManager node.

Configuring Maximum Attempts for Applications
Describes how to set the maximum number of restart attempts for all applications run by the MapR ResourceManager and the ApplicationMaster.

When an ApplicationMaster fails, the ResourceManager restarts the ApplicationMaster as long as the number of restart attempts does not exceed the max-attempt values set at the ResourceManager and ApplicationMaster level . By default, the maximum attempt value is set to 2.

- To configure the maximum number of ApplicationMaster attempt retries for all applications run by the ResourceManager:

  Set the value of `yarn.resourcemanager.am.max-attempts` in the `yarn-site.xml` file. The value defaults to 2.

- To configure the number of ApplicationMaster attempts allowed for the MapReduce ApplicationMaster:

  Set the value of `mapreduce.am.max-attempts` in the `mapred-site.xml` file. The value defaults to 2.

Configuring the MapR File System State Store
Describes the configuration of the MapR state store.

By default, the Resource Manager stores its state in the MapR filesystem. However, you can change the values for the following properties related to the MapR filesystem state store:

- To configure the URI to the state store location:

  Set the value of `yarn.resourcemanager.fs.state-store.uri` in the `yarn-site.xml` file. The value defaults to the ResourceManager volume (`/var/mapr/cluster/yarn/rm/system`).

- To configure the retry policy used by the state store client to connect with MapR file system:

  Set the value of `yarn.resourcemanager.fs.state-store.retry-policy-spec` in the `yarn-site.xml` file. The value defaults to (2000,500).

- To configure the number of completed applications retained by the state store:

  Set the value of `yarn.resourcemanager.state-store.max-completed-applications` in the `yarn-site.xml` file. The value defaults to 10000.

Enabling ZooKeeper Based State Store

By default, the Resource Manager stores its state in the MapR File System. However, you can use the Zookeeper based state store instead. To configure the ResourceManager to use the Zookeeper state store:

- Set the value of `yarn.resourcemanager.store.class` to `org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore` in the yarn-site.xml.
- Set the value of `yarn.resourcemanager.zk-address` to a comma-separated list of host:port pairs for each ZooKeeper server used by the ResourceManager. This property needs to be set in yarn-site.xml.

ResourceManager Configuration Properties

The following sections describe the properties that you can configure for the ResourceManager. The default values for these properties are defined in the yarn-default.xml or by MapR. You can configure overrides to the default by adding to or editing the properties in yarn-site.xml.

The following sections provide information about ResourceManager failover and recovery properties:

ResourceManager Failover Properties

The following table describes the configuration properties for ResourceManager failover:

Property	Description
yarn.resourcemanager.ha.custom-ha-enabled	When yarn.client.failover-proxy-provider is set to org.apache.hadoop.yarn.client.MapRZKBasedRMFailoverProxyProvider, this property must `true`. The default, set by configure.sh in yarn-site.xml when the cluster uses zero configuration failover for the ResourceManager, is `true`.
yarn.resourcemanager.ha.enabled	Enables high availability for the ResourceManager. The default, set by MapR in the yarn-site.xml, is `true`. This property must be set to `true` for failover to occur.
yarn.resourcemanager.ha.automatic-failover.enabled	When yarn.resourcemanager.ha.enabled is true, this property enables the ResourceManager to automatically failover. The default, set in yarn-default.xml, is `true`.
yarn.resourcemanager.ha.automatic-failover.embedded	When yarn.resourcemanager.ha.enabled is true, this property enables the ResourceManager to use the embedded automatic failover. The default, set in yarn-default.xml, is `true`.
yarn.resourcemanager.cluster-id	Specifies the cluster that the ResourceManager belongs to. This value is originally set by configure.sh in the yarn-site.xml and the value is required for failover to occur.
yarn.resourcemanager.ha.rm-ids	The ResourceManager service ID. Configure.sh adds this property to each node with the ResourceManager role.

Property	Description
yarn.resourcemanager.ha.id	Specifies the serviceID of the ResourceManager on the current node.
yarn.resourcemanager.zk-address	Specifies the zookeeper quorum that the ResourceManager belongs to. This value is originally set by configure.sh in the yarn-site.xml when you configure failover.
yarn.client.failover-proxy-provider	Specifies the ResourceManager failover implementation used by clients, ApplicationMasters, and NodeManagers. configure.sh sets this value based on the type of failover that you configure. • For automatic or manual failover, configure.sh sets this value to org.apache.hadoop.yarn.client.ConfiguredRMFailoverProxyProvider • For zero configuration failover, configure.sh sets this value to org.apache.hadoop.yarn.client.MapRZKBasedRMFailoverProxyProvider This value is set by configure.sh in yarn-site.xml when you configure failover. Otherwise, the default, set in yarn-default.xml is org.apache.hadoop.yarn.client.DefaultFailoverProxyProvider.
yarn.resourcemanager.scheduler.address[.<serviceID>]	The address of the scheduler interface This value, including the serviceID, is set by configure.sh in yarn-site.xml when you configure manual or automatic failover. For zero configuration failover, this property in not needed unless you have configure custom port values. When you specify the custom port number, the serviceID is not required.
yarn.resourcemanager.resource-tracker.address[.<serviceID>]	The address of the resource tracker interface. ResourceManager listens for container requests and heartbeats from the NodeManagers on this port. This value, including the serviceID, is set by configure.sh in yarn-site.xml when you configure manual or automatic failover. For zero configuration failover, this property in not needed unless you have configure custom port values. When you specify the custom port number, the serviceID is not required.
yarn.resourcemanager.address[.<serviceID>]	The address of the client interface. The ResourceManager listens for client requests on this port. This value, including the serviceID, is set by configure.sh in yarn-site.xml when you configure manual or automatic failover. For zero configuration failover, this property in not needed unless you have configure custom port values. When you specify the custom port number, the serviceID is not required.

last-updated: Apr 11, 2022

Property	Description
yarn.resourcemanager.admin.address[.<serviceID>]	The address of the administrative interface. ResourceManager listens for administrative requests from the yarn rmadmin command on this port. This value, including the serviceID, is set by configure.sh in yarn-site.xml when you configure manual or automatic failover. For zero configuration failover, this property in not needed unless you have configure custom port values. When you specify the custom port number, the serviceID is not required.
yarn.resourcemanager.webapp.address[.<serviceID>]	The address of the ResourceManager web UI. This value, including the serviceID, is set by configure.sh in yarn-site.xml when you configure manual or automatic failover. For zero configuration failover, this property in not needed unless you have configure custom port values. When you specify the custom port number, the serviceID is not required.
yarn.resourcemanager.webapp.https.address[.<serviceID>]	The address of the secure ResourceManager web UI. This value, including the serviceID, is set by configure.sh in yarn-site.xml when you configure manual or automatic failover. For zero configuration failover, this property in not needed unless you have configure custom port values. When you specify the custom port number, the serviceID is not required.
yarn.client.failover-max-attempts	The max number of times FailoverProxyProvider should attempt failover. The default is -1.
yarn.client.failover-sleep-base-ms	The sleep base (in milliseconds) to be used for calculating the exponential delay between failovers. The value defaults to the value set by yarn.resourcemanager.connect.retry-interval.ms, which is 30000 ms.
yarn.client.failover-sleep-max-ms	The maximum sleep time (in milliseconds) between failovers. The value defaults to the value set by yarn.resourcemanager.connect.retry-interval.ms, which is 30000 ms.
yarn.client.failover-retries	The number of times a client attempts to reconnect to a ResourceManager. The default, set in yarn-default.xml, is 0 (infinite).
yarn.client.failover-retries-on-socket-timeouts	The number of times a client attempts to reconnect to a ResourceManager on socket timeouts. The default, set in yarn-default.xml, is 0 (infinite).

ResourceManager Recovery Properties

The following table describes the configuration properties for ResourceManager recovery:

last-updated: Apr 11, 2022

Property	Description
yarn.resourcemanager.recovery.enabled	Enables the Resourcemanager to recovery based on the information in the ResourceManager state store. The default, set by configure.sh, is `true`.
yarn.resourcemanager.am.max-attempts	The maximum number of application attempts. This is a global setting for all ApplicationMaster nodes. You can configure an individual maximum number of application attempts for each ApplicationMaster node, but this property sets a global upper bound that overrides the individual node configuration. The default, set in yarn-default.xml, is 2.
mapreduce.am.max-attempts	The maximum number of MapReduce application attempts. If this value is larger than the value set by the ResourceManager, the ResourceManager value will override this value. The default number is set to 2, to allow at least one retry for AM. This property is set in mapred-default.xml.
yarn.resourcemanager.fs.state-store.uri	URI pointing to the location of the FileSystem path where the ResourceManager state is stored. The default value is configured to the path for the ResourceManager volume (`/var/mapr/cluster/yarn/rm/system`). If the FileSystem name is not provided, the system uses the value specified in the `fs.default.name` specified in the `core-site.xml` file.
yarn.resourcemanager.fs.state-store.retry-policy-spec	Specifies the retry policy for the MapR File System client. This policy is specified in pairs of values for the sleep time, in milliseconds, and number of retries. Each pair is enclosed in parentheses, such as `(1000,20), (2000,30)`. The previous example sleeps for 1000 milliseconds for twenty retries, then thirty more retries 2000 milliseconds apart. The default, set in yarn-default.xml, is `(2000,500)`.
yarn.resourcemanager.store.class	The class name of the state-store to be used for saving application/attempt state and the credentials. The available state-store implementations are `org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore`, a ZooKeeper based state-store implementation, and `org.apache.hadoop.yarn.server.resourcemanager.recovery.FileSystemRMStateStore`, a state-store implementation based on MapR File System. The default, yarn-default.xml, is `org.apache.hadoop.yarn.server.resourcemanager.recovery.FileSystemRMStateStore`.

Property	Description
yarn.resourcemanager.state-store.max-completed-applications	The maximum number of completed applications that the state store retains, which is a number less than or equal to ${yarn.resourcemanager.max-completed-applications}. The default value is 10000. This setting ensures that the applications kept in the state store are consistent with the applications in ResourceManager memory. Any value larger than ${yarn.resourcemanager.max-completed-applications} is reset to the default. The value of this property affects ResourceManager recovery performance.Typically, a smaller value optimizes performance for recovery.
yarn.resourcemanager.zk-address	A comma-separated list of Host:Port pairs. Each corresponds to a ZooKeeper server, such as 127.0.0.1:5181,127.0.0.1:5181,127.0.0.1:5181. These hosts are used by the ResourceManager to store state.
yarn.resourcemanager.zk-state-store.parent-path	The full path of the root znode where ResourceManager state is stored. The default value is/rmstore.
yarn.resourcemanager.zk-num-retries	Number of times the ResourceManager tries to connect to the ZooKeeper server when the connection is lost. The default value is 500.
yarn.resourcemanager.zk-retry-interval-ms	The interval between retries, in milliseconds, when connecting to a ZooKeeper server. The default value is 2000.
yarn.resourcemanager.zk-timeout-ms	The ZooKeeper session timeout in milliseconds. The ZooKeeper server uses this configuration to determine session expiration. Sessions expire when the server does not receive a heartbeat from the client within the session timeout period. The default value is 10000.
yarn.resourcemanager.zk-acl	ACLs that set permissions on ZooKeeper znodes. The default value is world:anyone:rwcda

Administrator's Reference

This section contains in-depth reference information for the administrator.

maprcli and REST API Syntax

This section provides information about the MapR command API. Most commands can be run on the command-line interface (CLI), or by making REST requests programmatically or in a browser.

To run CLI commands, use an ssh connection to any node in the cluster. To use the REST interface, make HTTP requests to a node that is running the WebServer service.

last-updated: Apr 11, 2022

Overview
Each command reference page includes the command syntax, a table that describes the parameters, and examples of command usage.

In each parameter table, required parameters are in **bold** text. For output commands, the reference pages include tables that describe the output fields. Values that do not apply to particular combinations are marked **NA**.

REST API Syntax
Describes the MapR REST API syntax format.

MapR REST calls use the following format:

```
https://<host>:<port>/rest/<command>[/<subcommand>...]?<parameters>
```

Construct the `<parameters>` list from the required and optional parameters, in the format `<parameter>=<value>` separated by the ampersand (&) character. Example:

```
https://r1n1.qa.sj.ca.us:8443/rest/volume/mount?name=test-volume&path=/test
```

Values in REST API calls must be URL-encoded. For readability, the values in this document use the actual characters, rather than the URL-encoded versions.

Authentication

To make REST calls using `curl` or `wget`, provide the username and password. To configure PAM for REST API, see PAM Configuration.

Curl Syntax

```
curl -k -u <username> https://<host>:<port>/rest/<command>...
```

Warning: To keep your password secure, do not provide it on the command line. Curl will prompt you for your password, and you can enter it securely.

Wget Syntax

```
wget --no-check-certificate --user <username> --ask-password https://
<host>:<port>/rest/<command>...
```

Warning: To keep your password secure, do not provide it on the command line. Use the `--ask-password` option instead; then wget will prompt you for your password and you can enter it securely.

To authenticate to the REST interface, use basic authentication or SPNEGO.

Basic Authentication

To authenticate using basic authentication, send a request with a basic authorization header, which has a user ID and password. For example, to authenticate using basic authentication, run the following command:

```
curl https://
<webserver-hostname>:8443/login -d
'username=root&password=mapr'
```

On the other hand, if you do not wish to reauthenticate with every request, you can save the cookie and send the cookie with every subsequent request. For example, to authenticate and save the cookie in a text

last-updated: Apr 11, 2022

file named `cookiejar.txt` in `/tmp` directory, run the following command:

```
curl -X POST -c /tmp/
cookiejar.txt "https://
<webserver-hostname>:8443/login -d"
'username=<name>&password=<pwd>'
```

To send the cookie with subsequent requests, for example to retrieve the list of nodes on the cluster, you can submit a request similar to the following:

```
curl -L -b /tmp/cookiejar.txt
"https://<webser-hostname>:8443/rest/
node/list/"
```

SPNEGO

To authenticate using SPNEGO, ensure that the apiserver nodes are configured for SPNEGO. After configuring, send a negotiate authorization header. For example, to authenticate with the SPNEGO token and save the cookie in a text file named `cookiejar.txt` in your home directory, run the following command:

```
curl --negotiate -u : -b ~/
cookiejar.txt -c ~/cookiejar.txt
https://<web server node>:8443/rest/
<API call> -k -v
```

The contents of the cookie is something similar to the following:

```
cat /tmp/cookiejar.txt
# Netscape HTTP Cookie File
# https://curl.haxx.se/docs/http-cookies.html
# This file was generated by libcurl! Edit at your own risk.

#HttpOnly_<webserver-hostname>  FALSE  /  TRUE   1509486224
MAPR.APISERVER.JSESSIONID  node014ukard563rhu1ns8umn2s6uft3709.node0
#HttpOnly_<webserver-hostname>  FALSE  /  FALSE  0
MAPR.APISERVER.SESSIONID   HZA9C20D084E614E36AA567F47FC9105A4
```

REST API Calls to Remote Cluster

If you have secure clusters and you wish to make REST API calls to a remote secure cluster, you can specify the `cluster` parameter in the request if you have your environment configured for remote access. To set up your environment for API calls to remote secure cluster:

1. Follow the steps for configuring secure clusters for running commands remotely.
2. Verify access to run remote commands.
3. Rename the ticket file generated using the `maprlogin password` command to `maprusertickt` and move the file to `/opt/mapr/conf` directory.

Command-Line Interface

Describes how the MapR CLI command syntax is documented.

The MapR CLI commands are documented using the following conventions:

- `[Square brackets]` indicate an optional parameter

last-updated: Apr 11, 2022

- `<Angle brackets>` indicate a value to enter

The following syntax example shows that the `volume mount` command requires the `-name` parameter, for which you must enter a list of volumes, and all other parameters are optional:

```
maprcli volume mount
      [ -cluster <cluster> ]
      -name <volume list>
      [ -path <path list> ]
```

For clarity, the syntax examples show each parameter on a separate line; in practical usage, the command and all parameters and options are typed on a single line. Example:

```
maprcli volume mount -name test-volume -path /test
```

Common Parameters
Describes parameters that are available for many commands.

The following parameters are available for many commands in both the REST and command-line contexts.

Parameter	Description
cluster	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.
zkconnect	A ZooKeeper connect string, which specifies a list of the hosts running ZooKeeper, and the port to use on each, in the format: `'<host>[:<port>][,<host>[:<port>]...]'` Default: `'localhost:5181'` In most cases the ZooKeeper connect string can be omitted, but it is useful in certain cases when the CLDB is not running.

Common Options
Describes options that are available for many commands.

The following options are available for most commands in the command-line context.

Option	Description
-noheader	When displaying tabular output from a command, omits the header row.
-long	Shows the entire value. This is useful when the command response contains complex information. When -long is omitted, complex information is displayed as an ellipsis (...).
-json	Displays command output in JSON format. When -json is omitted, the command output is displayed in tabular format.

Option	Description
-cli.loglevel	Specifies a log level for API output. Legal values for this option are: • DEBUG • INFO • ERROR • WARN • TRACE • FATAL

Filters
Describes the use of filters with MapR CLI commands.

Some MapR CLI commands use *filters*, which let you specify large numbers of nodes or volumes by matching specified values in specified fields rather than by typing each name explicitly.

Filters use the following format:

```
[<field><operator>"<value>"]<and>[<field><operator>"<value>"] ...
```

field	Field on which to filter. The field depends on the command with which the filter is used.
operator	An operator for that field: • == - Exact match • `!=` - Does not match • > - Greater than • < - Less than • >= - Greater than or equal to • <= - Less than or equal to
value	Value on which to filter. Wildcards (using *) are allowed for operators == and `!=`. There is a special value `all` that matches all values.

You can use the wildcard (*) for partial matches. For example, you can display all volumes whose owner is `root` and whose name begins with `test` as follows:

```
maprcli volume list -filter [n=="test*"]and[on=="root"]
```

Note: maprcli commands and REST APIs do not support OR conditions.

Response
Describes the different return responses.

The commands return responses in JSON or in a tabular format. When you run commands from the command line, the response is returned in tabular format unless you specify JSON using the -json option; when you run commands through the REST interface, the response is returned in JSON.

last-updated: Apr 11, 2022

Note: The columns returned by operations such as `get`, `list`, `info`, and so on are not sorted in any particular order.

Success

On a successful call, each command returns the error code zero (OK) and any data requested. When JSON output is specified, the data is returned as an array of records along with the status code and the total number of records. In the tabular format, the data is returned as a sequence of rows, each of which contains the fields in the record separated by tabs.

JSON

```
{
              "status":"OK",
              "total":<number of
records>,
              "data":[
              {
              <record>
              }
              ...
              ]
              }
```

Tabular

```
status
              0
```

Or

```
<heading>   <heading>   <heading> ...
              <field>     <field>
<field> ...
              ...
```

Error

When an error occurs, the command returns the error code and descriptive message.

JSON

```
{
            "status":"ERROR",
            "errors":[
            {
            "id":<error code>,
            "desc":"<command>: <error
message>"
            }
            ]
            }
```

Tabular

```
ERROR (<error code>) -  <command>:
<error message>
```

acerole validate
Verifies given user roles for ACEs exists in the `/opt/mapr/conf/m7_permissions_roles_refimpl.conf` file.

last-updated: Apr 11, 2022

This command returns `true` if role exists in the `/opt/mapr/conf/m7_permissions_roles_refimpl.conf` file and `false` if given role is not in the file. If the `MAPR_ROLES_LIB_ENABLE_TRACE` environment variable is set to `TRUE`, the command returns also the number of users assigned to the specified role and the number of roles in the file.

Syntax

```
/opt/mapr/bin/maprcli acerole validate -role <role to validate>
```

Parameters

Parameter	Description
`role`	The role to validate.

Examples

Verifies whether given role exists when the `MAPR_ROLES_LIB_ENABLE_TRACE` environment variable is not set:

```
$ maprcli acerole validate -role Role_1
maprcli acerole validate command returned : true
```

Verifies whether given role exists when the `MAPR_ROLES_LIB_ENABLE_TRACE` environment variable is set:

```
$ export MAPR_ROLES_LIB_ENABLE_TRACE=TRUE
$ echo $MAPR_ROLES_LIB_ENABLE_TRACE
TRUE
$ maprcli acerole validate -role Role_1
RoleMap: Added user 500 with role 'Role_1'
RoleMap: Added user 1000 with role 'Role_1'
RoleMap: found 2 users and 2 roles.
maprcli acerole validate command returned : true
```

acl
Describes the acl commands used to access control lists (ACLs).

Specifying Permissions

Specify permissions for a user or group with a string that lists the permissions for that user or group. To specify permissions for multiple users or groups, use a string for each, separated by spaces. The format is as follows:

- Users -

  ```
  <user>:<action>[,<action>...][<user>:<action>[,<action...]]
  ```

- Groups -

  ```
  <group>:<action>[,<action>...][<group>:<action>[,<action...]]
  ```

To use the `acl edit` command, you must have full control (`fc`) permission on the cluster or volume for which you are running the command.

The following tables list the permission codes used by the `acl` commands.

Cluster Permission Codes

Permission Code	Allowed Action
login	Log in to the MapR Control System, use the API and command-line interface, read access on cluster and volumes.
ss	Start/stop services.
cv	Create volumes.
a	Administrative access to cluster ACLs. Grants no other permissions.
fc	Full control over the cluster. This enables all cluster-related administrative options with the exception of changing the cluster ACLs.

Volume Permission Codes

Code	Allowed Action
dump	Dump the volume.
restore	Mirror or restore the volume.
m	Modify volume properties, create and delete snapshots.
d	Delete a volume.
a	Administrative access to volume ACLs.
fc	Full control (admin access and permission to change volume ACL).

Security Policy Permission Codes

Code	Allowed Action
a (admin)	View and modify the permissions on a security policy; cannot view or modify the security policy.
fc (full control)	View and modify the security policy, including data access ACEs; cannot view or modify the permissions on a security policy.
r (read)	View all parts of a security policy; cannot modify the security policy.

acl edit
Modifies a specific user's access to a cluster, volume, or security policy.

Permissions Required

The `acl edit` command grants one or more specific volume or cluster permissions to a user. To use the `acl edit` command, you must have administrative (a) permissions on the volume and cluster for which

last-updated: Apr 11, 2022

you are running the command. The permissions are specified as a comma-separated list of permission codes. See acl on page 1528.

Syntax

CLI

```
/opt/mapr/bin/maprcli acl edit
    [ -cluster <cluster name> ]
    [ -group <group> ]
    [ -name <name> ]
    -type cluster|volume|
    [ -user <user> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host:port>/rest/acl/edit?<parameters>`

Parameters

Parameter	Description
cluster	The cluster on which to run the command.
group	Groups and allowed actions for each group. See acl on page 1528. Format: `<group>:<action>[,<action>...] [ <group>:<action>[,<action...]]`. You must specify either a `user` or a `group`.
name	The object name. For a volume, specify the name of the volume in this parameter.
type	The object type (`cluster` or `volume`). When the type is `volume`, specify a volume name using the `name` parameter.
user	Users and allowed actions for each user. See acl on page 1528. Format: `<user>:<action>[,<action>...] [ <user>:<action>[,<action...]].` You must specify either a `user` or a `group`.

Examples

Give the user jsmith dump, restore, and delete permissions for "test-volume":

CLI

```
/opt/mapr/bin/maprcli acl edit -type
volume -name test-volume -user
jsmith:dump,restore,d
```

REST

```
https://10.10.82.22:8443/rest/acl/
edit?
```


last-updated: Apr 11, 2022

```
type=volume&name=test-volume&user=jsmi
th%3Adump,restore,d
```

acl set
Modifies the Access Control List (ACL) for a cluster, volume, or security policy.

The `acl set` command specifies the *ACL* for a cluster or volume. Any previous permissions are overwritten by the new values, and any permissions omitted are removed. To use the `acl set` command, you must have administrative (a) permissions on the volume and cluster for which you are running the command. The ACL permissions are specified as a comma-separated list of permission codes. See acl on page 1528. You must specify either a `user` or a `group`. When the `type` is `volume`, you must specify a volume name using the `name` parameter.

The `acl set` command removes any previous *ACL* values. To preserve some of the permissions, you should either use the `acl edit` command instead of `acl set`, or use `acl show` to list the values before overwriting them.

Syntax

CLI

```
/opt/mapr/bin/maprcli acl set
    [ -cluster <cluster name> ]
    [ -group <group> ]
    [ -name <name> ]
    -type cluster|volume|
    [ -user <user> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/acl/set?<parameters>`

Parameters

Parameter	Description
cluster	The cluster on which to run the command.
group	Groups and allowed actions for each group. See acl on page 1528. Format: `<group>:<action>[,<action>...] [ <group>:<action>[,<action...]]`
name	The object name. For a volume, specify the name of the volume in this parameter.
type	The object type (`cluster` or `volume`).
user	Users and allowed actions for each user. See acl on page 1528. Format: `<user>:<action>[,<action>...] [ <user>:<action>[,<action...]]`

last-updated: Apr 11, 2022

Examples

Give the user `root` full control of the `my.cluster.com` cluster and remove all permissions for all other users:

CLI

```
/opt/mapr/bin/maprcli acl set -type
cluster -user user10:fc
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/acl/set?
type=cluster&user=user10%3Afc' --user
mapr:mapr
{"timestamp":1525462091620,"timeofday"
:"2018-05-04 12:28:11.620 GMT-0700
PM","status":"OK","total":0,"data":[]}
```

Usage Example

```
# /opt/mapr/bin/maprcli acl show -type cluster
Allowed actions           Principal
[login, ss, cv, a, fc, cp] User mapr
[login, ss, cv, a, fc, cp] User root
[login, cp]  User fuser1

# /opt/mapr/bin/maprcli acl set -type cluster -cluster my.cluster.com -user
root:fc
# /opt/mapr/bin/maprcli acl show -type cluster
Principal  Allowed actions
User root  [login, ss, cv, a, fc, cp]
```

Warning: Notice that the specified permissions have overwritten the existing *ACL*.

Give multiple users specific permissions for the `egVol1` volume and remove all permissions for all other users:

CLI

```
# /opt/mapr/bin/maprcli acl
set -type volume -name
egVol1 -user m7user5:dump,restore,m
m7user4:fc -json
{
    "timestamp":1525462647371,
    "timeofday":"2018-05-04
12:37:27.371 GMT-0700 PM",
    "status":"OK",
    "total":0,
    "data":[

    ]
}
```

REST

```
# curl -k -X POST 'https://
10.10.82.24:8443/rest/acl/set?
type=volume&name=egVol1&user=m7user5%3
Adump,restore,m%20m7user4%3Afc' --user
 mapr:mapr
{"timestamp":1525463080941,"timeofday"
:"2018-05-04 12:44:40.941 GMT-0700
PM","status":"OK","total":0,"data":[]}
```


last-updated: Apr 11, 2022

acl show
Displays the ACL associated with an object (cluster or a volume).

Syntax

An ACL contains the list of users who can perform specific actions.

CLI

```
/opt/mapr/bin/maprcli acl show
     -type object type [cluster|
volume|securitypolicy]
    [ -name name ]
    [ -cluster cluster name ]
    [ -user userName whose ACL is
queried ]
    [ -group groupName whose ACL is
queried ]
    [ -output output format short|
long|terse (default short). default:
short ]
    [ -perm list of available
permissions Parameter takes no
value  ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/acl/show?<parameters>`

Parameters

Parameter	Description
cluster	The name of the cluster on which to run the command.
group	The group for which to display permissions.
name	The cluster or volume name.
output	The output format: • long • short • terse The default format is `short`.
perm	When you specify this option, `acl show` displays the permissions available for the object type specified in the `type` parameter.
type	Value can be one of `cluster` or `volume`. If type is `volume`, the volume `name` is required.
user	The user for whom to display permissions.

last-updated: Apr 11, 2022

Output

The actions that each user or group is allowed to perform on the cluster or the specified volume. For information about each allowed action, see acl on page 1528.

```
Principal   Allowed actions
User root   [login, ss, cv, a, fc]
Group root  [login, ss, cv, a, fc]
All users   [login]
```

Examples

Show the ACL for the cluster:

CLI

```
# /opt/mapr/bin/maprcli acl
show -type cluster -json
{
    "timestamp":1555494572399,
    "timeofday":"2019-04-17
02:49:32.399 GMT-0700 AM",
    "status":"OK",
    "total":2,
    "data":[
        {
            "Principal":"User mapr",
            "Allowed
actions":"[login, ss, cv, a, fc, cp]"
        },
        {
            "Principal":"User root",
            "Allowed
actions":"[login, ss, cv, a, fc, cp]"
        }
    ]
}
```

REST

```
curl -u mapr:mapr -X GET -k "https://
host:8443/rest/acl/show?type=cluster"
{"timestamp":1555494852652,"timeofday"
:"2019-04-17 02:54:12.652 GMT-0700
AM","status":"OK","total":2,"data":
[{"Principal":"User mapr","Allowed
actions":"[login, ss, cv, a, fc,
cp]"},{"Principal":"User
root","Allowed actions":"[login, ss,
cv, a, fc, cp]"}]}
```

Show the ACL for "test-volume":

CLI

```
# /opt/mapr/bin/maprcli acl
show -type volume -name sampleVol1
Allowed actions
Principal
[dump, restore, m, a, d, fc]  User
mapr
[dump, restore, m, d, fc]     User
foo
[dump, restore, a]            User
```

```
bar
[m, d]                          User abc
```

REST

```
# curl -u mapr:mapr -X GET -k
"https://host:8443/rest/acl/show?
type=volume&name=sampleVol1"
{"timestamp":1525461068100,"timeofday"
:"2018-05-04 12:11:08.100 GMT-0700
PM","status":"OK","total":4,"data":
[{"Principal":"User mapr","Allowed
actions":"[dump, restore, m, a, d,
fc]"},{"Principal":"User
foo","Allowed actions":"[dump,
restore, m, d, fc]"},
{"Principal":"User bar","Allowed
actions":"[dump, restore, a]"},
{"Principal":"User abc","Allowed
actions":"[m, d]"}]}
```

Show the permissions that can be set on a cluster:

CLI

```
# /opt/mapr/bin/maprcli acl
show -type cluster -perm
Permissions
Description
login        Login
access
ss           Start/stop services in
the cluster
cv           Create
volumes
a
Administrator
fc           Full
control
cp           Create security policies
```

REST

```
# curl -u mapr:mapr -X GET -k
"https://host:8443/rest/acl/show?
type=cluster&perm"
{"timestamp":1555497261931,"timeofday"
:"2019-04-17 03:34:21.931 GMT-0700
AM","status":"OK","total":6,"data":
[{"Permissions":"login","Description":
"Login access"},
{"Permissions":"ss","Description":"Sta
rt/stop services in the cluster"},
{"Permissions":"cv","Description":"Cre
ate volumes"},
{"Permissions":"a","Description":"Admi
nistrator"},
{"Permissions":"fc","Description":"Ful
l control"},
{"Permissions":"cp","Description":"Cre
ate security policies"}]}
```

alarm
Describes the alarm commands that perform functions related to system alarms.

Alarm Notification Fields

The following fields specify the configuration of alarm notifications.

Field	Description
alarm	The named alarm.
individual	Specifies whether individual alarm notifications are sent to the default email address for the alarm type: • 0 - do not send notifications to the default email address for the alarm type • 1 - send notifications to the default email address for the alarm type
email	A custom email address for notifications about this alarm type. If specified, alarm notifications are sent to this email address, regardless of whether they are sent to the default email address.

Alarm Types

See Alarms Reference.

Alarm History

To see a history of alarms that have been raised, look at the file `/opt/mapr/logs/cldb.log` on the master CLDB node. Example:

```
grep ALARM /opt/mapr/logs/cldb.log
```

alarm clear
Clears one or more alarms. Permissions required: `fc` or `a`.

Syntax

CLI

```
maprcli alarm clear
    -alarm <alarm>
    [ -cluster <cluster> ]
    [ -entity entity (hostname OR
volume name OR Ae name) ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/alarm/clear?<parameters>`

last-updated: Apr 11, 2022

Parameters

Parameter	Description
alarm	The named alarm to clear. See Alarm Types.
cluster	The cluster on which to run the command.
entity	The entity on which to clear the alarm.

Examples
Clear a specific alarm:

CLI

```
maprcli alarm clear -alarm
NODE_ALARM_DEBUG_LOGGING
```

REST

```
https://abc.sj.us:8443/rest/alarm/
clear?alarm=NODE_ALARM_DEBUG_LOGGING
```

alarm clearmulti
Clears all alarm occurences of specified multiple alarm types. Permissions required: fc or a.

Syntax

CLI

```
maprcli alarm clearmulti
   [ -cluster cluster_name ]
   -alarm alarm[:entity][:aetype]
<comma seperated alarms>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/alarm/clearmulti?<parameters>`

Parameters

Parameter	Description
alarm	Comma seperated list of alarm types to clear.
cluster	The cluster on which to run the command. Default is the current cluster.

Examples
Clear a specific alarm:

CLI

```
maprcli alarm clear -alarm
NODE_ALARM_DEBUG_LOGGING,VOLUME_ALARM_
COMPACTION_FAILURE
```


last-updated: Apr 11, 2022

REST

```
https://abc.sj.us:8443/rest/alarm/
clear?
alarm=NODE_ALARM_DEBUG_LOGGING,VOLUME_
ALARM_COMPACTION_FAILURE
```

alarm clearall
Clears all alarms. Permissions required: `fc` or `a`.

Syntax

CLI

```
maprcli alarm clearall
    [ -cluster <cluster> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/alarm/clearall?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.

Examples
Clear all alarms:

CLI

```
maprcli alarm clearall
```

REST

```
https://r1n1.sj.us:8443/rest/alarm/
clearall
```

alarm config load
Displays the configuration of alarm notifications. Permission required: `login`

Syntax

CLI

```
maprcli alarm config load
    [ -cluster <cluster> ]
    [ -output terse|verbose ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/alarm/config/load?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`output`	Whether the output should be terse or verbose.

Output

A list of configuration values for alarm notifications.

Output Fields

See Alarm Notification Fields.

Sample output

```
alarm                                  individual  email
CLUSTER_ALARM_UPGRADE_IN_PROGRESS      1
CLUSTER_ALARM_UNASSIGNED_VIRTUAL_IPS   1
CLUSTER_ALARM_LICENSE_NEAR_EXPIRATION  1
CLUSTER_ALARM_LICENSE_EXPIRED          1
CLUSTER_ALARM_CLUSTER_ALMOST_FULL      1
CLUSTER_ALARM_CLUSTER_FULL             1
CLUSTER_ALARM_LICENSE_MAXNODES_EXCEEDED 1
CLUSTER_ALARM_NEW_FEATURES_DISABLED    1
VOLUME_ALARM_SNAPSHOT_FAILURE          1
VOLUME_ALARM_MIRROR_FAILURE            1
VOLUME_ALARM_DATA_UNDER_REPLICATED     1
VOLUME_ALARM_DATA_UNAVAILABLE          1
VOLUME_ALARM_ADVISORY_QUOTA_EXCEEDED   1
VOLUME_ALARM_QUOTA_EXCEEDED            1
VOLUME_ALARM_NO_NODES_IN_TOPOLOGY      1
VOLUME_ALARM_TOPOLOGY_ALMOST_FULL      1
VOLUME_ALARM_TOPOLOGY_FULL             1
VOLUME_ALARM_INODES_EXCEEDED           1
NODE_ALARM_DEBUG_LOGGING               1
NODE_ALARM_DISK_FAILURE                1
NODE_ALARM_VERSION_MISMATCH            1
NODE_ALARM_TIME_SKEW                   1
NODE_ALARM_SERVICE_CLDB_DOWN           1
NODE_ALARM_SERVICE_FILESERVER_DOWN     1
NODE_ALARM_SERVICE_JT_DOWN             1
NODE_ALARM_SERVICE_TT_DOWN             1
NODE_ALARM_SERVICE_HBMASTER_DOWN       1
NODE_ALARM_SERVICE_HBREGION_DOWN       1
NODE_ALARM_SERVICE_NFS_DOWN            1
NODE_ALARM_SERVICE_WEBSERVER_DOWN      1
NODE_ALARM_SERVICE_HOSTSTATS_DOWN      0
NODE_ALARM_ROOT_PARTITION_FULL         1
NODE_ALARM_OPT_MAPR_FULL               1
NODE_ALARM_CORE_PRESENT                1
NODE_ALARM_HIGH_MFS_MEMORY             1
NODE_ALARM_PAM_MISCONFIGURED           1
NODE_ALARM_TT_LOCALDIR_FULL            1
NODE_ALARM_NO_HEARTBEAT                1
NODE_ALARM_MAPRUSER_MISMATCH           1
NODE_ALARM_DUPLICATE_HOSTID            1
NODE_ALARM_METRICS_WRITE_PROBLEM       1
NODE_ALARM_TOO_MANY_CONTAINERS         1
NODE_ALARM_M7_CONFIG_MISMATCH          1
NODE_ALARM_INCORRECT_TOPOLOGY_ALARM    1
AE_ALARM_AEADVISORY_QUOTA_EXCEEDED     1
```


last-updated: Apr 11, 2022

```
AE_ALARM_AEQUOTA_EXCEEDED                  1
NODE_ALARM_SERVICE_HUE_DOWN                1
NODE_ALARM_SERVICE_HTTPFS_DOWN             1
NODE_ALARM_SERVICE_BEESWAX_DOWN            1
NODE_ALARM_SERVICE_HIVEMETA_DOWN           1
NODE_ALARM_SERVICE_HS2_DOWN                1
NODE_ALARM_SERVICE_OOZIE_DOWN              1
NODE_ALARM_HB_PROCESSING_SLOW              1
NODE_ALARM_SERVICE_ELASTICSEARCH_DOWN      1
NODE_ALARM_SERVICE_ELASTICSEARCH_EXCP      1
VOLUME_ALARM_DATA_CONTAINERS_NONLOCAL      1
CLUSTER_ALARM_CLDB_HEAPSIZE                1
NODE_ALARM_SERVICE_NODEMANAGER_DOWN        1
VOLUME_ALARM_TABLE_REPL_LAG_HIGH           1
NODE_ALARM_MEMORY_SWAPPING                 1
NODE_ALARM_SERVICE_DRILL-BITS_DOWN         1
VOLUME_ALARM_TABLE_REPL_ERROR              1
NODE_ALARM_MEMORY_ALLOCATION_EXCEEDED      1
VOLUME_ALARM_TABLE_REPL_ASYNC              1
NODE_ALARM_SERVICE_RESOURCEMANAGER_DOWN    1
NODE_ALARM_SERVICE_HISTORYSERVER_DOWN      1
```

Examples

Display the alarm notification configuration:

CLI

```
maprcli alarm config load
```

REST

```
https://r1n1.sj.us:8443/rest/alarm/
config/load
```

alarm config save

Sets notification preferences for alarms. Permissions required: `fc` or `a`.

Alarm notifications can be sent to the default email address and a specific email address for each named alarm. If `individual` is set to `1` for a specific alarm, then notifications for that alarm are sent to the default email address for the alarm type. If a custom email address is provided, notifications are sent there regardless of whether they are also sent to the default email address.

Syntax

CLI

```
maprcli alarm config save
    [ -cluster <cluster> ]
    -values <values>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/alarm/config/save?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`values`	A comma-separated list of configuration values for one or more alarms, in the following format: `<alarm>,<individual>,<email>` See Alarm Notification Fields.

Examples

Send alert emails for the AE_ALARM_AEQUOTA_EXCEEDED alarm to the default email address and a custom email address:

CLI

```
maprcli alarm config save -values
"AE_ALARM_AEQUOTA_EXCEEDED,1,test@exam
ple.com"
```

REST

```
https://r1n1.sj.us:8443/rest/alarm/
config/save?
values=AE_ALARM_AEQUOTA_EXCEEDED,1,tes
t@example.com
```

alarm group

Alarm groups are groups of alarms for which email addresses of users/groups can be set (to send alert to when an alarm is raised) and removed.

Permissions required: fc or a

alarm group addalarms

Add alarms to a group.

Syntax

CLI

```
maprcli alarm group addalarms
    -groupname <group name>
    -alarms <alarm name>
```

REST API N/A

Parameters

Parameter	Description
groupname	The name of the alarm group to create. If necessary, run listGroup to retrieve the list of alarm groups.
alarms	The comma-separated list of alarms to add to the group.

last-updated: Apr 11, 2022

Examples

Add alarms to alarm group:

```
maprcli alarm group addAlarms
    -groupname cldb.alarm.group.info
    -alarms NODE_ALARM_HB_PROCESSING_SLOW,CLUSTER_ALARM_CLUSTER_ALMOST_FULL
```

alarm group addemails

Adds the email addresses of users/groups to send alert to when an alarm is raised.

Syntax

CLI

```
maprcli alarm group addEmails
    [ -cluster <cluster_name> ]
    -groupname <group name>
    -emails <email addresses>
```

REST API

N/A

Parameters

Parameter	Description
cluster	The cluster on which to run the command.
groupname	The name of the alarm group to add the email addresses to. When an alarm in the group is raised, an alert email will be sent to the users/groups. If necessary, run listgroup to retrieve the list of alarm groups.
emails	The comma-separated list of email addresses of users to send alert emails to when an alarm is raised.

Examples

Add email addresses to `cldb.alarm.group.error` group.

```
maprcli alarm group addEmails -groupname cldb.alarm.group.error -emails
abc@gmail.com,xyz@gmail.com
```

alarm group deletealarms

Delete alarms in an alarm group.

Syntax

CLI

```
maprcli alarm group deletealarms
    -groupname <group name>
    -alarms <alarm name>
```

REST API

N/A

Parameters

Parameter	Description
groupname	The name of the alarm group to remove alarms from.

Parameter	Description
alarms	The comma-separated list of alarms to remove from the group.

Examples

Delete NODE_ALARM_HB_PROCESSING_SLOW and CLUSTER_ALARM_CLUSTER_ALMOST_FULL alarms in the `cldb.alarm.group.info` group:

```
maprcli alarm group deleteAlarms
    -groupname cldb.alarm.group.info
    -alarms NODE_ALARM_HB_PROCESSING_SLOW,CLUSTER_ALARM_CLUSTER_ALMOST_FULL
```

alarm group deleteemails

Deletes the email addresses of users/groups.

Syntax

CLI

```
maprcli alarm group deleteEmails
    [ -cluster <cluster_name> ]
    -groupname <group name>
    -emails <email addresses>
```

REST API N/A

Parameters

Parameter	Description
cluster	The cluster on which to run the command.
groupname	The name of the alarm group to remove the email addresses from. If necessary, run listgroup to retrieve the list of alarm groups.
emails	The comma-separated list of email addresses of users/groups to remove.

Examples

Delete the given emails associated with the `cldb.alarm.group.error` group:

```
maprcli alarm group deleteEmails -groupname cldb.alarm.group.error -emails
xyz@gmail.com,abc@gmail.com
```

alarm group listgroup

Lists the alarm groups.

Note: The three dots in the output indicate multiple alarms in the group. Use `-json` to format the output.

Syntax

CLI

```
maprcli alarm group listGroup
    [ -cluster <cluster_name> ]
```


last-updated: Apr 11, 2022

```
    [ -start ]
    [ -limit ]
    [ -output <verbose> ]
```

REST API N/A

Parameters

Parameter	Description
cluster	The cluster on which to run the command.
start	The list offset at which to start. Default: 0
limit	The number of records to retrieve. Default value is 2147483647.
output	The output format: • terse • verbose Default value is verbose.

Examples

Return the list of alarms and associated email addresses.

```
# maprcli alarm group listGroup
alarm name emails group name
... abc@mapr.com cldb.alarm.group.error
... cldb.alarm.group.info
... cldb.alarm.group.warn
```

Return the list of alarms and associated email addresses in JSON format.

```
root@saradaqa4:~# maprcli alarm group listGroup -json
{
    "timestamp":1495018857252,
     "timeofday":"2017-05-17 11:00:57.252 GMT+0000",
     "status":"OK",
     "total":3,
     "data":[
  {
     "group name":"cldb.alarm.group.error",
     "emails":"abc@mapr.com",
     "alarm name":[
        "NODE_ALARM_DISK_FAILURE",
        "CLUSTER_ALARM_LICENSE_EXPIRED",
        "CLUSTER_ALARM_CLUSTER_FULL",
        "NODE_ALARM_NO_DISK_ATTACHED",
        "VOLUME_ALARM_SNAPSHOT_FAILURE",
        "VOLUME_ALARM_MIRROR_FAILURE",
        "NODE_ALARM_CORE_PRESENT",
        "NODE_ALARM_HIGH_MFS_MEMORY",
        "VOLUME_ALARM_DATA_UNAVAILABLE",
        "NODE_ALARM_MAPRUSER_MISMATCH",
        "VOLUME_ALARM_NO_NODES_IN_TOPOLOGY",
        "VOLUME_ALARM_TABLE_REPL_ERROR",
        "NODE_ALARM_NO_HEARTBEAT",
        "VOLUME_ALARM_QUOTA_EXCEEDED",
```

```
        "VOLUME_ALARM_TOPOLOGY_FULL",
        "NODE_ALARM_PAM_MISCONFIGURED",
        "NODE_ALARM_MEMORY_ALLOCATION_EXCEEDED",
        "NODE_ALARM_TOO_MANY_CONTAINERS",
        "NODE_ALARM_NUM_INSTANCES_MISMATCH",
        "AE_ALARM_AEQUOTA_EXCEEDED"
    ]
}
```

alarm list
Lists alarms in the system. Permissions required: `login`.

You can list all alarms, alarms by type (Cluster, Node or Volume), or alarms on a particular node or volume. To retrieve a count of all alarm types, pass `1` in the `summary` parameter. You can specify the alarms to return by filtering on type and entity. Use `start` and `limit` to retrieve only a specified window of data.

Syntax

CLI

```
maprcli alarm list
    [ -alarm alarm name ]
    [ -all list all raised alarms
including the ones which are muted
Parameter takes no value  ]
    [ -cluster cluster_name ]
    [ -entity entity (hostname OR
volume name OR Ae name) ]
   [ -entitylimit entitylimit ]
    [ -filter none. default: none ]


    [ -history list cleared up alarms
only Parameter takes no value  ]
    [ -limit limit. default:
2147483647 ]
    [ -muted list alarms configured
to be mute Parameter takes no value  ]
    [ -output output. default:
verbose ]
    [ -sortby <alarmname|
alarmdescription|alarmtype|alarmstate|
alarmraised|alarmcleared|alarmentity|
alarmmutetime|alarmmuteupto|
alarmmuteduration|alarmgroups> ]

    [ -start start. default: 0 ]
    [ -summary summary ]

    [ -type type (CLUSTER
OR NODE OR VOLUME OR AE) ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/alarm/list?<parameters>`

Parameters

Parameter	Description
`alarm`	The alarm name for which to return information.
`all`	The list of all raised and muted alarms.
`cluster`	The cluster on which to list alarms.
`entity`	The name of the cluster, node, volume, user, or group to check for alarms.
`entitylimit`	The number of alarm occurrences to return per alarm type. For example, if there are 10 alarms of type VOLUME_ALARM_COMPACTION_FAILURE and the entity limit is set to 4, then only the four latest alarm occurrences of type VOLUME_ALARM_COMPACTION_FAILURE are listed.
`filter`	A filter specifying alarms to list. See Filters for more information. Default: `none`
`history`	The list of all alarms cleared in the last 30 days. Muted alarms are not displayed in the output.
`limit`	The number of records to retrieve. Default: 2147483647
`muted`	The list of alarms that are muted.
`output`	Indicates whether the output should be `terse` or `verbose`. Default: `verbose`
`sortby`	Specifies one of the following attributes by which to sort the list of alarms: `alarmname`, `alarmdescripton`, `alarmtype`, `alarmstate`, `alarmraised`, `alarmcleared`, `alarmentity`, `alarmmutetime`, `alarmmuteupto`, `alarmmuteduration`, `alarmgroups`. By default, the list of alarms is sorted by `alarmtype`.
`start`	The list offset at which to start. Default: `0`
`summary`	Specifies the type of data to return: • `1` = count by alarm type • `0` = List of alarms Default: false (0)
`type`	The entity type: • `cluster` • `node` • `volume` • `ae`

Output

Information about one or more named alarms on the cluster, or for a specified node, volume, user, or group.

Output Fields

last-updated: Apr 11, 2022

Field	Description
alarm state	State of the alarm: • 0 = Clear • 1 = Raised
description	A description of the condition that raised the alarm.
entity	The name of the volume, node, user, or group.
alarm name	The name of the alarm.
alarm statechange time	The date and time when the alarm was most recently raised.

Sample Output

```
alarm state
description
entity                                alarm name
alarm statechange time
1            Volume desired replication is 1, current
replication is 0                mapr.qa-node173.qa.prv.local.logs
VOLUME_ALARM_DATA_UNDER_REPLICATED  1296707707872
1            Volume data
unavailable
mapr.qa-node173.qa.prv.local.logs    VOLUME_ALARM_DATA_UNAVAILABLE
1296707707871
1            Volume desired replication is 1, current
replication is 0                mapr.qa-node235.qa.prv.local.mapred
VOLUME_ALARM_DATA_UNDER_REPLICATED  1296708283355
1            Volume data
unavailable
mapr.qa-node235.qa.prv.local.mapred  VOLUME_ALARM_DATA_UNAVAILABLE
1296708283099
1            Volume desired replication is 1, current
replication is 0                mapr.qa-node175.qa.prv.local.logs
VOLUME_ALARM_DATA_UNDER_REPLICATED  1296706343256
```

Examples

List a summary of all alarms

CLI

```
maprcli alarm list -summary 1
```

REST

```
https://r1n1.sj.us:8443/rest/alarm/
list?summary=1
```

List cluster alarms

CLI

```
maprcli alarm list -type cluster
```

REST

```
https://r1n1.sj.us:8443/rest/alarm/
list?type=cluster
```

List all muted alarms

CLI

```
# maprcli alarm list -muted
mute duration muted time    mute
upto     entity  alarm name
15 mins       1495702964190
1495703864190 CLUSTER
CLUSTER_ALARM_LICENSE_NEAR_EXPIRATION
15 mins       1495702964192
1495703864192 vol3
VOLUME_ALARM_DATA_UNDER_REPLICATED
10 mins       1495702899201
1495703499201 vol2
VOLUME_ALARM_DATA_UNDER_REPLICATED
15 mins       1495702964188
1495703864188 vol1
VOLUME_ALARM_DATA_UNDER_REPLICATED
```

REST API

```
https://r1n1.sj.us:8443/rest/alarm/
list?muted
```

alarm mute
Mutes an active alarm for the specified amount of time.

Syntax

CLI

```
maprcli alarm mute
     -alarm alarm[:entity][:aetype]
[:mute_duration] <comma seperated
alarms>
     [ -cluster cluster_name ]
     [ -muteminutes <mute_period> ]
```

REST API

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/alarm/mute?<parameters>`

Parameters

Parameter	Description
`alarm`	The comma-separated list of the active alarms to mute. To mute an active alarm associated with an entity (such as volume, node, etc.) or type of accountable entity, you must also specify the name of the entity or the type of accountable entity separated by a colon (`:`).
`cluster`	The cluster on which to run the command.

Parameter	Description
`muteminutes`	The amount of time, in minutes, to mute the alarm. This can be specified separately or specified after the colon (`:<mute_period>`) immediately following the alarm name. For example, if multiple alarms are being muted for different periods of time, use colon (`:<mute_period>`) to specify the mute period for each alarm. Specifying this separately is optional if colon (`:<mute_period>`) is used to specify the amount of time to mute an alarm. For example, when specifying a list of alarms to mute, use this parameter to specify the amount of time to mute the alarm for which no time period is specified using colon (`:<mute_period>`). **Note:** Either the value for this parameter or colon (`:<mute_period>`) is required to specify the amount of time for which to mute the alarm.

Examples

Mute an active alarm for 10 minutes using one of the following:

```
maprcli alarm mute -alarm CLUSTER_ALARM_LICENSE_NEAR_EXPIRATION:10
```

```
maprcli alarm mute -alarm
CLUSTER_ALARM_LICENSE_NEAR_EXPIRATION -muteminutes 10
```

Mute an active volume alarm on volume1 for 10 minutes using one of the following:

```
maprcli alarm mute -alarm VOLUME_ALARM_DATA_UNDER_REPLICATED:volume1:10
```

```
maprcli alarm mute -alarm
VOLUME_ALARM_DATA_UNDER_REPLICATED:volume1 -muteminutes 10
```

Mute active volume alarm on volume1 for 10 minutes, active cluster alarm for 20 minutes, and active volume alarm on volume2 for 30 minutes using one of the following:

```
maprcli alarm mute -alarm
        VOLUME_ALARM_DATA_UNDER_REPLICATED:volume1:10,
        CLUSTER_ALARM_LICENSE_NEAR_EXPIRATION:20,
        VOLUME_ALARM_DATA_UNDER_REPLICATED:volume2:30
```

```
maprcli alarm mute -alarm
        VOLUME_ALARM_DATA_UNDER_REPLICATED:volume1:10,
        CLUSTER_ALARM_LICENSE_NEAR_EXPIRATION:20,
        VOLUME_ALARM_DATA_UNDER_REPLICATED:volume2
        -muteminutes 30
```

alarm names
Displays a list of alarm names. Permissions required: `login`

Syntax

CLI

```
maprcli alarm names
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/alarm/names`

Output

Attention: The list of alarms depends on the ecosystem components installed. Your output may vary depending on the ecosystem components that you have installed.

```
CLUSTER_ALARM_UPGRADE_IN_PROGRESS
CLUSTER_ALARM_UNASSIGNED_VIRTUAL_IPS
CLUSTER_ALARM_LICENSE_NEAR_EXPIRATION
CLUSTER_ALARM_LICENSE_EXPIRED
CLUSTER_ALARM_CLUSTER_ALMOST_FULL
CLUSTER_ALARM_CLUSTER_FULL
CLUSTER_ALARM_LICENSE_MAXNODES_EXCEEDED
CLUSTER_ALARM_NEW_FEATURES_DISABLED
CLUSTER_ALARM_TOO_MANY_SNAPSHOT_CONTAINERS
VOLUME_ALARM_SNAPSHOT_FAILURE
VOLUME_ALARM_MIRROR_FAILURE
VOLUME_ALARM_DATA_UNDER_REPLICATED
VOLUME_ALARM_DATA_UNAVAILABLE
VOLUME_ALARM_ADVISORY_QUOTA_EXCEEDED
VOLUME_ALARM_QUOTA_EXCEEDED
VOLUME_ALARM_NO_NODES_IN_TOPOLOGY
VOLUME_ALARM_TOPOLOGY_ALMOST_FULL
VOLUME_ALARM_TOPOLOGY_FULL
VOLUME_ALARM_INODES_EXCEEDED
VOLUME_ALARM_BECOME_MASTER_STUCK
VOLUME_ALARM_OFFLOAD_RECALL_FAILURE
VOLUME_ALARM_COMPACTION_FAILURE
NODE_ALARM_DEBUG_LOGGING
NODE_ALARM_DISK_FAILURE
NODE_ALARM_VERSION_MISMATCH
NODE_ALARM_TIME_SKEW
NODE_ALARM_SERVICE_CLDB_DOWN
NODE_ALARM_SERVICE_FILESERVER_DOWN
NODE_ALARM_SERVICE_JT_DOWN
NODE_ALARM_SERVICE_TT_DOWN
NODE_ALARM_SERVICE_HBMASTER_DOWN
NODE_ALARM_SERVICE_HBREGION_DOWN
NODE_ALARM_SERVICE_NFS_DOWN
NODE_ALARM_SERVICE_WEBSERVER_DOWN
NODE_ALARM_SERVICE_HOSTSTATS_DOWN
NODE_ALARM_ROOT_PARTITION_FULL
NODE_ALARM_OPT_MAPR_FULL
NODE_ALARM_CORE_PRESENT
NODE_ALARM_HIGH_MFS_MEMORY
NODE_ALARM_PAM_MISCONFIGURED
NODE_ALARM_TT_LOCALDIR_FULL
NODE_ALARM_NO_HEARTBEAT
NODE_ALARM_MAPRUSER_MISMATCH
NODE_ALARM_DUPLICATE_HOSTID
```

```
NODE_ALARM_METRICS_WRITE_PROBLEM
NODE_ALARM_TOO_MANY_CONTAINERS
NODE_ALARM_INCORRECT_TOPOLOGY_ALARM
NODE_ALARM_HIGH_MASTGATEWAY_MEMORY
NODE_ALARM_HIGH_NFS4_MEMORY
AE_ALARM_AEADVISORY_QUOTA_EXCEEDED
AE_ALARM_AEQUOTA_EXCEEDED
NODE_ALARM_SERVICE_HUE_DOWN
NODE_ALARM_SERVICE_HTTPFS_DOWN
NODE_ALARM_SERVICE_BEESWAX_DOWN
NODE_ALARM_SERVICE_HIVEMETA_DOWN
NODE_ALARM_SERVICE_HS2_DOWN
NODE_ALARM_SERVICE_OOZIE_DOWN
NODE_ALARM_HB_PROCESSING_SLOW
NODE_ALARM_SERVICE_ELASTICSEARCH_DOWN
NODE_ALARM_SERVICE_ELASTICSEARCH_EXCP
VOLUME_ALARM_DATA_CONTAINERS_NONLOCAL
VOLUME_ALARM_CANNOT_MIRROR
VOLUME_ALARM_DEGRADED_EC_STRIPES
VOLUME_ALARM_CRITICALLY_DEGRADED_EC_STRIPES
VOLUME_ALARM_EC_DATA_UNAVAILABLE
CLUSTER_ALARM_CLDB_HEAPSIZE
CLUSTER_ALARM_DARE_INCOMPATIBLE
CLUSTER_ALARM_DARE_COPY_MASTER_KEY
NODE_ALARM_SERVICE_NFS4_DOWN
NODE_ALARM_SERVICE_NODEMANAGER_DOWN
NODE_ALARM_NUM_INSTANCES_MISMATCH
VOLUME_ALARM_TABLE_INDEX_LAG_HIGH
VOLUME_ALARM_TABLE_INDEX_ENCODING_ERROR
NODE_ALARM_TINY_BUCKET_FLUSH
NODE_ALARM_NO_DISK_ATTACHED
NODE_ALARM_SERVICE_RESOURCEMANAGER_DOWN
VOLUME_ALARM_TABLE_LARGE_ROW_WARNING
NODE_ALARM_SERVICE_MASTGATEWAY_DOWN
VOLUME_ALARM_TABLE_REPL_ERROR
NODE_ALARM_MEMORY_ALLOCATION_EXCEEDED
VOLUME_ALARM_TABLE_REPL_LAG_HIGH
VOLUME_ALARM_TABLE_REPL_ASYNC
NODE_ALARM_SERVICE_HISTORYSERVER_DOWN
VOLUME_ALARM_TABLE_INDEX_ERROR
NODE_ALARM_MEMORY_SWAPPING
NODE_ALARM_SERVICE_APISERVER_DOWN
```

Examples

Display all alarm names:

CLI

```
maprcli alarm names
```

REST

```
https://r1n1.sj.us:8443/rest/alarm/
names
```

alarm raise
Raises a specified alarm or alarms. Permissions required: `fc` or `a`.

last-updated: Apr 11, 2022

Syntax

CLI

```
maprcli alarm raise
     -alarm <alarm>
    [ -cluster <cluster> ]
    [ -description <description> ]
    [ -entity <cluster, entity, host,
node, or volume> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/alarm/raise?<parameters>`

Parameters

Parameter	Description
`alarm`	The alarm type to raise. See Alarm Types.
`cluster`	The cluster on which to run the command.
`description`	A brief description.
`entity`	The entity on which to raise alarms.

Examples

Raise a specific alarm:

CLI

```
maprcli alarm raise -alarm
NODE_ALARM_DEBUG_LOGGING
```

REST

```
https://r1n1.sj.us:8443/rest/alarm/
raise?alarm=NODE_ALARM_DEBUG_LOGGING
```

alarm unmute
Unmute a muted alarm.

Syntax

CLI

```
maprcli alarm unmute
   -alarm [<alarm
name>[:<entity>[:<aetype>]]]+
   [ -cluster cluster_name ]
```

REST API

Request Type	POST

Request URL	`http[s]://<host>:<port>/rest/alarm/unmute?<parameters>`

Parameters

Parameter	Description
`alarm`	The comma-separated list of the muted alarms to unmute. To unmute an alarm associated with an entity (such as volume, node, etc.) or type of accountable entity, you can specify also the name of the entity or the type of accountable entity separated by a colon (:).
`cluster`	The name of the cluster on which to run the command.

audit
Describes commands used to audit operations related to cluster management and data access.

audit cluster
Enables and disables auditing of operations that are related to the administration of a MapR cluster.

Only the `mapr` user for the cluster can run this command. For more information about the `mapr` user, see Managing Users and Groups.

For information about auditing cluster-administration operations, see Auditing of Activity Related to Cluster Administration.

Syntax

CLI

```
maprcli audit cluster
   -enabled <true | false>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/audit/cluster?<parameters>`

Parameters

Parameter	Description
`enabled`	The value `true` enables auditing, the value `false` disables it.

audit data
Enables and disables auditing of filesystem and table operations.

For a list of these operations, see Auditing of Filesystem Operations and Table Operations.

All administrative users for the cluster can run this command. For more information, see Managing Users and Groups on page 752.

Syntax

CLI

```
maprcli audit data
[ -cluster <cluster name> ]
[ -enabled <true | false> ]
[ -maxsize <GB> ]
[ -retention <number of days> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/audit/data?<parameters>`

Parameters

Parameter	Description
`cluster`	The path and name of a remote MapR cluster.
`enabled`	The value `true` enables auditing, the value `false` disables it.
`maxsize`	The size in GB at which an alarm is sent to the dashboard in the MapR Control Service. The alarm is to notify the cluster administrator that the audit log is becoming large enough that the administrator might want to take action. For more information about this parameter, the alarm, and possible actions to take, see Managing Audit Logs for Filesystem and Table Operations. The audit log continues to grow until the administrator takes action or until the retention period ends. The default value is 32.
`retention`	The period of time in days for which to keep the data in the audit log for the data access. After this period elapses, the content of the file is deleted and new entries are added to the file until the next retention period elapses.

audit info

Displays whether auditing of cluster-management operations and auditing data-access operations are enabled. Also, displays the maxSize and retention values for these two levels of auditing.

Only the `mapr` user for the cluster can run this command. For more information about the `mapr` user, see Managing Users and Groups.

Syntax

CLI

```
maprcli audit info
[ -cluster <cluster_name> ]
-json
```

REST

Request Type	POST

last-updated: Apr 11, 2022

Request URL	`http[s]://<host>:<port>/rest/audit/info?<parameters>`

Parameter

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`json`	Displays command output in JSON format. When `-json` is omitted, there is no output.

Example Output

This output shows that auditing of operations on data and auditing of cluster-level operations are both enabled. For descriptions of `maxSizeGB` and `retentionDays`, see the commands maprcli audit cluster and maprcli audit data.

```
]# maprcli audit info -json
{
        "timestamp":1434458923034,
        "timeofday":"2015-06-16 12:48:43.034 GMT+0000",
        "status":"OK",
        "total":1,
        "data":[
                {
                        "data":{
                                "enabled":"1",
                                "maxSizeGB":"32",
                                "retentionDays":"30"
                        },
                        "cluster":{
                                "enabled":"1",
                                "maxSizeGB":"NA",
                                "retentionDays":"NA"
                        }
                }
        ]
}
```

blacklist
Describes commands used to blacklist and to list blacklisted users.

blacklist user
Blocks a user on a specific cluster.

This action cancels all existing tickets for the specified user. There is no REST equivalent command. For information about blocking, see How Tickets Work.

Syntax

CLI

```
maprcli blacklist user
    -name <user name>
   [ -blacklisttime <millis from
```

```
epoch> or <MM/DD/YYYY> ]
   [ -cluster <cluster name> ]
```

REST N/A

Parameters

Parameter	Description
name	Username to block.
blacklisttime	Invalidates all user's tickets that were raised prior to the specified date (in the format <MM/DD/YYYY>). Alternatively, you can specify the time in milliseconds from epoch time (the number of milliseconds that have elapsed since Jan 1, 1970 midnight UTC).
cluster	Name of the cluster from which to block the user.

Example

Block the *rogueuser* user name from the cluster my.cluster.com:

CLI

```
maprcli blacklist user -name
rogueuser -cluster my.cluster.com
```

Block the *rogueuser* user's tickets that were raised prior to 1st September 2020 from the cluster my.cluster.com:

CLI

```
maprcli blacklist
user -name rogueuser -cluster
my.cluster.com -blacklisttime
09/01/2020
```

Block the *rogueuser* user's tickets that were raised prior to 1605418200155 milliseconds from epoch, from the cluster my.cluster.com:

CLI

```
maprcli blacklist
user -name rogueuser -cluster
my.cluster.com -blacklisttime
1605418200155
```

The value `1605418200155` corresponds to the time November 15th 2020, 11:00:00 am IST+05:30. Therefore, all *rogueuser* tickets that were raised prior to November 15th 2020, 11:00:00 am IST+05:30 are blocked.

Related Log File

The log file `/opt/mapr/logs/cldbaudit.log.json` contains the log of the block operation including the updated block time. For example:

```
{"timestamp":
{"$date":"2020-11-13T08:37:36.524Z"},"resource":"mapruser4","operation":"bla
cklist",
```

```
"username":"root","uid":0,"clientip":"10.10.50.42","properties":

[{"property":"blacklisttime","oldvalue":"1605254599376","newvalue":"16058757
66173"}],
             "status":0}{"timestamp":
{"$date":"2020-11-13T08:37:45.020Z"},"resource":"cluster",
             "operation":"listBlacklist","username":"root","uid":0,
             "clientip":"10.10.50.42","status":0}
```

Here the old block list time was *1605254599376* milliseconds (November 13, 2020 1:33:19 PM IST) and is now updated to *1605875766173* milliseconds (Friday, November 20, 2020 6:06:06 PM IST).

blacklist listusers
Lists blocked users on a specific cluster.

By default, this command lists users that have been blocked from the cluster where the command is run. There is no REST equivalent command. For information about blocking, see How Tickets Work.

Syntax

CLI

```
maprcli blacklist listusers
      [ -cluster <cluster name> ]
```

REST

N/A

Parameters

Parameter	Description
cluster	Name of the cluster for which the blocked users must be listed.

Examples

Show blocked users for the cluster my.cluster.com:

CLI

```
maprcli blacklist listusers -cluster
my.cluster.com
```

cluster
Manages cluster features, gateways, and cluster-wide settings.
cluster feature enable
Allows features to be enabled. Used after upgrading.

Note: Run `cluster feature list` on page 1558 command to retrieve the list of features that can be enabled.

Syntax

CLI

```
maprcli cluster feature enable
  [ -name <feature name >]
  [ -force ]
  [ -all ]
```

REST

```
N/A
```

Parameters

Parameter	Description
`name`	Name of the feature.
`force`	Forces the enabled and disabled dependency features. No value is taken.
`all`	Enables all of the features. No value is taken.

Note: Once a feature is enabled, it can not be disabled.

Examples

To enable all features, use the `cluster feature enable -all` command.

```
maprcli cluster feature enable -all
```

To enable a specific feature, use the `cluster feature enable -name` command. For example:

```
maprcli cluster feature enable -name mfs.feature.audit.support
```

cluster feature list
Allows features to be listed. Used after upgrading.

Syntax

CLI

```
maprcli cluster feature list
              [ -name <Feature name >]
              [ -enabled ]
              [ -disabled ]
```

REST

```
N/A
```

Parameters

Parameter	Description
`name`	Lists the named feature. Displays whether the feature is enabled or disabled along with the feature name.
`enabled`	Lists only the enabled features. No value is taken.
`disabled`	Lists only the disabled features. No value is taken.

Tip: The three dots in the output indicate multiple entries. Use `-json` to format the output.

Examples

Lists the disabled features.

```
maprcli cluster feature list -disabled
dependency
name
description    enabled

mfs.feature.audit.support
     false
```

Lists all the features.

```
maprcli cluster feature list

dependency

name

description

              enabled


cldb.feature.policiesmap.incache.enabl
ed

                          true


cldb.feature.multi.compression


                          true


cldb.feature.volumenumcntrs.incache.en
abled

                          true
{"dependency":
{"name":"cldb.reduce.container.size","
enabled":true}}
mfs.feature.enforce.min.replication
           Support for Enforced Min
Replication For
IO
           true


mfs.feature.db.repl.support


                          true
...

mfs.feature.storage.tiering.support
           Support for MapR Automated
Storage
Tiering.
                 true
```


last-updated: Apr 11, 2022

```
cldb.feature.compression.zlib


                          true
...

mfs.feature.db.streams.v6.support
          Support for Replication
Autosetup with Directcopy, ChangeData
Replication with Changelog  true


bulk.container.create.support


                          true


mfs.feature.db.regionmerge.support


                          true


mfs.feature.metrics.support
          Support for volume
metrics
                                 true
...

mfs.feature.fileace.support
          Support for file-level
access control
expressions.
             true


mfs.feature.name.container.size.contro
l          Support for limiting the
name container data
size
       true


mfs.feature.dare
          Support for Data At Rest
Encryption
                            true
...

mfs.feature.db.streams.v6dot1.support
          Support for Table Get/
Scan, Secondary Indexes for
Arrays
  true


mfs.feature.db.spillv2.support


                          true
```

```
cldb.feature.compression.lz4

                          true

mfs.feature.filecipherbit.support

                          true

mfs.feature.bulkwrite

                          true

cldb.feature.mapr.user.enabled

                          true

mfs.feature.db.bulkload.support

                          true

cldb.feature.separate.cldbvol.rpcs

                          true
...

mfs.feature.db.json.support
           Support for MapR-DB JSON
tables and
MapR-Streams.
                 true
...

mfs.feature.pbs
           Support for Policy Based
Security. Enabled by default in 6.2.0
and
later.
                       true

mfs.feature.fastacr.support

                          true
{"dependency":
{"name":"cldb.reduce.container.size","
enabled":true}}
cldb.feature.cid.reuse
           Support for container
identity
```


last-updated: Apr 11, 2022

```
reuse.
                    true


mfs.feature.streams.connect.support
          Support for Kafka Connect
in the Distributed
mode
       true
...

mfs.feature.container.sharding.support
          Support for Container
Sharding
                              true


mfs.feature.db.ace.support


                          true
...

mfs.feature.fastinodescan.support
          Support for fast scanning
of inodes during
mirror.
         true


mfs.feature.tables


                          true
...

mfs.feature.snapshot.restore.support
          Support for Restoration of
a volume to
snapshot.
            true


mfs.feature.rwmirror.support


                          true
...

mfs.feature.hardlinks.support
          Support for
hardlinks.

  true


cldb.feature.setgid


                          true
```

```
mfs.feature.snapshotdb.lite
           Support For (Switch to)
SnapshotDB
Lite
                 true


mfs.feature.sercmd.support


                          true


cldb.lbs.support
           Support for Label based
storage
                             true
{"dependency":
{"name":"mfs.feature.rwmirror.support"
,"enabled":true}}
mfs.feature.volume.upgrade


                          true


mfs.feature.devicefile.support


                          true


cldb.reduce.container.size


                          true
{"dependency":
{"name":"cldb.reduce.container.size","
enabled":true}}
mfs.feature.external.ip
           Support for Reporting of
External
IP
                  true


mfs.feature.audit.support


                          true


cldb.feature.volumenumsnapshots.incach
e.enabled

                          true


mfs.feature.disk.flush
           Support for Disk
Flush
```


last-updated: Apr 11, 2022

Tip: Use `-json` to format the output.

cluster gateway delete

Deletes the list of MapR gateways from a source MapR cluster.

Source MapR clusters can use such lists to locate the gateways that enable replication of table data to a particular MapR cluster or indexing of table data in a particular Elasticsearch cluster. You create lists of gateways by running the `cluster gateway set` on page 1573 command.

There are three methods of specifying the location of gateways to a MapR cluster that is a source for table replication or indexing in Elasticsearch. If a source MapR cluster relies on DNS records to find out where gateways are located, or the cluster relies on the `mapr-clusters.conf` file to locate gateways, there is no list for the `cluster gateway delete` command to delete.

Note: Deleting a list of gateways with the `maprcli cluster gateway delete` command does not uninstall the listed gateways from the MapR cluster where they are located.

Syntax

CLI

```
/opt/mapr/bin/maprcli cluster gateway
delete
    [ -cluster <cluster on which
command needs to be run> ]
     -dstcluster <cluster name>
```

REST

```
http[s]://<host>:<port>/rest/cluster/
gateway/delete?dstcluster=<path>
```

Parameters

Parameter	Description
`cluster`	If you are not on the source cluster, provide the name of the source cluster on which this command should be run.
`dstcluster`	The name of the cluster on which the gateways are located. If you are replicating table data to another MapR cluster, specify the name of that destination cluster. This destination cluster could be the source cluster if you are performing intra-cluster replication. If you are indexing table data in an Elasticsearch cluster, specify the name of the source MapR cluster because that is where the gateways are located.

Example

Deletes a list of gateways that is stored on a source MapR cluster. The gateways are being used for table replication are located in the destination MapR cluster `newyork`.

CLI

```
/opt/mapr/bin/maprcli cluster gateway
delete -dstcluster newyork
```


last-updated: Apr 11, 2022

REST

```
https://<host>:<port>/rest/cluster/
gateway/delete?dstcluster=newyork
```

Related concepts

Administering MapR Gateways on page 1150
A MapR gateway mediates one-way communication between a source MapR cluster and a destination cluster. You can replicate MapR Database tables (binary and JSON) and MapR Event Store For Apache Kafka streams. MapR gateways also apply updates from JSON tables to their secondary indexes and propagate Change Data Capture (CDC) logs.

Configuring Gateways for Table and Stream Replication on page 1152
Configuring gateways involves installing the `mapr-gateway` package on nodes on a MapR destination cluster and then configuring the MapR source cluster to communicate with the destination cluster. The MapR source cluster is configured by specifying the destination cluster's CLDB node and gateway nodes.

gateway.conf on page 2197

Gateways for Replicating MapR Database Tables on page 621
In MapR Database table replication, MapR Database replicates updates to tables (binary and JSON) on source MapR clusters to replicas of those tables on destination MapR clusters. Gateways are services that receive these updates and apply them to the replicas. These gateways also propagate updates from JSON tables to their secondary indexes.

Related tasks

Specifying the Location of Gateways on page 782
Describes how to set the location of the MapR gateways using either the Control System or the CLI.

Related reference

cluster gateway get on page 1565
Lists the MapR gateways that a source MapR cluster is using.

cluster gateway list on page 1567
Lists all of the gateways that a source MapR cluster is using.

cluster gateway local on page 1569
Lists the gateways that are configured on the MapR cluster where this command is run.

cluster gateway resolve on page 1571
Lists the gateways configured on a MapR cluster that are up and running at the time that the command is issued.

cluster gateway set on page 1573
Specifies the locations of the MapR gateways that a source MapR cluster can use for table replication to a destination MapR cluster, or for indexing table data in an Elasticsearch cluster.

Related information

Managing Gateways on page 1154
Describes the commands for listing gateways, checking status of gateways, managing gateways if they fail, and troubleshooting gateways.

cluster gateway get

Lists the MapR gateways that a source MapR cluster is using.

The source MapR cluster could be using the MapR gateways either for replication of table data to a destination MapR cluster or for the indexing of data in an Elasticsearch cluster.

This list of gateways is created by the `cluster gateway set on page 1573` command.

last-updated: Apr 11, 2022

Syntax

CLI

```
/opt/mapr/bin/maprcli cluster gateway
get
    [ -cluster <sourceCluster> ]
    -dstcluster
<destinationCluster>
```

REST

```
http[s]://<host>:<port>/
rest/cluster/gateway/get?
dstcluster=<path>&cluster=<path>
```

Parameters

Parameter	Description
`cluster`	If you are not on the source cluster, provide the name of the source cluster on which this command should run.
`dstcluster`	The name of the cluster on which the gateways are located. If you are replicating table data to another MapR cluster, specify the name of that destination cluster. This destination cluster could be the source cluster if you are performing intra-cluster replication. If you are indexing table data in an Elasticsearch cluster, specify the name of the source MapR cluster because that is where the gateways are located.

Example

Gets the list of gateways that is stored on a source MapR cluster. The gateways are being used for table replication and are located in the destination MapR cluster `sfcluster`.

CLI

```
/opt/mapr/bin/maprcli cluster gateway
get -dstcluster sfcluster
```

REST

```
https://<host>:<port>/rest/cluster/
gateway/get?dstcluster=sfcluster
```

Example Output

```
cluster gatewayConfig
sfcluster gw1 gw2
```

Related concepts

Administering MapR Gateways on page 1150
A MapR gateway mediates one-way communication between a source MapR cluster and a destination cluster. You can replicate MapR Database tables (binary and JSON) and MapR Event Store For Apache Kafka streams. MapR gateways also apply updates from JSON tables to their secondary indexes and propagate Change Data Capture (CDC) logs.

Configuring Gateways for Table and Stream Replication on page 1152
Configuring gateways involves installing the `mapr-gateway` package on nodes on a MapR destination cluster and then configuring the MapR source cluster to communicate with the destination cluster. The MapR source cluster is configured by specifying the destination cluster's CLDB node and gateway nodes.

gateway.conf on page 2197

Gateways for Replicating MapR Database Tables on page 621
In MapR Database table replication, MapR Database replicates updates to tables (binary and JSON) on source MapR clusters to replicas of those tables on destination MapR clusters. Gateways are services that receive these updates and apply them to the replicas. These gateways also propagate updates from JSON tables to their secondary indexes.

Related tasks

Specifying the Location of Gateways on page 782
Describes how to set the location of the MapR gateways using either the Control System or the CLI.

Related reference

cluster gateway delete on page 1564
Deletes the list of MapR gateways from a source MapR cluster.

cluster gateway list on page 1567
Lists all of the gateways that a source MapR cluster is using.

cluster gateway local on page 1569
Lists the gateways that are configured on the MapR cluster where this command is run.

cluster gateway resolve on page 1571
Lists the gateways configured on a MapR cluster that are up and running at the time that the command is issued.

cluster gateway set on page 1573
Specifies the locations of the MapR gateways that a source MapR cluster can use for table replication to a destination MapR cluster, or for indexing table data in an Elasticsearch cluster.

Related information

Managing Gateways on page 1154
Describes the commands for listing gateways, checking status of gateways, managing gateways if they fail, and troubleshooting gateways.

cluster gateway list

Lists all of the gateways that a source MapR cluster is using.

The source MapR cluster is using gateways either for replication of table data to destination MapR clusters or for the indexing of table data in Elasticsearch clusters.

This list is created by the `cluster gateway set` on page 1573 command.

Syntax

CLI

```
/opt/mapr/bin/maprcli cluster gateway
list
    [ -cluster <sourceCluster> ]
```

REST

```
http[s]://<host>:<port>/rest/cluster/
gateway/list
```

Parameters

Parameter	Description
`cluster`	If you are not on the source cluster, provide the name of the source cluster on which this command should run.

Example

Lists all the gateways that a source MapR cluster can use when replicating table data in MapR clusters or indexing data in Elasticsearch clusters. In this example, assuming `newyork` to be the name of a MapR cluster that is a destination for table replication, the output shows two gateways that are available for replicating to this cluster.

CLI

```
/opt/mapr/bin/maprcli cluster gateway
list
```

REST

```
https://<host>:<port>/rest/cluster/
gateway/list
```

Example Output

```
cluster gatewayConfig
newyork gw1 gw2
```

Related concepts

Administering MapR Gateways on page 1150
A MapR gateway mediates one-way communication between a source MapR cluster and a destination cluster. You can replicate MapR Database tables (binary and JSON) and MapR Event Store For Apache Kafka streams. MapR gateways also apply updates from JSON tables to their secondary indexes and propagate Change Data Capture (CDC) logs.

Configuring Gateways for Table and Stream Replication on page 1152
Configuring gateways involves installing the `mapr-gateway` package on nodes on a MapR destination cluster and then configuring the MapR source cluster to communicate with the destination cluster. The MapR source cluster is configured by specifying the destination cluster's CLDB node and gateway nodes.

gateway.conf on page 2197

Gateways for Replicating MapR Database Tables on page 621
In MapR Database table replication, MapR Database replicates updates to tables (binary and JSON) on source MapR clusters to replicas of those tables on destination MapR clusters. Gateways are services that receive these updates and apply them to the replicas. These gateways also propagate updates from JSON tables to their secondary indexes.

Related tasks

Specifying the Location of Gateways on page 782
Describes how to set the location of the MapR gateways using either the Control System or the CLI.

Related reference

cluster gateway delete on page 1564
Deletes the list of MapR gateways from a source MapR cluster.

cluster gateway get on page 1565
Lists the MapR gateways that a source MapR cluster is using.

cluster gateway local on page 1569
Lists the gateways that are configured on the MapR cluster where this command is run.

cluster gateway resolve on page 1571
Lists the gateways configured on a MapR cluster that are up and running at the time that the command is issued.

cluster gateway set on page 1573
Specifies the locations of the MapR gateways that a source MapR cluster can use for table replication to a destination MapR cluster, or for indexing table data in an Elasticsearch cluster.

Related information

Managing Gateways on page 1154

Describes the commands for listing gateways, checking status of gateways, managing gateways if they fail, and troubleshooting gateways.

cluster gateway local
Lists the gateways that are configured on the MapR cluster where this command is run.

Syntax

CLI

```
/opt/mapr/bin/maprcli cluster gateway
local
        [ -cluster cluster on which
command to be run ]
        [ -format dns/text. default:
text ]
```

REST

```
http[s]://<host>:<port>/rest/cluster/
gateway/local
```

Parameters

Parameter	Description
`format`	The output format. Value types: `dns` or `text`. Default: `text`. **Note:** • If the output is formatted as DNS, you can copy and paste the output into a DNS record in the zone file for your domain. The source MapR cluster can then locate the gateways by doing a DNS lookup. • If the output is formatted as text, you can copy and paste that text into the `-gateways` parameter of the `maprcli cluster gateway set` command. Running this command is an alternative way of specifying the location of these gateways to the source MapR cluster.
`cluster`	If you are not on a MapR cluster where one or more gateways are configured, provide the name of the cluster. • When replicating MapR Database table data to one or more replicas on this cluster, then the cluster is a destination MapR cluster. • When indexing MapR Database table data in one or more Elasticsearch cluster, then the current cluster is a source MapR cluster where the tables being indexed are located.

Example

This example shows text output of the list of gateways that are configured on a MapR cluster:

last-updated: Apr 11, 2022

CLI	`/opt/mapr/bin/maprcli cluster gateway local`
REST	`https://<host>:<port>/rest/cluster/gateway/local`
Example Output	`gatewayinfo centos23 centos22`

This example shows DNS output of the list of gateways that are configured on a MapR cluster:

CLI	`/opt/mapr/bin/maprcli cluster gateway local -format dns`
REST	`https://<host>:<port>/rest/cluster/gateway/local?format=dns`
Example Output	`gatewaydnsinfo ; TXT Record addresses gateway.mycluster IN TXT "centos23 centos22"`

Related concepts

Administering MapR Gateways on page 1150
A MapR gateway mediates one-way communication between a source MapR cluster and a destination cluster. You can replicate MapR Database tables (binary and JSON) and MapR Event Store For Apache Kafka streams. MapR gateways also apply updates from JSON tables to their secondary indexes and propagate Change Data Capture (CDC) logs.

Configuring Gateways for Table and Stream Replication on page 1152
Configuring gateways involves installing the `mapr-gateway` package on nodes on a MapR destination cluster and then configuring the MapR source cluster to communicate with the destination cluster. The MapR source cluster is configured by specifying the destination cluster's CLDB node and gateway nodes.

gateway.conf on page 2197

Gateways for Replicating MapR Database Tables on page 621
In MapR Database table replication, MapR Database replicates updates to tables (binary and JSON) on source MapR clusters to replicas of those tables on destination MapR clusters. Gateways are services that receive these updates and apply them to the replicas. These gateways also propagate updates from JSON tables to their secondary indexes.

Related tasks

Specifying the Location of Gateways on page 782
Describes how to set the location of the MapR gateways using either the Control System or the CLI.

Related reference

cluster gateway delete on page 1564
Deletes the list of MapR gateways from a source MapR cluster.

cluster gateway get on page 1565
Lists the MapR gateways that a source MapR cluster is using.

cluster gateway list on page 1567
Lists all of the gateways that a source MapR cluster is using.

cluster gateway resolve on page 1571
Lists the gateways configured on a MapR cluster that are up and running at the time that the command is issued.

cluster gateway set on page 1573
Specifies the locations of the MapR gateways that a source MapR cluster can use for table replication to a destination MapR cluster, or for indexing table data in an Elasticsearch cluster.

Related information

Managing Gateways on page 1154
Describes the commands for listing gateways, checking status of gateways, managing gateways if they fail, and troubleshooting gateways.

cluster gateway resolve

Lists the gateways configured on a MapR cluster that are up and running at the time that the command is issued.

Run this command on a source MapR cluster to find out how many gateways are available for table replication to a destination MapR cluster, or for indexing table data in an Elasticsearch cluster.

This command uses the following criteria to get the list:

- If you specified the locations of the gateways with the `cluster gateway set` on page 1573 command, the `maprcli cluster gateway resolve` command returns the list of the gateways.
- If you specified the locations of the gateways only with a DNS record, this command performs a DNS lookup for gateways on the specified MapR cluster and returns the list that it finds.
- If you did not specify the locations of the gateways using the previously listed methods, this command assumes that gateways are located on the CLDB nodes configured in the `mapr-clusters.conf` file on the MapR cluster where the command is run.

Note: Unresponsive gateways are not included in the list.

For more information about gateways, see MapR Gateways.

Syntax

CLI

```
/opt/mapr/bin/maprcli cluster gateway
resolve
    [ -cluster <cluster on which the
command is to be run> ]
    -dstcluster <destination cluster
name>
```

REST

```
http[s]://<host>:<port>/rest/
cluster/gateway/resolve?
dstcluster=<clustername>
```

Parameters

Parameter	Description
`cluster`	If you are not on the source MapR cluster, provide the name of the source cluster on which this command should run.

last-updated: Apr 11, 2022

Parameter	Description
`dstcluster`	The name of the cluster for which you want to list the available gateways. If you are replicating table data to another MapR cluster, specify the name of that destination cluster. This destination cluster can be the source cluster if you are performing intra-cluster replication. If you are indexing table data in an Elasticsearch cluster, specify the name of the source MapR cluster because that is where the gateways are located.

Example

This example shows that only one gateway is up and running on the MapR cluster `cluster1`. The IP address of this gateway was found in a DNS record, as indicated by the `Source` field.

CLI

```
/opt/mapr/bin/maprcli cluster gateway
resolve -dstcluster cluster1 -json
```

REST

```
https://<host>:<port>/rest/cluster/
gateway/resolve?dstcluster=cluster1
```

Example Output

```
{
        "timestamp":1424266395862,
        "timeofday":"2015-02-18
01:33:15.862 GMT+0000",
        "status":"OK",
        "total":1,
        "data":[
                {

"GatewayHosts":"10.10.20.12:7660",
                        "Source":"DNS"

                }
        ]
}
```

Related concepts

Administering MapR Gateways on page 1150
A MapR gateway mediates one-way communication between a source MapR cluster and a destination cluster. You can replicate MapR Database tables (binary and JSON) and MapR Event Store For Apache Kafka streams. MapR gateways also apply updates from JSON tables to their secondary indexes and propagate Change Data Capture (CDC) logs.

Configuring Gateways for Table and Stream Replication on page 1152
Configuring gateways involves installing the `mapr-gateway` package on nodes on a MapR destination cluster and then configuring the MapR source cluster to communicate with the destination cluster. The MapR source cluster is configured by specifying the destination cluster's CLDB node and gateway nodes.

gateway.conf on page 2197

Gateways for Replicating MapR Database Tables on page 621
In MapR Database table replication, MapR Database replicates updates to tables (binary and JSON) on source MapR clusters to replicas of those tables on destination MapR clusters. Gateways are services that

receive these updates and apply them to the replicas. These gateways also propagate updates from JSON tables to their secondary indexes.

Related tasks

Specifying the Location of Gateways on page 782
Describes how to set the location of the MapR gateways using either the Control System or the CLI.

Related reference

cluster gateway delete on page 1564
Deletes the list of MapR gateways from a source MapR cluster.

cluster gateway get on page 1565
Lists the MapR gateways that a source MapR cluster is using.

cluster gateway list on page 1567
Lists all of the gateways that a source MapR cluster is using.

cluster gateway local on page 1569
Lists the gateways that are configured on the MapR cluster where this command is run.

cluster gateway set on page 1573
Specifies the locations of the MapR gateways that a source MapR cluster can use for table replication to a destination MapR cluster, or for indexing table data in an Elasticsearch cluster.

Related information

Managing Gateways on page 1154
Describes the commands for listing gateways, checking status of gateways, managing gateways if they fail, and troubleshooting gateways.

cluster gateway set

Specifies the locations of the MapR gateways that a source MapR cluster can use for table replication to a destination MapR cluster, or for indexing table data in an Elasticsearch cluster.

In addition to this method, there are two other methods that you can use to specify the locations of gateways that a source MapR cluster can use when replicating to a particular MapR cluster or when indexing in an Elasticsearch cluster. See Configuring Gateways for Table and Stream Replication for details about them.

Syntax

CLI

```
/opt/mapr/bin/maprcli cluster gateway
set
   [ -cluster <cluster on which
command to be run> ]
   -dstcluster <cluster name>
   -gateways <space-separated list of
hostnames>
```

REST

```
http[s]://<host>:<port>/
rest/cluster/gateway/set?
dstcluster=<path>&gateways=<list
of gateways>
```

Parameters

Parameter	Description
`cluster`	If you are not on the source MapR cluster, provide the name of that cluster.

last-updated: Apr 11, 2022

Parameter	Description
`dstcluster`	The name of the MapR cluster in which the gateways are located. If you are replicating table data to another MapR cluster, specify the name of that destination cluster. This destination cluster could be the source cluster if you are performing intra-cluster replication. If you are indexing table data in an Elasticsearch cluster, specify the name of the source MapR cluster because that is where the gateways are located.
`gateways`	A space-delimited list of gateway hostnames or IP addresses. Place double quotation marks around the list of gateways, as in this example: `-gateways "gateway1 gateway2"`

Example

This example specifies the hostnames of two gateways that are in the MapR cluster `newyork . This command could be used in any of these situations:`

- The cluster `newyork` is the destination cluster for table replication from the source MapR cluster.
- The cluster `newyork` is both a source and destination cluster for intra-cluster table replication.
- The cluster `newyork` is a source MapR cluster that contains tables being indexed in one or more Elasticsearch clusters.

CLI

```
/opt/mapr/bin/maprcli cluster
gateway set -dstcluster
newyork -gateways "gw1.bigcompany.com
gw2.bigcompany.com"
```

REST

```
https://<host>:<port>/rest/cluster/
gateway/set?
dstcluster=newyork&gateways=gw1.bigcom
pany.com%20gw2.bigcompany.com
```

Related concepts

Administering MapR Gateways on page 1150
A MapR gateway mediates one-way communication between a source MapR cluster and a destination cluster. You can replicate MapR Database tables (binary and JSON) and MapR Event Store For Apache Kafka streams. MapR gateways also apply updates from JSON tables to their secondary indexes and propagate Change Data Capture (CDC) logs.

Configuring Gateways for Table and Stream Replication on page 1152
Configuring gateways involves installing the `mapr-gateway` package on nodes on a MapR destination cluster and then configuring the MapR source cluster to communicate with the destination cluster. The MapR source cluster is configured by specifying the destination cluster's CLDB node and gateway nodes.

gateway.conf on page 2197

Gateways for Replicating MapR Database Tables on page 621
In MapR Database table replication, MapR Database replicates updates to tables (binary and JSON) on source MapR clusters to replicas of those tables on destination MapR clusters. Gateways are services that receive these updates and apply them to the replicas. These gateways also propagate updates from JSON tables to their secondary indexes.

Related tasks

Specifying the Location of Gateways on page 782
Describes how to set the location of the MapR gateways using either the Control System or the CLI.

Related reference

cluster gateway delete on page 1564
Deletes the list of MapR gateways from a source MapR cluster.

cluster gateway get on page 1565
Lists the MapR gateways that a source MapR cluster is using.

cluster gateway list on page 1567
Lists all of the gateways that a source MapR cluster is using.

cluster gateway local on page 1569
Lists the gateways that are configured on the MapR cluster where this command is run.

cluster gateway resolve on page 1571
Lists the gateways configured on a MapR cluster that are up and running at the time that the command is issued.

Related information

Managing Gateways on page 1154
Describes the commands for listing gateways, checking status of gateways, managing gateways if they fail, and troubleshooting gateways.

cluster info

Returns minimum and maximum values for the following attributes:

- `VolumeSize` — The size of the volume.
- `VolumeQuotaSize` — The hard quota (disk space) for the volume.
- `VolumeAdvisoryQuota` — The advisory quota (disk space) for the volume.
- `VolumeLogicalUsedSize` — The logical size used by the volume.
- `VolumeNumContainers` — The number of replicas for the volume.
- `VolumeGuranteedNumContainers` — The number of guaranteed replicas for the volume.
- `VolumeNumNamespaceContainers` — The number of replicas for the name container associated with the volume.
- `VolumeGuranteedNumNamespaceContainers` — The number of guaranteed replicas for the name container associated with the volume.
- `VolumeNumSnapshots` — The number of snapshots of the volume.
- `VolumeCoalesceInterval` — The coalesce interval setting for the volume.
- `VolumeMaxInodesAlarmThreshold` — The threshold for triggering the VOLUME_ALARM_INODES_EXCEEDED alarm.
- `VolumeMaxNsSizeMbAlarmThreshold` — The threshold for triggering the VOLUME_ALARM_INODES_EXCEEDED alarm.
- `VolumeReReplicationTimeOut` — The timeout value for re-replication.
- `StoragePoolCapacitySize` — The total amount of disk space on the storage pool.

- `StoargePoolUsedSize` — The amount of used space on the storage pool.
- `StoragePoolAvailableSize` — The amount of available space on the storage pool.

Syntax

CLI	`maprcli cluster info -getminmax <attributes>`
REST API	N/A

Parameters

Parameter	Description
`getminmax`	The comma-separated list of attributes to return minimum and maximum values for. To retrieve the minimum and maximum values for all the attributes, use the keyword `all`.

Examples

Retrieve the minimum and maximum values for all the attributes:

```
maprcli cluster info -getminmax all
```

Output

```
# maprcli cluster info -getminmax all
min     unit max     name
1804262 MB   2703486 StoragePoolAvailableSize
0       MB   32768   VolumeAdvisoryQuota
2       Num  6       VolumeNumContainers
0       MB   47152   VolumeLogicalUsedSize
60      Min  60      VolumeCoalesceInterval
0       MB   0       VolumeMaxNsSizeMbAlarmThreshold
0       MB   0       VolumeSharedSize
1826966 MB   2738572 StoragePoolCapacitySize
2       Num  6       VolumeNumNamespaceContainers
0       MB   6020    VolumeOwnedSize
22704   MB   35086   StoragePoolUsedSize
0       Num  0       VolumeNumSnapshots
0       Sec  0       VolumeReReplicationTimeOut
0       MB   6020    VolumeSize
1       Num  2       VolumeGuranteedNumNamespaceContainers
0       Sec  5       VolumeMaxInodesAlarmThreshold
1       Num  2       VolumeGuranteedNumContainers
0       MB   0       VolumeQuotaSize
```

Retrieve the minimum and maximum values for the given attributes:

```
# maprcli cluster info
    -getminmax
VolumeNumContainers,VolumeGuranteedNumContainers,VolumeGuranteedNumNamespace
Containers,VolumeNumNamespaceContainers
```

Output

```
# maprcli cluster info -getminmax
VolumeNumContainers,VolumeGuranteedNumContainers,VolumeGuranteedNumNamespace
Containers,VolumeNumNamespaceContainers
min unit max name
1   Num  2   VolumeGuranteedNumNamespaceContainers
2   Num  6   VolumeNumContainers
2   Num  6   VolumeNumNamespaceContainers
1   Num  2   VolumeGuranteedNumContainers
```

cluster mapreduce get
Displays the cluster-wide default for the MapReduce mode.

Warning: This command is deprecated alongside MapReduce v1.

Syntax

CLI

```
maprcli cluster mapreduce get
```

REST

```
http[s]://<host>:<port>/rest/cluster/
mapreduce/get
```

Output Fields

Field	Description
`default_mode`	Displays either `yarn` or `classic`.
`mapreduce_version`	Displays the hadoop version associated with the default_mode.

Sample Output

```
default_mode  mapreduce_version
classic       0.20.2
```

Examples

CLI

```
maprcli cluster mapreduce get
```

REST

```
https://r1n1.sj.us:8443/rest/cluster/
mapreduce/get
```

cluster mapreduce set
Sets the cluster-wide MapReduce mode.

Warning: This command is deprecated alongside MapReduce v1.

Syntax

CLI

```
maprcli cluster mapreduce set -mode
yarn
```

REST

```
http[s]://<host>:<port>/rest/cluster/
mapreduce/set?<parameters>
```

Parameters

Parameter	Description
`mode`	The MapReduce mode of the cluster. Enter `yarn` to use Resource Manager and Node Manager to run MapReduce jobs or applications.

Examples

Sets the MapReduce mode for the cluster to classic.

CLI

```
maprcli cluster mapreduce set -mode
yarn
```

REST

```
https://r1n1.sj.us:8443/rest/cluster/
mapreduce/set?mode=yarn
```

cluster queryservice
Describes the commands to enable/disable and view the settings for the OJAI Distributed Query Service.

Enable the OJAI Distributed Query Service on page 505 if you want the following functionality when querying MapR Database JSON tables:

- Advanced secondary index selection
- Sorts of large data sets
- Parallel query execution

Permissions Required

If you enable the OJAI Distributed Query Service during installation, then you must be user 'mapr' to run these commands. If you disable the service, and later re-enable it, then the command needs to be run by the user that re-enabled the service.

cluster queryservice getconfig
Retrieves the configuration of the OJAI Distributed Query Service.

Permissions Required

Only the user that enabled the OJAI Distributed Query Service can run this command. If the service was enabled during installation, user 'mapr' must run the command.

Syntax

CLI

```
maprcli cluster queryservice
getconfig -cluster < cluster-name >
```

REST

Not available

Parameters

Parameter	Description
cluster	(Optional) Name of the cluster that is using secondary indexes to query MapR Database JSON table

Example

```
maprcli cluster queryservice getconfig -cluster my.cluster.com
```

cluster queryservice setconfig
Enables or disables the OJAI Distributed Query Service. When enabling the service, you can specify the configuration of the service.

Permissions Required

Only the user that enabled the OJAI Distributed Query Service can run this command. If the service was enabled during installation, user 'mapr' must run the command.

Syntax

CLI

```
maprcli cluster queryservice
setconfig
              [ -cluster <
cluster-name > ]
              -enabled < true | false
>
              -clusterid < cluster-id
of MapR Drill cluster >
              -storageplugin < Name
of MapR Drill Storage plug-in >
              -znode < Root Zookeeper
node user by MapR Drill cluster >
```

REST

Not available

Parameters

Parameter	Description
cluster	(Optional) Name of the cluster that is using secondary indexes to query MapR Database JSON tables
enabled	(Required) Whether the OJAI Distributed Query Service is enabled Values: true or false

Parameter	Description
clusterid	(Required) Cluster ID of your MapR Drill cluster Refer to the value of the `cluster-id` parameter in the `drill-distrib.conf` file. You can find this file in the `/opt/mapr/drill/drill-<version number>/conf` directory.
storageplugin	(Required) Name of the MapR Drill Storage plug-in instance used to run OJAI queries (usually **dfs**)
znode	(Required) Name of the root Zookeeper node used by the MapR Drill cluster (usually **/drill**)

Example

```
maprcli cluster queryservice setconfig \
   -enabled true \
   -clusterid mycluster \
   -storageplugin dfs \
   -znode /drill
```

config
Lists configuration values for the MapR cluster.

Configuration Fields

The following fields are configurable.

`cldb.balancer.disk.max.switches.in.nodes.percentage`
Default Value: 10

The maximum number of containers that can be balanced in parallel by the disk balancer. The value is a percentage of the number of nodes in the system.

`cldb.disk.balancer.enable`
Default Value: 1 (Disk Balancer is enabled)

Enables (1) or disables (0) the Disk Balancer.

`cldb.balancer.disk.sleep.interval.sec`
Default Value: 120

The sleep interval (in seconds) between two successive runs of the Disk Balancer.

`cldb.balancer.disk.threshold.percentage`
Default Value: 70

Percentage of used space that causes containers in a storage pool to be distributed across other less used storage pools.

`cldb.balancer.logging`
Default Value: 0

Disables (0) or enables (1) the logging of messages in the Disk Balancer and Role Balancer.

`cldb.balancer.role.max.switches.in.nodes.percentage`
Default Value: 10

The percentage (of the number of nodes in the system) to use to determine the maximum number of containers whose roles (Masters and Tails) are balanced in parallel by the Role Balancer.

For example, suppose there are 500 nodes and the value of this parameter is 10(%). The number of containers whose roles are balanced in parallel is (10/100)*500=50.

cldb.balancer.role.paused

Default Value: 1

Enables (0) or Disables (1) the Role Balancer.

cldb.balancer.role.sleep.interval.sec

Default Value: 900

The sleep interval (in seconds) between two successive runs of the Role Balancer.

cldb.balancer.startup.interval.sec

Default Value: 1800

The initial startup delay (in seconds) of the Role Balancer for existing clusters.

cldb.cluster.almost.full.percentage

Default Value: 90

The percentage at which the `CLUSTER_ALARM_CLUSTER_ALMOST_FULL` alarm is triggered.

cldb.container.alloc.selector.algo

Default Value: 0

The allocation algorithm to use when creating new containers. The value can be one of:

- 0 - indicates Round Robin algorithm if the number of nodes is less than or equal to 100, Randomized algorithm otherwise.
- 1 - indicates Round Robin algorithm. If selected, containers are allocated across nodes in a topology in a round robin fashion.
- 2 - indicates Randomized algorithm. If selected, containers are allocated across nodes in a randomized way.

cldb.container.assign.buffer.sizemb

Default Value: 1024

The size of the container (in MB) that should be used as a buffer. When allocating a new container, this size is deducted from the maximum container size.

Note: When you modify the value of `cldb.container.sizemb`, check and update the value of `cldb.container.assign.buffer.sizemb` to prevent new containers from being created when existing containers are not full.

cldb.container.create.diskfull.threshold

Default Value: 85

The percentage of space on a file server to use to classify the file server as full.

cldb.container.sizemb

Default Value: 32768

The maximum size for containers (in MB). This is a soft limit.

Note: When `cldb.container.sizemb` value is modified, check and update the value of `cldb.container.assign.buffer.sizemb` to prevent new containers from being created when existing containers are not full.

cldb.default.chunk.sizemb

Default Value: 256

The size of each chunk (in MB) that make up a file in the MapR filesystem.

cldb.default.volume.topology

Default Value: /data

last-updated: Apr 11, 2022

	The default topology for new volumes.
cldb.dialhome.metrics.file.rotation.period	*Default Value:* 365 The retention period of the files (in days) that is used to record Dialhome metrics. Files that are past their retention period are automatically deleted.
cldb.disable.alarm.history	*Default Value:* 0 (false) Set this to 1 (true) to disable CLDB alarm history, as tracking and fetching the alarm history can degrade the performance of CLDB on large clusters.
cldb.fs.mark.rereplicate.sec	*Default Value:* 3600 The number of seconds that a node can fail to heartbeat before it is considered dead. Once a node is considered dead, the CLDB re-replicates any data contained on the node.
cldb.fs.reregistration.wait.time	*Default Value:* 15 The amount of time (in minutes) to wait before checking for inactive nodes. **Note:** Reduce the value to raise the No Heartbeat Alarm on page 2233 without delay, after CLDB failover. To avoid spurious alarms, do not reduce this value below 5 (minutes).
cldb.log.fileserver.timeskew.interval.mins	*Default Value:* 60 The frequency (in minutes) at which CLDB should log messages about the time skew on the file server.
cldb.max.parallel.resyncs.star	*Default Value:* 3 The number of container replicas that can resync in parallel from the source for low-latency (star-replicated) volumes.
cldb.mfs.heartbeat.timeout.multiple	*Default Value:* 10 Specifies a multiple heartbeat timeout. For small clusters, the heartbeat interval is 1 second and the multiple is 10 by default, which makes the heartbeat timeout 10 seconds.
cldb.min.fileservers	*Default Value:* 1 The number of file servers hosting the CLDB volume that is required for the master CLDB to complete the bootstrap process.
cldb.replication.manager.critical.paused	*Default Value:* 0 Disables (0) or enables (1) the processing of critically under-replicated containers. If enabled, the critically under-replicated containers are processed on a priority basis to increase the number of copies.
cldb.replication.manager.max.resyncs.in.nodes.percentage	*Default Value:* 1200 The number of containers that can be replicated in parallel, expressed as a percentage of the number of active nodes. If the value is 1200, the number of containers that can be replicated is 12 times the number of active nodes.
cldb.replication.manager.over.paused	*Default Value:* 0 Disables (0) or enables (1) the processing of over-replicated containers. Over-replicated containers are processed to delete extra copies, which is when

	the number of copies is more than the desired replication factor.
cldb.replication.manager.start.mins	*Default Value:* 15 The delay (in minutes) between CLDB startup and replication manager startup, to allow all nodes to register and heartbeat.
cldb.replication.max.in.transit.containers.per.sp	*Default Value:* 4 The maximum number of containers that can be in transit on a storage pool (SP). Containers that serve either as the source or destination of a resync operation are considered as being in 'transit'.
cldb.replication.sleep.interval.sec	*Default Value:* 15 The sleep duration (in seconds) between consecutive runs of the Replication Manager.
cldb.replication.tablescan.interval.sec	*Default Value:* 120 The sleep duration (in seconds) between consecutive runs of the Replication Scanner. While the Replication Scanner classifies containers into different buckets, the Manager thread either replicates or removes additional copies.
cldb.rm.wait.rack.violated.fork.copy.mins	*Default Value:* 720 The buffer time (in minutes) after which all container copies found on the same rack are fixed.
cldb.rm.wait.fork.on.same.rack.mins	*Default Value:* 180 The time (in minutes) to defer creating containers on the same rack, for critically under-replicated containers, if there are at least two copies of the containers.
cldb.security.user.ticket.duration.seconds	*Default Value:* 1209600 The length of time (in seconds) before the user ticket (generated using the `maprlogin password` command) expires.
cldb.security.user.ticket.max.duration.seconds	*Default Value:* 2592000 The maximum amount of time (in seconds) allowed for the user ticket (generated using the `maprlogin password` command).
cldb.security.user.ticket.renew.duration.seconds	*Default Value:* 2592000 The length of time (in seconds) to renew the user ticket (generated using the `maprlogin password` command).
cldb.security.user.ticket.renew.max.duration.seconds	*Default Value:* 7776000 The maximum duration allowed for a user ticket (generated using `maprlogin password` command) renewal.
cldb.topology.almost.full.percentage	*Default Value:* 90 The threshold percentage that is used to raise alarms when the used space on the nodes of a topology exceed a certain percentage of total space.
cldb.volume.epoch	*Default Value:* Not Applicable The starting epoch of a new Container. Epoch is used internally in the selection of the master container.

last-updated: Apr 11, 2022

cldb.volumes.namespace.default.min.replication	*Default Value:* 2 The minimum replication factor for the name container. Containers with fewer copies than this value are replicated on a priority basis. **Note:** To modify, run the `maprli volume modify -name <volume name> -nsminreplication <replication factor>` command.
cldb.volumes.namespace.default.replication	*Default Value:* 3 The desired replication factor for the name container. **Note:** To modify, run the `maprli volume modify -name <volume name> -nsreplication <replication factor>` command.
mapr.fs.nocompression	*Default Value:* "bz2,gz,tgz,tbz2, zip,z,Z,mp3,jpg, jpeg,mpg,mpeg,avi, gif,png,lzo,jar" The file types that should not be compressed. See File Extensions of Compressed Files on page 989.
mapr.fs.permissions.supergroup	*Default Value:* root The *super group* of the MapR filesystem layer.
mapr.fs.permissions.superuser	*Default Value:* mapr The *super user* of the MapR filesystem layer.
mapr.targetversion	*Default Value:* Not Applicable The configuration variable to set the current version of the MapR distribution. Failing to set this variable on an upgrade causes alarms to be missed when all the nodes in a cluster are not at the same version of the software.
mfs.db.parallel.copyregions	*Default Value:* Not Applicable The number of parallel copy regions per MFS instance. Setting this field to a larger value increases the parallelism for data transfers during index updates, CDC propagation, and table replication. A larger value increases the transfer rate and reduces the initial synchronization time, but uses more system resources. The latter may impact the response time and performance of applications that read data from the same nodes.
mfs.high.memory.alarm.threshold	*Default Value:* 110 (percentage of allocated memory) On initialization, the MapR filesystem is allocated a certain amount of memory. There is some additional headroom that can be used if the MapR filesystem is under memory pressure. However, if the MapR filesystem exceeds the high memory threshold (default 10% over the allocated memory, that is 110%), the High FileServer Memory Alarm on page 2230 is raised. This threshold can be 8% to 30% over the allocated memory (that is 108% to 130%) .
mfs.feature.db.json.support	*Default Value:* • 1 for new MapR installations • 0 for upgraded MapR installations

last-updated: Apr 11, 2022

	Disables (0) or enables (1) MapR streams and support in MapR Database for JSON documents and tables..
mfs.feature.devicefile.support	*Default Value:* 1 Disables (0) or enables (1) usage of Named Pipes over NFS.
pernode.numcntrs.alarm.thr	*Default Value:* 50000 The maximum number of Read/Write (RW) containers on each node beyond which performance may not be optimal. The optimal number for RW and snapshot containers combined is 10 times the value of this parameter.

config load
Displays information about the cluster configuration.

Syntax

CLI

```
maprcli config load
    [ -cluster <cluster> ]
    [ -keys <keys> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/config/load?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster for which to display values.
`keys`	This parameter is used to specify which information to display. Comma-separated fields are used to display values; see the Configuration Fields table.

Output

Information about the cluster configuration. See the Configuration Fields table.

Sample Output

```
{
        "status":"OK",
        "total":1,
        "data":[
                {
                        "mapr.webui.http.port":"8080",
                        "mapr.fs.permissions.superuser":"root",
                        "mapr.smtp.port":"25",
                        "mapr.fs.permissions.supergroup":"supergroup"
                }
        ]
}
```


last-updated: Apr 11, 2022

Examples

Display several keys:

CLI

```
/opt/mapr/bin/maprcli config
load -keys
mapr.webui.http.port,mapr.webui.https.
port,mapr.webui.https.keystorepath,map
r.webui.https.keystorepassword,mapr.we
bui.https.keypassword,mapr.webui.timeo
ut
```

REST

```
https://abc.sj.us:8443/rest/config/
load?
keys=mapr.webui.http.port,mapr.webui.h
ttps.port,mapr.webui.https.keystorepat
h,mapr.webui.https.keystorepassword,ma
pr.webui.https.keypassword,mapr.webui.
timeout
```

config save

Saves configuration information, specified as key/value pairs. Permissions required: `fc` or `a`.

See the Configuration Fields table.

Warning: Changing cluster configuration may have an impact on the way the cluster functions. Make sure you understand the change well or else make the change under the guidance of MapR support.

Syntax

CLI

```
maprcli config save
         [ -cluster cluster name ]
         -test test only. default: 0
         -values JSON Object to
comprise all config properties to save
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/config/save?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`values`	A JSON object containing configuration fields; see the Configuration Fields table.
`test`	Set this to 1 to test SMTP configuration without actually saving the values. The system sends a test email to check if the configuration is correct. This parameter is applicable only for SMTP configuration.

last-updated: Apr 11, 2022

Examples

Configure MapR SMTP settings:

CLI

```
/opt/mapr/bin/maprcli config
save -values
'{"mapr.smtp.provider":"gmail","mapr.s
mtp.server":"smtp.gmail.com","mapr.smt
p.sslrequired":"true","mapr.smtp.port"
:"465","mapr.smtp.sender.fullname":"Ab

Cd","mapr.smtp.sender.email":"xxx@gmai
l.com","mapr.smtp.sender.username":"xx
x@gmail.com","mapr.smtp.sender.passwor
d":"abc"}'
```

REST

```
https://abc.sj.us:8443/rest/config/
save?
values={"mapr.smtp.provider":"gmail","
mapr.smtp.server":"smtp.gmail.com","ma
pr.smtp.sslrequired":"true","mapr.smtp
.port":"465","mapr.smtp.sender.fullnam
e":"Ab
Cd","mapr.smtp.sender.email":"xxx@gmai
l.com","mapr.smtp.sender.username":"xx
x@gmail.com","mapr.smtp.sender.passwor
d":"abc"}
```

Note: The maximunm number of volumes that can be balanced at a time is 1000.

Related concepts

config on page 1580
Lists configuration values for the MapR cluster.

dashboard info

Displays a summary of information about the cluster.

Syntax

CLI

```
/opt/mapr/bin/maprcli dashboard info
    [ -cluster <cluster name> ]
    [ -multi_cluster_info true|false]
    [ -version true|false ]
    [ -zkconnect <ZooKeeper connect
string> ]
    -json
```

Note: The `-json` option is required.

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/dashboard/info[?<parameters>]`

last-updated: Apr 11, 2022

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. By default, the cluster is the local cluster.
`multi_cluster_info`	Specifies whether to display cluster information from multiple clusters. Values: `true` or `false`. Default: `false`.
`version`	Specifies whether to display the version. Values: `true` or `false`. Default: `false`.
`zkconnect`	Common Parameters on page 1525
`json`	Formats the output.

Output

A summary of information about the services, volumes, MapReduce applications, health, and utilization of the cluster.

Output Fields

Field	Description
timestamp	The time at which the `dashboard info` data was retrieved, expressed as a Unix epoch time.
timeofday	The local time and date of the query.
status	The success status of the `dashboard info` command.
total	The number of clusters for which data was queried in the `dashboard info` command.
version	The MapR software version running on the cluster.
cluster	Determines the following information about the cluster: • `name` — the cluster name • `secure` — whether the cluster is secure or not. Value: `true` (if enabled) or `false` (if disabled) • `dare` — whether the cluster is enabled for data at rest encryption or not. Value: `true` (if enabled) or `false` (if disabled) • • `ip` — the IP address of the active CLDB • `id` — the cluster ID • `nodesUsed` — number of nodes in the cluster • `totalNodesAllowed` — number of allowed nodes

last-updated: Apr 11, 2022

Field	Description
volumes	The number and size (in GB) of volumes that are: • Mounted • Unmounted

last-updated: Apr 11, 2022

Field	Description

utilization	The following utilization information:
	• CPU — utilization, total and active. CPU utilization % is calculated as (100% - idle%) on each node and then averaged across all nodes where hoststats is running.
	• Memory — total and active in MB.
	• Disk space — total and active in GB.
	• Compression — compressed and uncompressed data size
	• Tiering — following cluster level tiering information:
	  • `logicalUsed` — The total gigabytes of disk space used (before compression) for the tiered volumes in the cluster.
	  This value does not include erasure-coded backend volumes and cache-volumes.
	  • `replicatedLogicalUsed` — The logical size in gigabytes, of disk used by the tiered volumes and all associated replicas.
	  This value is calculated as follows: *disk space (before compression) used by volume * number of replicas*. This value does not include erasure-coded backend volumes and cache-volumes.
	  • `replicatedTotalUsed` — The total space after compression (in GB) used by tiered volumes and associated snapshots and replicas.
	  This value is calculated as follows: *total space used by volume * number of replicas*. This value does not include EC-backend volumes and cache-volumes since their `replicatedTotalUsed` is already accounted for by the front-end and parent volumes respectively.
	  • `metaDBUsedMB` — The disk space (in MB) used by the metadata volume associated with the tier.
	  • `replicatedMetaDBUsedMB` — The disk space (in MB) used by the replicas of the metadata volume associated with the tier.
	  • `offloaded` — The total physical data (in GB) offloaded to the cold tier.
	  This value is calculated as follows: *amount of data purged from the hot tier (MapR cluster) + amount of data recalled to the hot tier (MapR cluster)*.
	  • `recalled` — The total physical data (in GB) recalled from the cold tier.
	  This value is calculated as follows: *amount of data recalled to the hot tier (MapR cluster) + total amount of disk space used by the cache-volume*.
	  • `cvTotalUsed` — The total disk space (in GB) used by all the cache-volumes. At the volume level, for mirror volumes, this is the total size of the cache volume and its replicas.
	  • `replicatedCvTotalUsed` — The total disk space (in GB) used by all the cache-volumes and associated replicas.

Field	Description
clusterReplication	The following cluster replication information: • `bytesReceived` • `bytesSend`
streamThroughput	The following stream throughput information: • `bytesProduced` • `bytesConsumed`
label_stats	Information about the labels registered and assigned.
services	The number of active, stopped, failed, and total installed services on the cluster, for example: • API server • CLDB • Fileserver • File Migrate • ResourceManager • NodeManager • NFSv4 • hoststats • MAST Gateway
yarn	The following mapreduce information: • Running applications • Queued applications • Number of NodeManagers • Total memory • Total VCores • Total disks • Used memory • Used VCores • Used disks

Examples

Display dashboard information:

CLI

```
# maprcli dashboard info -json
{
```


last-updated: Apr 11, 2022

```
     "timestamp":1599138960056,
        "timeofday":"2020-09-03
06:16:00.056 GMT-0700 AM",
        "status":"OK",
        "total":1,
        "data":[
                {

"version":"6.2.0.0.20200823204949.GA",
                        "cluster":{

"name":"my.cluster.com",

"secure":true,

"dare":false,

"globalPolicyMaster":true,

"ip":"10.163.167.212",

"id":"2812007637544940359",

"nodesUsed":1,

"totalNodesAllowed":-1
                        },
                        "volumes":{

"mounted":{

  "total":17,

  "size":6605
                                },

"unmounted":{

  "total":2,

  "size":1
                                }
                        },
                        "mirrors":{
                                "num
jobs":0,

"active containers":0,

"resync containers":0,

"mirrored datasize mb":0,

"remaining datasize mb":0,

"completion pcnt":0
                        },
                        "utilization":
{
                                "cpu":
{
```

```
  "util":21,
  "total":8,
  "active":1
                                    },
"memory":{
  "total":23911,
  "active":23075
                                    },
"disk_space":{
  "total":287,
  "active":6
                                    },
"compression":{
  "compressed":6,
  "uncompressed":6
                                    },
"tiering":{
  "logicalUsed":0,
  "replicatedLogicalUsed":0,
  "replicatedTotalUsed":0,
  "metaDBUsedMB":0,
  "replicatedMetaDBUsedMB":0,
  "offloaded":0,
  "recalled":0,
  "cvTotalUsed":0,
  "replicatedCvTotalUsed":0,
  "ecOffloaded":0,
  "ecRecalled":0,
  "ecTotalUsed":0
                                    }
                          },
"clusterReplication":{
"bytesReceived":0,
"bytesSend":0
                          },
```

```
"streamThroughput":{
"bytesProduced":46746456005,
"bytesConsumed":46748653475
                        },
                        "services":{
"hbaserest":{
  "active":1,
  "stopped":0,
  "failed":0,
  "total":1
                                },
"hbasethrift":{
  "active":1,
  "standby":0,
  "stopped":0,
  "failed":0,
  "total":1
                                },
"fileserver":{
  "active":1,
  "stopped":0,
  "failed":0,
  "total":1
                                },
"grafana":{
  "active":1,
  "stopped":0,
  "failed":0,
  "total":1
                                },
"cldb":{
  "active":1,
  "stopped":0,
```

```
  "failed":0,
  "total":1
                                    },
"mastgateway":{
  "active":1,
  "stopped":0,
  "failed":0,
  "total":1
                                    },
"opentsdb":{
  "active":1,
  "stopped":0,
  "failed":0,
  "total":1
                                    },
"gateway":{
  "active":1,
  "stopped":0,
  "failed":0,
  "total":1
                                    },
"hoststats":{
  "active":1,
  "stopped":0,
  "failed":0,
  "total":1
                                    },
"collectd":{
  "active":1,
  "stopped":0,
  "failed":0,
  "total":1
                                    },
"apiserver":{
```

```
  "active":1,

  "stopped":0,

  "failed":0,

  "total":1
                                  }
                            }
                      }
          ]
}
```

REST

```
curl -u mapr:mapr -X GET  -k "https://
host:8443/rest/dashboard/info"
{"timestamp":1599139171576,"timeofday"
:"2020-09-03 06:19:31.576 GMT-0700
AM","status":"OK","total":1,"data":
[{"version":"6.2.0.0.20200823204949.GA
","cluster":
{"name":"my.cluster.com","secure":true
,"dare":false,"globalPolicyMaster":tru
e,"ip":"10.163.167.212","id":"28120076
37544940359","nodesUsed":1,"totalNodes
Allowed":-1},"volumes":{"mounted":
{"total":17,"size":6659},"unmounted":
{"total":2,"size":1}},"mirrors":{"num
jobs":0,"active containers":0,"resync
containers":0,"mirrored datasize
mb":0,"remaining datasize
mb":0,"completion
pcnt":0},"utilization":{"cpu":
{"util":30,"total":8,"active":2},"memo
ry":
{"total":23911,"active":23166},"disk_s
pace":
{"total":287,"active":6},"compression"
:
{"compressed":6,"uncompressed":6},"tie
ring":
{"logicalUsed":0,"replicatedLogicalUse
d":0,"replicatedTotalUsed":0,"metaDBUs
edMB":0,"replicatedMetaDBUsedMB":0,"of
floaded":0,"recalled":0,"cvTotalUsed":
0,"replicatedCvTotalUsed":0,"ecOffload
ed":0,"ecRecalled":0,"ecTotalUsed":0}}
,"clusterReplication":
{"bytesReceived":0,"bytesSend":0},"str
eamThroughput":
{"bytesProduced":46802771481,"bytesCon
sumed":46804959666},"services":
{"hbaserest":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"hbasethrift":
{"active":1,"standby":0,"stopped":0,"f
ailed":0,"total":1},"fileserver":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"grafana":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"cldb":
{"active":1,"stopped":0,"failed":0,"to
```

```
tal":1},"mastgateway":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"opentsdb":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"gateway":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"hoststats":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"collectd":
{"active":1,"stopped":0,"failed":0,"to
tal":1},"apiserver":
{"active":1,"stopped":0,"failed":0,"to
tal":1}}}]}
```


dialhome

The `dialhome` commands are used to change the Dial Home status of the cluster:

dialhome ackdial
Acknowledges the most recent Dial Home on the cluster. Permissions required: login

Syntax

CLI

```
maprcli dialhome ackdial
    [ -forDay <date> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/dialhome/ackdial[?<parameters>]`

Parameters

Parameter	Description
`forDay`	Date for which the recorded metrics were successfully dialed home. Accepted values: UTC timestamp or a UTC date in MM/DD/YY format. Default: yesterday

Sample Output

```
# maprcli dialhome ackdial -forDate 5/26/15
dialhome ackdial
     -forDay Date for which the recorded metrics were successfully dialed
home. Accepted values: UTC timestamp in millisecond or a UTC date in
MM/DD/YY format. default: 5/31/15
```

Examples

Acknowledge Dial Home:

CLI

```
maprcli dialhome ackdial
```


last-updated: Apr 11, 2022

REST

```
https://abc.sj.us:8443/rest/dialhome/
ackdial
```

dialhome enable
Enables Dial Home on the cluster. Permissions required: `fc` or a

Syntax

CLI

```
maprcli dialhome enable
    -enable 0|1
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/dialhome/enable`

Parameters

Parameter	Description
`enable`	Specifies whether to enable or disable Dial Home: • 0 - Disable • 1 - Enable

Output

A success or failure message.

Sample output

```
# maprcli dialhome enable -enable 1
enabled
1

# maprcli dialhome status
enabled
1
```

Examples

Enable Dial Home:

CLI

```
maprcli dialhome enable -enable 1
```

REST

```
https://abc.sj.us:8443/rest/dialhome/
enable?enable=1
```

dialhome lastdialed
Displays the date of the last successful Dial Home call. Permissions required: `fc` or a.

Syntax

CLI

```
maprcli dialhome lastdialed
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/dialhome/lastdialed`

Output

The date of the last successful Dial Home call.

Sample output

```
$ maprcli dialhome lastdialed
date
1322438400000
```

Examples

Show the date of the most recent Dial Home:

CLI

```
maprcli dialhome lastdialed
```

REST

```
https://abc.sj.us:8443/rest/dialhome/lastdialed
```

dialhome metrics

Returns a compressed metrics object. Permissions required: login.

Syntax

CLI

```
maprcli dialhome metrics [ -forDay <date> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/dialhome/metrics`

Parameters

Parameter	Description
`forDay`	Date for which the recorded metrics were successfully dialed home. Accepted values: UTC timestamp or a UTC date in MM/DD/YY format. Default: yesterday

last-updated: Apr 11, 2022

Output

Sample output

```
$ maprcli dialhome metrics
metrics
[B@48067064
```

Examples

Show the Dial Home metrics:

CLI

```
maprcli dialhome metrics
```

REST

```
https://abc.sj.us:8443/rest/dialhome/
metrics
```

dialhome status
Displays the Dial Home status. Permissions required: login.

Syntax

CLI

```
maprcli dialhome status
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/dialhome/status`

Output

The current Dial Home status.

Sample output

```
$ maprcli dialhome status
enabled
1
```

Examples

Display the Dial Home status:

CLI

```
maprcli dialhome status
```

REST

```
https://abc.sj.us:8443/rest/dialhome/
status
```

disk
Lists disk parameters.

Disk Fields

The following table shows the fields displayed in the output of the disk list and disk listall commands. You can choose which fields (columns) to display and sort in ascending or descending order by any single field.

availablespace	Terse Name: dsa Description: Disk space available, in MB.
diskname	Terse Name: n Description: Name of the disk or partition
error	Terse Name: err Description: Disk error message, in English. Only sent if **status** is `1`.  **Note:** This message is **not** translated.
failuretime	Terse Name: ft Description: Disk failure time, This field is applicable only for MapR disks. Only sent if **status** is `1`.
firmwareversion	Terse Name: fw Description: Firmware version
fstype	Terse Name: fs Description: File system type
hostname	Terse Name: hn Description: Hostname of the node that owns this disk/partition
modelnum	Terse Name: mn Description: The model number of the disk
mount	Terse Name: mt Description: Disk mount status • `0` = unmounted • `1` = mounted
powerstatus	Terse Name: pst Description: Disk power status: • `0` = Active/idle (normal operation) • `1` = Standby (low power mode) • `2` = Sleeping (lowest power mode, drive is completely shut down)
serialnum	Terse Name: sn Description: Serial number of the disk
status	Terse Name: st Description: Disk status: • `0` = Good • `1` = Bad disk

last-updated: Apr 11, 2022

	• 2 = Offline disk
totalspace	Terse Name: dst Description: Total disk space, in MB
usedspace	Terse Name: dsu Description: Disk space used, in MB
vendor	Terse Name: ven Description: Name of the disk vendor

Related concepts

node on page 1694
Manages nodes in the cluster

Related reference

disk add on page 1602
Adds one or more disks to the specified node. Permissions required: `fc` or `a`.

volume create on page 1931
Creates a volume.

node list on page 1705
Lists nodes in the cluster.

dump volumeinfo on page 1637
Returns information about volumes and the associated containers. For JSON formatted output, use the -json option from the command line.

configure.sh on page 2053
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

disk add

Adds one or more disks to the specified node. Permissions required: `fc` or `a`.

Syntax

CLI

```
maprcli disk add
    -disks <disk names>
    -host <host>
  [ -cluster <cluster> ]

  [ -stripeWidth <stripe-width> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/disk/add?<parameters>`

Parameters

Parameter: `cluster`	*Default Value*: No default value Possible Values: Any valid cluster.

	Description: The cluster on which to add disks. If not specified, the default is the current cluster.
Parameter: `disks`	*Default Value*: No default value Possible Values: Any valid disk names Description: A comma-separated list of disk names. Examples: • /dev/sdc • /dev/sdd,/dev/sde,/dev/sdf
Parameter: `host`	*Default Value*: No default value Possible Values: Any valid host or IP Description: The hostname or IP address of the machine on which to add the disk.
Parameter: `stripeWidth`	*Default Value*: No default value Possible Values: Any integer Description: The number of disks per storage pool.

Output

Output Fields

Field	Description
`ip`	The IP address of the machine that owns the disk(s).
`disk`	The name of a disk or partition. Example **sca** or **sca/sca1**
`all`	The string `all`, meaning all unmounted disks for this node.

Examples

Add a disk:

CLI

```
maprcli disk add -disks /dev/
sda1 -host 10.250.1.79
```

REST

```
https://abc.sj.us:8443/rest/disk/add?
disks=["/dev/sda1"]&host="10.250.1.79"
```

Related concepts

node on page 1694
Manages nodes in the cluster

Related reference

volume create on page 1931
Creates a volume.

node list on page 1705
Lists nodes in the cluster.

configure.sh on page 2053

last-updated: Apr 11, 2022

Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

disk list
Lists the disks on a node.

Syntax

CLI

```
maprcli disk list
  -host name/ip
  [ -system 1/0 ]
  [ -output <terse|verbose>. default:
verbose ]


  [ -sortby <hostname|diskname|
mount|vendor|modelnum|serialnum|
firmwareversion|totalspace|usedspace|
            availablespace|
fstype|powerstatus|status|errormsg|
storagepoolid|failuretime> ]
  [ -sortorder <asc|desc> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/disk/list?<parameters>`

Parameters

Parameter	Description
`host`	The node on which to list the disks.
`output`	Whether the output should be `terse` or `verbose`. Default is `verbose`.
`sortby`	Specifies one of the following attributes to sort the list of disks by: `hostname`, `diskname`, `mount`, `vendor`, `modelnum`, `serialnum`, `firmwareversion`, `totalspace`, `usedspace`, `availablespace`, `fstype`, `powerstatus`, `status`, `errormsg`, `storagepoolid`, `failuretime`
`sortorder`	The order to sort the results by. Value can be: • `asc` - for ascending order • `desc` - for descending order

Parameter	Description
`system`	Show only operating system disks: • 0 - shows only MapR File System disks • 1 - shows only operating system disks • Not specified - shows both MapR File System and operating system disks

Output

Information about the specified disks. See the Disk Fields table.

Sample Output

```
# maprcli disk list -host 10.10.82.23 -output terse
mn            pst      sp  fw   mt  fs       dsu  n          st  dsa
dst     hn           vn
Virtual_disk  running      1.0  1   ext4     77   /dev/sda1  0   423
500     10.10.82.23  VMware
Virtual_disk  running      1.0  0                /dev/sda2  0
15883   10.10.82.23  VMware
Virtual_disk  running  1   1.0  0   MapR-FS  608  /dev/sdb   0   101792
102400  10.10.82.23  VMware
                           0                    /dev/dm-0  0
7864    10.10.82.23
                           0   swap             /dev/dm-1  0
8016    10.10.82.23
```

Examples

List disks on a host:

CLI

```
maprcli disk list -host 10.10.100.22
```

REST

```
https://abc.sj.us:8443/rest/disk/list?
host=10.10.100.22
```

Lists disks in ascending order sorted by fstype:

CLI

```
# maprcli disk list -host
atsqa4-161.qa.lab -sortby
fstype -sortorder asc
modelnum      mount  totalspace
diskname   hostname
firmwareversion  vendor
availablespace  storagepoolid
powerstatus  usedspace  fstype
status
ST91000640NS  1      1024        /dev/
sda1  atsqa4-161.qa.lab
SN03             ATA
895
running      129        ext3
0
ST91000640NS  0
```


last-updated: Apr 11, 2022

```
953869      /dev/sdb
atsqa4-161.qa.lab  SN03
ATA     953530
1               running
339         MapR-FS  0
ST91000640NS  0
953869      /dev/sdc
atsqa4-161.qa.lab  SN03
ATA     953530
1               running
339         MapR-FS  0
ST91000640NS  0
953869      /dev/sde
atsqa4-161.qa.lab  SN03
ATA     953371
2               running
498         MapR-FS  0
ST91000640NS  0
953869      /dev/sdf
atsqa4-161.qa.lab  SN03
ATA     953371
2               running
498         MapR-FS  0
ST91000640NS  0
953869      /dev/sdd
atsqa4-161.qa.lab  SN03
ATA     953530
1               running
339         MapR-FS  0
               0      13024       /dev/
dm-0
atsqa4-161.qa.lab

                         swap
0
ST91000640NS  0      952844      /dev/
sda2  atsqa4-161.qa.lab
SN03
ATA
 running
0
               0      51200       /dev/
dm-1
atsqa4-161.qa.lab


0
               0      358400      /dev/
dm-2  atsqa4-161.qa.lab
```

Related concepts

node on page 1694
Manages nodes in the cluster

disk on page 1600
Lists disk parameters.

Related reference

disk add on page 1602
Adds one or more disks to the specified node. Permissions required: `fc` or `a`.

volume create on page 1931

last-updated: Apr 11, 2022

Creates a volume.

node list on page 1705
Lists nodes in the cluster.

dump volumeinfo on page 1637
Returns information about volumes and the associated containers. For JSON formatted output, use the -json option from the command line.

configure.sh on page 2053
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

disk listall
Lists all disks.

Syntax

CLI

```
maprcli disk listall
    [ -cluster <cluster> ]
    [ -limit <limit>]
    [ -output terse|verbose ]
    [ -start <offset>]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/disk/listall[?<parameters>]`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`limit`	The number of rows to return, beginning at start. Default: 0
`output`	Always the string `terse`.
`start`	The offset from the starting row according to sort. Default: 0

Output

Information about all disks. See the Disk Fields table.

Sample Output

```
# maprcli disk listall -output terse
mn            pst      sp  fw   mt  fs       dsu  n          st  dsa
dst     hn            vn
Virtual_disk  running      1.0  1   ext4     77   /dev/sda1  0   423
500     centos22.lab  VMware
Virtual_disk  running      1.0  0                /dev/sda2  0
15883   centos22.lab  VMware
Virtual_disk  running  1   1.0  0   MapR-FS  478  /dev/sdb   0   101922
102400  centos22.lab  VMware
```


last-updated: Apr 11, 2022

```
                                0                 /dev/dm-0  0
7864    centos22.lab
                                0   swap          /dev/dm-1  0
8016    centos22.lab
Virtual_disk  running      1.0  1   ext4     77   /dev/sda1  0   423
500     centos23.lab  VMware
Virtual_disk  running      1.0  0                 /dev/sda2  0
15883   centos23.lab  VMware
Virtual_disk  running  1   1.0  0   MapR-FS  608  /dev/sdb   0   101792
102400  centos23.lab  VMware
                                0                 /dev/dm-0  0
7864    centos23.lab
                                0   swap          /dev/dm-1  0
8016    centos23.lab
Virtual_disk  running      1.0  1   ext4     77   /dev/sda1  0   423
500     centos29.lab  VMware
Virtual_disk  running      1.0  0                 /dev/sda2  0
15883   centos29.lab  VMware
Virtual_disk  running  1   1.0  0   MapR-FS  757  /dev/sdb   0   15627
16384   centos29.lab  VMware
                                0                 /dev/dm-0  0
7864    centos29.lab
                                0   swap          /dev/dm-1  0
8016    centos29.lab
```

Examples

List all disks:

CLI

```
maprcli disk listall
```

REST

```
https://abc.sj.us:8443/rest/disk/
listall
```

disk remove

Removes a disk from MapR File System. Permissions required: `fc` or `a`.

The `disk remove` command does not remove a disk containing unreplicated data, unless forced. To force disk removal, specify `-force` with the value `1` or `true`.

Note:

- Use the `-force 1`, or the equivalent `-force true` option only if you are sure that you do not need the data on the disk. This option removes the disk without regard to the replication factor or other data protection mechanisms, and may result in permanent data loss.
- Removing a disk in the storage pool that contains Container ID 1 stops the cluster. Container ID 1 contains CLDB data for the master CLDB. Run `disk remove` without the `-force 1`, or the equivalent `-force true` option first, and examine the warning messages to make sure that you are not removing the disk with Container ID 1. If you try to remove a disk associated with the storage pool that contains Container ID 1, you receive an error message similar to the following:

```
ERROR (151) -  Failed operation for disk /dev/sdb, Operation may
bring
down cluster due to loss of cluster meta-data.
```

Tip: If necessary, run the following command for information on the disk associated with the storage pool that contains Container ID 1:

```
/opt/mapr/server/mrconfig info dumpcontainers  | grep cid:1
```

The command output may look similar to the following:

```
cid:1 volid:1 sp:SP1:/dev/sdb
spid:82380c287085486f0058112ecf016b76
prev:0 next:0 issnap:0 isclone:0 deleteinprog:0 fixedbyfsck:0
stale:0
querycldb:0 resyncinprog:0 shared:0 owned:206 logical:206
snapusage:0
snapusageupdated:1 ismirror:0 isrwmirrorcapable:0 role:1
maxUniq:2100150
isResyncSnapshot:0 snapId:0 port:5660
```

To safely remove such a disk, first perform a CLDB Failover to make one of the other CLDB nodes the primary CLDB, and then remove the disk as normal.

Syntax

CLI

```
/opt/mapr/bin/maprcli disk remove
    -host name/ip
    -disks comma-separated list of
disks
    [ -force <true|false OR 1|0>.
Required to remove the disk when
errors have been reported; otherwise,
the command behaves like a test
remove when errors are reported.
If -force is set to false and there
are no errors, the disk is removed.
Default: false ]
    [ -cluster cluster_name ]
```

REST

Request Type	POST

last-updated: Apr 11, 2022

Request URL	`http[s]://<host>:<port>/rest/disk/remove?<parameters>`

Parameters

Parameter	Description
`host`	The hostname or ip address of the node from which to remove the disk.
`disks`	A list of disks in the form: `["disk"]or["disk","disk","disk"...]or[]`
`force`	Whether to force disk removal. • `0` or `false` (default) - do not remove the disk or disks if there is unreplicated data on the disk • `1` or `true` - remove the disk or disks regardless of data loss, or other consequences
`cluster`	The cluster on which to run the command.

Output

Output Fields

Field	Description
`disk`	The name of a disk or partition. Example: `sca` or `sca/sca1`
`all`	The string `all`, meaning all unmounted disks attached to the node.
`disks`	A comma-separated list of disks which have non-replicated volumes. For example, "sca" or "sca/sca1,scb"

Examples

Remove a disk:

CLI

```
/opt/mapr/bin/maprcli disk
remove -disks /dev/sda
```

REST

```
https://abc.sj.us:8443/rest/disk/
remove?disks=/dev/sda
```

dump
Returns key information about volumes, containers, storage pools, and MapR cluster services for debugging and troubleshooting.

dump balancerinfo

Returns detailed information about the storage pools on a cluster. If there are any active container moves, the command returns information about the source and destination storage pools.

The `maprcli dump balancerinfo` command enables you to see how much space is used in storage pools and to track active container moves. For best results, use the `-json` option when running `dump balancerinfo` from the command line.

The *disk space balancer* is a tool that balances disk space usage on a cluster by moving containers between storage pools. Whenever a storage pool is over 70% full (or a threshold defined by the `cldb.balancer.disk.threshold.percentage` parameter), the disk space balancer distributes containers to other storage pools that have lower utilization than the average for that cluster. The disk space balancer aims to ensure that the percentage of space used on all the disks in the node is similar. For more information, see Disk Space Balancer.

Syntax

CLI

```
maprcli dump balancerinfo
    [-cluster <cluster name>]
```

REST

N/A

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.

Output

The `maprcli dump balancerinfo` command returns detailed information about the storage pools on a cluster. If there are any active container moves, the command returns information about the source and destination storage pools.

```
# maprcli dump balancerinfo -cluster my.cluster.com -json
{
        "timestamp":1337036566035,
        "status":"OK",
        "total":187,
        "data":[
{
                        "spid":"4bc329ce06752062004fa1a537abcdef",
                        "fsid":5410063549464613987,
                        "ip:port":"10.50.60.72:5660-",
                        "capacityMB":1585096,
                        "usedMB":1118099,
                        "percentage":70,
                        "fullnessLevel":"AboveAverage",
                        "inTransitMB":0,
                        "outTransitMB":31874
                },
{
                        "spid":"761fec1fabf32104004fad9630ghijkl",
                        "fsid":3770844641152008527,
                        "ip:port":"10.50.60.73:5660-",
```


last-updated: Apr 11, 2022

```
                        "capacityMB":1830364,
                        "usedMB":793679,
                        "percentage":47,
                        "fullnessLevel":"BelowAverage",
                        "inTransitMB":79096,
                        "outTransitMB":0
                },

                ....


{
                        "containerid":4034,
                        "sizeMB":16046,
                        "From fsid":5410063549464613987,
                        "From IP:Port":"10.50.60.72:5660-",
                        "From SP":"4bc329ce06752062004fa1a537abcefg",
                        "To fsid":3770844641152008527,
                        "To IP:Port":"10.50.60.73:5660-",
                        "To SP":"761fec1fabf32104004fad9630ghijkl"
                },
```

Output fields

Field	Description
spid	The unique ID number of the storage pool.
fsid	The unique ID number of the file server. The FSID identifies an MapR File System instance or a node that has MapR File System running in the cluster. Typically, each node has a group of storage pools, so the same FSID will correspond to multiple SPIDs.
ip:port	The host IP address and MapR File System port.
capacityMB	The total capacity of the storage pool (in MB).
usedMB	The amount of space used on the storage pool (in MB).
percentage	The percentage of the storage pool currently utilized. A ratio of the space used (*usedMB*) to the total capacity (*capacityMB*) of the storage pool.
fullnessLevel	The fullness of the storage pool relative to the fullness of the rest of the cluster. Possible values are *OverUsed*, *AboveAverage*, *Average*, *BelowAverage*, and *UnderUsed*. For more information, see Monitoring storage pool space usage below.
inTransitMB	The amount of data (in MB) that the disk space balancer is currently moving into a storage pool.
outTransitMB	The amount of data (in MB) that the disk space balancer is currently moving out of a storage pool.

The following fields are returned only if the disk space balancer is actively moving one or more containers at the time the command is run.

Field	Description
containerid	The unique ID number of the container.
sizeMB	The amount of data (in MB) being moved.
From fsid	The FSID (file server ID number) of the source file server.
From IP:Port	The IP address and port number of the source node.
From SP	The SPID (storage pool ID) of the source storage pool.
To fsid	The FSID (file server ID number) of the destination file server.
To IP:Port	The IP address and port number of the destination node.
To SP	The SPID (storage pool ID number) of the destination storage pool.

Examples

Monitoring storage pool space usage

You can use the `maprcli dump balancerinfo` command to monitor space usage on storage pools.

```
# maprcli dump balancerinfo -json

....
{
                        "spid":"4bc329ce06752062004fa1a537abcefg",
                        "fsid":5410063549464613987,
                        "ip:port":"10.50.60.72:5660-",
                        "capacityMB":1585096,
                        "usedMB":1118099,
                        "percentage":70,
                        "fullnessLevel":"AboveAverage",
                        "inTransitMB":0,
                        "outTransitMB":31874
                },
```

Tracking active container moves

Using the `maprcli dump balancerinfo` command you can monitor the activity of the disk space balancer. Whenever there are active container moves, the command returns information about the source and destination storage pools.

```
# maprcli dump balancerinfo -json
      ....
      {
           "containerid":7840,
           "sizeMB":15634,
           "From fsid":8081858704500413174,
           "From IP:Port":"10.50.60.64:5660-",
           "From SP":"9e649bf0ac6fb9f7004fa19d20rstuvw",
           "To fsid":3770844641152008527,
           "To IP:Port":"10.50.60.73:5660-",
           "To SP":"fefcc342475f0286004fad963flmnopq"
       }
```


last-updated: Apr 11, 2022

The example shows that a container (7840) is being moved from a storage pool on node 10.50.60.64 to a storage pool on node 10.50.60.73.

> **Tip:** You can use the storage pool IDs (SPIDs) to search the CLDB and MapR File System logs for activity (balancer moves, container moves, creates, deletes, etc.) related to specific storage pools.

dump balancermetrics
Returns a cumulative count of container moves and MB of data moved between storage pools.

The `maprcli dump balancermetrics` command returns a cumulative count of container moves and MB of data moved between storage pools. You can run this command periodically to determine how much data has been moved by the disk space balancer between two intervals. For best results, use the `-json` option when running `dump balancermetrics` from the command line.

The *disk space balancer* is a tool that balances disk space usage on a cluster by moving containers between storage pools. Whenever a storage pool is over 70% full (or it reaches a threshold defined by the `cldb.balancer.disk.threshold.percentage` parameter), the disk space balancer distributes containers to other storage pools that have lower utilization than the average for that cluster. The disk space balancer aims to ensure that the percentage of space used on all the disks in the node is similar. For more information, see Disk Space Balancer.

Syntax

CLI

```
maprcli dump balancermetrics
    [-cluster <cluster name>]
```

REST

N/A

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.

Output

The `maprcli dump balancermetrics` command returns a cumulative count of container moves and MB of data moved between storage pools since the current CLDB became the master CLDB.

```
# maprcli dump balancermetrics -json
{
    "timestamp":1337770325979,
    "status":"OK",
    "total":1,
    "data":[
        {
            "numContainersMoved":10090,
            "numMBMoved":3147147,
            "timeOfLastMove": "Wed May 23 03:51:44 PDT 2012"
        }
    ]
}
```

Output fields

Field	Description
numContainersMoved	The number of containers moved between storage pools by the disk space balancer.
numMBMoved	The total MB of data moved between storage pools on the cluster.
timeOfLastMove	The date and time of most recent container move.

Example

CLI

```
maprcli dump balancermetrics -cluster
10.10.82.23 -json
```

dump cldbnodes

Lists the nodes that contain *container location database* (CLDB) data.

The CLDB is a service running on one or more MapR nodes that maintains the location of cluster containers, services, and other information. The CLDB automatically replicates its data to other nodes in the cluster, preserving at least two (and generally three) copies of the CLDB data. If the CLDB process dies, it is automatically restarted on the node.

Syntax

CLI

```
maprcli dump cldbnodes
    [-cluster <cluster name>]
    -zkconnect <ZooKeeper Connect
String>
    -json | -long
```

Note: For best results, use the `-json` option from the command line.

REST

N/A

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.
`zkconnect`	A ZooKeeper connect string, which specifies a list of the hosts running ZooKeeper, and the port to use on each, in the format: '<host>[:<port>][,<host>[:<port>]...]'. To obtain zookeeper connection strings, use the `maprcli node listzookeepers` command.
`json | long`	This command returns multiple levels of data. You need to specify either JSON format or "long" format to see the full output.

last-updated: Apr 11, 2022

Output

The `maprcli dump cldbnodes` command returns the IP address and port number of the CLDB nodes on the cluster.

```
$ maprcli dump cldbnodes -zkconnect centos23.lab:5181 -json
{
    "timestamp":1433270634424,
    "timeofday":"2015-06-02 06:43:54.424 GMT+0000",
    "status":"OK",
    "total":1,
    "data":[
        {
            "valid":[
                "10.10.82.23:5660-",
                "10.10.82.28:5660-",
                "10.10.82.29:5660-",
                "10.10.82.22:5660-"
            ]
        }
    ]
}
```

Example

CLI

```
maprcli dump cldbnodes -zkconnect
centos23.lab:5181 -json
```

dump containerinfo
Returns detailed information about one or more specified containers.

A *container* is a unit of sharded storage in a MapR cluster. Every container in a MapR volume is either a *name container* or a *data container*.

> **Tip:** For an explanation of sharding, see the Configuring the Chunk Size topic.

The name container is the first container in a volume and holds that volume's namespace and file chunk locations. Depending on its replication role, a name container may be either a *master container* (part of the original copy of the volume) or a *replica container* (one of the replicas in the replication chain).

Every data container is either a *master container*, an *intermediate container*, or a *tail container*.

Syntax

CLI

```
maprcli dump containerinfo
    [-cluster <cluster name>]
    -ids <id1,id2,id3 ...>
```

Note: For best results, use the `-json` option from the command line.

REST

N/A

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.
`ids`	Specifies one or more container IDs. Container IDs are comma separated. The `maprcli dump containers` command provides the container ID required for `-ids` parameter.

Output

The `maprcli dump containerinfo` command returns information about one or more containers.

```
# maprcli dump containerinfo -ids 1  -json
{
        "timestamp":1507024362685,
        "timeofday":"2017-10-03 02:52:42.685 GMT-0700 AM",
        "status":"OK",
        "total":1,
        "data":[
                {
                        "ContainerId":1,
                        "Epoch":9,
                        "Master":"10.10.105.35:5660--9-VALID",
                        "ActiveServers":{
                                "IP":[

                                        "10.10.105.35:5660--9-VALID",
                                        "10.10.105.36:5660--9-VALID",
                                        "10.10.105.37:5660--9-VALID"
                                ],
                                "ExtIP":[

"10.10.104.35:5660-10.10.104.35:5692",

"10.10.104.36:5660-10.10.104.36:5692",

"10.10.104.37:5660-10.10.104.37:5692"
                                ]
                        },
                        "InactiveServers":{

                        },
                        "UnusedServers":{

                        },
                        "OwnedSizeMB":"0 MB",
                        "SharedSizeMB":"0 MB",
                        "LogicalSizeMB":"0 MB",
                        "TotalSizeMB":"0 MB",
                        "NameContainer":"true",
                        "CreatorContainerId":0,
                        "CreatorVolumeUuid":"",
                        "UseActualCreatorId":false,
                        "VolumeName":"mapr.cldb.internal",
                        "VolumeId":1,
                        "VolumeReplication":3,
```


last-updated: Apr 11, 2022

```
                        "NameSpaceReplication":3,
                        "VolumeMounted":false,
                        "AccessTime":"September 29, 2017"
                }
        ]
}
```

Output fields

Field	Description
ContainerID	The unique ID number for the container.
Epoch	A sequence number that indicates the most recent copy of the container. The CLDB uses the epoch to ensure that an out-of-date copy cannot become the master for the container.
Master	The physical IP address and port number of the *master copy*. The master copy is part of the original copy of the volume.
ActiveServers	The physical IP address and port number of each active node on which the container resides.
InactiveServers	The physical IP address and port number of each inactive node on which the container resides.
UnusedServers	The physical IP address and port number of servers from which no "heartbeat" has been received for quite some time.
OwnedSizeMB	The size on disk (in MB) dedicated to the container.
SharedSizeMB	The size on disk (in MB) shared by the container.
LogicalSizeMB	The logical size on disk (in MB) of the container.
TotalSizeMB	The total size on disk (in MB) allocated to the container. Combines the Owned Size and Shared Size.
Mtime	The time of the last modification to the contents of the container.
NameContainer	Indicates if the container is the *name container* for the volume. If `true`, the container holds the volume's namespace information and file chunk locations.
VolumeName	The name of the volume.
VolumeId	The unique ID number of the volume.
VolumeReplication	The *replication factor*, the number of copies of a volume excluding the original.
VolumeMounted	Indicates whether the volume is mounted. If `true`, the volume is currently mounted. If `false`, the volume is not mounted.

last-updated: Apr 11, 2022

Example

CLI

```
maprcli dump containerinfo -ids
2049  -json
```

dump cldbmetainfo

Prints metadata from the *container location database* (CLDB) tables.

Syntax

CLI

```
maprcli dump cldbmetainfo
    -json
```

Note: For formatted results, use the `-json` option from the command line.

REST

N/A

Parameters

Parameter	Description
`json`	Returns formatted output

Output

The `maprcli dump cldmetainfo` command lists meta information from the CLDB tables, For an explanation of the output fields, see fid stat on page 1657.

```
$ maprcli dump cldbmetainfo -json
{
    "timestamp":1433270634424,
    "timeofday":"2020-06-02 06:43:54.424 GMT+0000",
    "status":"OK",
    "total":2,
    "data":[
    {
                        "name":"cntrSzTable7",
                        "type":"FTKvstore",
                        "parent fid":"<parentCID>.32.131332",
                        "fid":"1.97.131462",
                        "size":7,
                        "nblocks":1,
                        "lblocks":0,
                        "compression":"off",
                        "deleteFlags":"DeleteTypeNone",
                        "atime":1581839467,
                        "mtime":1581839467,
                        "mode":"660",
                        "uid":1000,
                        "gid":1000,
                        "nlink":1,
                        "xattrInum":0,
                        "version":3149249,
                        "networkencryption":false,
                        "diskflush":false,
                        "nlevels":1
                },
                {
```


last-updated: Apr 11, 2022

```
                        "name":"containerLocationTable12",
                        "type":"FTKvstore",
                        "parent fid":"<parentCID>.32.131332",
                        "fid":"1.58.131384",
                        "size":0,
                        "nblocks":0,
                        "lblocks":0,
                        "compression":"off",
                        "deleteFlags":"DeleteTypeNone",
                        "atime":1581839467,
                        "mtime":1581839467,
                        "mode":"660",
                        "uid":1000,
                        "gid":1000,
                        "nlink":1,
                        "xattrInum":0,
                        "version":1048603,
                        "networkencryption":false,
                        "diskflush":false,
                        "nlevels":0
                }
        ]
}
```

Example

CLI

```
maprcli dump cldbmetainfo -json
```

dump containers
Returns information about containers in a cluster.

This command provides information about containers based on the following `-type` criteria:

- offline - Returns information about containers that have no valid copies online. This command is useful when you need to find out exactly what data is offline (for example, when a "volume data unavailable" alarm is raised).
- resync - Returns information about containers that are resynchronizing.
- bm - Returns information about containers that are becoming master but are not yet master.
- unused - Returns information about containers that are unused.
- waiting - Returns information about containers that are waiting for a role.

A *container* is a unit of sharded storage in a MapRMapR Data Platform cluster. Every container in a MapRMapR Data Platform volume is either a *name container* or a *data container*. The name container is the first container in a volume and holds that volume's namespace and file chunk locations. Depending on its replication role, a name container may be either a *master container* (part of the original copy of the volume) or a *replica container* (one of the replicas in the replication chain).

Every data container is either a *master container*, an *intermediate container*, or a *tail container*.

Syntax

CLI

```
maprcli dump containers
    [ -cluster <cluster_name> ]
     -type offline|resync|bm|waiting|
unused
```

Note: For best results, use the `-json` option from the command line.

REST N/A

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.
`type`	Specifies the type of information that is returned about the containers: • `offline` - Returns information about containers that have no valid copies online. • `resync` - Returns information about containers that are resynchronizing. • `bm` - Returns information about containers that are becoming master but are not yet master. • `unused` - Returns information about containers that are unused. • `waiting` - Returns information about containers that are waiting for a role.

Output fields

Field	Description
ContainerID	The unique ID number for the container.
Epoch	A sequence number that indicates the most recent copy of the container. The CLDB uses the epoch to ensure that an out-of-date copy cannot become the master for the container.
Master	The physical IP address and port number of the *primary copy*. The primary copy is part of the original copy of the volume.
ActiveServers	The physical IP address and port number of each active node on which the container resides.
InactiveServers	The physical IP address and port number of each inactive node on which the container resides.
UnusedServers	The physical IP address and port number of servers from which no "heartbeat" has been received for quite some time.
OwnedSizeMB	The size on disk (in MB) dedicated to the container.

last-updated: Apr 11, 2022

Field	Description
SharedSizeMB	The size on disk (in MB) shared by the container.
LogicalSizeMB	The logical size on disk (in MB) of the container.
TotalSizeMB	The total size on disk (in MB) allocated to the container. Combines the Owned Size and Shared Size.
Mtime	The time of the last modification to the contents of the container.
NameContainer	Indicates if the container is the *name container* for the volume. If `true`, the container holds the volume's namespace information and file chunk locations.
VolumeName	The name of the volume.
VolumeId	The unique ID number of the volume.
VolumeReplication	The *replication factor*, the number of copies of a volume excluding the original.
VolumeMounted	Indicates whether the volume is mounted. If `true`, the volume is currently mounted. If `false`, the volume is not mounted.

Example

CLI

```
maprcli dump containers -type
offline -cluster my.cluster -json
```

Output Samples

The following `maprcli dump containers -type offline` command returns information about all offline containers.

```
# maprcli dump containers -type offline -json
{
        "timestamp":1348174731389,
        "status":"OK",
        "total":11,
        "data":[
                {
                        "ContainerId":2060,
                        "Epoch":3,
                        "Master":"unknown ip (0)-0-VALID",
                        "ActiveServers":{

                        },
                        "InactiveServers":{

                        },
                        "UnusedServers":{
                                "IP:Port":"10.10.20.39:5660--3"
                        },
                        "OwnedSizeMB":"0 MB",
                        "SharedSizeMB":"0 MB",
```

```
                        "LogicalSizeMB":"0 MB",
                        "TotalSizeMB":"0 MB",
                        "NameContainer":"true"
                },
                {
                        "ContainerId":2185,
                        "Epoch":3,
                        "Master":"unknown ip (0)-0-VALID",
                        "ActiveServers":{

                        },
                        "InactiveServers":{

                        },
                        "UnusedServers":{
                                "IP:Port":"10.10.20.39:5660--3"
                        },
                        "OwnedSizeMB":"0 MB",
                        "SharedSizeMB":"0 MB",
                        "LogicalSizeMB":"0 MB",
                        "TotalSizeMB":"0 MB",
                        "NameContainer":"false"
                },
...
```

The following `maprcli dump containers -type resync` command returns information about containers that are resyncronizing.

```
# maprcli dump containers -type resync -json
{
        "timestamp":1438666159569,
        "timeofday":"2015-08-03 10:29:19.569 GMT-0700",
        "status":"OK",
        "total":1,
        "data":[
                {
                        "InstanceCount":1,
                        "ContainerId":2242,
                        "Epoch":4,
                        "Master":"10.10.103.30:5660--4-VALID",
                        "ActiveServers":{
                                "IP:Port":[
                                        "10.10.103.30:5660--4-VALID",
                                        "10.10.103.28:5660--4-VALID",
                                        "10.10.103.29:5660--3-RESYNC"
                                ]
                        },
                        "InactiveServers":{

                        },
                        "UnusedServers":{

                        },
                        "OwnedSizeMB":"0 MB",
                        "SharedSizeMB":"0 MB",
                        "LogicalSizeMB":"0 MB",
                        "TotalSizeMB":"0 MB",
                        "NameContainer":"true",
                        "CreatorContainerId":0,
                        "CreatorVolumeUuid":""
                }
        ]
```


last-updated: Apr 11, 2022

dump replicationmanagerinfo

Returns information about which containers are under or over replicated in a specified volume.

For each container, the command displays the current state of that container.

Syntax

CLI

```
maprcli dump replicationmanagerinfo
     [-cluster <cluster name>]
     -volumename <volume name>
```

Note: For best results, use the `-json` option from the command line.

REST

N/A

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.
`volumename`	Specifies the name of the volume. To obtain a volume name, use the `maprcli volume list` command.

Output

The `maprcli dump replicationmanagerinfo` returns information about volumes and the containers on those volumes including the nodes on which the containers have been replicated and the space allocated to each container. If replication activity is not underway when the maprcli command is executed, no container information is included. If replication activity is underway, details of containers are listed.

```
# maprcli dump replicationmanagerinfo -volumename mapr.metrics  -json
{
    "timestamp":1433449934381,
    "timeofday":"2015-06-04 08:32:14.381 GMT+0000",
    "status":"OK",
    "total":1,
    "data":[
        {
            "VolumeName":"mapr.metrics",
            "VolumeId":54955151,
            "VolumeTopology":"/data",
            "VolumeUsedSizeMB":0,
            "VolumeReplication":3,
            "VolumeMinReplication":2,
            "MirrorThrottle":true,
            "AccessTime":"Thu Jun 04 16:57:58 UTC 2015",
            "limitSpread":true
        },
                {
                        "ContainerId":2053,
                        "Epoch":9,

"Master":"10.250.1.15:5660-172.16.122.1:5660-192.168.115.1:5660--9-VALID",
```


last-updated: Apr 11, 2022

```
                        "ActiveServers":{


"IP:Port":"10.250.1.15:5660-172.16.122.1:5660-192.168.115.1:5660--9-VALID"
                        },
                        "InactiveServers":{

                        },
                        "UnusedServers":{

                        },
                        "OwnedSizeMB":"1 MB",
                        "SharedSizeMB":"0 MB",
                        "LogicalSizeMB":"1 MB",
                        "Mtime":"Mon Apr 30 16:40:41 PDT 2012",
                        "NameContainer":"true"
                }
        ]
}
```

Output fields

Field	Description
VolumeName	Indicates the name of the volume.
VolumeId	Indicates the ID number of the volume.
VolumeTopology	The volume topology corresponds to the node topology of the rack or nodes where the volume resides. By default, new volumes are created with a topology of / (root directory). For more information, see Volume Topology.
VolumeUsedSizeMB	The size on disk (in MB) of the volume.
VolumeReplication	The desired replication factor, the number of copies of a volume excluding the original. The default value is *3*.
VolumeMinReplication	The minimum replication factor, the number of copies of a volume (excluding the original) that should be maintained by the MapR cluster for normal operation. When the replication factor falls below this minimum, writes to the volume are disabled. The default value is *2*.
MirrorThrottle	Specifies whether mirror throttling is enabled (true) or disabled (false). Throttling is set on the source volume and applies to all its mirrors. This property was introduced in version 4.0.2.
AccessTime	A value that can be used to determine which volumes are accessed regularly. This value is updated every 6 hours with the last time that an operation occurred on the volume. The access time is not updated for changes to volume properties, creation of a snapshot, or synchronization between a volume and a mirror. However, the volume access time is updated the first time you upgrade to a MapR version that includes this property. This property was introduced in version 4.0.2.

Field	Description
limitSpread	An internal flag for MapR volumes to control the growth of volume in terms of number of containers. When this flag is set, cldb tries to limit the number of new containers created depending on the present size of volume. If volume size (data in volume) is small, cldb tries to reuse space in existing containers to avoid the creation of new containers. This helps reduce the wasting of containers IDs in an environment that has small volumes.
ContainerId	The unique ID number for the container.
Epoch	A sequence number that indicates the most recent copy of the container. The CLDB uses the epoch to ensure that an out-of-date copy cannot become the master for the container.
Master	The physical IP address and port number of the *master copy*. The master copy is part of the original copy of the volume.
ActiveServers	The physical IP address and port number of each active node on which the container resides.
InactiveServers	The physical IP address and port number of each inactive node on which the container resides.
UnusedServers	The physical IP address and port number of each on which the container does not reside.
OwnedSizeMB	The size on disk (in MB) dedicated to the container.
SharedSizeMB	The size on disk (in MB) shared by the container.
LogicalSizeMB	The logical size on disk (in MB) of the container.
Mtime	Indicates the time of the last modification to the container's contents.
NameContainer	Indicates if the container is the *name container* for the volume. If `true`, the container is the volume's first container and replication occurs simultaneously from the master to the intermediate and tail containers.

Example

CLI

```
maprcli dump
replicationmanagerinfo -cluster
docs41cluster -volumename
mapr.metrics  -json
```

dump replicationmanagerqueueinfo
Returns information that enables you to check the status of containers in various replication manager queues like under-replicated containers, and over-replicated containers, etc.

last-updated: Apr 11, 2022

Syntax

CLI

```
maprcli dump
replicationmanagerqueueinfo
    [-cluster <cluster name>]
    -queue <queue>
```

Note: For best results, use the `-json` option from the command line.

REST

N/A

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.
queue	The name of the queue. Valid values are *0*, *1*, *2*, or *5*. Queue 0 includes containers that have copies below the minimum replication factor for the volume. Queue 1 includes containers that have copies below the replication for the volume, but above the minimum replication factor. Queue 2 includes containers that are over-replicated. Queue 5 includes containers which are not rack aware.

Output

The `maprcli dump replicationmanagerqueueinfo` command returns information about one of these queues: 0, 1, 2, or 5. Depending on the queue value entered, the command displays information about containers that are under-replicated or over-replicated. You can use this information to decide if you need to change the replication factor for that volume.

```
# maprcli dump replicationmanagerqueueinfo -queue 0
Mtime                        LogicalSizeMB  UnusedServers  ActiveServers
TotalSizeMB  NameContainer  InactiveServers  ContainerId
Master                      Epoch  SharedSizeMB  OwnedSizeMB
Thu May 17 10:32:59 PDT 2012  0 MB                          ...
0 MB         false                          2065
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                          ...
0 MB         false                          2064
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
                              0 MB                          ...
0 MB         true                           1
10.250.1.103:5660--8-VALID  8      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                          ...
0 MB         false                          2066
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  1 MB                          ...
0 MB         false                          2069
10.250.1.103:5660--5-VALID  5      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  1 MB                          ...
0 MB         false                          2068
10.250.1.103:5660--5-VALID  5      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                          ...
```


last-updated: Apr 11, 2022

```
0 MB         false                                2071
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         false                                2070
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         false                                2073
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         false                                2072
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         false                                2075
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         false                                2074
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         false                                2077
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         false                                2076
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:36:30 PDT 2012  0 MB                              ...
0 MB         true                                 2049
10.250.1.103:5660--7-VALID  7      0 MB          0 MB
Thu May 17 10:36:36 PDT 2012  0 MB                              ...
0 MB         true                                 2050
10.250.1.103:5660--7-VALID  7      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         true                                 2051
10.250.1.103:5660--6-VALID  6      0 MB          0 MB
Thu May 17 10:37:06 PDT 2012  0 MB                              ...
0 MB         true                                 2053
10.250.1.103:5660--6-VALID  6      0 MB          0 MB
Fri May 18 14:33:44 PDT 2012  0 MB                              ...
0 MB         true                                 2054
10.250.1.103:5660--5-VALID  5      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         true                                 2055
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         true                                 2056
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         false                                2057
10.250.1.103:5660--5-VALID  5      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         false                                2058
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         false                                2059
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         false                                2060
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         false                                2061
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
0 MB         false                                2062
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
Thu May 17 10:32:59 PDT 2012  0 MB                              ...
```

```
0 MB         false                                 2063
10.250.1.103:5660--3-VALID  3      0 MB          0 MB
```

Output fields

Field	Description
ContainerID	The unique ID number of the container.
Epoch	A sequence number that indicates the most recent copy of the container. The CLDB uses the epoch to ensure that an out-of-date copy cannot become the master for the container.
Master	The physical IP address and port number of the *master copy*. The master copy is part of the original copy of the volume.
ActiveServers	The physical IP address and port number of each active node on which the container resides.
InactiveServers	The physical IP address and port number of each inactive node on which the container resides.
UnusedServers	The physical IP address and port number of servers from which no "heartbeat" has been received for quite some time.
OwnedSizeMB	The size on disk (in MB) dedicated to the container.
SharedSizeMB	The size on disk (in MB) shared by the container.
LogicalSizeMB	The logical size on disk (in MB) of the container.
TotalSizeMB	The total size on disk (in MB) allocated to the container. Combines the Owned Size and Shared Size.
Mtime	The time of the last modification to the contents of the container.
NameContainer	Indicates if the container is the *name container* for the volume. If `true`, the container holds the volume's namespace information and file chunk locations.

Example

CLI

```
maprcli dump
replicationmanagerqueueinfo -queue
0 -json
```

dump rereplicationinfo
Returns information about the ongoing re-replication of replica containers.

This information includes the destination IP address and port number, the ID number of the destination file server, and the ID number of the destination storage pool.

Re-replication occurs whenever the number of available replica containers drops below the number prescribed by that volume's replication factor. Re-replication may occur for a variety of reasons including replica container corruption, node unavailability, hard disk failure, or an increase in replication factor.

last-updated: Apr 11, 2022

Syntax

CLI

```
maprcli dump rereplicationinfo
    [-cluster <cluster name>]
```

Note: For best results, use the `-json` option from the command line.

REST

N/A

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.

Output

The `maprcli dump rereplicationinfo` command returns information about the ongoing re-replication of replica containers including the destination IP address and port number, the ID number of the destination file server, and the ID number of the destination storage pool.

```
# maprcli dump rereplicationinfo -json
{
        "timestamp":1338222709331,
        "status":"OK",
        "total":7,
        "data":[
                {
                        "containerid":2158,
                        "replica":{
                                "sizeMB":15467,
                                "To fsid":9057314602141502940,
                                "To IP:Port":"192.0.2.28:5660-",
                                "To SP":"03b5970f41abbe48004f828abaabcdef"
                        }
                },
                {
                        "containerid":3367,
                        "replica":{
                                "sizeMB":658,
                                "To fsid":3684488804112157043,
                                "To IP:Port":"192.0.2.33:5660-",
                                "To SP":"3b86b4ce5bfd6bbf004f87e9b6ghijkl"
                        }
                },
                {
                        "containerid":3376,
                        "replica":{
                                "sizeMB":630,
                                "To fsid":3684488804112157043,
                                "To IP:Port":"192.0.2.33:5660-",
                                "To SP":"3b86b4ce5bfd6bbf004f87e9b6ghijkl"
                        }
                },
                {
                        "containerid":3437,
```


last-updated: Apr 11, 2022

```
                        "replica":{
                                "sizeMB":239,
                                "To fsid":6776586767180745590,
                                "To IP:Port":"192.0.2.32:5660-",
                                "To SP":"6cd440fad0426db7004f828b2amnopqr"
                        }
                },
                {
                        "containerid":8833,
                        "replica":{
                                "sizeMB":7327,
                                "To fsid":9057314602141502940,
                                "To IP:Port":"192.0.2.28:5660-",
                                "To SP":"33885e3c5be9a04d004f828abcstuvwx"
                        }
                }
        ]

}
```

Output fields

Field	Description
sizeMB	The amount of data (in MB) being moved.
To fsid	The ID number (FSID) of the destination file server.
To IP:Port	The IP address and port number of the destination node.
To SP	The ID number (SPID) of the destination storage pool.

Example

CLI

```
maprcli dump rereplicationinfo -json
```

dump rereplicationmetrics
Displays information about containers that were copied by the replication manager.

This command displays the following fields :

- `numContainersCopied` - The number of containers that were copied by the replication manager to maintain the volume's replication factor or topology since the current CLDB was the master.
- `numMBCopied` -The cumulative size of the containers that were copied by the replication manager to maintain the volume's replication factor or topology since the current CLDB was the master.

Syntax

CLI

```
maprcli dump rereplicationmetrics
  [-cluster <cluster name>]
```

REST

N/A

Parameters

Parameter	Description
`cluster`	Cluster name.

Example

CLI maprcli dump rereplicationmetrics

Example Output

```
# maprcli dump rereplicationmetrics
numContainersCopied  numMBCopied
0                    0
```

dump rolebalancerinfo
Returns information about active replication role switches.

Use the `dump rolebalancerinfo` command to see if the replication role balancer is currently switching the replication roles of any containers in a cluster. For example, if too many data containers with the master or intermediate roles exist within a storage pool, the replication role balancer switches the role of some of these containers to the tail role to evenly spread the load across nodes during the replication process. If the role balancer is not currently switching the roles of any containers, the command returns a message stating that there are no active role switches.

You can include some additional parameters with the `dump rolebalancerinfo` command, such as the `volumeinfo` parameter, which provides information about how the replication role balancer balanced container roles across each storage pool in a particular volume.

See Replication Role Balancer for more information about how the replication role balancer works.

For the best readability, use the `-json` option at the end of the command.

Syntax

CLI

```
maprcli dump rolebalancerinfo
        [ -cluster cluster_name ]
        [ -namecntrinfo Get
NameContainers Info Parameter takes
no value  ]
        [ -stats Gets RoleBalancer
AcitveSwitches Info Parameter takes
no value  ]
        [ -volumeinfo Gets Balancing
Info for Volumes(s) Parameter takes
no value  ]
        [ -volumename Specifies the
name of the volumes ]
```

REST N/A

Parameters

Parameter	Description
cluster	The cluster on which to run the command. When you omit this parameter, the command runs on the same cluster where it is issued. In a multi-cluster environment, use this parameter to specify a particular cluster.

stats	Provides a list of active switches for the role balancer. The command returns the same information with or without this parameter.
volumeinfo	Provides the volume balancing information and details how the container roles are balanced across each storage pool in a volume. Requires the `volumename` parameter.
volumename	The name of the volume. To obtain volume names, use the `maprcli volume list` command.
namecntrinfo	Provides information about how the name containers are distributed across the storage pools in the cluster, including how many name containers are master and tail containers. Useful when the replication role balancer is configured to balance container roles by count instead of size.

Output

The following example shows the output of the `dump rolebalancerinfo` command when the replication role balancer switches a container to the tail role:

```
# maprcli dump rolebalancerinfo -json
{
      "timestamp":1452150159265,
      "timeofday":"2016-01-07 07:02:39.265 GMT+0000",
      "status":"OK",
      "total":1,
      "data":[
             {
                   "containerid":57482,
                   "Tail IP:Port":"10.10.104.37:5660-10.10.105.37:5660-",
                   "Updates blocked Since":"Thu Jan 07 07:02:24 UTC 2016"
             }
        ]
   }
```

The following example shows the `dump rolebalancerinfo -volumeinfo -volumename` command:

```
# maprcli dump rolebalancerinfo -volumeinfo -volumename vol2 -json
{
    "timestamp":1452218225547,
    "timeofday":"2016-01-08 01:57:05.547 GMT+0000",
    "status":"OK",
    "total":1,
    "data":[
        {
            "VolumeBalancingInfo":{
                "Volume":"vol2",
                "Assign Cache Containers Count":60,
                "Assign Cache Containers Size":951171,
                "Zero Size Containers Count":5,
                "Storage Pools":[
                    {
                        "SpId":"e471499d52ce710e00566942c1075a69",
                        "HostAddress":"10.10.104.34(2)",
                        "NumContainers":17,
                        "NumMasters":7,
                        "NumTails":4,
                        "SizeOfContainers":213690,
                        "SizeOfMasters":93769,
                        "DesiredSizeOfMasters":71230,
                        "SizeOfTails":76001,
                        "DesiredSizeOfTails":71230,
                        "Assign Cache Containers Count":6,
```

```
                        "Assign Cache Containers Size":71783
                    },
                    {
                        "SpId":"6a7222578e9cb90300566942e00bfb3e",
                        "HostAddress":"10.10.104.35(2)",
                        "NumContainers":21,
                        "NumMasters":6,
                        "NumTails":9,
                        "SizeOfContainers":373222,
                        "SizeOfMasters":141546,
                        "DesiredSizeOfMasters":124407,
                        "SizeOfTails":115855,
                        "DesiredSizeOfTails":124407,
                        "Assign Cache Containers Count":3,
                        "Assign Cache Containers Size":62491
                    },
                    {
                        "SpId":"ca379b6cfc888270005669428c02c6e3",
                        "HostAddress":"10.10.104.32(2)",
                        "NumContainers":5,
                        "NumMasters":0,
                        "NumTails":2,
                        "SizeOfContainers":107549,
                        "SizeOfMasters":0,
                        "DesiredSizeOfMasters":35849,
                        "SizeOfTails":48267,
                        "DesiredSizeOfTails":35849,
                        "Assign Cache Containers Count":0,
                        "Assign Cache Containers Size":0
                    }

                ]
            }
        }
    ]
}
```


Output

Field	Description
containerID	The unique ID number of the container.
Tail IP:Port	The IP address and port number of the tail container node.
Updates blocked Since	During a replication role switch, updates to that container are blocked during the date and time indicated.
Volume	Name of the volume.
Assign Cache Containers Count	The total number of containers in the assign cache. The assign cache is a list of containers that the CLDB has reserved on a particular file server node. The CLDB allocates containers from the cache when it receives a request from the file server for more containers.
Assign Cache Containers Size	The total size of the containers in the assigned cache. The assign cache is a list of containers that the CLDB has reserved on a particular file server node. The CLDB allocates containers from the cache when it receives a request from the file server for more containers.
Zero Size Containers Count	The number of empty cache containers.
Storage Pools	The list of storage pools in the specified volume.

SpId	The ID of the storage pool located within the specified volume.
HostAddress	The server on which the storage pool resides.
NumContainers	The total number of containers that reside on the storage pool in the specified volume.
NumMasters	The total number of master containers that reside on the storage pool in the specified volume.
NumTails	The total number of tail containers that reside on the storage pool in the specified volume.
SizeOfContainers	The total size of the containers that reside on the storage pool in the specified volume.
SizeOfMasters	The total size of the master containers that reside on the storage pool in the specified volume.
DesiredSizeOfMasters	The cumulative size of master replicas on a specific storage pool within a volume. Typically, this is 1/ReplicationFactor of all containers on a storage pool for a particular volume. For example, if the replication factor is set to 3, then ⅓ of all containers on a storage pool should have the master container role.
SizeOfTails	The total size of the tail containers that reside on the storage pool in the specified volume.
DesiredSizeOfTails	The cumulative size of tail replicas on a specific storage pool within a volume. Typically, this is 1/ReplicationFactor of all containers on a storage pool for a particular volume. For example, if the replication factor is set to 3, then ⅓ of all containers on a storage pool should have the tail container role.

Example

CLI

```
maprcli dump rolebalancerinfo -json
```

dump rolebalancermetrics

Returns the cumulative number of times that the replication role balancer has switched the replication role of name containers and data containers on the cluster.

The `maprcli dump rolebalancermetrics` command enables you to view the number of times that the replication role balancer has switched the replication role of the name containers and data containers to ensure that containers are balanced across the nodes in the cluster. For best results, use the `-json` option when running `dump rolebalancermetrics` from the command line.

The *replication role balancer* is a tool that switches the replication roles of containers to ensure that every node has an equal share of master and replica containers (for name containers) and an equal share of master, intermediate, and tail containers (for data containers).

The replication role balancer changes the replication role of the containers in a cluster so that network bandwidth is spread evenly across all nodes during the replication process. A container's replication role determines how it is replicated to the other nodes in the cluster. For *name containers* (the volume's first container), replication occurs simultaneously from the master to all replica containers. For *data containers*, replication proceeds from the master to the intermediate container(s) until it reaches the tail containers. For more information, see Replication Role Balancer.

last-updated: Apr 11, 2022

Syntax

CLI

```
maprcli dump rolebalancermetrics
    [-cluster <cluster name>]
```

REST

N/A

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.

Output

The `maprcli dump rolebalancerinfo` command returns the cumulative number of times that the replication role balancer has switched the replication role of name containers and data containers on the cluster.

```
# maprcli dump rolebalancermetrics -json
{
        "timestamp":1433372048169,
    "timeofday":"2015-06-03 10:54:08.169 GMT+0000",
        "status":"OK",
        "total":1,
        "data":[
                {
                        "numNameContainerSwitches":60,
                        "numDataContainerSwitches":28,
                        "timeOfLastMove":"Wed May 23 05:48:00 PDT 2015"
                }
        ]
}
```

Output fields

Field	Description
numNameContainerSwitches	The number of times that the replication role balancer has switched the replication role of name containers.
numDataContainerSwitches	The number of times that the replication role balancer has switched the replication role of data containers.
timeOfLastMove	The date and time of the last replication role change.

Example

CLI

```
maprcli dump rolebalancermetrics -json
```

dump supportdump
Collects logs and other information about the node to help troubleshoot issues.

Syntax

CLI

```
maprcli dump supportdump [ -cluster
<cluster name> ] [ -nodes <node
names> ] [ -params <parameter
string> ] [ -zkconnect <ZK connect
string> ]
```

REST

N/A

Parameters

Parameter	Description
cluster	Cluster name.
nodes	Node names for which support dump is needed. Space separated. Default: all
params	Parameter string to create a dump.
zkconnect	ZK connection string.

Output

```
# maprcli dump supportdump
node
centos29.lab
centos23.lab
centos28.lab
centos22.lab
```

Example

CLI

maprcli dump supportdump

dump volumeinfo

Returns information about volumes and the associated containers. For JSON formatted output, use the -json option from the command line.

A *volume* is a logical unit that allows you to apply policies to a set of files, directories, and sub-volumes. Using volumes, you can enforce disk usage limits, set replication levels, establish ownership and accountability, and measure the cost generated by different projects or departments. For more information, see Administering Volumes on page 856.

Syntax

CLI

```
maprcli dump volumeinfo
    [-cluster <cluster name>]
    -volumename <volume name>
```

REST

N/A

Parameters

cluster

The cluster on which to run the command. If this parameter is omitted, the command is run on the

last-updated: Apr 11, 2022

	cluster on which it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.
volumename	The name of the volume. To obtain volume names, use the volume list on page 1979 `maprcli volume list` command. This parameter is mandatory.

Output

The `maprcli volume info` returns information about the volume and the containers associated with that volume. Volume information includes the ID, volume name, and replication factor. For each container on the specified volume, the command returns information about nodes and storage. See the following Example on page 1640 for sample output.

AccessTime	Indicates the volumes that are accessed regularly. This value is updated every 6 hours with the last time that an operation occurred on the volume. The access time is not updated for changes to volume properties, creation of a snapshot, or synchronization between a volume and a mirror. However, the volume access time is updated the first time you upgrade to a MapR version that includes this property. This property was introduced in MapR version 4.0.2.
ActiveServers	The IP address and port number of each active node on which the container resides.
allowGrant	Indicates whether (`true`) or not (`false`) a parent volume grants permission for a child volume to inherit its properties.
Audited	Indicates whether (`1`) or not (`0`) auditing is enabled for the volume.
AuditVolume	Indicates whether (`1`) or not (`0`) the volume accommodates audit logs.
CoalesceInterval	The interval of time to elapse after the first instance of an operation on a node is recorded in audit logs, if auditing is enabled. Subsequent identical operations performed on the same node from the same client are ignored during the interval.
ContainerId	The unique ID number of the container.
CreatorContainerId	The container ID of the read-write container. The container ID is retained in all mirrors of those containers (in all mirrors of the volume). The container ID enables the identification of the correct containers to source from, when mirror sources are changed in a mirror chain.
CreatorVolumeUuid	A randomly generated unique ID that is shared by all mirrors of a volume, and all containers of them. You can use this ID to avoid undesirable chaining of containers when mirror sources are changed in a mirror chain.
dareEnabled	Indicates whether (`1`) or not (`0`) data-at-rest encryption (DARE) is enabled for the volume.
DisabledDataAuditOperations	The list of operations excluded from auditing.
EnabledDataAuditOperations	The list of operations selected for auditing.
enforcementMode	The data access enforcement mode.

Epoch	A sequence number that indicates the most recent copy of the container. The CLDB uses the epoch to ensure that an out-of-date copy cannot become the master for the container.
fixCreatorId	An internal flag for MapR volumes to fix the creator container ID.
ForceAudit	Indicates whether (`1`) or not (`0`) to force audit of operations on all files, tables, and streams in the volume.
InactiveServers	The IP address and port number of each inactive node on which the container resides.
limitSpread	An internal flag for MapR volumes to control the growth of a volume in terms of number of containers. When this flag is set, CLDB tries to limit the number of new containers created, depending on the present size of a volume. If a volume size (data in volume) is small, CLDB tries to reuse space in existing containers to avoid the creation of new containers. This reuse helps reduce wastage of containers IDs in an environment that has small volumes.
LogicalSizeMB	The logical size on disk (in MB) of the container.
Master	The IP address and port number of the *master copy*. The master copy is part of the original copy of the volume.
MetricsEnabled	Indicates whether (`1`) or not (`0`) metrics collection is enabled for the volume.
MirrorThrottle	Specifies whether mirror throttling is enabled (true) or disabled (false). Throttling is set on the source volume and applies to all its mirrors. This property was introduced in MapR version 4.0.2.
Mtime	Indicates the time when the last modification was made to the contents of the container.
NameContainer	Indicates if the container is the *name container* for the volume. If `true`, the container is the first container of the volume. Replication then occurs simultaneously from the master to the intermediate and tail containers.
NameSpaceMinReplication	The minimum replication factor or the number of copies of the name container associated with the volume that should be maintained by the MapR cluster for normal operation. When the replication factor falls below this minimum value, writes to the volume are disabled. The default value is *2*.
NameSpaceReplication	The desired replication factor or the number of copies of the name container associated with the volume. The default value is *3*. The maximum value is *6*.
NumInodesInUse	Indicates the uumber of inodes used by the container.
OwnedSizeMB	The size on disk (in MB) dedicated to the container.
ReReplicationTimeOutSec	The timeout (in seconds) period until CLDB starts re-replicating the containers on the node of the volume, when CLDB stops receiving a heartbeat from the node.
securityPolicyTags	The list of security policy tags to be associated with this volume.
SharedSizeMB	The size on disk (in MB) shared by the container.

last-updated: Apr 11, 2022

TenantUser	Displays the name of the tenant user, if any.
TotalSizeMB	The total size on disk (in MB) allocated to the container. Combines the Owned Size and Shared Size.
UnusedServers	The IP address and port number of servers from which no "heartbeat" has been received for quite some time.
VolumeId	The unique ID number of the volume.
VolumeMinReplication	The minimum replication factor. Indicates the number of copies of a volume (including the original) that should be maintained by the MapR cluster for normal operation. When the replication factor falls below this minimum value, writes to the volume are disabled. The default value is *2*. A replication factor of 2 indicates that the number of copies of a volume is 2 (original +1 copy). A replication factor of 3 indicates that the number of copies of a volume is 3 (original + 2 copies).
VolumeName	The name of the volume.
VolumeReplication	The desired replication factor. Indicates the number of copies of a volume. The default value is *3*. The maximum value is *6*.
VolumeTopology	The volume topology corresponds to the node topology of the rack or nodes where the volume resides. By default, new volumes are created with a topology of `/` (root directory). For more information, see Setting Up Node Topology on page 805.
VolumeUsedSizeMB	The size on disk (in MB) of the volume.
WireSecurityEnabled	Indicates whether (`1`) or not (`0`) wire-level security is enabled.

Example
Dump volume information

CLI

```
# /opt/mapr/bin/maprcli dump
volumeinfo -cluster
docs41cluster -volumename
sampleVol  -json
{
    "timestamp":1435363982346,
    "timeofday":"2015-06-26
05:13:02.346 GMT-0700",
    "status":"OK",
    "total":2,
    "data":[
        {
            "VolumeName":"sampleVol",
            "VolumeId":47274128,
            "VolumeTopology":"/data",
            "VolumeUsedSizeMB":0,
            "VolumeReplication":3,
            "VolumeMinReplication":2,
            "NameSpaceReplication":3,

"NameSpaceMinReplication":2,

"ReReplicationTimeOutSec":0,
            "MirrorThrottle":true,
```


last-updated: Apr 11, 2022

```
            "AccessTime":"Fri Jun 26
09:38:30 PDT 2015",
            "AuditVolume":"0",
            "Audited":"0",
            "ForceAudit":"0",
            "CoalesceInterval":60,

"EnabledDataAuditOperations":"setattr,
chown,chperm,chgrp,getxattr,listxattr,
setxattr,removexattr,read,write,create
,delete,mkdir,readdir,rmdir,createsym,
lookup,rename,createdev,truncate,table
cfcreate,tablecfdelete,tablecfmodify,t
ablecfScan,tableget,tableput,tablescan
,tablecreate,tableinfo,tablemodify,get
perm,getpathforfid,hardlink,filescan,f
ileoffload,filerecall,filetierjobstatu
s,filetierjobabort",

"DisabledDataAuditOperations":"getattr
,filetieroffloadevent,filetierrecallev
ent",
            "WireSecurityEnabled":"1",


            "limitSpread":true,
            "allowGrant":false,
            "fixCreatorId":false,
            "MetricsEnabled":"0",
            "dareEnabled":1

        },
        {
            "ContainerId":2049,
            "Epoch":3,

"Master":"10.10.100.126:5660-10.10.101
.126:5660-172.17.42.1:5660--3-VALID",
            "ActiveServers":{


"IP:Port":"10.10.100.126:5660-10.10.10
1.126:5660-172.17.42.1:5660--3-VALID"
            },
            "InactiveServers":{

            },
            "UnusedServers":{

            },
            "OwnedSizeMB":"0 MB",
            "SharedSizeMB":"0 MB",
            "LogicalSizeMB":"0 MB",
            "TotalSizeMB":"0 MB",
            "NumInodesInUse":41,
            "Mtime":"Fri Jun 26
13:27:35 PDT 2015",
            "NameContainer":"true",
            "CreatorContainerId":0,

"CreatorVolumeUuid":"-8225749748229459
176:-4287758954200211096",
```

```
         "UseActualCreatorId":true
        }
    ]
}
```

REST

```
# curl -k -X GET 'https://
abc.sj.us:8443/rest/dump/volumeinfo?
volumename=sampleVol' --user mapr:mapr
{"timestamp":1531074195026,"timeofday"
:"2018-07-08 11:23:15.026 GMT-0700
AM","status":"OK","total":2,"data":
[{"VolumeName":"sampleVol","VolumeId":
245584625,"VolumeTopology":"/
data","VolumeUsedSizeMB":0,"VolumeRepl
ication":3,"VolumeMinReplication":2,"N
ameSpaceReplication":3,"NameSpaceMinRe
plication":2,"ReReplicationTimeOutSec"
:0,"MirrorThrottle":true,"AccessTime":
"July 7,
2018","AuditVolume":"0","Audited":"0",
"ForceAudit":"0","CoalesceInterval":60
,"EnabledDataAuditOperations":"setattr
,chown,chperm,chgrp,getxattr,listxattr
,setxattr,removexattr,read,write,creat
e,delete,mkdir,readdir,rmdir,createsym
,lookup,rename,
createdev,truncate,tablecfcreate,table
cfdelete,tablecfmodify,tablecfScan,tab
leget,tableput,tablescan,tablecreate,t
ableinfo,tablemodify,getperm,
getpathforfid,hardlink,filescan,fileof
fload,filerecall,filetierjobstatus,fil
etierjobabort","DisabledDataAuditOpera
tions":"getattr,filetieroffloadevent,f
iletierrecallevent","WireSecurityEnabl
ed":"1",

"limitSpread":true,
"allowGrant":false,"fixCreatorId":fals
e,"MetricsEnabled":"0",dareEnabled":0}
,{"ContainerId":2068,"Epoch":3,
"Master":"10.10.82.24:5660--3-VALID","
ActiveServers":
{"IP:Port":"10.10.82.24:5660--3-VALID"
},"InactiveServers":
{},"UnusedServers":
{},"OwnedSizeMB":"0
MB","SharedSizeMB":"0
MB","LogicalSizeMB":"0
MB","TotalSizeMB":"0
MB","NumInodesInUse":256,"Mtime":"July
 7,
2018","NameContainer":"true","CreatorC
ontainerId":2068,"CreatorVolumeUuid":
"-8225749748229459176:-428775895420021
1096","UseActualCreatorId":true}]}
```

Related concepts

node on page 1694
Manages nodes in the cluster

Related reference

disk add on page 1602
Adds one or more disks to the specified node. Permissions required: `fc` or `a`.

volume create on page 1931
Creates a volume.

node list on page 1705
Lists nodes in the cluster.

configure.sh on page 2053
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

dump volumenodes
Returns information about the nodes on a volume.

Syntax

CLI

```
maprcli dump volumenodes
    [-cluster <cluster name>]
    -volumename <volume name>
```

Note: For best results, use the `-json` option from the command line.

REST

N/A

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.
`volumename`	The name of the volume. To obtain volume names, use the `maprcli volume list` command.

Output

The `maprcli dump volumenodes` command returns the IP address and port number of volume nodes.

```
# maprcli dump volumenodes -volumename mapr.hbase -json
{
    "timestamp":1433372931725,
    "timeofday":"2015-06-03 11:08:51.725 GMT+0000",
    "status":"OK",
    "total":1,
    "data":[
        {
            "Servers":{
                "IP:Port":[
                    "10.10.82.23:5660--3-VALID",
                    "10.10.82.28:5660--3-VALID",
                    "10.10.82.29:5660--3-VALID"
                ]
            }
```


last-updated: Apr 11, 2022

```
        }
    ]
}
```

Output fields

Field	Description
IP:Port	The IP address and MapR File System port.

Example

CLI

```
maprcli dump volumenodes -volumename
mapr.hbase -json
```

dump zkinfo
Returns the ZooKeeper znodes.

Note: This command is used by the `mapr-support-collect.sh` script to gather cluster diagnostics for troubleshooting.

This command enables you to view a snapshot of the data stored in Zookeeper as a result of cluster operations

ZooKeeper prevents service coordination conflicts by enforcing a rigid set of rules and conditions, provides cluster-wide information about running services and their configuration, and provides a mechanism for almost instantaneous service failover. Warden will not start any services unless ZooKeeper is reachable and more than half of the configured ZooKeeper nodes are live.

The `mapr-support-collect.sh` script calls the `maprcli dump supportdump` command to gather cluster diagnostics for troubleshooting. For more information, see mapr-support-collect.sh.

Syntax

CLI

```
maprcli dump zkinfo
    [-cluster <cluster name>]
    [-zkconnect <connect string>]
```

Note: For best results, use the `-json` option from the command line.

REST

N/A

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.
`zkconnect`	A ZooKeeper connect string, which specifies a list of the hosts running ZooKeeper, and the port to use on each, in the format: '<host>[:<port>][,<host>[:<port>]...]'. To obtain zookeeper connection strings, use the `maprcli node listzookeepers` command.

Output

The `maprcli dump zkinfo` command is run as part of support dump tools to view the current state of the Zookeeper service. The command should always be run using the `-json` option, since output in the default tabular format is not useful. Command output displays the data stored in the ZooKeeper hierarchical tree of znodes.

```
# maprcli dump zkinfo -json
{
        "timestamp":1335825202157,
        "status":"OK",
        "total":1,
        "data":[
                {
                "/_Stats":"\ncZxid = 0,ctime = Wed Dec 31 16:00:00 PST
1969,mZxid = 0,mtime = Wed Dec 31 16:00:00 PST 1969,pZxid = 516,cversion
= 12,dataVersion = 0,aclVersion = 0,ephemeralOwner = 0,dataLength =
0,numChildren = 13",
                        "/":[
                                {
                                    ....
                                }
        ]
}
```

Output fields

You can use the `maprcli dump zkinfo` command as you would use a database snapshot. The `/services`, `/services_config`, `/servers`, and `/*_locks` znodes are used by Warden to store and exchange information.

Field	Description
services	The `/services` directory is used by Warden to store and exchange information about services.
datacenter	The `/datacenter` directory contains CLDB "vital signs" that you can use to identify the CLDB master, the most recent epoch, and other key data. For more information, see Moving CLDB Data.
services_config	The `/services_config` directory is used by Warden to store and exchange information.
zookeeper	The `/zookeeper` directory stores information about the ZooKeeper service.
servers	The `/servers` directory is used by Warden to store and exchange information.
nodes	The `/nodes` directory (znode) stores key information about the nodes.

Moving CLDB Data
Describes how to move CLDB data to another node.

In a Community Edition-licensed cluster, CLDB data must be recovered from a failed CLDB node and installed on another node. The cluster can continue normally as soon as the CLDB is started on another node.

For more information, see CLDB Failover on page 1500.

Use the `maprcli dump zkinfo` command to identify the latest epoch of the CLDB, identify the nodes where replicates of the CLDB are stored, and select one of those nodes to serve the new CLDB node. Perform the following steps on any cluster node:

1. Log in as `root` or use `sudo` for the following commands.
2. Issue the `maprcli dump zkinfo` command using the `-json` flag.
   `# maprcli dump zkinfo -json`
   The output displays the ZooKeeper znodes.
3. In the `/datacenter/controlnodes/cldb/epoch/1` directory, locate the CLDB with the latest epoch.

   ```
   { "/datacenter/controlnodes/cldb/epoch/1/KvStoreContainerInfo":"
   Container ID:1 VolumeId:1
   Master:10.250.1.15:5660-172.16.122.1:5660-192.168.115.1:5660--13-VALID
   Servers: 10.250.1.15:5660-172.16.122.1:5660-192.168.115.1:5660--13-VALID
   Inactive Servers: Unused Servers: Latest epoch:13" }
   ```

   The Latest Epoch field identifies the current epoch of the CLDB data. In this example, the latest epoch is *13*.
4. Select a CLDB from among the copies at the latest epoch. For example, `10.250.2.41:5660--13-VALID` indicates that the node has a copy at epoch 13 (the latest epoch).

entity
Manages *entities* (users and groups).
entity info
Displays information about an entity.

Syntax

CLI

```
maprcli entity info
    [ -cluster <cluster> ]
    -name <entity name>
    [ -output terse|verbose ]
    -type <type>
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/entity/info?<parameters>`

Parameters

Parameter	Description
cluster	The cluster on which to run the command.
name	The *entity* name. Obtain the entity name by running the `maprcli entity list` command.
output	Whether to display terse or verbose output.

Parameter	Description
type	The entity type. Obtain the entity type by running the `maprcli entity list` command.

Output

Sample Output

```
DiskUsage  EntityQuota  EntityType  EntityName  VolumeCount
EntityAdvisoryquota  EntityId
864415     0            0           root        208
0                   0
```

Output Fields

Field	Short Name	Description
DiskUsage	dsu	Disk space used by the user or group
EntityQuota	qta	The user or group quota
EntityType	t	The entity type
EntityName	n	The entity name
VolumeCount	vct	The number of volumes associated with the user or group
EntityAdvisoryquota	aqt	The user or group advisory quota
EntityId	id	The ID of the user or group

Examples

Display information for the user 'root':

CLI

```
maprcli entity info -type 0 -name root
```

REST

```
https://abc.sj.us:8443/rest/entity/
info?type=0&name=root
```

entity list
Lists and displays information about entities.

Syntax

CLI

```
maprcli entity list
    [ -alarmedentities true|false ]
    [ -cluster <cluster> ]
    [ -columns <columns> ]
    [ -filter <filter> ]
    [ -limit <rows> ]
    [ -output terse|verbose ]
    [ -sortby ]
    [ -start <start> ]
```

REST

Request Type	GET

Request URL	`http[s]://<host>:<port>/rest/entity/list[?<parameters>]`

Parameters

Parameter	Description
`alarmedentities`	Specifies whether to list only entities that have exceeded a quota or advisory quota.
`cluster`	The cluster on which to run the command.
`columns`	A comma-separated list of fields to return in the query. See the Fields table below.
`filter`	A filter specifying entities to display. See Filters for more information.
`limit`	The number of rows to return, beginning at start. Default: 0
`output`	Specifies whether output should be `terse` or `verbose`.
`sortby`	Specifies one of the following attributes to sort the list of entities by: `entityname`, `entitytype`, `entityid`, `entityemail`, `entityquota`, `entityadvisoryquota`, `entitydiskusage`, `entityvolumecount`. By default, the list of entities sorted by `entityname`.
`start`	The offset from the starting row according to sort. Default: 0

Output

Information about the users and groups. Only users and groups with associated volumes are returned in the output.

Table

Field	Short Name	Description
EntityType	t	Entity type • `0` = User • `1` = Group
EntityName	n	User or Group name
EntityId	id	User or Group id
EntityQuota	qta	Quota, in MB. `0` = no quota.
EntityAdvisoryquota	aqt	Advisory quota, in MB. `0` = no advisory quota.
VolumeCount	vct	The number of volumes this entity owns.
DiskUsage	dsu	Disk space used for all entity's volumes, in MB.

last-updated: Apr 11, 2022

Sample Output

```
DiskUsage  EntityQuota  EntityType  EntityName  VolumeCount
EntityAdvisoryquota  EntityId
5859220    0            0           root        209
0                   0
```

Examples

List all entities:

CLI

```
maprcli entity list
```

REST

```
https://abc.sj.us:8443/rest/entity/
list
```

Filter entities by entity name:

CLI

```
maprcli entity list -filter
"[EntityName==mapr]"
```

REST

```
https://abc.sj.us:8443/rest/entity/
list?filter=[EntityName%3D%3Dmapr]
```

entity modify

Modifies a user or group quota or email address. Permissions required: `fc` or `a`.

Syntax

CLI

```
maprcli entity modify
    [ -advisoryquota <advisory quota>
    [ -cluster <cluster> ]
    [ -email <email>]
    [ -entities <entities> ]
    -name <entityname>
    [ -quota <quota> ]
    -type <type>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/entity/modify?<parameters>`

Parameters

Parameter	Description
`advisoryquota`	The advisory quota.
`cluster`	The cluster on which to run the command.

Parameter	Description
`email`	Email address.
`entities`	A comma-separated list of entities, in the format `<type>:<name>`. Example: `0:<user1>,0:<user2>,1:<group1>,1:<group2>...`
`name`	The *entity* name.
`quota`	The quota for the entity.
`type`	The entity type: • 0=user • 1-group

Examples

Modify the email address for the user 'root':

CLI

```
maprcli entity modify -name
root -type 0 -email test@example.com
```

REST

```
https://abc.sj.us:8443/rest/entity/
modify?
name=root&type=0&email=test@example.co
m
```

Related tasks

Setting Quota Defaults for Users and Groups on page 781
Explains how to set disk space quotas for users and groups.

Related reference

rlimit set on page 1736
Sets the resource usage limit for the cluster's disk resource.

entity remove

Removes an entity (specified by name and type).

Note: Entity can be removed only when there are no resources associated with the entity.

Syntax

CLI

```
maprcli entity remove
    -name <entity name>
    -type <type>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/entity/remove?<parameters>`

Parameters

Parameter	Description
`name`	The *entity* name. Obtain the entity name by running the `maprcli entity list` command.
`type`	The entity type. Value can be: • `0` - for user • `1` - for group If necessary, obtain the entity type by running the `maprcli entity list` command.

Example

Remove an entity by name and type:

```
maprcli entity remove -name mapruser1 -type 1
```

fid
Displays information about MapR Database or file-system components that are identified by a FID.
fid dump
Displays detailed information for MapR Database or file-system components that are identified by an FID.

Note: Only the root user and the MAPR_USER user (user under which MapRMapR Data Platform services runs) have permissions to run this command.

Syntax

CLI

```
maprcli fid dump
   [ -cluster <cluster_name> ]
   -fid <file identified for the
element>
```

REST

N/A

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.
`fid`	The file identifier for the element (region, kvstore, etc.) for which you want detailed information. The output of `maprcli table region list` lists the FIDs for the table's regions.

Note: You can run this command on any FID available on the MapRMapR Data Platform filesystem.

Tablet Map
Displays output for a tablet map includes the key for each tablet and its corresponding FID.

last-updated: Apr 11, 2022

Each tablet contains a range of data starting with the key associated with the tablet and ending before the key associated with the next tablet

FID for a Tablet Map

Describes how to determine the FID for a tablet map.

To determine the FID for a tablet map:

1. Run `hadoop mfs -ls <table path>` to determine the table FID. The table FID is the FID that displays after the "p."
   Example:

```
[mapr@hostname ~]$ hadoop mfs -ls /testdst
                        Found 1 items
                        tr-------- Z U 3 mapr mapr 2 2015-02-18 15:24 0 /
testdst
                        p 2049.49.131220 hostname:5660
                        r 2061.32.131258 hostname:5660
```

2. Run maprcli fid dump on the table FID to determine the tablet map FID.
   Example:

```
                        [mapr@hostname ~]$ maprcli fid dump -fid
2049.49.131220
                        value key
                        {"value":{"fid":"<parentCID>.51.131224"}} schema
                        {"value":{"fid":"<parentCID>.50.131222"}}
tabletmap
```

3. Construct the tablet map FID for the maprcli fid dump command by replacing <parentCID> with the set of numbers before the first period four numbers in the table FID.
   Example: `2049.50.131222`

Output Example for a Tablet Map

Example command and output.

```
maprcli fid dump -fid 2049.50.131222 -json
{
 "timestamp":1425579595296,
 "timeofday":"2015-03-05 06:19:55.296 GMT+0000",
 "status":"OK",
 "total":4,
 "data":[
  {
   "key":"",
   "value":{
    "fid":"2116.59.131462"
   }
  },
  {
   "key":"user3155781742051747178",
   "value":{
    "fid":"2114.49.131348"
   }
  },
  {
   "key":"user5238840414188136300",
   "value":{
    "fid":"2118.49.131394"
```

```
    }
   },
   {
    "key":"user7257930685533675764",
    "value":{
     "fid":"2115.59.131316"
    }
   }
  ]
}
```

Tablet
Describes output for a tablet.

The `maprcli fid dump` output for a tablet includes key and value pairs for the following:

- **startkey.** The first key value in the tablet.
- **pmap.** Each partition.
- **endkey.** The last key value in the tablet.

Output Fields for a Tablet

This table describes a majority of the output values for each partition (pmap) in the tablet.

Field	Description
`key`	The partition key.
`segfid`	The FID of the segment map associated with this partition.
`isFrozen`	A Boolean value that indicates if a partition is in a frozen state or not. Internally, a partition is sometimes temporarily marked as frozen in order for certain operations to complete.
`inSplit`	A Boolean value that indicates if a partition split is in progress for this partition.
`useBucketDesc`	This property is for internal use only.
`lastFlushedBucketFid`	The FID of the bucket file (WAL) which was last flushed for this partition.
`numLogicalBlocks`	The number of logical blocks (8K) for this partition.
`numPhysicalBLocks`	The number of physical blocks (8K) for this partition.
`numRows`	The number of rows stored in this partition.
`numRowsWithDelete`	The number of rows which are marked for delete in this partition.
`numRemoteBlocks`	The number of disk blocks which are not local to this partition. When a region splits, a partition moves from one node to another and it is possible to temporarily have some remote blocks.
`numSpills`	The number of spills.
`numSegments`	The number of segments.

Output Example for a Tablet


```
maprcli fid dump -fid 2116.59.131462 -json
{
    "timestamp":1425579636931,
    "timeofday":"2015-03-05 06:20:36.931 GMT+0000",
    "status":"OK",
    "total":6,
    "data":[
        {
            "key":"endkey.user3155781742051747178",
            "value":{

            }
        },
        {
            "key":"pmap.",
            "value":{
                "segfid":"<parentCID>.1065.133486",
                "isFrozen":false,
                "inSplit":false,
                "useBucketDesc":true,
                "lastFlushedBucketFid":"2116.901.133158",
                "numLogicalBlocks":34921,
                "numPhysicalBlocks":21976,
                "numRows":9332,
                "numRowsWithDelete":0,
                "numRemoteBlocks":0,
                "numSpills":137,
                "numSegments":74
            }
        },
        {
            "key":"pmap.user1523186274532578170",
            "value":{
                "segfid":"<parentCID>.1066.133488",
                "isFrozen":false,
                "inSplit":false,
                "useBucketDesc":true,
                "lastFlushedBucketFid":"2116.902.133160",
                "numLogicalBlocks":37011,
                "numPhysicalBlocks":23260,
                "numRows":9868,
                "numRowsWithDelete":0,
                "numRemoteBlocks":168,
                "numSpills":147,
                "numSegments":78
            }
        },
        {
            "key":"pmap.user2078250355776544396",
            "value":{
                "segfid":"<parentCID>.445.132238",
                "isFrozen":false,
                "inSplit":false,
                "useBucketDesc":true,
                "lastFlushedBucketFid":"2116.447.132242",
                "numLogicalBlocks":71124,
                "numPhysicalBlocks":44797,
                "numRows":18991,
                "numRowsWithDelete":0,
                "numRemoteBlocks":172,
                "numSpills":300,
```

```
                "numSegments":152
            }
        },
        {
            "key":"postSplitCopy",
            "value":{
                "raw":"dummy"
            }
        },
        {
            "key":"startkey.",
            "value":{
            }
        }
    ]
}
```

Note: The postSpitCopy key and value are for internal use only.

Segment Map
Describes output of a segment map.

The `maprcli fid dump` output of a segment map includes a map of row keys and the corresponding segment FID.

Output Fields for a Segment Map

Field	Description
key	The row key
value	The FID corresponding to the segment associated with this key.

Output Example for a Segment Map

```
maprcli fid dump -fid 2116.1065.133486 -json

{
    "timestamp":1425579702407,
    "timeofday":"2015-03-05 06:21:42.407 GMT+0000",
    "status":"OK",
    "total":74,
    "data":[
        {
            "key":"",
            "value":{
                "fid":"<parentCID>.943.133242"
            }
        },
        {
            "key":"user1006417450462802131",
            "value":{
                "fid":"<parentCID>.945.133246"
            }
        },
…
    ]
}
```

Segment
Describes output for a segment.

The output of `maprcli fid dump` for a segment includes details about each spill.

Output Fields for a Segment

Field	Description
`key`	The index of the spill.
`numRemoteBlocks`	The number of remote blocks.
`numspills`	The number of spills.
`value`	The property contains the following values: • fid: The FID of the spill containing row and value data. • smeSize: The spill map entry size. • keyIdxOffset: The offsets and length inside the spill for the index • keyIdxLength: The length inside the spill for the index • ldbIdxLength: The length of the index portion in the spill. • bloomBitsPerKey: The number of bits used in the bloom filter per key. • numLogicalBlocks. The number of logical blocks in the spill. • numPhysicalBlocks: The number of physical blocks in the spill. • numRows: The number of rows in the spill. • numRowsWithDelete: The number of rows which are marked for delete in the spill. • families:Information about the location of different column family data in the spill and the time range of that data.

Output Example for a Segment

```
maprcli fid dump -fid  2116.945.133246  -json
{
    "timestamp":1425579733821,
    "timeofday":"2015-03-05 06:22:13.821 GMT+0000",
    "status":"OK",
    "total":1,
    "data":[
        {
            "key":0,
            "numRemoteBlocks":0,
            "numSpills":0,
            "numSegments":0,
            "value":{
                "fid":"<parentCID>.946.133248",
                "smeSize":55,
                "keyIdxOffset":12,
                "keyIdxLength":3587,
                "ldbIdxLength":20,
```

```
                "bloomBitsPerKey":80,
                "numLogicalBlocks":369,
                "numPhysicalBlocks":232,
                "numRows":99,
                "numRowsWithDelete":0,
                "families":{
                    "id":1,
                    "offset":524288,
                    "length":2976835,
                    "minTimeStamp":1425578650850,
                    "maxTimeStamp":1425578856492
                }
            }
        }
    ]
}
```


fid stat

Displays statistics for MapR Database or filesystem components that are identified by a FID.

Only the root user and the MAPR_USER user (user name under which MapR services runs) have permissions to run this command.

Note: This command is similar to the UNIX `stat` command.

Syntax

CLI

```
/opt/mapr/bin/maprcli fid stat
   [ -cluster <cluster name> ]
   -fid <file identifier for the
element>
```

REST

N/A

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.
`fid`	The file identifier for the element (region, kvstore, etc.) for which you want detailed information. The output of the `maprcli table region list` command lists the FIDs for the regions of the table.

Output Fields

Columns	Description
compression	The compression setting, either `on` or `off`. If `on`, this parameter displays the compression type.
deleteFlags	An internal delete flag that is set on the FID for transactions involving multiple nodes. If deleted, this parameter denotes the deletion type, either `self` or `recursive delete`.

last-updated: Apr 11, 2022

Columns	Description
diskflush	Indicates whether persistent flush is enabled (`true`) or not (`false`) for this inode.
gid	The group ID
lblocks	Number of logical B-Tree blocks
mode	The UNIX style permission mode bits for the FID
mtime	Last modification time
nblocks	Total number of B-Tree blocks used
networkencryption	Indicates whether wire encryption is enabled or disabled
nlevels	Number of B-Tree levels
nlink	Number of links to this inode
parent	The parent FID
size	The size of the FID. Depending on the type of FID, it can be the actual size (in bytes), or the number of entries
subtype	The subtype of the FID
type	The type of the FID. For example, regular file, dir, filelet, kvstore, fidmap etc.
uid	The user ID of the owner
version	The current version of the FID.
xattrInum	The extended attribute of the FID

Example

Displays statistics for a specified FID:

```
[user@hostname ~]$ maprcli fid stat -fid 2062.32.131252  -json
{
        "timestamp":1586935733623,
        "timeofday":"2020-04-15 12:28:53.623 GMT-0700 AM",
        "status":"OK",
        "total":1,
        "data":[
                {
                        "type":"FTDirectory",
                        "subtype":"FSTInval",
                        "parent":"<parentCID>.35.131200",
                        "size":1,
                        "nblocks":1,
                        "lblocks":0,
                        "compression":"off",
                        "deleteFlags":"DeleteTypeNone",

                        "mtime":1583751630,
                        "mode":"755",
                        "uid":1000,
                        "gid":1000,
                        "nlink":3,
                        "xattrInum":0,
                        "version":1048589,
                        "networkencryption":true,
                        "diskflush":false,
```

```
                        "nlevels":1
                }
        ]
}
```

file
Lets you perform tiering operations at the file level.

file offload
Initiates offload of a file using a MAST Gateway.

Permissions Required

The user running the command must have (mode bit or *ACE*) permissions to write to the file.

Syntax

CLI

```
/opt/mapr/bin/maprcli file offload
       -name <file_name>
```

REST

Request Type	POST
Request URL	`http[s]://<host:port>/rest/file/offload?<parameters>`

Parameters

Parameter	Description
`name`	The name (including the path) of the file to offload.

Error Message

The OP_TIMEOUT message that indicates that the operation timed out, is returned if the connection to the gateway is lost.

Example

Offload file named `test1` in volume named `vol1`:

CLI

```
/opt/mapr/bin/maprcli file
offload -name /vol1/test1
 {
    "timestamp":1520277246831,
    "timeofday":"2018-03-05
07:14:06.831 GMT+0000",
    "status":"OK",
    "total":1,
    "data":[
        {
            "status":12,
            "message":"File transfer
request queued.",
```

```
"gateway":"10.10.88.200:8660",

"jobid":"0x37d7c7738cd0991f.0xe35d5f0e
5b24cda.0x4"
        }
    ]
 }
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/file/offload?
name=/vol1/test1' --user mapr:mapr

{"timestamp":1520277246831,"timeofday"
:"2018-03-05 07:14:06.831
GMT+0000","status":"OK","total":1,
   "data":
[{"status":12,"message":"File
transfer request
queued.","gateway":"10.10.88.200:8660"
,

"jobid":"0x37d7c7738cd0991f.0xe35d5f0e
5b24cda.0x4"}]}
```

Offload a file named `mfs` in volume named `vol1`:

CLI

```
/opt/mapr/bin/maprcli file
offload -name /vol1/mfs -json
  {
    "timestamp":1534141379576,
    "timeofday":"2018-08-12
11:22:59.576 GMT-0700 PM",
    "status":"ERROR",
    "errors":[
        {
            "id":6,
            "desc":"Lost connection
to gateway."
        }
    ]
  }
```

REST

```
curl -k -X GET 'https://
abc.sj.us:8443/rest/file/
tierjobstatus?name=/vol1/mfs' --user
mapr:mapr

{"timestamp":1534141379576,"timeofday"
:"2018-08-12 11:22:59.576 GMT-0700
PM","status":"ERROR",
   "errors":[{"id":6,"desc":"Lost
connection to gateway."}]}
```

file recall
Initiates recall of a file from a storage tier using a MAST Gateway.

Permissions Required

The user running the command must have (mode bit or *ACE*) permissions to write to the file.

Syntax

CLI

```
/opt/mapr/bin/maprcli file recall
     -name <file_name>
```

REST

Request Type	GET
Request URL	`http[s]://<host:port>/rest/file/recall?<parameters>`

Parameters

Parameter	Description
`name`	The name (including the path) of the file to recall.

Example

Recall file named file1 in volume named vol1:

CLI

```
/opt/mapr/bin/maprcli file
recall -name /vol1/test1 -json
{
  "timestamp":1516337242973,
  "timeofday":"2018-01-19
04:47:22.973 GMT+0000",
  "status":"OK",
  "total":1,
  "data":[{
       "status":12,
       "message":"File transfer
request queued.",
       "gateway":"10.10.88.198:8660",

"jobid":"0xb76f872c64fe4677.0x3673092f
759a500d.0x1"
  }]
}
```

REST

```
curl -k -X GET 'https://
abc.sj.us:8443/rest/file/recall?
<parameters>' --user mapr:mapr
{"timestamp":1516337242973,"timeofday"
:"2018-01-19 04:47:22.973
GMT+0000","status":"OK","total":1,"dat
a":[{"status":12,"message":"File
transfer request
queued.","gateway":"10.10.88.198:8660"
,"jobid":"0xb76f872c64fe4677.0x3673092
f759a500d.0x1"}]}
```


last-updated: Apr 11, 2022

file tierjobabort

Initiates abort of an ongoing offload or recall operation.

Permissions Required

The user running the command must have (mode bit or *ACE*) permissions to write to the file.

Syntax

CLI

```
/opt/mapr/bin/maprcli file
tierjobabort
    -name <file_name>
    [ -job <jobID> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host:port>/rest/file/tierjobabort?<parameters>`

Parameters

Parameter	Description
`name`	The name (including the path) of the file being offloaded/recalled.
`job`	The ID of the job, which was specified with the offload or recall command, to abort. This must be specified to ensure that the correct offload or recall task is aborted. If this is not specified, the command picks a job to abort in the following order: **1.** Job that is in “already aborting progress” status. **2.** Job that is in “running jobs with latest jobid” status. If there are no jobs in progress, the command returns “no active transfer in progress” error.

Examples

CLI

```
/opt/mapr/bin/maprcli file
tierjobabort -name /c1/file5G -json
{
  "timestamp":1557734728770,
  "timeofday":"2019-05-13
01:05:28.770 GMT-0700 AM",
   "status":"OK",
   "total":1,
    "data":[
              {
               "status":10,
               "message":"File
transfer being aborted.",

"gateway":"10.10.103.79:8660",
```


last-updated: Apr 11, 2022

```
"jobid":"0x140dea11228a3211.0x18565bc5
d5e4e4fe.0x2"
             }
            ]
}
```

REST

```
curl -k -X POST 'https://host:port/
rest/file/tierjobabort?name=/c1/
file5G' --user mapr:mapr
{"timestamp":1557738947905,"timeofday"
:"2019-05-13 02:15:47.905 GMT-0700
AM","status":"OK","total":1,"data":
[{"status":10,"message":"File
transfer being
aborted.","gateway":"host:port","jobid
":"0x140dea11228a3211.0x18565bc5d5e4e4
fe.0x3"}]}
```

file tierjobstatus

Checks the status of a previous offload or recall operation.

Permissions Required

The user running the command must have (mode bit or *ACE*) permissions to write to the file.

Syntax

CLI

```
/opt/mapr/bin/maprcli file
tierjobstatus
            -name <file_name>
            [ -job <jobID> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host:port>/rest/file/tierjobstatus?<parameters>`

Parameters

Parameter	Description
`name`	The name (including the path) of the file.
`job`	The ID of the job specified with the offload or recall command.

Output

The command returns ***one*** of the following messages:

FTOS_SUCCESS

Indicates that the file tiering operation was successful. For example:

last-updated: Apr 11, 2022

CLI

```
/opt/mapr/bin/
maprcli file
tierjobstatus  -n
ame /v5/
nfile2 -json
   {

"timestamp":15335
55093521,

"timeofday":"201
8-08-06
04:31:33.521
GMT-0700 AM",

"status":"OK",
    "total":1,
    "data":[
            {

"status":0,

"message":"File
offload
completed.",

"gateway":"10.10.
104.21:8660",

"op":"Offload",

"completedFids":2
,

"failedFids":0,

"totalFids":2
             }
            ]
    }
```

REST

```
curl -k -X GET
'https://
abc.sj.us/rest/
file/
tierjobstatus?
name=/v5/
nfile2' -- user
mapr:mapr

{"timestamp":1533
555093521,"timeof
day":"2018-08-06
04:31:33.521
GMT-0700
AM","status":"OK"
,"total":1,
 "data":
[{"status":0,"mes
sage":"File
```

```
offload
completed.","gate
way":"10.10.104.2
1:8660","op":"Off
load",

"completedFids":2
,"failedFids":0,"
totalFids":2}]}
```

OP_FAIL

Indicates that the operation failed. For example:

CLI

```
/opt/mapr/bin/
maprcli file
tierjobstatus -na
me /
volume_cold_aws/
sampleFile2 -json
 {

"timestamp":15339
37284242,

"timeofday":"201
8-08-10
02:41:24.242
GMT-0700 PM",

"status":"ERROR",
   "errors":[
             {

"id":2,

"desc":"File
offload failed."
            }
           ]
 }
```

REST

```
curl -k -X GET
'https://
abc.sj.us:8443/
rest/file/
tierjobstatus?
name=/
volume_cold_aws/
sampleFile2' --us
er mapr:mapr

{"timestamp":1533
937284242,"timeof
day":"2018-08-10
02:41:24.242
GMT-0700
PM","status":"ERR
OR",
 "errors":
[{"id":2,"desc":"
```

```
File offload
failed."}]}
```

INVALID_FILE

Indicates that the specified file does not exist. For example:

CLI

```
/opt/mapr/bin/
maprcli file
tierjobstatus -na
me /ecvol1/
file3_24 -json
  {

"timestamp":15341
88250720,

"timeofday":"201
8-08-13
12:24:10.720
GMT-0700 PM",

"status":"ERROR",
    "errors":[
              {

"id":3,

"desc":"Tierfile
transfer failed,
Could not open
file /ecvol1/
file3_24"
             }
            ]
  }
```

REST

```
curl -k -X GET
'https://
abc.sj.us:8443/
rest/file/
tierjobstatus?
name=/ecvol1/
file3_24' --user
mapr:mapr

{"timestamp":1534
188250720,"timeof
day":"2018-08-13
12:24:10.720
GMT-0700
PM","status":"ERR
OR",
  "errors":
[{"id":3,"desc":"
Tierfile
transfer failed,
Could not open
```

```
file /ecvol1/
file3_24"}]}
```

FILE_EMPTY

Indicates that the file contains no data and is empty. For example:

CLI

```
/opt/mapr/bin/
maprcli file
tierjobstatus -na
me /vol1/
test1 -json
 {

"timestamp":15341
41220360,

"timeofday":"201
8-08-12
11:20:20.360
GMT-0700 PM",

"status":"ERROR",
  "errors":[
            {

"id":5,

"desc":"File
empty."
           }
          ]
 }
```

REST

```
curl -k -X GET
'https://
abc.sj.us:8443/
rest/file/
tierjobstatus?
name=/vol1/
test1' --user
mapr:mapr

{"timestamp":1534
142083085,"timeof
day":"2018-08-12
11:34:43.085
GMT-0700
PM","status":"ERR
OR",
 "errors":
[{"id":5,"desc":"
File empty."}]}
```

NO_GATEWAY

Indicates that there is no MAST Gateway available. For example:

CLI

```
/opt/mapr/bin/
maprcli file
tierjobstatus -na
me /ecvol1/
file2 -json
 {

"timestamp":15341
85984585,

"timeofday":"201
8-08-13
11:46:24.585
GMT-0700 AM",

"status":"ERROR",
  "errors":[
             {

"id":6,

"desc":"Lost
connection to
gateway."
             }
           ]
 }
```

REST

```
curl -k -X GET
'https://
abc.sj.us:8443/
rest/file/
tierjobstatus?
name=/ecvol1/
file2' --user
mapr:mapr

{"timestamp":1534
185984585,"timeof
day":"2018-08-13
11:46:24.585
GMT-0700
AM","status":"ERR
OR",
  "errors":
[{"id":6,"desc":"
Lost connection
to gateway."}]}
```

HAS_LOCAL_DATA

Indicates that the data is still on the cluster. For example:

CLI

```
/opt/mapr/bin/
maprcli file
tierjobstatus -na
me /vol1/
mfs1 -json
  File has local
```

```
data.
  {

"timestamp":15341
41820011,

"timeofday":"201
8-08-12
11:30:20.011
GMT-0700 PM",

"status":"OK",
    "total":1,
    "data":[
               {

"status":8,

"message":"File
has local data."
                }
            ]
  }
```

REST

```
curl -k -X GET
'https://
abc.sj.us:8443/
rest/file/
tierjobstatus?
name=/vol1/
mfs1' --user
mapr:mapr

{"timestamp":153414
1975490,"timeofday"
:"2018-08-12
11:32:55.490
GMT-0700
PM","status":"OK",
  "total":1,"data":
[{"status":8,"messa
ge":"File has
local data."}]}
```

FTOS_ABORTED

Indicates that the file tiering operation was aborted. For example:

CLI

```
/opt/mapr/bin/
maprcli file
tierjobstatus -na
me /v3/dataVol/
file5 -json
 {

"timestamp":15338
45080525,

"timeofday":"201
8-08-09
01:04:40.525
```

```
GMT-0700 PM",
   "status":"OK",
   "total":1,
   "data":[
          {

"status":9,

"message":"Transf
er aborted.",

"gateway":"10.10.
25.22:8660",

"op":"Offload",

"completedFids":9
,

"failedFids":0,

"totalFids":9
          }
        ]
 }
```

REST

```
curl -k -X GET
'https://
abc.sj.us:8443/
rest/file/
tierjobstatus?
name=/v3/dataVol/
file5' --user
mapr:mapr

{"timestamp":1533
845080525,"timeof
day":"2018-08-09
01:04:40.525
GMT-0700
PM","status":"OK"
,

"total":1,"data":
[{"status":9,"mes
sage":"Transfer
aborted.","gatewa
y":"10.10.25.22:8
660",

"op":"Offload","c
ompletedFids":9,"
failedFids":0,"to
talFids":9}]}
```

FTOS_ABORT_IN_PROGRESS

Indicates that the file tiering operation is being aborted. For example:

CLI

```
/opt/mapr/bin/
maprcli file
tierjobstatus -na
me /v3/dataVol/
file5 -json
 {

"timestamp":15338
45004549,

"timeofday":"201
8-08-09
01:03:24.549
GMT-0700 PM",
   "status":"OK",
   "total":1,
   "data":[
            {

"status":10,

"message":"File
transfer being
aborted.",

"gateway":"10.10.
25.22:8660",

"op":"Offload",

"completedFids":5
,

"failedFids":0,

"totalFids":9
          }
        ]
 }
```

REST

```
curl -k -X GET
'https://
abc.sj.us:8443/
rest/file/
tierjobstatus?
name=/v3/dataVol/
file5' --user
mapr:mapr

{"timestamp":1533
845004549,"timeof
day":"2018-08-09
01:03:24.549
GMT-0700
PM","status":"OK"
,

"total":1,"data":
[{"status":10,"me
ssage":"File
```


last-updated: Apr 11, 2022

```
transfer being
aborted.","gatewa
y":"10.10.25.22:8
660",

"op":"Offload","c
ompletedFids":5,"
failedFids":0,"to
talFids":9}]}
```

FTOS_TRANSFER_IN_PROGRESS

Indicates that offload or recall of file data is currently in progress. For example:

CLI

```
/opt/mapr/bin/
maprcli file
tierjobstatus -na
me /v3/dataVol/
file5 -json
 {

"timestamp":15338
44965363,

"timeofday":"201
8-08-09
01:02:45.363
GMT-0700 PM",
   "status":"OK",
   "total":1,
   "data":[
            {

"status":11,

"message":"File
transfer in
progress.",

"gateway":"10.10.
25.22:8660",

"op":"Offload",

"completedFids":2
,

"failedFids":0,

"totalFids":9
            }
         ]
 }
```

REST

```
curl -k -X GET
'https://
abc.sj.us:8443/
rest/file/
tierjobstatus?
name=/v3/dataVol/
```

```
file5' --user
mapr:mapr

{"timestamp":1533
844965363,"timeof
day":"2018-08-09
01:02:45.363
GMT-0700
PM","status":"OK"
,

"total":1,"data":
[{"status":11,"me
ssage":"File
transfer in
progress.","gatew
ay":"10.10.25.22:
8660",

"op":"Offload","c
ompletedFids":2,"
failedFids":0,"to
talFids":9}]}
```

FTOS_REQ_QUEUED

Indicates that the file is queued for offload. For example:

CLI

```
/opt/mapr/bin/
maprcli file
tierjobstatus -na
me /v5/
egFile2 -json
  {

"timestamp":15341
87988469,

"timeofday":"201
8-08-13
12:19:48.469
GMT-0700 PM",

"status":"OK",
    "total":1,
    "data":[
            {

"status":12,

"message":"File
transfer request
queued.",

"gateway":"10.10.
25.29:8660",

"op":"Offload",

"completedFids":0
,
```

```
"failedFids":0,

"totalFids":0
            }
           ]
  }
```

REST

```
curl -k -X GET
'https://
abc.sj.us:8443/
rest/file/
tierjobstatus?
name=/v5/
egFile2' --uesr
mapr:mapr

{"timestamp":1534
187988469,"timeof
day":"2018-08-13
12:19:48.469
GMT-0700
PM","status":"OK"
,

"total":1,"data":
[{"status":12,"me
ssage":"File
transfer request
queued.","gateway
":"10.10.25.29:86
60",

"op":"Offload","c
ompletedFids":0,"
failedFids":0,"to
talFids":0}]}
```

FTOS_JOB_NOT_AVAILABLE

Indicates that the job ID associated with the specified file tiering operation is not available or is invalid. For example:

CLI

```
/opt/mapr/bin/
maprcli file
tierjobstatus -na
me /v5/
nfile2 -json
 {

"timestamp":15338
41993320,

"timeofday":"201
8-08-09
12:13:13.320
GMT-0700 PM",

"status":"ERROR",
   "errors":[
```

```
              {

"id":13,

"desc":"File has
no active
transfer in
progress."
              }
            ]
 }
```

REST

```
curl -k -X GET
'https://
abc.sj.us:8443/
rest/file/
tierjobstatus?
name=/v5/
nfile2' --user
mapr:mapr

{"timestamp":1533
841993320,"timeof
day":"2018-08-09
12:13:13.320
GMT-0700
PM","status":"ERR
OR",
  "errors":
[{"id":13,"desc":
"File has no
active transfer
in progress."}]}
```

FTOS_EPERM

Indicates that the user cannot perform the tiering operation. For example:

CLI

```
/opt/mapr/bin/
maprcli file
tierjobstatus -na
me /ecvol1/
file3_1 -json
  {

"timestamp":15341
88598543,

"timeofday":"201
8-08-13
12:29:58.543
GMT-0700 PM",

"status":"ERROR",
    "errors":[
                {

"id":14,

"desc":"File
```

```
transfer request
permission
denied."
              }
            ]
  }
```

REST

```
curl -k -X GET
'https://
abc.sj.us:8443/
rest/file/
tierjobstatus?
name=/ecvol1/
file3_1' --user
mapr:mapr

{"timestamp":1534
188598543,"timeof
day":"2018-08-13
12:29:58.543
GMT-0700
PM","status":"ERR
OR",
   "errors":
[{"id":14,"desc":
"File transfer
request
permission
denied."}]}
```

file tierstatus
Checks the status of the file offload operation and returns information on whether or not the file has any local data.

This command does not require a MAST Gateway.

Syntax

CLI

```
maprcli file tierstatus
         -name <file_name>
```

REST

Request Type	GET
Request URL	`http[s]://<host:port>/rest/file/tierstatus?<parameters>`

Parameters

Parameter	Description
`name`	The name (including the path) of the file.

Output

The output of this command varies based on whether or not data is local, was offloaded, or was recalled. The output returns *one* of the following messages:

- Data was completely offloaded:

```
File does not have local data
```

- Data could not be completely offloaded or data was recalled:

```
File has local data
```

- File is not configured for tiering:

```
File is not on a tiered volume
```

Examples

Retrieve the status of file named new2test4 in volume name testvol2:

CLI

```
maprcli file tierstatus -name /
testvol2/new2test4 -json
 File does not have local data.
  {
        "timestamp":1514877988773,
        "timeofday":"2018-01-01
11:26:28.773 GMT-0800",
        "status":"OK",
        "total":1,
        "data":[
                {
                    "status":1,
                    "message":"File
does not have local data."
                }
        ]
  }
```

REST

Send a request of type GET. For example:

```
curl -k -X GET 'https://
abc.sj.us:8443/rest/file/tierstatus?
name=/testvol2/new2test4' --user
mapr:mapr

{"timestamp":1514877988773,"timeofday"
:"2018-01-01 11:26:28.773
GMT-0800","status":"OK","total":1,
  "data":[{"status":1,"message":"File
does not have local data."}]}
```

Retrieve the status of file named new2test3 in volume named testvol2:

CLI

```
maprcli file tierstatus -name /
testvol2/new2test3 -json
 File has local data.
```


last-updated: Apr 11, 2022

```
  {
        "timestamp":1514878021374,
        "timeofday":"2018-01-01
11:27:01.374 GMT-0800",
        "status":"OK",
        "total":1,
        "data":[
                {
                    "status":0,
                    "message":"File
has local data."
                }
        ]
  }
```

REST

Send a request of type GET. For example:

```
curl -k -X GET 'https://
abc.sj.us:8443/rest/file/tierstatus?
name=/testvol2/new2test3' --user
mapr:mapr

{"timestamp":1514878021374,"timeofday"
:"2018-01-01 11:27:01.374
GMT-0800","status":"OK","total":1,
  "data":[{"status":0,"message":"File
has local data."}]}
```

Retrieve the status of file named file0 in volume named dir1 inside a volume called std_volume:

CLI

```
/opt/mapr/bin/maprcli
file tierstatus -name /std_volume/
dir1/file0 -json
  File is not on a tiered volume.
 {
  "timestamp":1609831337961,
  "timeofday":"2021-01-04
11:22:17.961 GMT-0800 PM",
  "status":"ERROR",
  "errors":[
           {
            "id":4,
            "desc":"File is not on a
tiered volume."
           }
          ]
 }
```

REST

Send a request of type GET. For example:

```
curl -k -X
GET 'https://abc.sj.us:8443/rest/
file/tierstatus?name=/std_volume/dir1/
file0' --user mapr:mapr
```

job
Manages Hadoop jobs running on the cluster.

job linklogs

Creates symbolic links to all the logs relating to the activity of a specific job.

The `maprcli job linklogs` command works with the Centralized Logging to provide a job-centric view or an application-centric view of all log files generated during job or application execution.

The output of `job linklogs` is a directory populated with symbolic links to all log files related to the specified job(s) or to the application. The command can be performed during or after a job or application is processed.

Syntax

CLI

```
maprcli job linklogs
    -jobid <jobPattern>
    -todir <desinationDirectory>
    [-jobconf <pathToJobXml>]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/job/linklogs?<parameters>`

Parameters

Parameter	Description
`jobid`	For MapReduce version 2, specify the application ID.
`todir`	The target directory for the symbolic links to the log files.
`jobconf`	For MapReduce version 2, this parameter is not applicable.

Output

For MapReduce version 2, the following directory will be created in the location specified by `todir` for the application ID that you specify for the `jobid` parameter:

- `<applicationId>/hosts/<host>/` contains symbolic links to log directories of tasks executed for <applicationId> on <host>

Examples

Link logs for all jobs named "wordcount1" and dump output to /myvolume/joblogviewdir:

CLI

```
maprcli job linklogs -jobid
job_*_wordcount1 -todir /myvolume/
joblogviewdir
```

REST

```
https://abc.sj.us:8443/
rest/job/linklogs?
jobid=job_*_wordcount1&todir=/
myvolume/joblogviewdir
```


last-updated: Apr 11, 2022

license
Manages MapR licenses.
license add
Adds a license. Permissions required: `fc` or `a`.

You can specify the license either by passing the license string itself to `license add`, or by specifying a file containing the license string. In a multinode cluster, add the license to one node (any node). Adding the same license to more than one node returns an error.

Syntax

CLI

```
maprcli license add
    [ -cluster cluster name ]
    [ -is_file true|false. default:
false ]
    -license long_license_string
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/license/add?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`is_file`	Specifies whether the `license` specifies a file. If `false`, the `license` parameter contains a long license string.
`license`	The license to add to the cluster. If `-is_file` is true, `license` specifies the file name of a license file. Otherwise, `license` contains the license string itself.

Examples

Note: After obtaining a valid license file from your MapR sale representative, copy the license file to a cluster node, for example in the path `/tmp/license.txt`.

To add a license from a file:

CLI

```
maprcli license add -is_file
true -license /tmp/license.txt
```

REST

```
https://abc.sj.us:8443/rest/license/
add?
is_file=true&license=%2Ftmp%2Flicense.
txt
```

Related concepts

Upgrading and Your License on page 295

If you are upgrading from MapR version 5.0 or earlier, the Base License file must be manually updated on all nodes in your cluster.

Related tasks

Viewing the Licenses on the Cluster on page 778
List the licenses on the cluster using either the Control System or the CLI.

Adding a License on page 777
Add a license through the Control System or the CLI.

Removing a License on page 780
Describes how to remove a license using the Control System and the CLI.

Related reference

license addcrl on page 1681
Adds a certificate revocation list (CRL). Permissions required: `fc` or `a`.

license apps on page 1682
Displays the features authorized for the current license. Permissions required: `login`

license list on page 1684
Lists licenses on the cluster. Permissions required: `login`. For best results, use the `-json` option when running the command.

license listcrl on page 1686
Lists certificate revocation lists (CRLs) on the cluster. Permissions required: `login`.

license remove on page 1687
Removes a license. Permissions required: `fc` or `a`.

license showid on page 1689
Displays the cluster ID for use when creating a new license. Permissions required: `login`.

license addcrl

Adds a certificate revocation list (CRL). Permissions required: `fc` or `a`.

Syntax

CLI

```
maprcli license addcrl
    [ -cluster <cluster> ]
    -crl <crlstring>
    [ -is_file true|false. default:
false ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/license/addcrl?<parameters>`

Parameters

Parameter	Description
cluster	The cluster on which to run the command.

Parameter	Description
crl	The CRL to add to the cluster. If file is set, `crl` specifies the filename of a CRL file. Otherwise, `crl` contains the CRL string itself.
is_file	Specifies whether the license is contained in a file.

Examples

CLI

```
maprcli license addcrl
   -crl crl.txt
   -is_file true
```

REST

```
https://centos26.lab:8443/
rest/license/addcrl?
crl=crl.txt&is_file=true
```

Related concepts

Upgrading and Your License on page 295
If you are upgrading from MapR version 5.0 or earlier, the Base License file must be manually updated on all nodes in your cluster.

Related tasks

Viewing the Licenses on the Cluster on page 778
List the licenses on the cluster using either the Control System or the CLI.

Adding a License on page 777
Add a license through the Control System or the CLI.

Removing a License on page 780
Describes how to remove a license using the Control System and the CLI.

Related reference

license add on page 1680
Adds a license. Permissions required: `fc` or `a`.

license apps on page 1682
Displays the features authorized for the current license. Permissions required: `login`

license list on page 1684
Lists licenses on the cluster. Permissions required: `login`. For best results, use the `-json` option when running the command.

license listcrl on page 1686
Lists certificate revocation lists (CRLs) on the cluster. Permissions required: `login`.

license remove on page 1687
Removes a license. Permissions required: `fc` or `a`.

license showid on page 1689
Displays the cluster ID for use when creating a new license. Permissions required: `login`.

license apps

Displays the features authorized for the current license. Permissions required: `login`

last-updated: Apr 11, 2022

Syntax

CLI

```
maprcli license apps
[ -cluster <cluster> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/license/apps[?<parameters>]`

Parameters

Parameter	Description
cluster	The cluster on which to run the command.

Output

Sample Output

```
# maprcli license apps
capability             grace  featuredata
NFS                    false  unlimited
NFS_MULTINODE          false
NFS_HA                 false
MULTI_CLUSTER          false
CLDB_HA                false
JOBTRACKER_HA          false
SNAPSHOT               false
MIRRORING              false
DATA_PLACEMENT         false
MAXNODES               false  unlimited
OPTIMIZED_SHUFFLE      false
JM_CHARTS              false
JM_HISTOGRAMS          false
MAPR_TABLES            false
MAPR_TABLES_FULL       false
POSIX_CLIENT           false
POSIX_CLIENT_BASE      true
POSIX_CLIENT_GOLD      false
POSIX_CLIENT_PLATINUM  false
MAPR_STREAMS           false
MAPR_STREAMS_FULL      false
JM_CHARTS              false
JM_HISTOGRAMS          false
```

Example

CLI

```
maprcli license apps
```

REST

```
https://abc.sj.us:8443/rest/license/apps
```


last-updated: Apr 11, 2022

Related concepts

Upgrading and Your License on page 295
If you are upgrading from MapR version 5.0 or earlier, the Base License file must be manually updated on all nodes in your cluster.

Related tasks

Viewing the Licenses on the Cluster on page 778
List the licenses on the cluster using either the Control System or the CLI.

Adding a License on page 777
Add a license through the Control System or the CLI.

Removing a License on page 780
Describes how to remove a license using the Control System and the CLI.

Related reference

license add on page 1680
Adds a license. Permissions required: `fc` or `a`.

license addcrl on page 1681
Adds a certificate revocation list (CRL). Permissions required: `fc` or `a`.

license list on page 1684
Lists licenses on the cluster. Permissions required: `login`. For best results, use the `-json` option when running the command.

license listcrl on page 1686
Lists certificate revocation lists (CRLs) on the cluster. Permissions required: `login`.

license remove on page 1687
Removes a license. Permissions required: `fc` or `a`.

license showid on page 1689
Displays the cluster ID for use when creating a new license. Permissions required: `login`.

license list

Lists licenses on the cluster. Permissions required: `login`. For best results, use the `-json` option when running the command.

Syntax

CLI

```
maprcli license list
[ -cluster <cluster> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/license/list[?<parameters>]`

Parameters

Parameter	Description
cluster	The cluster on which to run the command.

Note: If you use the `-json` option with this command, and you pipe the output into another program, you may find that the result cannot be parsed by certain JSON libraries, such as the JSON library for Python. To work around this problem, you can replace the single-escape characters (\) in the JSON output that the `license list` command returns, with double-escape characters (\\).

Output

Sample Output

```
# maprcli license list -json
{
    "timestamp":1433543033194,
    "timeofday":"2015-06-05 10:23:53.194 GMT+0000",
    "status":"OK",
    "total":2,
    "data":[
        {
            "id":"88aEvYonv5HqJgaFrGfsKis5puQ=",
            "description":"Base MapR POSIX Client for fast secure file
access",
            "nfscliendnodes":"10",
            "isAdditioanlFeature":true,
            "deletable":false,
            "grace":true,
            "license":"version: \"4.0\"\ncustomerid:
\"BaseLicenseUser\"\nissuer: \"MapR Technologies,
             Inc.\"\nlicType: AdditionalFeaturesBase\ndescription: \"Base
MapR POSIX Client for fast secure file access\"\nenforcement:
HARD\ncapabilities {\n  feature: NFS_CLIENT_BASE\n  name: \"MapR POSIX
CLIENT\"\n  permission: ALLOW\n  featureData {\n    maxNfsClientNodes:
\"10\"\n  }\n}\nhash: \"88aEvYonv5HqJgaFrGfsKis5puQ=\"\n"
        },
        {
            "id":"iSs4C9+yb9WSbE1lHJGy5KW0m3E=",
            "description":"MapR Base Edition",
            "maxnodes":"unlimited",
            "isAdditioanlFeature":false,
            "deletable":false,
            "grace":true,
            "license":"version: \"4.0\"\ncustomerid:
\"BaseLicenseUser\"\nissuer: \"MapR Technologies,
             Inc.\"\nlicType: Base\ndescription: \"MapR Base
Edition\"\nenforcement: HARD\ncapabilities {\n  feature: MAXNODES\n
name: \"Max Nodes in Cluster\"\n  permission: ALLOW\n  featureData
{\n    maxNodes: \"unlimited\"\n  }\n}\ncapabilities {\n  feature:
MAPR_TABLES\n  name: \"MapR Tables\"\n  permission: ALLOW\n}\nhash:
\"iSs4C9+yb9WSbE1lHJGy5KW0m3E=\"\n"
        }
    ]
}
```

Examples

CLI

```
maprcli license list -json
```

REST

```
https://abc.sj.us:8443/rest/license/
list
```

Related concepts

Upgrading and Your License on page 295
If you are upgrading from MapR version 5.0 or earlier, the Base License file must be manually updated on all nodes in your cluster.

Related tasks

Viewing the Licenses on the Cluster on page 778
List the licenses on the cluster using either the Control System or the CLI.

Adding a License on page 777
Add a license through the Control System or the CLI.

Removing a License on page 780
Describes how to remove a license using the Control System and the CLI.

Related reference

license add on page 1680
Adds a license. Permissions required: `fc` or `a`.

license addcrl on page 1681
Adds a certificate revocation list (CRL). Permissions required: `fc` or `a`.

license apps on page 1682
Displays the features authorized for the current license. Permissions required: `login`

license listcrl on page 1686
Lists certificate revocation lists (CRLs) on the cluster. Permissions required: `login`.

license remove on page 1687
Removes a license. Permissions required: `fc` or `a`.

license showid on page 1689
Displays the cluster ID for use when creating a new license. Permissions required: `login`.

license listcrl

Lists certificate revocation lists (CRLs) on the cluster. Permissions required: `login`.

Syntax

CLI

```
maprcli license listcrl
[ -cluster <cluster> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/license/listcrl[?<parameters>]`

Parameters

Parameter	Description
cluster	The cluster on which to run the command.

Examples

CLI

```
maprcli license listcrl
   -cluster my.test.cluster
```

REST

```
https://abc.sj.us:8443/rest/license/
listcrl?cluster=my.test.cluster
```

Related concepts

Upgrading and Your License on page 295
If you are upgrading from MapR version 5.0 or earlier, the Base License file must be manually updated on all nodes in your cluster.

Related tasks

Viewing the Licenses on the Cluster on page 778
List the licenses on the cluster using either the Control System or the CLI.

Adding a License on page 777
Add a license through the Control System or the CLI.

Removing a License on page 780
Describes how to remove a license using the Control System and the CLI.

Related reference

license add on page 1680
Adds a license. Permissions required: `fc` or `a`.

license addcrl on page 1681
Adds a certificate revocation list (CRL). Permissions required: `fc` or `a`.

license apps on page 1682
Displays the features authorized for the current license. Permissions required: `login`

license list on page 1684
Lists licenses on the cluster. Permissions required: `login`. For best results, use the `-json` option when running the command.

license remove on page 1687
Removes a license. Permissions required: `fc` or `a`.

license showid on page 1689
Displays the cluster ID for use when creating a new license. Permissions required: `login`.

license remove

Removes a license. Permissions required: `fc` or `a`.

Syntax

CLI

```
maprcli license remove
[ -cluster <cluster> ]
-license_id <license>
```

REST

Request Type	POST

Request URL	`http[s]://<host>:<port>/rest/license/remove?<parameters>`

Parameters

Parameter	Description
cluster	The cluster on which to run the command.
license_id	The license to remove.

Examples

CLI

```
maprcli license remove -license_id
5119043355327235351
```

REST

```
https://10.10.82.23:8443/rest/license/
remove?license_id=5119043355327235351
```

Related concepts

Upgrading and Your License on page 295
If you are upgrading from MapR version 5.0 or earlier, the Base License file must be manually updated on all nodes in your cluster.

Related tasks

Viewing the Licenses on the Cluster on page 778
List the licenses on the cluster using either the Control System or the CLI.

Adding a License on page 777
Add a license through the Control System or the CLI.

Removing a License on page 780
Describes how to remove a license using the Control System and the CLI.

Related reference

license add on page 1680
Adds a license. Permissions required: `fc` or `a`.

license addcrl on page 1681
Adds a certificate revocation list (CRL). Permissions required: `fc` or `a`.

license apps on page 1682
Displays the features authorized for the current license. Permissions required: `login`

license list on page 1684
Lists licenses on the cluster. Permissions required: `login`. For best results, use the `-json` option when running the command.

license listcrl on page 1686
Lists certificate revocation lists (CRLs) on the cluster. Permissions required: `login`.

license showid on page 1689
Displays the cluster ID for use when creating a new license. Permissions required: `login`.

last-updated: Apr 11, 2022

license showid

Displays the cluster ID for use when creating a new license. Permissions required: `login`.

Syntax

CLI

```
maprcli license showid
[ -cluster <cluster> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/license/showid[?<parameters>]`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.

Output

Sample Output

```
# maprcli license showid
id
5119043355327235351
```

Examples

CLI

```
maprcli license showid
```

REST

```
https://abc.sj.us:8443/rest/license/
showid?showNodes=true
```

Related concepts

Upgrading and Your License on page 295
If you are upgrading from MapR version 5.0 or earlier, the Base License file must be manually updated on all nodes in your cluster.

Related tasks

Viewing the Licenses on the Cluster on page 778
List the licenses on the cluster using either the Control System or the CLI.

Adding a License on page 777
Add a license through the Control System or the CLI.

Removing a License on page 780
Describes how to remove a license using the Control System and the CLI.

Related reference

license add on page 1680

last-updated: Apr 11, 2022

Adds a license. Permissions required: `fc` or `a`.

license addcrl on page 1681
Adds a certificate revocation list (CRL). Permissions required: `fc` or `a`.

license apps on page 1682
Displays the features authorized for the current license. Permissions required: `login`

license list on page 1684
Lists licenses on the cluster. Permissions required: `login`. For best results, use the `-json` option when running the command.

license listcrl on page 1686
Lists certificate revocation lists (CRLs) on the cluster. Permissions required: `login`.

license remove on page 1687
Removes a license. Permissions required: `fc` or `a`.

nfsmgmt
Refreshes NFS exports and server cache.
nfsmgmt refreshexports
Refreshes the list of clusters and mount points available to mount with NFS. Permissions required: `fc` or `a`.

Syntax

CLI

```
[ -nfshost <ip or hostname>  ]
[ -nfsport <port> ]
[ -isusermode <true | false> ]
```

REST N/A

Parameters

Parameter	Description
`nfshost`	The hostname of the node that is running the MapR NFS server. Default: 127.0.0.1
`nfsport`	The port to use. Default: 9998
`isusermode`	Specifies whether the mode for creating the ticket for NFS is in user mode or not. Options: True or False. The ticket can not be created for nfs in user mode. Default: false

Example

CLI

```
maprcli nfsmgmt refreshexports -nfshost
10.10.82.29 -nfsport 9998
```

nfsmgmt refreshgidcache
Deletes the GID list (`uidGidCache_` entries) in NFS server cache. Permissions required: `fc` or `a`.

Useful for immediately reflecting the groups update.

Syntax

CLI

```
    [ -nfshost <ip or hostname>  ]
    [ -nfsport <port> ]
    [ -isusermode <true | false> ]
```

REST

N/A

Parameters

Parameter	Description
nfshost	The hostname of the node that is running the MapR NFS server. Default: 127.0.0.1
nfsport	The port to use. Default: 9998
isusermode	Specifies whether the mode for nfs is in user mode or not. Options: True or False. The ticket can not be created for nfs in user mode. Default: false

Example

CLI

```
maprcli nfsmgmt refreshgidcache -nfshost
10.10.82.29 -nfsport 9998
```

nfs4mgmt
Manages NFSv4 server.

nfs4mgmt add-export
Adds an export.

Syntax

CLI

```
/opt/mapr/bin/maprcli nfs4mgmt
add-export
       [ -nfshost ip/hostname.
default: 127.0.0.1 ]
       [ -nfsport port. default:
9995 ]
       -exportid export id. default:
0
       -conffile conf file path
```

REST

N/A

Parameters

Parameter	Description
conffile	The path to the NFSv4 configuration file.
exportid	The export ID as specified in the configuration file. The default value is 0.
nfshost	The NFS server host. Value can be the IP address or the hostname of the NFS server host. The default value is 127.0.0.1.

last-updated: Apr 11, 2022

Parameter	Description
`nfsport`	The NFS server port. The default value is 9995.

Examples

nfs4mgmt list-exports
Returns the list of exports.

Syntax

CLI

```
/opt/mapr/bin/maprcli nfs4mgmt
list-exports
        [ -nfshost ip/hostname.
default: 127.0.0.1 ]
        [ -nfsport port. default:
9995 ]
```

REST

N/A

Parameters

Parameter	Description
`nfshost`	The NFS server host. Value can be the IP address or the hostname of the NFS server host. The default value is 127.0.0.1.
`nfsport`	The NFS server port. The default value is 9995.

Output

Example

Troubleshooting

Issue

Sometimes, you might see the following error when you run the `list-exports` command:

```
Error
org.freedesktop.DBus.Error.ServiceUnkn
own: The name org.ganesha.nfsd was
not provided by any .service files
```

Resolution

Modify the `/etc/dbus-1/system.conf file` as follows:

last-updated: Apr 11, 2022

1. Remove the following:

```
<deny
send_destination="org.freedesktop.D
Bus"

send_interface="org.freedesktop.DBu
s"

send_member="UpdateActivationEnviro
nment"/>
```

2. Add the following:

```
<allow send_interface="*"/>
<allow receive_interface="*"/>
<allow own="*"/>
```

nfs4mgmt remove-export
Removes an export.

Syntax

CLI

```
/opt/mapr/bin/maprcli mfs4mgmt
remove-export
       [ -nfshost ip/hostname.
default: 127.0.0.1 ]
       [ -nfsport port. default:
9995 ]
       -exportid export id. default:
0
```

REST

N/A

Parameters

Parameter	Description
`exportid`	The export ID to remove as specified in the configuration file. The default value is 0.
`nfshost`	The NFS server host. Value can be the IP address or the hostname of the NFS server host. The default value is 127.0.0.1.
`nfsport`	The NFS server port. The default value is 9995.

Examples

nfs4mgmt update-export
Updates an export based on configuration changes.

Syntax

CLI

```
/opt/mapr/bin/maprcli mfs4mgmt
update-export
```

```
        [ -nfshost ip/hostname.
default: 127.0.0.1 ]
        [ -nfsport port. default:
9995 ]
        -exportid export id. default:
0
        -conffile conf file path
```

REST N/A

Parameters

Parameter	Description
`conffile`	The path to the NFSv4 configuration file.
`exportid`	The export ID to update as specified in the configuration file. The default value is 0.
`nfshost`	The NFS server host. Value can be the IP address or the hostname of the NFS server host. The default value is 127.0.0.1.
`nfsport`	The NFS server port. The default value is 9995.

Examples

node
Manages nodes in the cluster

Fields

The following table lists the data fields that provide information about each node. Each field has two names:

- Field name - displayed in the output of the `node list` command
- Short name - used to specify the columns displayed using the `columns` parameter

The short name is also used when specifying rows with a filter, for example when specifying nodes on which to perform an action with the `node services` command.

Field Name	Short Name	Description
blockMovesIn	bmi	Block moves in.
blockMovesOut	bmo	Block moves out.
bytesReceived	br	Bytes received by the node since the last CLDB heartbeat.
bytesSent	bs	Bytes sent by the node since the last CLDB heartbeat.
clienthealth	clhealth	The status of the client. Value can be one of the following: • Active • Inactive

Field Name	Short Name	Description
clienttype	cltype	The type of client. For example, posixclientgold, posixclientbasic, posixclientplatinum, LOOPBACK_NFS, NFS_V3, NFS_V4.
configuredservice	csvc	Services that are configured as roles on the node.
CorePresentAlarm	ncp	Timestamp when the Core Present alarm was raised.
cpus	cpc	The total number of CPUs on the node.
davail	dsa	Disk space available on the node, in GB.
DiskFailureAlarm	fda	Timestamp when Disk Failure on page 2226 alarm was raised.
disks	dsc	Total number of disks on the node.
dreadK	drk	Disk Kbytes read since the last heartbeat.
dreads	dro	Disk read operations since the last heartbeat.
DRILLDOWNALARM	nadrill	Timestamp when "Drill Service Down" alarm was raised.
dtotal	dst	Total disk space on the node, in GB.
dused	dsu	Disk space used on the node, in GB.
dwriteK	dwk	Disk Kbytes written since the last heartbeat.
dwrites	dwo	Disk write ops since the last heartbeat.
ESServerDown	naes	Timestamp when "Elasticsearch Server Down" alarm was raised.
faileddisks	nfd	Number of failed MapR File System disks on the node. • 0 = Clear • 1 = Raised
fs-heartbeat	fhb	Time since the last heartbeat to the CLDB, in seconds.
GatewayServiceDown	nagwsd	Timestamp when "Gateway Service Down" alarm was raised.
HbaseThriftServiceDown	hbasethrift	Timestamp when "HBase Thrift Service Down" alarm was raised.
HbProcessingSlow	hbpsa	Timestamp when Heartbeat Processing Slow on page 2227 alarm was raised.

last-updated: Apr 11, 2022

Field Name	Short Name	Description
health	h	Overall node health, calculated from various alarm states: • 0 = Healthy • 1 = Needs attention • 2 = Degraded • 3 = Maintenance • 4 = Critical
healthDesc	hd	The health description.
HighMfsMemoryAlarm	nhmm	Timestamp when MapR File System High Memory alarm was raised.
HomeMapRFullAlarm	hmf	Timestamp when Installation Directory Full alarm was raised.
hostname	hn	The host name. In the output for the `clientsonly` option, this is the hostname where the client is running.
id	id	The node ID.
IncorrectTopologyAlarm	ita	Timestamp when Incorrect Topology alarm was raised.
InstanceMismatch	nanim	Timestamp when Instance Mismatch alarm was raised.
Insufficient memory for buckets	tbwarning	Timestamp when Tiny Bucket Flush alarm was raised.
ip	ip	A list of IP addresses associated with the node. In the output for the `clientsonly` option, this is the IP address of the host where the client is running.
JobHistoryServerDown	nasjhsd	Timestamp when "Job History Server Down" alarm was raised.
jt-heartbeat	jhb	Time since the last heartbeat to the JobTracker, in seconds.
lasthb	lhb	Time since the last heartbeat from the client host.
LogLevelAlarm	lla	Timestamp when Excess Logs alarm was raised.
MapRfs disks	nmd	Number of disks for use by MapR File System
MemoryAllocationAlarm	maa	Timestamp when Memory Allocation alarm was raised.
MemorySwapping	nams	Timestamp when Memory Usage alarm was raised.
mtotal	mt	Total memory, in MB.
mused	mu	Memory used, in MB.

last-updated: Apr 11, 2022

Field Name	Short Name	Description
NodeDuplicateHostIdAlarm	ndh	Timestamp when Duplicate Host ID on page 2226 alarm was raised.
NodeManagerDown	nanmd	Timestamp when "Node Manager Down" alarm was raised.
NodeMaprUserMismatchAlarm	nma	Timestamp when MapR User Mismatch on page 2230 alarm was raised.
NodeNoHeartbeatAlarm	nha	Timestamp when No Heartbeat alarm was raised.
NodeTooManyContainersAlarm	nmc	Timestamp when Node Too Many Containers on page 2234 alarm was raised.
NoDiskAttached	nanda	Timestamp when No Disk Attached alarm was raised.
numInstances	ni	Number of configured MapR File System instances.
numReportedInstances	nri	The number of running instances reported by MapR File System to CLDB.
numResyncSlots	nrs	The number of resync slots.
numGetsInLastTenSeconds	ngl10s	Number of table get operations in last 10 seconds.
numGetsInLastMinute	ngl1m	Number of table get operations in last 1 minute.
numGetsInLastFiveMinutes	ngl5m	Number of table get operations in last 5 minutes.
numGetsInLastFifteenMinutes	ngl15m	Number of table get operations in last 15 minutes.
numPutsInLastTenSeconds	npl10s	Number of table put operations in last 10 seconds.
numPutsInLastMinute	npl1m	Number of table put operations in last 1 minute.
numPutsInLastFiveMinutes	npl5m	Number of table put operations in last 5 minutes
numPutsInLastFifteenMinutes	npl15m	Number of table put operations in last 15 minutes.
numScansInLastTenSeconds	nsl10s	Number of table scan operations in last 10 seconds.
numScansInLastMinute	nsl1m	Number of table scan operations in last 1 minute.
numScansInLastFiveMinutes	nsl5m	Number of table scan operations in last 5 minutes.
numScansInLastFifteenMinutes	nsl15m	Number of table scan operations in last 15 minutes.

last-updated: Apr 11, 2022

Field Name	Short Name	Description
PamMisconfiguredAlarm	pma	PAM misconfigured alarm (NODE_ALARM_PAM_MISCONFIGURED): • 0 = Clear • 1 = Raised
ResourceManagerDown	narmd	Timestamp when "Resource Manager Down" alarm is raised.
RootPartitionFullAlarm	rpf	Timestamp when Root Partition Full alarm was raised.
rpcin	rpi	RPC bytes received since the last heartbeat.
rpcout	rpo	RPC bytes sent since the last heartbeat.
rpcs	rpc	Number of RPCs since the last heartbeat.
service	svc	A comma-separated list of services running on the node: • cldb - CLDB • fileserver - MapR File System • nfs - NFS Gateway Example: "cldb,fileserver,nfs"
ServiceBeeswaxDownNotRunningAlarm	sbwa	Timestamp when "Beeswax Service Down" alarm was raised.
ServiceCLDBDownNotRunningAlarm	sca	Timestamp when CLDB alarm was raised.
ServiceFileserverDownNotRunningAlarm	sfsa	Timestamp when Fileserver alarm was raised.
ServiceHiveDownNotRunningAlarm	shsma	Timestamp when HiveMeta Service Down alarm was raised.
ServiceHoststatsDownNotRunningAlarm	sha	Timestamp when Hoststats alarm was raised.
ServiceHs2DownNotRunningAlarm	shsa	Timestamp when HS2 Service Down alarm was raised.
ServiceHttpfsDownNotRunningAlarm	shfsa	Timestamp when "Httpfs Service Down" alarm is raised.
ServiceHueDownNotRunningAlarm	shuea	Timestamp when "Hue Service Down" alarm was raised.
ServiceNFSDownNotRunningAlarm	sna	Timestamp when NFS alarm was raised.
ServiceOozieDownNotRunningAlarm	sooza	Timestamp when Oozie Service Down alarm was raised.
ServicesWebserverDownNotRunningAlarm	swa	Timestamp when Webserver alarm was raised.

last-updated: Apr 11, 2022

Field Name	Short Name	Description
spsPerInstance	nsp	Number of storage pools per file server instance.
TimeSkewAlarm	tsa	Timestamp when Time Skew alarm was raised.
racktopo	rp	The rack path.
uptime	cpt	Date when the node came up.
utilization	cpu	CPU use percentage since the last heartbeat.
VersionMismatchAlarm	vma	Timestamp when Version alarm was raised.
vip	vip	The virtuap IP address

Related concepts

node on page 1694
Manages nodes in the cluster

Related reference

disk add on page 1602
Adds one or more disks to the specified node. Permissions required: `fc` or `a`.

volume create on page 1931
Creates a volume.

node list on page 1705
Lists nodes in the cluster.

configure.sh on page 2053
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

node allow-into-cluster

Allows host IDs to join the cluster after duplicates have been resolved.

When the CLDB detects duplicate nodes with the same host ID, all nodes with that host ID are removed from the cluster and prevented from joining it again. After making sure that all nodes have unique host IDs, you can use the `node allow-into-cluster` command to un-ban the host ID that was previously duplicated among several nodes.

Syntax

CLI

```
maprcli node allow-into-cluster
    [ -hostids <host IDs> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/node/allow-into-cluster[?<parameters>]`

Parameters

Parameter	Description
hostids	A comma-separated list of host IDs.

Examples

Allow former duplicate host IDs node1 and node2 to join the cluster:

CLI

```
maprcli node
allow-into-cluster -hostids
node1,node2
```

REST

```
https://abc.sj.us:8443/rest/node/
allow-into-cluster?hostids=node1,node2
```

node cldbmaster
Returns the address of the master CLDB node.

The `node cldbmaster` API returns the server ID and hostname of the node serving as the CLDB master node.

Syntax

CLI

```
maprcli node cldbmaster
    [ -cluster <cluster name> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/node/cldbmaster[?<parameters>]`

Parameters

Parameter	Description
`cluster`	The name of the cluster for which to return the CLDB master node information.

Examples

Return the CLDB master node information for the cluster my.cluster.com:

CLI

```
maprcli node cldbmaster -cluster
my.cluster.com
```

```
{
        "timestamp":1622099062802,
        "timeofday":"2021-05-27
```

```
07:04:22.802 GMT+0000 AM",
        "status":"OK",
        "total":1,
        "data":[
                {

"cldbmaster":"ServerID:
3523090783455785824 HostName:
m2-mapreng-vmm167214.xxxx"
                }
        ]
}
```

REST

```
curl -k -X
POST 'https://10.163.167.214:8443/
rest/node/cldbmaster?
cluster=my.cluster.com' --user
mapr:mapr
```

```
{"timestamp":1622099484367,"timeofday"
:"2021-05-27 07:11:24.367 GMT+0000
AM","status":"OK","total":1,"data":
[{"cldbmaster":"ServerID:
3523090783455785824 HostName:
m2-mapreng-vmm167214.mip.xxx"}]}
```

node failover
Fails over master containers and VIPs to another node.

When this command runs, all master and intermediate containers are moved off the node and VIPs are re-assigned.

Syntax

CLI

```
maprcli node failover
            [ -nodes <node hostname> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/node/failover[?<parameters>]`

Parameters

Parameter	Description
nodes	The hostname of the node going down.

Examples

Notify CLDB of the node, exampleHost, going down:

CLI

```
maprcli node failover -nodes
exampleHost
```

REST

```
https://abc.sj.us:8443/rest/node/
failover?nodes=exampleHost
```

node heatmap

Displays a heatmap for the specified nodes.

Syntax

CLI

```
maprcli node heatmap
    [ -cluster <cluster> ]
    [ -filter <filter> ]
    [ -view <view> ]
    -json | -long
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/node/heatmap[?<parameters>]`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`filter`	A filter specifying snapshots to preserve. See Filters for more information.

Parameter	Description

`view`	Name of the heatmap view to show:
	• `status` = Node status (the default view):
	  • `0` = Healthy
	  • `1` = Needs attention
	  • `2` = Degraded
	  • `3` = Maintenance
	  • `4` = Critical
	• `cpu` = CPU utilization, as a percent from 0-100.
	• `memory` = Memory utilization, as a percent from 0-100.
	• `diskspace` = MapR filesystem disk space utilization, as a percent from 0-100.
	• `NODE_*` = Status of various alarms: `0` if clear, `1` if raised. For example: `NODE_ALARM_DISK_FAILURE` or `NODE_ALARM_SERVICE_CLDB_DOWN`. You can return a complete list of supported alarm parameters by running:
	`maprcli node heatmap -view`
	See the lists of parameters below this table.

`-json` | `-long`	This command returns multiple levels of data. You must specify either JSON format or "long" format to see the full output.

Alarm Parameters

You can view the status of a number of different alarms, including the status of alarms for services down and alarms for other conditions on the cluster.

Service Down Alarms

`NODE_ALARM_SERVICE_CLDB_DOWN` `NODE_ALARM_SERVICE_FILESERVER_DOWN`
`NODE_ALARM_SERVICE_JT_DOWN` `NODE_ALARM_SERVICE_TT_DOWN`
`NODE_ALARM_SERVICE_HBMASTER_DOWN` `NODE_ALARM_SERVICE_HBREGION_DOWN`
`NODE_ALARM_SERVICE_WEBSERVER_DOWN` `NODE_ALARM_SERVICE_NFS_DOWN`
`NODE_ALARM_SERVICE_HOSTSTATS_DOWN` `NODE_ALARM_SERVICE_OOZIE_DOWN`
`NODE_ALARM_SERVICE_HUE_DOWN` `NODE_ALARM_SERVICE_HTTPFS_DOWN`
`NODE_ALARM_SERVICE_BEESWAX_DOWN` `NODE_ALARM_SERVICE_HIVEMETA_DOWN`
`NODE_ALARM_SERVICE_HS2_DOWN`

Other Alarms

`NODE_ALARM_DEBUG_LOGGING` `NODE_ALARM_DISK_FAILURE` `NODE_ALARM_VERSION_MISMATCH`
`NODE_ALARM_TIME_SKEW` `NODE_ALARM_ROOT_PARTITION_FULL`
`NODE_ALARM_OPT_MAPR_FULL` `NODE_ALARM_CORE_PRESENT` `NODE_ALARM_HIGH_MFS_MEMORY`
`NODE_ALARM_PAM_MISCONFIGURED` `NODE_ALARM_TT_LOCALDIR_FULL`
`NODE_ALARM_NO_HEARTBEAT` `NODE_ALARM_MAPRUSER_MISMATCH`
`NODE_ALARM_DUPLICATE_HOSTID` `NODE_ALARM_METRICS_WRITE_PROBLEM`
`NODE_ALARM_TOO_MANY_CONTAINERS`

Output

In general, the heatmap output looks like this (in JSON format).

```
{
            status:"OK",
            data: [{
            "{{rackTopology}}" : {
            "{{nodeName}}" : {{heatmapValue}},
            "{{nodeName}}" : {{heatmapValue}},
            "{{nodeName}}" : {{heatmapValue}},
            ...
            },
            "{{rackTopology}}" : {
            "{{nodeName}}" : {{heatmapValue}},
            "{{nodeName}}" : {{heatmapValue}},
            "{{nodeName}}" : {{heatmapValue}},
            ...
            },
            ...
            }]
            }
```

Table

Field	Description
rackTopology	The topology for a particular rack.
nodeName	The name of the node in question.
heatmapValue	The value of the metric specified in the view parameter for this node, as an integer.

Examples

Display a heat map with the node status (default view) for the default rack:

```
maprcli node heatmap -json
            {
            "timestamp":1422567293873,
            "timeofday":"2015-01-29 01:34:53.873 GMT-0800",
            "status":"OK",
            "total":1,
            "data":[
            {"/data/default-rack":{
            "centos24":2}
            }]
            }
```

The equivalent REST API command would be:

```
https://r1n1.sj.us:8443/rest/node/heatmap
```

Display memory usage for the default rack:

```
maprcli node heatmap -view memory -json
            {
            "timestamp":1422585976631,
            "timeofday":"2015-01-29 06:46:16.631 GMT-0800",
            "status":"OK",
```


last-updated: Apr 11, 2022

```
            "total":1,
            "data":[
            {"/data/default-rack":{
            "centos24":71}
            }]
            }
```

The equivalent REST API command would be:

```
https://r1n1.sj.us:8443/rest/node/heatmap?view=memory
```

Display the value of NODE_ALARM_DISK_FAILURE for the default rack:

```
maprcli node heatmap -view NODE_ALARM_DISK_FAILURE -long
            /data/default-rack
            {"centos24":0}
```

node list
Lists nodes in the cluster.

You can retrieve information for a set of nodes in several ways:

- To list only nodes with raised alarms, set `alarmednodes` to `1`.
- To list only NFS nodes, set `nfsnodes` to `1`.
- To view only a few nodes from the list, use the `start` and `limit` options to select only a portion of the results.
- To list nodes that match certain criteria, pass a filter to the `filter` parameter. See the node on page 1694 table for the filter options. See the Filters on page 1526 page for information on filters.

Using the `node list` command without the `-clientsonly true` or the `-nfsnodes true` option, does not list edge nodes. To include edge nodes, use the `-nfsnodes true` or the `-clientsonly true` option.

Syntax

CLI

```
/opt/mapr/bin/maprcli node list
    [ -alarmednodes 0|1 ]
    [ -cluster <cluster> ]
    [ -clientsonly true|false ]
    [ -columns <columns>|all]
    [ -filter <filter> ]
    [ -limit <limit> ]
    [ -nfsnodes true|false ]
    [ -output terse|verbose ]
    [ -sortby <attribute> ]
    [ -sortorder asc|desc ]
    [ -start <offset> ]
    [ -zkconnect <ZooKeeper Connect
String> ]
```

REST

Request Type	GET

last-updated: Apr 11, 2022

Request URL	`http[s]://<host>:<port>/rest/node/list[?<parameters>]`

Parameters

Parameter	Description
`alarmednodes`	When set to `1`, displays only nodes with raised alarms. You cannot use this parameter if `nfsnodes` is set.
`cluster`	The cluster on which to run the command.
`clientsonly`	Set this parameter to `true` to return the list of nodes running unique platinum FUSE-based POSIX clients, and NFSv3, and NFSv4 services. The command returns the following fields: `clienttype`, `clienthealth`, `hostname`, `ip`, `lasthb`, `id`. For more information, see the fields table. If you set this parameter to `false`, which is the default value, this parameter returns node-level information for all the services running on each node.
`columns`	A comma-separated list of fields to return in the query, specified by the short names. When specifying this option, the `ip` and `hostname` columns are always returned in the query.
`filter`	A filter specifying nodes on which to start or stop services. See the Fields table on the node page for the fields available to filter. See the maprcli and REST API Syntax page for information on filters.
`limit`	The number of rows to return, beginning at start. Default: 0
`nfsnodes`	Set this to `true` to display POSIX (edge) nodes. When set to `false`, edge nodes are not displayed. Cannot be used if `alarmednodes` is set. When you set the `cldb.ignore.posix.only.hb.alarm` parameter to `1`, dead edge nodes are displayed for 4 minutes from the time they went down, as CLDB removes all the dead edge nodes after 4 minutes. However, when you set `cldb.ignore.posix.only.hb.alarm` parameter to `0`, dead edge nodes are displayed for 24 hours.
`output`	Specifies whether the output should be terse or verbose.

Parameter	Description
sortby	Specifies one of the following attributes by which to sort the list of nodes: nodeid, nodeip, nodehostname, noderackpath, nodeswitchpath, nodestatus, nodeservices, nodefshb, nodejthb, nodedisktotal, nodediskused, nodediskavail, noderpc, noderpcin, noderpcout, nodediskcount, nodediskreadops, nodediskreadkbytes, nodediskwriteops, nodediskwritekbytes, nodecpucount, nodecpuutil, nodememtotal, nodememused, nodefaileddisks, nodevirtualip, nodevirtualipend, nodenetmaskvnodemacaddress, nodegateway, nodebytesreceived, nodebytessent, nodecpuuptime, nodemaprdiskcount, nodestatusdesc, nodeblockmovesout, nodeblockmovesinvnodemaxcontainersthreshold, nodenuminstances, nodenumspsperinstance, nodenfsstate, nodeisposixclient, nodeisloopbacknfs, nodeisloopbacknfsrunning
start	The offset from the starting row according to sort. Default: 0
zkconnect	ZooKeeper Connect String

Output

Information about the nodes. See the fields for more information.

Sample Output

```
# /opt/mapr/bin/maprcli node list -json
{
    "timestamp":1555342212112,
    "timeofday":"2019-04-15 08:30:12.112 GMT-0700 AM",
    "status":"OK",
    "total":1,
    "data":[
        {
            "id":"7146221175287263104",
            "ip":[
                "10.10.82.29",
                "172.17.0.1"
            ],
            "hostname":"doc29.lab",
            "racktopo":"/data/default-rack/doc29.lab",
            "health":2,
            "healthDesc":"One or more services is down",

"service":"resourcemanager,fileserver,cldb,nfs4,mastgateway,nodemanager,gateway,hoststats,apiserver,posixclientbasic",

"configuredservice":"resourcemanager,filemigrate,fileserver,cldb,nfs4,mastgateway,nodemanager,gateway,hoststats,apiserver,posixclientbasic",
            "fs-heartbeat":0,
            "jt-heartbeat":2,
            "dtotal":272,
            "dused":0,
            "davail":272,
            "rpcs":0,
            "rpcin":345,
```


last-updated: Apr 11, 2022

```
            "rpcout":652,
            "disks":5,
            "MapRfs disks":3,
            "faileddisks":0,
            "dreads":0,
            "dreadK":0,
            "dwrites":1,
            "dwriteK":8,
            "cpus":8,
            "utilization":25,
            "uptime":"Mon Nov 20 15:03:37 PST 2017",
            "mtotal":23949,
            "mused":11996,
            "ttmapSlots":0,
            "ttmapUsed":0,
            "ttReduceSlots":0,
            "ttReduceUsed":0,
            "bytesReceived":168,
            "bytesSent":180,
            "numResyncSlots":16,
            "blockMovesOut":false,
            "blockMovesIn":false,
            "numInstances":"1",
            "numReportedInstances":"1",
            "spsPerInstance":"0",
            "numPutsInLastTenSeconds":0,
            "numPutsInLastMinute":0,
            "numPutsInLastFiveMinutes":0,
            "numPutsInLastFifteenMinutes":0,
            "numGetsInLastTenSeconds":0,
            "numGetsInLastMinute":0,
            "numGetsInLastFiveMinutes":0,
            "numGetsInLastFifteenMinutes":0,
            "numScansInLastTenSeconds":0,
            "numScansInLastMinute":0,
            "numScansInLastFiveMinutes":0,
            "numScansInLastFifteenMinutes":0,
            "LogLevelAlarm":0,
            "ServiceCLDBDownNotRunningAlarm":0,
            "ServiceFileserverDownNotRunningAlarm":0,
            "ServiceJTDownNotRunningAlarm":0,
            "ServiceTTDownNotRunningAlarm":0,
            "ServiceHBMasterDownNotRunningAlarm":0,
            "ServiceHBRegionDownNotRunningAlarm":0,
            "ServiceNFSDownNotRunningAlarm":0,
            "ServiceNFS4DownNotRunningAlarm":0,
            "ServiceWebserverDownNotRunningAlarm":0,
            "ServiceHoststatsDownNotRunningAlarm":0,
            "DiskFailureAlarm":0,
            "VersionMismatchAlarm":0,
            "TimeSkewAlarm":0,
            "HbProcessingSlow":1554758472188,
            "RootPartitionFullAlarm":0,
            "HomeMapRFullAlarm":0,
            "CorePresentAlarm":0,
            "HighMfsMemoryAlarm":0,
            "PamMisconfiguredAlarm":0,
            "TTLocaldirFullAlarm":0,
            "NodeNoHeartbeatAlarm":0,
            "NodeMaprUserMismatchAlarm":0,
            "NodeDuplicateHostIdAlarm":0,
            "NodeMetricsWriteProblemAlarm":0,
            "NodeTooManyContainersAlarm":0,
            "IncorrectTopologyAlarm":0,
```


last-updated: Apr 11, 2022

```
            "ServiceHueDownNotRunningAlarm":0,
            "ServiceHttpfsDownNotRunningAlarm":0,
            "ServiceBeeswaxDownNotRunningAlarm":0,
            "ServiceHiveDownNotRunningAlarm":0,
            "ServiceHs2DownNotRunningAlarm":0,
            "ServiceOozieDownNotRunningAlarm":0,
            "NodeManagerDown":0,
            "InstanceMismatch":0,
            "ResourceManagerDown":0,
            "Insufficient memory for buckets":0,
            "NoDiskAttached":0,
            "MemoryAllocationAlarm":0,
            "FileMigrateServerDown":1555340598576,
            "MemorySwapping":0,
            "ApiServerDown":0
        }
    ]
}
```

Fields

For definitions of the output fields, and short names for use with filters, see the fields table.

Examples

List all nodes:

For neatly formatted results, use the `-json` option when listing all nodes or a large subset of node information.

CLI

```
/opt/mapr/bin/maprcli node list -json
{
    "timestamp":1555342212112,
    "timeofday":"2019-04-15
08:30:12.112 GMT-0700 AM",
    "status":"OK",
    "total":1,
    "data":[
        {

"id":"7146221175287263104",
            "ip":[
                "10.10.82.29",
                "172.17.0.1"
            ],
            "hostname":"doc29.lab",
            "racktopo":"/data/
default-rack/doc29.lab",
            "health":2,
            "healthDesc":"One or more
services is down",

"service":"resourcemanager,fileserver,
cldb,nfs4,mastgateway,nodemanager,gate
way,hoststats,apiserver,posixclientbas
ic",

"configuredservice":"resourcemanager,f
ilemigrate,fileserver,cldb,nfs4,mastga
teway,nodemanager,gateway,hoststats,ap
iserver,posixclientbasic",
```


last-updated: Apr 11, 2022

```
            "fs-heartbeat":0,
            "jt-heartbeat":2,
            "dtotal":272,
            "dused":0,
            "davail":272,
            "rpcs":0,
            "rpcin":345,
            "rpcout":652,
            "disks":5,
            "MapRfs disks":3,
            "faileddisks":0,
            "dreads":0,
            "dreadK":0,
            "dwrites":1,
            "dwriteK":8,
            "cpus":8,
            "utilization":25,
            "uptime":"Mon Nov 20
15:03:37 PST 2017",
            "mtotal":23949,
            "mused":11996,
            "ttmapSlots":0,
            "ttmapUsed":0,
            "ttReduceSlots":0,
            "ttReduceUsed":0,
            "bytesReceived":168,
            "bytesSent":180,
            "numResyncSlots":16,
            "blockMovesOut":false,
            "blockMovesIn":false,
            "numInstances":"1",

"numReportedInstances":"1",
            "spsPerInstance":"0",

"numPutsInLastTenSeconds":0,
            "numPutsInLastMinute":0,

"numPutsInLastFiveMinutes":0,

"numPutsInLastFifteenMinutes":0,

"numGetsInLastTenSeconds":0,
            "numGetsInLastMinute":0,

"numGetsInLastFiveMinutes":0,

"numGetsInLastFifteenMinutes":0,

"numScansInLastTenSeconds":0,
            "numScansInLastMinute":0,

"numScansInLastFiveMinutes":0,

"numScansInLastFifteenMinutes":0,
            "LogLevelAlarm":0,

"ServiceCLDBDownNotRunningAlarm":0,

"ServiceFileserverDownNotRunningAlarm"
:0,
```

```
"ServiceJTDownNotRunningAlarm":0,

"ServiceTTDownNotRunningAlarm":0,

"ServiceHBMasterDownNotRunningAlarm":0
,

"ServiceHBRegionDownNotRunningAlarm":0
,

"ServiceNFSDownNotRunningAlarm":0,

"ServiceNFS4DownNotRunningAlarm":0,

"ServiceWebserverDownNotRunningAlarm":
0,

"ServiceHoststatsDownNotRunningAlarm":
0,
            "DiskFailureAlarm":0,
            "VersionMismatchAlarm":0,
            "TimeSkewAlarm":0,

"HbProcessingSlow":1554758472188,

"RootPartitionFullAlarm":0,
            "HomeMapRFullAlarm":0,
            "CorePresentAlarm":0,
            "HighMfsMemoryAlarm":0,
            "PamMisconfiguredAlarm":0,
            "TTLocaldirFullAlarm":0,
            "NodeNoHeartbeatAlarm":0,

"NodeMaprUserMismatchAlarm":0,

"NodeDuplicateHostIdAlarm":0,

"NodeMetricsWriteProblemAlarm":0,

"NodeTooManyContainersAlarm":0,

"IncorrectTopologyAlarm":0,

"ServiceHueDownNotRunningAlarm":0,

"ServiceHttpfsDownNotRunningAlarm":0,

"ServiceBeeswaxDownNotRunningAlarm":0,

"ServiceHiveDownNotRunningAlarm":0,

"ServiceHs2DownNotRunningAlarm":0,

"ServiceOozieDownNotRunningAlarm":0,
            "NodeManagerDown":0,
            "InstanceMismatch":0,
            "ResourceManagerDown":0,
            "Insufficient memory for
buckets":0,
            "NoDiskAttached":0,
            "MemoryAllocationAlarm":0,
```

```
"FileMigrateServerDown":1555340598576,
            "MemorySwapping":0,
            "ApiServerDown":0
        }
    ]
}
```

REST

```
curl -u mapr:mapr -X GET -k "https://
host:8443/rest/node/list"
```

List the health and configured service of all nodes:

The following examples show the use of short forms for the `column` parameter.

CLI

```
/opt/mapr/bin/maprcli node
list -columns
service,health,configuredservice -json
/opt/mapr/bin/maprcli node
list -columns svc,h,csvc -json
{
    "timestamp":1555343115082,
    "timeofday":"2019-04-15
08:45:15.082 GMT-0700 AM",
    "status":"OK",
    "total":1,
    "data":[
        {
            "ip":[
                "10.10.82.29",
                "172.17.0.1"
            ],
            "hostname":"doc29.lab",
            "health":2,

"service":"resourcemanager,fileserver,
cldb,nfs4,mastgateway,nodemanager,gate
way,hoststats,apiserver,posixclientbas
ic",

"configuredservice":"resourcemanager,f
ilemigrate,fileserver,cldb,nfs4,mastga
teway,nodemanager,gateway,hoststats,ap
iserver,posixclientbasic"
        }
    ]
}
```

REST

```
curl -u mapr:mapr -X GET -k "https://
host:8443/rest/node/list?
columns=service%2Chealth%2Cconfigureds
ervice"
curl -u mapr:mapr -X GET -k "https://
host:8443/rest/node/list?
columns=svc%2Ch%2Ccsvs"
{"timestamp":1555482645387,"timeofday"
:"2019-04-16 11:30:45.387 GMT-0700
PM","status":"OK","total":1,"data":
[{"ip":
```

```
["10.10.82.29","172.17.0.1"],"hostname
":"doc29.lab","health":2,"service":"",
"configuredservice":""}]}
```

List the number of slots on all nodes:

CLI

```
/opt/mapr/bin/maprcli node
list -columns
ip,ttmapSlots,ttmapUsed,ttReduceSlots,
ttReduceUsed -json
{
    "timestamp":1555483525095,
    "timeofday":"2019-04-16
11:45:25.095 GMT-0700 PM",
    "status":"OK",
    "total":1,
    "data":[
        {
            "ip":[
                "10.10.82.29",
                "172.17.0.1"
            ],
            "hostname":"doc29.lab",
            "ttmapSlots":0,
            "ttmapUsed":0,
            "ttReduceSlots":0,
            "ttReduceUsed":0
        }
    ]
}
```

REST

```
curl -u mapr:mapr -X GET -k "https://
host:8443/rest/node/list?
columns=ip%2CttmapSlots%2CttmapUsed%2C
ttReduceSlots%2CttReduceUsed"
{"timestamp":1555483675606,"timeofday"
:"2019-04-16 11:47:55.606 GMT-0700
PM","status":"OK","total":1,"data":
[{"ip":
["10.10.82.29","172.17.0.1"],"hostname
":"doc29.lab","ttmapSlots":0,"ttmapUse
d":0,"ttReduceSlots":0,"ttReduceUsed":
0}]}
```

List nodes on a particular subnet:

CLI

```
/opt/mapr/bin/maprcli node
list -filter '[ip==10.*]' -json
{
    "timestamp":1555483749837,
    "timeofday":"2019-04-16
11:49:09.837 GMT-0700 PM",
    "status":"OK",
    "total":1,
    "data":[
        {

"id":"1470287842321938805",
```

```
            "ip":[
                "10.10.82.29",
                "172.17.0.1"
            ],
            "hostname":"doc29.lab",
            "racktopo":"/data/
default-rack/doc29.lab",
            "health":2,
            "healthDesc":"One or more
services is down",

"service":"resourcemanager,fileserver,
cldb,nfs4,mastgateway,nodemanager,host
stats,gateway,apiserver",

"configuredservice":"resourcemanager,f
ileserver,cldb,nfs4,mastgateway,nodema
nager,hoststats,gateway,apiserver",
            "fs-heartbeat":0,
            "jt-heartbeat":2,
            "dtotal":272,
            "dused":0,
            "davail":272,
            "rpcs":0,
            "rpcin":489,
            "rpcout":940,
            "disks":5,
            "MapRfs disks":3,
            "faileddisks":0,
            "dreads":0,
            "dreadK":0,
            "dwrites":0,
            "dwriteK":0,
            "cpus":8,
            "utilization":27,
            "uptime":"Tue Apr 16
04:00:39 PDT 2019",
            "mtotal":23947,
            "mused":10646,
            "ttmapSlots":0,
            "ttmapUsed":0,
            "ttReduceSlots":0,
            "ttReduceUsed":0,
            "bytesReceived":0,
            "bytesSent":0,
            "numResyncSlots":16,
            "blockMovesOut":false,
            "blockMovesIn":false,
            "numInstances":"1",

"numReportedInstances":"1",
            "spsPerInstance":"0",

"numPutsInLastTenSeconds":0,
            "numPutsInLastMinute":0,

"numPutsInLastFiveMinutes":0,

"numPutsInLastFifteenMinutes":0,

"numGetsInLastTenSeconds":0,
            "numGetsInLastMinute":0,
```

```
"numGetsInLastFiveMinutes":0,

"numGetsInLastFifteenMinutes":0,

"numScansInLastTenSeconds":0,
            "numScansInLastMinute":0,

"numScansInLastFiveMinutes":0,

"numScansInLastFifteenMinutes":0,
            "LogLevelAlarm":0,

"ServiceCLDBDownNotRunningAlarm":0,

"ServiceFileserverDownNotRunningAlarm"
:0,

"ServiceJTDownNotRunningAlarm":0,

"ServiceTTDownNotRunningAlarm":0,

"ServiceHBMasterDownNotRunningAlarm":0
,

"ServiceHBRegionDownNotRunningAlarm":0
,

"ServiceNFSDownNotRunningAlarm":0,

"ServiceNFS4DownNotRunningAlarm":0,

"ServiceWebserverDownNotRunningAlarm":
0,

"ServiceHoststatsDownNotRunningAlarm":
0,
            "DiskFailureAlarm":0,
            "VersionMismatchAlarm":0,
            "TimeSkewAlarm":0,
            "HbProcessingSlow":0,

"RootPartitionFullAlarm":0,
            "HomeMapRFullAlarm":0,
            "CorePresentAlarm":0,
            "HighMfsMemoryAlarm":0,
            "PamMisconfiguredAlarm":0,
            "TTLocaldirFullAlarm":0,
            "NodeNoHeartbeatAlarm":0,

"NodeMaprUserMismatchAlarm":0,

"NodeDuplicateHostIdAlarm":0,

"NodeMetricsWriteProblemAlarm":0,

"NodeTooManyContainersAlarm":0,

"IncorrectTopologyAlarm":0,

"ServiceHueDownNotRunningAlarm":0,
```

```
"ServiceHttpfsDownNotRunningAlarm":0,

"ServiceBeeswaxDownNotRunningAlarm":0,

"ServiceHiveDownNotRunningAlarm":0,

"ServiceHs2DownNotRunningAlarm":0,

"ServiceOozieDownNotRunningAlarm":0,
            "NodeManagerDown":0,
            "GatewayServiceDown":0,
            "InstanceMismatch":0,
            "ResourceManagerDown":0,
            "Insufficient memory for
buckets":0,
            "NoDiskAttached":0,
            "MemoryAllocationAlarm":0,

"FileMigrateServerDown":1555482296140,
            "MemorySwapping":0,
            "ApiServerDown":0
        }
    ]
}
```

REST

```
curl -u mapr:mapr -X GET -k "https://
host:8443/rest/node/list?
filter=%5Bip%3D%3D10.*%5D"
{"timestamp":1555483809698,"timeofday"
:"2019-04-16 11:50:09.698 GMT-0700
PM","status":"OK","total":1,"data":
[{"id":"1470287842321938805","ip":
["10.10.82.29","172.17.0.1"],"hostname
":"doc29.lab","racktopo":"/data/
default-rack/
doc29.lab","health":2,"healthDesc":"On
e or more services is
down","service":"","configuredservice"
:"","fs-heartbeat":0,"jt-heartbeat":2,
"dtotal":272,"dused":0,"davail":272,"r
pcs":0,"rpcin":489,"rpcout":942,"disks
":5,"MapRfs
disks":3,"faileddisks":0,"dreads":0,"d
readK":0,"dwrites":0,"dwriteK":0,"cpus
":8,"utilization":1,"uptime":"Tue Apr
16 04:00:39 PDT
2019","mtotal":23947,"mused":10560,"tt
mapSlots":0,"ttmapUsed":0,"ttReduceSlo
ts":0,"ttReduceUsed":0,"bytesReceived"
:672,"bytesSent":792,"numResyncSlots":
16,"blockMovesOut":false,"blockMovesIn
":false,"numInstances":"1","numReporte
dInstances":"1","spsPerInstance":"0","
numPutsInLastTenSeconds":0,"numPutsInL
astMinute":0,"numPutsInLastFiveMinutes
":0,"numPutsInLastFifteenMinutes":0,"n
umGetsInLastTenSeconds":0,"numGetsInLa
stMinute":0,"numGetsInLastFiveMinutes"
:0,"numGetsInLastFifteenMinutes":0,"nu
mScansInLastTenSeconds":0,"numScansInL
astMinute":0,"numScansInLastFiveMinute
```

```
s":0,"numScansInLastFifteenMinutes":0,
"LogLevelAlarm":0,"ServiceCLDBDownNotR
unningAlarm":0,"ServiceFileserverDownN
otRunningAlarm":0,"ServiceJTDownNotRun
ningAlarm":0,"ServiceTTDownNotRunningA
larm":0,"ServiceHBMasterDownNotRunning
Alarm":0,"ServiceHBRegionDownNotRunnin
gAlarm":0,"ServiceNFSDownNotRunningAla
rm":0,"ServiceNFS4DownNotRunningAlarm"
:0,"ServiceWebserverDownNotRunningAlar
m":0,"ServiceHoststatsDownNotRunningAl
arm":0,"DiskFailureAlarm":0,"VersionMi
smatchAlarm":0,"TimeSkewAlarm":0,"HbPr
ocessingSlow":0,"RootPartitionFullAlar
m":0,"HomeMapRFullAlarm":0,"CorePresen
tAlarm":0,"HighMfsMemoryAlarm":0,"PamM
isconfiguredAlarm":0,"TTLocaldirFullAl
arm":0,"NodeNoHeartbeatAlarm":0,"NodeM
aprUserMismatchAlarm":0,"NodeDuplicate
HostIdAlarm":0,"NodeMetricsWriteProble
mAlarm":0,"NodeTooManyContainersAlarm"
:0,"IncorrectTopologyAlarm":0,"Service
HueDownNotRunningAlarm":0,"ServiceHttp
fsDownNotRunningAlarm":0,"ServiceBeesw
axDownNotRunningAlarm":0,"ServiceHiveD
ownNotRunningAlarm":0,"ServiceHs2DownN
otRunningAlarm":0,"ServiceOozieDownNot
RunningAlarm":0,"NodeManagerDown":0,"G
atewayServiceDown":0,"InstanceMismatch
":0,"ResourceManagerDown":0,"Insuffici
ent memory for
buckets":0,"NoDiskAttached":0,"MemoryA
llocationAlarm":0,"FileMigrateServerDo
wn":1555482296140,"MemorySwapping":0,"
ApiServerDown":0}]}
```


List the nodes running the clients:

CLI

```
/opt/mapr/bin/maprcli node
list -clientsonly true -json
clienttype
clienthealth  hostname
ip                       lasthb  id
posixclientgold
Active        atsqa4-119.qa.lab
10.10.88.119,172.17.0.1  28
5412384279424088014
NFS_V3
Active        qa108-181.qa.lab
10.10.108.181            1
711699521447755347
posixclientbasic
Active        qa108-182.qa.lab
10.10.108.182            5
5689202715616988402
posixclientplatinum
Active        qa108-183.qa.lab
10.10.108.183            15
5679519305469912939
LOOPBACK_NFS
Active        qa108-184.qa.lab
```

```
10.10.108.184            1
7236866912027931 55
NFS_V4
Active        qa108-185.qa.lab
10.10.108.185            1
7808496860582738296
posixclientbasic
Active        qa108-186.qa.lab
10.10.108.186            25
2792316733179447508
posixclientplatinum
Active        qa108-187.qa.lab
10.10.108.187            11
5678398615695393161
LOOPBACK_NFS
Active        qa108-188.qa.lab
10.10.108.188            1
5524477677754836725
NFS_V3
Active        qa108-189.qa.lab
10.10.108.189            1
3396225116726542411
NFS_V4
Active        qa108-190.qa.lab
10.10.108.190            2
1203052391917747224
```

REST

```
curl -u mapr:mapr -X GET -k "https://
host:8443/rest/node/list?
clientsonly=true"
{"timestamp":1531171868890,"timeofday"
:"2018-07-09 02:31:08.890 GMT-0700
PM","status":"OK","total":1,"data":
[{"id":"5412384279424088014","hostname
":"atsqa4-119.qa.lab","ip":"10.10.88.1
19,172.17.0.1","clienttype":"posixclie
ntgold","clienthealth":"Active","lasth
b":28}]}
```

Related concepts

node on page 1694
Manages nodes in the cluster

Related reference

disk add on page 1602
Adds one or more disks to the specified node. Permissions required: `fc` or `a`.

volume create on page 1931
Creates a volume.

dump volumeinfo on page 1637
Returns information about volumes and the associated containers. For JSON formatted output, use the -json option from the command line.

configure.sh on page 2053
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

node listcldbs

Returns the hostnames of the nodes in the cluster that are running the CLDB service.

Syntax

CLI

```
maprcli node listcldbs
    [ -cluster <cluster name> ]
    [ -cldb  <cldb hostname|ip:port> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/node/listcldbs[?<parameters>]`

Parameters

Parameter	Description
`cluster`	The name of the cluster for which to return the list of CLDB node hostnames.
`cldb`	The hostname or IP address and port number of a CLDB node.

Examples

Return the list of CLDB nodes for the cluster my.cluster.com:

CLI

```
# maprcli node listcldbs -cluster
my.cluster.com -json
{
    "timestamp":1529445021408,
    "timeofday":"2018-06-19
02:50:21.408 GMT-0700 PM",
    "status":"OK",
    "total":1,
    "data":[
        {

"CLDBs":"in111-22.qa.lab,in111-24.qa.l
ab,in111-21.qa.lab"
        }
    ]
}
```

REST

```
# curl -k -X GET 'https://
abc.sj.us:8443/rest/node/listcldbs?
cluster=my.cluster.com' --user
mapr:mapr
{"timestamp":1529445190525,"timeofday"
:"2018-06-19 02:53:10.525 GMT-0700
PM","status":"OK","total":1,"data":
[{"CLDBs":"in111-22.qa.lab,in111-24.qa
.lab,in111-21.qa.lab"}]}
```


last-updated: Apr 11, 2022

node listcldbzks
Returns the hostnames of the nodes in the cluster that are running the CLDB service and the IP addresses and port numbers for the nodes in the cluster that are running the ZooKeeper service.

Syntax

CLI

```
maprcli node listcldbzks
    [ -cluster <cluster name> ]
    [ -cldb  <cldb hostname|ip:port> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/node/listcldbzks[?<parameters>]`

Parameters

Parameter	Description
`cluster`	The name of the cluster for which to return the CLDB and ZooKeeper information.
`cldb`	The hostname or IP address and port number of a CLDB node.

Examples

Return CLDB and ZooKeeper node information for the cluster my.cluster.com:

CLI

```
maprcli node listcldbzks -cluster
my.cluster.com
{
    "timestamp":1529445399193,
    "timeofday":"2018-06-19
02:56:39.193 GMT-0700 PM",
    "status":"OK",
    "total":1,
    "data":[
        {

"CLDBs":"in111-22.qa.lab,in111-24.qa.l
ab,in111-21.qa.lab",

"Zookeepers":"in111-21.qa.lab:5181,in1
11-22.qa.lab:5181,in111-24.qa.lab:5181
"
        }
    ]
}
```

REST

```
# curl -k -X GET 'https://
abc.sj.us:8443/rest/node/listcldbzks?
cluster=my.cluster.com' --user
```

```
mapr:mapr
{"timestamp":1529445324540,"timeofday"
:"2018-06-19 02:55:24.540 GMT-0700
PM","status":"OK","total":1,"data":
[{"CLDBs":"in111-22.qa.lab,in111-24.qa
.lab,in111-21.qa.lab","Zookeepers":"in
111-21.qa.lab:5181,in111-22.qa.lab:518
1,in111-24.qa.lab:5181"}]}
```

node listzookeepers
Returns the hostnames of the nodes in the cluster that are running the ZooKeeper service.

Syntax

CLI

```
maprcli node listzookeepers
    [ -cluster <cluster name> ]
    [ -cldb  <cldb hostname|ip:port> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/node/listzookeepers[?<parameters>]`

Parameters

Parameter	Description
`cluster`	The name of the cluster for which to return the list of zookeeper node hostnames.
`cldb`	The hostname or IP address and port number of a valid CLDB node. The other CLDB nodes and zookeeper nodes can be discovered from this node.

Examples

Return the list of zookeeper nodes for the cluster my.cluster.com

If you know that the CLDB service is running on a node with hostname `host1`, you can enter:

CLI

```
maprcli node listzookeepers -cluster
my.cluster.com -cldb host1 -json
{
    "timestamp":1529451245796,
    "timeofday":"2018-06-19
04:34:05.796 GMT-0700 PM",
    "status":"OK",
    "total":1,
    "data":[
        {

"Zookeepers":"in111-21.qa.lab:5181,in1
```

```
11-22.qa.lab:5181,in111-24.qa.lab:5181
"
        }
    ]
}
```

REST

```
https://abc.sj.us:8443/rest/node/
listzookeepers?
cluster=my.cluster.com&cldb=host1
{"timestamp":1529451245796,"timeofday"
:"2018-06-19 04:34:05.796 GMT-0700
PM","status":"OK","total":1,"data":
[{"Zookeepers":"in111-21.qa.lab:5181,i
n111-22.qa.lab:5181,in111-24.qa.lab:51
81"}]}
```

node maintenance

Places a node into a maintenance mode for a specified duration.

Important: Stop CLDB if it is running on the node, before putting that node in maintenance mode. Else, the maintenance mode operation is not permitted. Run: `maprcli node services -name cldb -action stop -nodes mapr-<node>`

Note: You cannot put a master CLDB node in Maintenance mode.

For the duration of the maintenance mode, the cluster's CLDB does not consider the data of this node as lost and does not trigger a resync of the data on this node. See Administering Nodes on page 797 for more information.

Syntax

CLI

```
/opt/mapr/bin/maprcli node maintenance
    [ -cluster <cluster> ]
    [ -serverids <serverids> ]
    -nodes <node names>
    -timeoutminutes <timeout in
minutes>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/node/maintenance?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`serverids`	List of server IDs
`nodes`	List of nodes, space separated.

last-updated: Apr 11, 2022

Parameter	Description
`timeoutminutes`	Duration of the maintenance mode in minutes

Examples

CLI

```
/opt/mapr/bin/maprcli
node maintenance -nodes
centos22.lab -timeoutminutes 20
```

REST

```
curl -u mapr:mapr -X POST -k 'https://
abc.sj.us:8443/rest/node/maintenance?
nodes=centos22.lab&timeoutminutes=20'
```

node metrics

Retrieves metrics information for nodes in a cluster.

Warning: This command is deprecated. See Using MapR Data Platform Monitoring (Spyglass Initiative) on page 1330 for information about viewing metrics and logs for nodes, services, and applications.

This command retrieves and displays various metrics related to the operation of nodes. The data displayed comes from the files that each node updates periodically that are stored in the node local volume on each node in the cluster.

Syntax

CLI

```
maprcli node metrics
         -nodes <hostname>
         -start <start time>
         -end <end time>
        [ -interval <interval
timestamp> ]
        [ -events true|false ]
        [ -columns <column names> ]
        [ -cluster <cluster name> ]
```

Parameters

Parameter	Description
nodes	A space-separated list of host names. The host name must be either the specific hostname (use the `maprcli node list -columns hostname` command to obtain the hostname value) or the name "hostname" if using the command line on the actual node. The IP address and "localhost" can not be used.
start	The start of the time range. Can be a UTC timestamp (in this case, a Java millisecond timestamp) or a UTC date in MM/DD/YY or MM/DD/YYYY format.
end	The end of the time range. Can be a UTC timestamp (in this case, a Java millisecond timestamp) or a UTC date in MM/DD/YY or MM/DD/YYYY format.

last-updated: Apr 11, 2022

Parameter	Description
interval	Data measurement interval in seconds. The minimum value is 10 seconds.
events	Specify TRUE to return node events only. The default value of this parameter is FALSE.
columns	Comma-separated list of column names to return.
cluster	Cluster name.

Column Name Parameters

The `node metrics` API always returns the `NODE` (node name) and `TIMESTAMP` (integer timestamp) columns. Use the `-columns` flag to specify a comma-separated list of column names to return.

Warning: The `CPUNICE`, `CPUUSER`, and `CPUSYSTEM` parameters return information in *jiffies*. This unit measures one tick of the system timer interrupt and is usually equivalent to 10 milliseconds, but may vary depending on your particular node configuration. The reporting interval is the maximum possible value. In addition, the CPU* parameters accumulate and do not reset from report to report. Call `sysconf(_SC_CLK_TCK)` to determine the exact value for your node.

`CPUNICE`	Amount of CPU time used by processes with a positive nice value.	
`CPUUSER`	Amount of CPU time used by user processes.	
`CPUSYSTEM`	Amount of CPU time used by system processes.	
`LOAD5PERCENT`	Percentage of time this node spent at load 5 or below	
`LOAD1PERCENT`	Percentage of time this node spent at load 1 or below	
`MEMORYCACHED`	Memory cache size in bytes	
`MEMORYSHARED`	Shared memory size in bytes	
`MEMORYBUFFERS`	Memory buffer size in bytes	
`MEMORYUSED`	Memory used in megabytes	
`PROCRUN`	Number of processes running	
`RPCCOUNT`	Number of MapR RPC calls	
`RPCINBYTES`	Number of bytes passed in by MapR RPC calls	
`RPCOUTBYTES`	Number of bytes passed out by MapR RPC calls	
`SERVAVAILSIZEMB`	Server storage available in megabytes	

last-updated: Apr 11, 2022

SERVUSEDSIZEMB	Server storage used in megabytes	
SWAPFREE	Free swap space in bytes	
TTMAPUSED	Number of TaskTracker slots used for map tasks	
TTREDUCEUSED	Number of TaskTracker slots used for reduce tasks	

Three column name parameters return data that is too granular to display in a standard table. Use the `-json` option to return this information as a JSON object.

Parameter	Description	Metrics Returned
CPUS	Activity on this node's CPUs. Each CPU on the node is numbered from zero, `cpu0` to `cpuN`. Metrics returned are for each CPU.	`CPUIDLE`: Amount of CPU time spent idle. Reported as *jiffies*. `CPUIOWAIT`: Amount of CPU time spent waiting for I/O operations. Reported as *jiffies*. `CPUTOTAL`: Total amount of CPU time. Reported as *jiffies*.
DISKS	Activity on this node's disks. Metrics returned are for each partition.	`READOPS`: Number of read operations. `READKB`: Number of kilobytes read. `WRITEOPS`: Number of write operations. `WRITEKB`: Number of kilobytes written.
NETWORK	Activity on this node's network interfaces. Metrics returned are for each interface.	`BYTESIN`: Number of bytes received. `BYTESOUT`: Number of bytes sent. `PKTSIN`: Number of packets received. `PKTSOUT`: Number of packets sent.

Examples

Retrieving the percentage of time that a node spent at the 1 and 5 load levels between dates

```
$ maprcli node metrics
        -nodes centos24.lab
        -start 08/02/15
        -end 08/03/15
        -interval 7200
        -columns LOAD5PERCENT,LOAD1PERCENT
```

Sample Output

```
NODE          LOAD5PERCENT  LOAD1PERCENT  TIMESTAMPSTR
TIMESTAMP
centos24.lab  15            9             Sat Aug 01 17:00:08 PDT 2015
1438473608000
centos24.lab  20            20            Sat Aug 01 19:00:13 PDT 2015
1438480813000
centos24.lab  14            9             Sat Aug 01 21:00:18 PDT 2015
1438488018000
centos24.lab  13            11            Sat Aug 01 23:00:24 PDT 2015
1438495224000
centos24.lab  11            1             Sun Aug 02 01:00:29 PDT 2015
1438502429000
centos24.lab  14            8             Sun Aug 02 03:00:34 PDT 2015
```


last-updated: Apr 11, 2022

```
1438509634000
centos24.lab  13            22            Sun Aug 02 05:00:39 PDT 2015
1438516839000
centos24.lab  24            46            Sun Aug 02 07:00:44 PDT 2015
1438524044000
centos24.lab  18            21            Sun Aug 02 09:00:49 PDT 2015
1438531249000
centos24.lab  10            2             Sun Aug 02 11:00:54 PDT 2015
1438538454000
centos24.lab  24            24            Sun Aug 02 13:00:59 PDT 2015
1438545659000
centos24.lab  8             0             Sun Aug 02 15:01:04 PDT 2015
1438552864000
centos24.lab  14            10            Sun Aug 02 17:01:09 PDT 2015
1438560069000
centos24.lab  10            2             Sun Aug 02 19:01:14 PDT 2015
1438567274000
centos24.lab  17            21            Sun Aug 02 21:01:19 PDT 2015
1438574479000
centos24.lab  15            8             Sun Aug 02 23:01:24 PDT 2015
1438581684000
centos24.lab  28            66            Mon Aug 03 01:01:29 PDT 2015
1438588889000
centos24.lab  16            28            Mon Aug 03 03:01:34 PDT 2015
1438596094000
centos24.lab  20            26            Mon Aug 03 05:01:40 PDT 2015
1438603300000
centos24.lab  22            39            Mon Aug 03 07:01:45 PDT 2015
1438610505000
centos24.lab  16            18            Mon Aug 03 09:01:50 PDT 2015
1438617710000
centos24.lab  16            17            Mon Aug 03 11:01:55 PDT 2015
1438624915000
centos24.lab  18            35            Mon Aug 03 13:02:00 PDT 2015
1438632120000
centos24.lab  11            10            Mon Aug 03 15:02:05 PDT 2015
1438639325000
```

Retrieving time percentage at load 1 and 5 levels and CPU usage between timestamps

```
$ maprcli node metrics
        -nodes centos24.lab
        -start 1438502429000
        -end 1438581684000
        -interval 28800
        -columns LOAD5PERCENT,LOAD1PERCENT,CPUS
        -json
```

Sample JSON output

```
{
    "timestamp":1438819022412,
    "timeofday":"2015-08-05 04:57:02.412 GMT-0700",
    "status":"OK",
    "total":3,
    "data":[
        {
            "NODE":"centos24.lab",
            "TIMESTAMPSTR":"Sat Aug 01 18:00:01 PDT 2015",
            "TIMESTAMP":1438477201000,
            "CPUS":{
                "cpu0":{
                    "CPUIDLE":491625764,
```


last-updated: Apr 11, 2022

```
                    "CPUIOWAIT":48455544,
                    "CPUTOTAL":571787058
                }
            },
            "LOAD1PERCENT":8,
            "LOAD5PERCENT":18
        },
        {
            "NODE":"centos24.lab",
            "TIMESTAMPSTR":"Sun Aug 02 02:00:01 PDT 2015",
            "TIMESTAMP":1438506001000,
            "CPUS":{
                "cpu0":{
                    "CPUIDLE":494046587,
                    "CPUIOWAIT":48483715,
                    "CPUTOTAL":574608277
                }
            },
            "LOAD1PERCENT":26,
            "LOAD5PERCENT":23
        },
        {
            "NODE":"centos24.lab",
            "TIMESTAMPSTR":"Sun Aug 02 10:00:01 PDT 2015",
            "TIMESTAMP":1438534801000,
            "CPUS":{
                "cpu0":{
                    "CPUIDLE":496468384,
                    "CPUIOWAIT":48512056,
                    "CPUTOTAL":577430149
                }
            },
            "LOAD1PERCENT":6,
            "LOAD5PERCENT":11
        }
    ]
}
```


Retrieving data at the 1 and 5 load levels to the last even hour

```
maprcli node metrics
     -nodes $(ls /mapr/se1/var/mapr/local/)
     -start $(date -u -d '2 minutes ago' +%s000)
     -end $(date -u -d 'now' +%s000)
     -interval 60
     -columns CPUNICE,CPUUSER,CPUSYSTEM,LOAD1PERCENT,LOAD5PERCENT
     true
    -json
```

Sample Output


```
{
    "timestamp":1436395101882,
    "timeofday":"2015-07-08 10:38:21.882 GMT+0000",
    "status":"OK",
    "total":150,
    "data":[
        {
            "NODE":"se-node10.se.lab",
            "TIMESTAMPSTR":"Wed Jul 08 22:00:09 UTC 2015",
            "TIMESTAMP":1436392809000
        },
        {
```

```
            "NODE":"se-node10.se.lab",
            "TIMESTAMPSTR":"Wed Jul 08 22:01:10 UTC 2015",
            "TIMESTAMP":1436392870000
        },
        {
            "NODE":"se-node10.se.lab",
            "TIMESTAMPSTR":"Wed Jul 08 22:02:13 UTC 2015",
            "TIMESTAMP":1436392933000
        },
...
        {
            "NODE":"se-node13.se.lab",
            "TIMESTAMPSTR":"Wed Jul 08 22:38:10 UTC 2015",
            "TIMESTAMP":1436395090000
        }
    ]
}
```

node move
Moves one or more nodes to a different topology. Permissions required: `fc` or `a`.

Syntax

CLI

```
maprcli node move
    [ -cluster <cluster> ]
     -serverids <server IDs>
     -topology <topology>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/node/move?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`serverids`	The comma-separated list of server IDs of the nodes to move. If you insert spaces between server IDs, the command only operates on the first server ID in the list.
`topology`	The new topology.

To obtain existing topology, run

```
maprcli node topo
```

To obtain the server ID, run

```
maprcli node list -columns id
```

Sample output from `maprcli node list -columns id` is shown below. The resulting server ID(s) can be copied and pasted into the `maprcli node move` command.

```
id                   hostname     ip
547819249997313015   node-34.lab  10.10.40.34,10.10.88.34
2130988050310536949  node-36.lab  10.10.40.36,10.10.88.36
8683110801227243688  node-37.lab  10.10.40.37,10.10.88.37
5056865595028557458  node-38.lab  10.10.40.38,10.10.88.38
3111141192171195352  node-39.lab  10.10.40.39,10.10.88.39
```

Examples

CLI

```
maprcli node move
   -topology /newData
   -serverids 547819249997313015
```

REST

```
https://abc.sj.us:8443/rest/node/move?
topology=%2FnewData&serverids=54781924
9997313015
```

node remove

Removes one or more server nodes from the system. Permissions required: `fc` or `a`.

After issuing the `node remove` command, wait several minutes to ensure that the node has been properly and completely removed.

Syntax

CLI

```
maprcli node remove
    [ -filter "<filter>" ]
    [ -hostids <host IDs> ]
    [ -nodes <node names> ]
    [ -service <fileserver or
nfsserver> ]
    [ -zkconnect <ZooKeeper Connect
String> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/node/remove[?<parameters>]`

Parameters

Parameter	Description
`filter`	A filter specifying nodes on which to start or stop services. See Filters for more information.
`hostids`	A list of host IDs, separated by spaces.

last-updated: Apr 11, 2022

Parameter	Description
`nodes`	A list of node names, separated by spaces.
`service`	Service to be removed. Either fileserver or nfsserver.
`zkconnect`	ZooKeeper Connect String. Example: 'host:port,host:port,host:port,...'. To obtain zookeeper connection strings, use the `maprcli node listzookeepers` command. Default: localhost:5181

Examples

CLI

```
maprcli node remove -nodes 10.20.30.40
```

REST

```
https://abc.sj.us:8443/rest/node/
remove?nodes=10.20.30.40
```

node services

Starts, stops, or restarts services on one or more server nodes. Permissions required: `ss`, `fc` or `a`.

To start or stop services, you must specify the service name, the action (start, stop, or restart), and the nodes on which to perform the action. You can specify the nodes in one of two ways:

- Use the `nodes` parameter to specify a space-delimited list of node names.
- Use the `filter` parameter to specify all nodes that match a certain pattern. See Filters for more information.

Syntax

CLI

```
/opt/mapr/bin/maprcli node services
    [ -action start|stop|restart|
enable|disable ]
    [ -apiserver start|stop|restart|
enable|disable ]
    [ -cldb start|stop|restart|enable|
disable ]
    [ -cluster <cluster> ]
    [ -fileserver start|stop|restart|
enable|disable ]
    [ -filter <filter> ]

    [ -name <service> ]
    [ -nfs start|stop|restart|enable|
disable ]
    [ -nfs4 start|stop|restart|enable|
disable ]
    [ -nodes <node names> ]

    [ -zkconnect <ZooKeeper Connect
String> ]
```

REST

Request Type	POST

last-updated: Apr 11, 2022

Request URL	`http[s]://<host>:<port>/rest/node/services[?<parameters>]`

Parameters

To perform an action on a service, on a particular set of nodes, you must specify the following three parameters:

- `action`- the action to perform:
  - Start, stop, restart a service.
  - Disable a service to prevent it from starting when Warden restarts and enable a service to allow the service to start when Warden restarts.

  Note: Suspend and resume are not supported.
- `node` or `filter` - the nodes on which to perform the action; either a list of nodes, or a filter that matches a set of nodes
- `name` - the service on which to perform the action

The following table lists the parameters available with the `node services` command.

Parameter	Description
`action`	An action to perform on a service specified in the `name` parameter: Values: start, stop, suspend, resume, restart, enable, or disable
`apiserver`	Starts, stops, or restarts the apiserver. Values: start, stop, restart, enable, or disable
`cldb`	Starts, stops, or restarts the cldb service. Values: start, stop, suspend, resume, restart, enable, or disable
`cluster`	The cluster on which to run the command.
`fileserver`	Starts, stops, or restarts the fileserver service. Values: start, stop, suspend, resume, restart, enable, or disable
`filter`	A filter specifying nodes on which to start or stop services. For fields to use with the filter, see the node on page 1694 table. See Filters on page 1526 for more information about filters. **Note:** You must specify either the `filter` parameter or the `nodes` parameter.
`name`	A service on which to perform an action specified by the `action` parameter. Any service can be specified with this option, but the following services can be specified only with the `name` option: `collectd`, `elasticsearch`, `fluentd`, `grafana`, `historyserver`, `hivemeta`, `hoststats`, `hs2`, `httpfs`, `hue`, `kibana`, `nodemanager`, `opentsdb`, `oozie`, and `resourcemanager`.

last-updated: Apr 11, 2022

Parameter	Description
`nfs`	Starts, stops, or restarts the nfs service. Values: start, stop, suspend, resume, restart, enable, or disable
`nfs4`	Starts, stops, or restarts the NFSv4 service. Values: start, stop, restart, enable, or disable
`nodes`	A list of node names, separated by spaces.  **Note:** Either this or `filter` is required.
`zkconnect`	The ZooKeeper Connect String.

Examples

Start the NodeManager Service

```
/opt/mapr/bin/maprcli node services -name nodemanager -nodes
abc.sj.us -action start
```

Stop the ResourceManager Service

```
/opt/mapr/bin/maprcli node services -name resourcemanager -nodes
abc.sj.us -action stop
```

Stop the ResourceManager Service

```
/opt/mapr/bin/maprcli node services -name resourcemanager -nodes
abc.sj.us -action restart
```

Restart NFS server

```
/opt/mapr/bin/maprcli node services -nodes abc.sj.us -nfs restart
```

Restart NFS server using a filter

Using a filter is common, especially in HBase environments, where full restarts of region and master servers are needed.

```
/opt/mapr/bin/maprcli node services -filter ["csvc==nfs"] -nfs restart
```

Start the Hue Service

```
/opt/mapr/bin/maprcli node services -name hue -action start -nodes <node n>
```

Restart the hoststats service

Restart the hoststats service after making changes to the MapRMapR Data Platform Metrics database. You do not need to restart warden.

```
/opt/mapr/bin/maprcli node services -name hoststats -action restart -nodes
<nodes>

OR

/opt/mapr/bin/maprcli node services -name hoststats -action restart -filter
'[csvc==hoststats]'
```

node topo
Lists cluster topology information.

Lists internal nodes only (switches/racks/etc) and not leaf nodes (server nodes).

Syntax

CLI

```
maprcli node topo
    [ -cluster <cluster> ]
    [ -path <path> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/node/topo?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`path`	The path on which to list node topology.

Output

Node topology information.

Sample output

```
{
    "timestamp":1433545849048,
    "timeofday":"2015-06-05 11:10:49.048 GMT+0000",
    "status":"OK",
    "total":4,
    "data":[
        {
            "path":"/"
        },
        {
            "path":"/data"
        },
        {
            "path":"/data/default-rack"
        },
        {
            "path":"/default-rack"
        }
    ]
}
```

Output Fields

Field	Description
path	The physical topology path to the node.

last-updated: Apr 11, 2022

Examples

CLI

```
# maprcli node topo
path
/
/data
/data/default-rack
/default-rack
```

REST

```
# curl -k -X GET 'https://
abc.sj.us:8443/rest/node/topo' --user
mapr:mapr
{"timestamp":1529382835319,"timeofday"
:"2018-06-18 09:33:55.319 GMT-0700
PM","status":"OK","total":4,"data":
[{"path":"/"},{"path":"/data"},
{"path":"/data/default-rack"},
{"path":"/default-rack"}]}
```

node toposize
Lists disk space utilization for each topology.

Syntax

CLI

```
maprcli node toposize
    [ -cluster <cluster> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/node/toposize[?<parameters>]`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.

Output

Field	Description
`AvailableSpace`	The amount of available disk space (in GB).
`path`	The cluster rack path to the node (or topology of the node).
`TotalSpace`	The amount of total disk space (in GB).
`UsedSpace`	The amount of utilized disk space (in GB).

Example

Retrieve topology-based disk utilization information:

last-updated: Apr 11, 2022

CLI

```
# maprcli node toposize
path
TotalSpace(GB)  UsedSpace(GB)
AvailableSpace(GB)
/
1584            18              1563
/abcd
432             6               425
/abcd/test
288             0               288
/abcd/test/test1
144             0               144
/cldb
576             6               569
/cldb/test
432             0               432
/cldb/test/test1
288             0               288
/cldb/test/test1/test2
144             0               144
/ecvol
288             0               288
/ecvol/test
288             0               288
/ecvol/test/test1
144             0               144
/ecvol/test/test1/test2
144             0               144
/ecvol/test/test1/test2/test3
144             0               144
/ecvol/test/test1/test2/test3/test4
144             0               144
```

REST

```
# curl -k -X GET 'https://
abc.sj.us:8443/rest/node/
toposize' --user mapr:mapr
{"timestamp":1533655046390,"timeofday"
:"2018-08-07 08:17:26.390 GMT-0700
AM","status":"OK","total":3,"data":
[{"path":"/","TotalSpace(GB)":273,"Use
dSpace(GB)":0,"AvailableSpace(GB)":272
},{"path":"/
data","TotalSpace(GB)":273,"UsedSpace(
GB)":0,"AvailableSpace(GB)":272},
{"path":"/data/
default-rack","TotalSpace(GB)":273,"Us
edSpace(GB)":0,"AvailableSpace(GB)":27
2}]}
```

rlimit
Manages resource usage limits for the cluster.
rlimit get
Returns the resource usage limit for the cluster's disk resource.

last-updated: Apr 11, 2022

Syntax

CLI

```
maprcli rlimit get
    -resource disk
    [ -cluster <cluster name> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/rlimit/get?<parameters>`

Parameters

Parameter	Description
`resource`	The type of resource to get the usage limit for. Currently only the value `disk` is supported.
`cluster`	The name of the cluster whose usage limit is being queried.

Examples

Return the disk usage limit for the cluster my.cluster.com:

CLI

```
# maprcli rlimit get -resource
disk -cluster ksTest -json
{
    "timestamp":1529382231966,
    "timeofday":"2018-06-18
09:23:51.966 GMT-0700 PM",
    "status":"OK",
    "total":1,
    "data":[
        {
            "limit":"251947MB",
            "clusterSize":"279942MB",
            "currentUsage":"3MB"
        }
    ]
}
```

REST

```
# curl -k -X GET 'https://
abc.sj.us:8443/rest/rlimit/get?
cluster=ksTest' --user mapr:mapr
{"timestamp":1529382231966,"timeofday"
:"2018-06-18 09:23:51.966 GMT-0700
PM","status":"OK","total":1,"data":
[{"limit":"251947MB","clusterSize":"27
9942MB","currentUsage":"3MB"}]}
```

rlimit set

Sets the resource usage limit for the cluster's disk resource.

last-updated: Apr 11, 2022

Syntax

CLI

```
maprcli rlimit set
    -resource disk
    [ -cluster <cluster name> ]
    -value <limit>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/rlimit/set?<parameters>`

Parameters

Parameter	Description
`resource`	The type of resource for which to set the usage limit. Currently `disk` is the only value that is supported.
`cluster`	The name of the cluster whose usage limit is being set.
`value`	The value of the limit being set. You can express the value as KB, MB, GB, or TB.

Examples

Set the disk usage limit for the cluster my.cluster.com to 80TB:

CLI

```
maprcli rlimit set -resource
disk -cluster my.cluster.com -value
80TB
```

REST

```
https://abc.sj.us:8443/rest/rlimit/
get?
resource=disk&cluster=my.cluster.com&v
alue=80TB
```

Related tasks

Setting Quota Defaults for Users and Groups on page 781
Explains how to set disk space quotas for users and groups.

Related reference

entity modify on page 1649
Modifies a user or group quota or email address. Permissions required: `fc` or `a`.

schedule
Manages schedules.

Schedule Fields

The schedule object contains the following fields:

last-updated: Apr 11, 2022

Field	Value
id	The ID of the schedule.
name	The name of the schedule.
inuse	Indicates whether the schedule is associated with an action.
rules	An array of JSON objects specifying how often the scheduled action occurs. See Rule Fields below.

Rule Fields

The following table shows the fields to use when creating a rules object.

Field	Values
frequency	How often to perform the action: • `once` - Once • `yearly` - Yearly • `monthly` - Monthly • `weekly` - Weekly • `daily` - Daily • `hourly` - Hourly • `semihourly` - Every 30 minutes • `quarterhourly` - Every 15 minutes • `fiveminutes` - Every 5 minutes • `minute` - Every minute
retain	How long to retain the data resulting from the action. For example, if the schedule creates a snapshot, the retain field sets the snapshot's expiration. The retain field consists of an integer and one of the following units of time: • mi - minutes • h - hours • d - days • w - weeks • m - months • y - years **Note:** For offload schedule, set the value for this to `0`.

last-updated: Apr 11, 2022

Field	Values
time	The time of day to perform the action, in 24-hour format: HH
date	The date on which to perform the action: • For single occurrences, specify month, day and year: MM/DD/YYYY • For yearly occurrences, specify the month and day: MM/DD • For monthly occurrences occurrences, specify the day: DD Daily and hourly occurrences do not require the date field.

Example

The following example JSON shows a schedule called "snapshot," with three rules.

```
     {
            "id":8,
            "name":"snapshot",
            "inuse":0,
            "rules":[
                {
                    "frequency":"monthly",
                    "date":"8",
                    "time":14,
                    "retain":"1m"
                },
                {
                    "frequency":"weekly",
                    "date":"sat",
                    "time":14,
                    "retain":"2w"
                },
                {
                    "frequency":"hourly",
                    "retain":"1d"
                }
            ]
        }
```

schedule create

Creates a schedule. Permissions required: `fc` or `a`.

A schedule can be associated with a volume to automate mirror syncing, snapshot creation, and data offload. See volume create and volume modify.

Syntax

CLI

```
maprcli schedule create
    [ -cluster <cluster> ]
    -schedule <JSON>
```

REST

Request Type	POST

last-updated: Apr 11, 2022

Request URL	`http[s]://<host>:<port>/rest/schedule/create?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`schedule`	A JSON object describing the schedule. See Schedule Objects for more information.

Examples

Scheduling a Single Occurrence

CLI

```
maprcli schedule create -schedule
'{"name":"Schedule-1","rules":
[{"frequency":"once","retain":"1w","ti
me":13,"date":"12/5/2010"}]}'
```

REST

```
https://abc.sj.us:8443/rest/schedule/
create?
schedule={"name":"Schedule-1","rules":
[{"frequency":"once","retain":"1w","ti
me":13,"date":"12/5/2010"}]}
```

A Schedule with Several Rules

CLI

```
maprcli schedule create -schedule
'{"name":"Schedule-2","rules":
[{"frequency":"weekly","date":"sun","t
ime":7,"retain":"2w"},
{"frequency":"daily","time":14,"retain
":"1w"},
{"frequency":"hourly","retain":"1w"},
{"frequency":"yearly","date":"11/5","t
ime":14,"retain":"1w"}]}'
```

REST

```
https://abc.sj.us:8443/rest/schedule/
create?
schedule={"name":"Schedule-1","rules":
[{"frequency":"weekly","date":"sun","t
ime":7,"retain":"2w"},
{"frequency":"daily","time":14,"retain
":"1w"},
{"frequency":"hourly","retain":"1w"},
{"frequency":"yearly","date":"11/5","t
ime":14,"retain":"1w"}]}
```

schedule list

Lists the schedules on the cluster.

last-updated: Apr 11, 2022

Syntax

CLI

```
maprcli schedule list
    [ -cluster <cluster> ]
    [ -output terse|verbose ]
    [ -sortby ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/schedule/list[?<parameters>]`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`output`	Specifies whether the output should be terse or verbose.
`sortby`	Specifies one of the following attributes to sort the list of schedules by: `scheduleid`, `schedulename`, `schedulerulefrequency`, `scheduleruledate`, `scheduleruletime`, `scheduleruleminutes`, `scheduleruleretaintime`, `scheduleinuse`. By default, the list of schedules sorted by `schedulename`.

Output

A list of all schedules on the cluster. See Schedule Objects for more information.

Sample Output

```
# maprcli schedule list
id  name            inuse  rules
1   Critical data   0      ...
2   Important data  0      ...
3   Normal data     0      ...
```

Examples

List schedules:

CLI

```
# maprcli schedule list -json
{
    "timestamp":1531004445284,
    "timeofday":"2018-07-07
04:00:45.284 GMT-0700 PM",
    "status":"OK",
    "total":4,
    "data":[
        {
            "id":4,
            "name":"Automatic Tiering
Scheduler",
```


last-updated: Apr 11, 2022

```
            "inuse":0,
            "description":"Automatic
Scheduler for EC and Cold Tier: It
uses internal policies to schedule
the task",
            "rules":{

            }
        },
        {
            "id":1,
            "name":"Critical data",
            "inuse":0,
            "rules":[
                {

"frequency":"hourly",
                    "retain":"24h"
                },
                {

"frequency":"daily",
                    "time":0,
                    "retain":"7d"
                },
                {

"frequency":"weekly",
                    "date":"sun",
                    "time":0,
                    "retain":"12w"
                }
            ]
        },
        {
            "id":2,
            "name":"Important data",
            "inuse":0,
            "rules":[
                {

"frequency":"daily",
                    "time":6,
                    "retain":"24h"
                },
                {

"frequency":"daily",
                    "time":12,
                    "retain":"24h"
                },
                {

"frequency":"daily",
                    "time":18,
                    "retain":"24h"
                },
                {

"frequency":"daily",
                    "time":0,
                    "retain":"7d"
```

```
                },
                {

"frequency":"weekly",
                    "date":"sun",
                    "time":0,
                    "retain":"4w"
                },
                {

"frequency":"monthly",
                    "date":"1",
                    "time":0,
                    "retain":"2m"
                }
            ]
        },
        {
            "id":3,
            "name":"Normal data",
            "inuse":0,
            "rules":[
                {

"frequency":"daily",
                    "time":0,
                    "retain":"7d"
                },
                {

"frequency":"weekly",
                    "date":"sun",
                    "time":0,
                    "retain":"4w"
                },
                {

"frequency":"monthly",
                    "date":"1",
                    "time":0,
                    "retain":"2m"
                }
            ]
        }
    ]
}
```

REST

```
# curl -k -X GET 'https://
abc.sj.us:8443/rest/schedule/
list' --user mapr:mapr
{"timestamp":1531004578264,"timeofday"
:"2018-07-07 04:02:58.264 GMT-0700
PM","status":"OK","total":4,"data":
[{"id":4,"name":"Automatic Tiering
Scheduler","inuse":0,"description":"Au
tomatic Scheduler for EC and Cold
Tier: It uses internal policies to
schedule the task","rules":{}},
{"id":1,"name":"Critical
data","inuse":0,"rules":
[{"frequency":"hourly","retain":"24h"}
```

```
,
{"frequency":"daily","time":0,"retain"
:"7d"},
{"frequency":"weekly","date":"sun","ti
me":0,"retain":"12w"}]},
{"id":2,"name":"Important
data","inuse":0,"rules":
[{"frequency":"daily","time":6,"retain
":"24h"},
{"frequency":"daily","time":12,"retain
":"24h"},
{"frequency":"daily","time":18,"retain
":"24h"},
{"frequency":"daily","time":0,"retain"
:"7d"},
{"frequency":"weekly","date":"sun","ti
me":0,"retain":"4w"},
{"frequency":"monthly","date":"1","tim
e":0,"retain":"2m"}]},
{"id":3,"name":"Normal
data","inuse":0,"rules":
[{"frequency":"daily","time":0,"retain
":"7d"},
{"frequency":"weekly","date":"sun","ti
me":0,"retain":"4w"},
{"frequency":"monthly","date":"1","tim
e":0,"retain":"2m"}]}]}
```


schedule modify
Modifies an existing schedule, specified by ID. Permissions required: `fc` or `a`.

To find a schedule's ID:

1. Use the schedule list command to list the schedules.
2. Select the schedule to modify.
3. Pass the selected schedule's ID in the -id parameter to the `schedule modify` command.

Syntax

CLI

```
maprcli schedule modify
    [ -cluster <cluster> ]
    -id <schedule ID>
    [ -name <schedule name> ]
    [ -rules <JSON>]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/schedule/modify?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`id`	The ID of the schedule to modify.
`name`	The new name of the schedule.
`rules`	A JSON object describing the rules for the schedule. If specified, replaces the entire existing rules object in the schedule. For information about the fields to use in the JSON object, see Rule Fields.

Examples

Modify a schedule

CLI

```
maprcli schedule modify -id 0 -name
Newname -rules
'[{"frequency":"weekly","date":"sun","
time":7,"retain":"2w"},
{"frequency":"daily","time":14,"retain
":"1w"}]'
```

REST

```
https://abc.sj.us:8443/rest/schedule/
modify?
id=0&name=Newname&rules=[{"frequency":
"weekly","date":"sun","time":7,"retain
":"2w"},
{"frequency":"daily","time":14,"retain
":"1w"}]
```

schedule remove
Removes a schedule.

A schedule can only be removed if it is not associated with any volumes. See volume modify.

Syntax

CLI

```
maprcli schedule remove
    [ -cluster <cluster> ]
    -id <schedule ID>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/schedule/remove?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.

last-updated: Apr 11, 2022

Parameter	Description
`id`	The ID of the schedule to remove.

Examples

Remove schedule with ID 0:

CLI

```
maprcli schedule remove -id 0
```

REST

```
https://abc.sj.us:8443/rest/schedule/
remove?id=0
```

security

Configures security options.

genkey
Generates keys and certificates.

This command is for internal use only. You must use `configure.sh` with the `genkeys` option instead.

genticket
Generates tickets.

This command is for internal use only. You must use the maprlogin utility instead.

service list
Lists all services on the specified node, the memory allocated for each service, the state of each service, and log path for each service.

Syntax

CLI

```
/opt/mapr/bin/maprcli service list
    -node <node name>
    [ -cluster <cluster name> ]
    [ -zkconnect <ZooKeeper connect
string> ]
    [ -output terse|verbose ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/service/list?<parameters>`

Parameters

Parameter	Description
cluster	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.

last-updated: Apr 11, 2022

Parameter	Description
node	The node for which to list services. Default: localhost. If this is not specified, the `/etc/hosts` file must include the IP address or hostname for the localhost.
output	Whether the output should be terse or verbose. Default: verbose.
zkconnect	A ZooKeeper connect string, which specifies a list of the hosts running ZooKeeper, and the port to use on each, in the format: '<host>[:<port>][,<host>[:<port>]...]'. To obtain zookeeper connection strings, use the `maprcli node listzookeepers` command.

Output Fields

Field	Description
memallocated	The amount of system memory allocated to the service.
name	Service name.
state	Current state of the service. See Service States on page 1747.
logpath	Path to the log files for the service.
displayname	Display name of the service in the Control System.

Service States

The following table lists the service states with their descriptions:

State	Description
0	Not configured. The package for the service is not installed and/or the service is not configured (`configure.sh` has not run). This state is also returned for all the services if you run the command without specifying a node.
1	Configured. The package for the service is installed and configured.
2	Running. The service is installed, started by the warden, and is currently running.
3	Stopped. The service is installed and `configure.sh` has run, but the service is not running.
4	Failed. The service is installed and configured, but not running.
5	Stand by. The service is installed and is in standby mode, waiting to take over in case of failure of another instance.

Examples

CLI Example

The following output is an example of the service information returned when you run the `service list` command without specifying the node:

```
# /opt/mapr/bin/maprcli service list -json
{
```


last-updated: Apr 11, 2022

```
    "timestamp":1555048050131,
    "timeofday":"2019-04-11 10:47:30.131 GMT-0700 PM",
    "status":"OK",
    "total":10,
    "data":[
        {
            "name":"fileserver",
            "state":0,
            "logpath":"/opt/mapr/logs/mfs.log",
            "displayname":"FileServer"
        },
        {
            "name":"resourcemanager",
            "state":0,
            "logpath":"/opt/mapr/hadoop/hadoop-2.7.4/logs",
            "displayname":"ResourceManager"
        },
        {
            "name":"filemigrate",
            "state":0,
            "logpath":"/opt/mapr/filemigrate/filemigrate-1.0.0/logs",
            "displayname":"FileMigrate"
        },
        {
            "name":"cldb",
            "state":0,
            "logpath":"/opt/mapr/logs/cldb.log",
            "displayname":"CLDB"
        },
        {
            "name":"nfs4",
            "state":0,
            "logpath":"/opt/mapr/logs/nfs4/nfs4server.log",
            "displayname":"NFS4 Gateway"
        },
        {
            "name":"mastgateway",
            "state":0,
            "logpath":"/opt/mapr/logs/mastgateway.log",
            "displayname":"MASTGatewayService"
        },
        {
            "name":"nodemanager",
            "state":0,
            "logpath":"/opt/mapr/hadoop/hadoop-2.7.4/logs",
            "displayname":"NodeManager"
        },
        {
            "name":"gateway",
            "state":0,
            "logpath":"/opt/mapr/logs/gateway.log",
            "displayname":"GatewayService"
        },
        {
            "name":"hoststats",
            "state":0,
            "logpath":"/opt/mapr/logs/hoststats.log",
            "displayname":"HostStats"
        },
        {
            "name":"apiserver",
            "state":0,
            "logpath":"/opt/mapr/apiserver/logs/apiserver.log",
            "displayname":"APIServer"
```

```
        }
    ]
```

REST Example

The following output is an example of the service information returned when you issue the `service list` REST API call, without specifying a node:

```
# curl -k -X GET 'https://abc.sj.us:8443/rest/service/list' --user mapr:mapr
{"timestamp":1529380971417,"timeofday":"2018-06-18
09:02:51.417 GMT-0700 PM","status":"OK","total":9,"data":
[{"name":"fileserver","state":0,"logpath":"/opt/
mapr/logs/mfs.log","displayname":"FileServer"},
{"name":"resourcemanager","state":0,"logpath":"/opt/mapr/
hadoop/hadoop-2.7.0/logs","displayname":"ResourceManager"},
{"name":"cldb","state":0,"logpath":"/opt/mapr/logs/
cldb.log","displayname":"CLDB"},{"name":"nfs4","state":0,"logpath":"/opt/
mapr/logs/nfs4/nfs4server.log","displayname":"NFS4
Gateway"},{"name":"mastgateway","state":0,"logpath":"/opt/
mapr/logs/mastgateway.log","displayname":"MASTGatewayService"},
{"name":"nodemanager","state":0,"logpath":"/opt/mapr/
hadoop/hadoop-2.7.0/logs","displayname":"NodeManager"},
{"name":"gateway","state":0,"logpath":"/opt/mapr/
logs/gateway.log","displayname":"GatewayService"},
{"name":"hoststats","state":0,"logpath":"/opt/mapr/
logs/hoststats.log","displayname":"HostStats"},
{"name":"apiserver","state":0,"logpath":"/opt/mapr/apiserver/logs/
apiserver.log","displayname":"APIServer"}]}
```

The following output is an example of the service information returned when you run the `service list` command after specifying a node:

```
/opt/mapr/bin/maprcli service list -node 10.10.82.29 -json
{
    "timestamp":1555049507312,
    "timeofday":"2019-04-11 11:11:47.312 GMT-0700 PM",
    "status":"OK",
    "total":10,
    "data":[
        {
            "name":"fileserver",
            "state":2,
            "memallocated":"8382.0",
            "logpath":"/opt/mapr/logs/mfs.log",
            "displayname":"FileServer"
        },
        {
            "name":"resourcemanager",
            "state":2,
            "memallocated":"2395.0",
            "logpath":"/opt/mapr/hadoop/hadoop-2.7.4/logs",
            "displayname":"ResourceManager"
        },
        {
            "name":"filemigrate",
            "state":4,
            "logpath":"/opt/mapr/filemigrate/filemigrate-1.0.0/logs",
            "displayname":"FileMigrate"
        },
        {
            "name":"cldb",
            "state":2,
            "memallocated":"1916.0",
```

```
            "logpath":"/opt/mapr/logs/cldb.log",
            "displayname":"CLDB"
        },
        {
            "name":"nfs4",
            "state":2,
            "memallocated":"2048.0",
            "logpath":"/opt/mapr/logs/nfs4/nfs4server.log",
            "displayname":"NFS4 Gateway"
        },
        {
            "name":"mastgateway",
            "state":2,
            "memallocated":"2395.0",
            "logpath":"/opt/mapr/logs/mastgateway.log",
            "displayname":"MASTGatewayService"
        },
        {
            "name":"nodemanager",
            "state":2,
            "memallocated":"325.0",
            "logpath":"/opt/mapr/hadoop/hadoop-2.7.4/logs",
            "displayname":"NodeManager"
        },
        {
            "name":"gateway",
            "state":2,
            "memallocated":"239.0",
            "logpath":"/opt/mapr/logs/gateway.log",
            "displayname":"GatewayService"
        },
        {
            "name":"hoststats",
            "state":2,
            "memallocated":"Auto",
            "logpath":"/opt/mapr/logs/hoststats.log",
            "displayname":"HostStats"
        },
        {
            "name":"apiserver",
            "state":2,
            "memallocated":"1000.0",
            "logpath":"/opt/mapr/apiserver/logs/apiserver.log",
            "displayname":"APIServer"
        }
    ]
}
```


REST Example

The following output is an example of the service information returned when you issue the `service list` REST API call, after specifying a node:


```
curl -k -X GET 'https://abc.sj.us:8443/rest/service/list' --user mapr:mapr
  {"timestamp":1529380971417,"timeofday":"2018-06-18
09:02:51.417 GMT-0700 PM","status":"OK","total":9,"data":
[{"name":"fileserver","state":0,"logpath":"/opt/
mapr/logs/mfs.log","displayname":"FileServer"},
{"name":"resourcemanager","state":0,"logpath":"/opt/mapr/
hadoop/hadoop-2.7.0/logs","displayname":"ResourceManager"},
{"name":"cldb","state":0,"logpath":"/opt/mapr/logs/
cldb.log","displayname":"CLDB"},{"name":"nfs4","state":0,"logpath":"/opt/
mapr/logs/nfs4/nfs4server.log","displayname":"NFS4
```

```
Gateway"},{"name":"mastgateway","state":0,"logpath":"/opt/
mapr/logs/mastgateway.log","displayname":"MASTGatewayService"},
{"name":"nodemanager","state":0,"logpath":"/opt/mapr/
hadoop/hadoop-2.7.0/logs","displayname":"NodeManager"},
{"name":"gateway","state":0,"logpath":"/opt/mapr/
logs/gateway.log","displayname":"GatewayService"},
{"name":"hoststats","state":0,"logpath":"/opt/mapr/
logs/hoststats.log","displayname":"HostStats"},
{"name":"apiserver","state":0,"logpath":"/opt/mapr/apiserver/logs/
apiserver.log","displayname":"APIServer"}]}
```


Note: When you configure high availability for the ResourceManager, the status of the standby ResourceManager service differs based on the selected failover implementation. When the cluster uses manual or automatic failover for the ResourceManager, standby ResourceManagers have a state equal to 2 (running). When the cluster uses zero configuration failover for the ResourceManager, standby ResourceManagers have a state equal to 5 (stand by).

setloglevel
Sets log level on individual services.
setloglevel cldb
Sets the log level on the CLDB service. Permissions required: `fc` or `a`.

Syntax

CLI

```
maprcli setloglevel cldb
    -classname <class>
    -loglevel DEBUG|ERROR|FATAL|INFO|
TRACE|WARN
    -node <node>
    -port <port>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/setloglevel/cldb?<parameters>`

Parameters

Parameter	Description
`classname`	The name of the class for which to set the log level. The class can be at the package level or a specific class. Contact MapR Support for this parameter.

Parameter	Description
loglevel	The log level to set. Default: INFO • DEBUG • ERROR • FATAL • INFO • TRACE • WARN
node	The node on which to set the log level.
port	The CLDB port. Default: 7222

Examples

CLI

```
maprcli setloglevel cldb
    -classname
com.mapr.fs.cldb.CLDBServer
    -loglevel debug
    -node abc.sj.us
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/setloglevel/cldb?
classname=com.mapr.fs.cldb.CLDBServer&
loglevel=debug&node=abc.sj.us' --user
mapr:mapr
{"timestamp":1529380341288,"timeofday"
:"2018-06-18 08:52:21.288 GMT-0700
PM","status":"OK","total":0,"data":[]}
```

setloglevel fileserver

Sets the log level on the FileServer service. Permissions required: fc or a.

Syntax

CLI

```
maprcli setloglevel fileserver
    -classname <class>
    -loglevel DEBUG|ERROR|FATAL|INFO|
TRACE|WARN
    -node <node>
    -port <port>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/setloglevel/fileserver?<parameters>`

Parameters

Parameter	Description
classname	The name of the class for which to set the log level. The classname is listed under the `maprcli trace info` command. Contact MapR Support for this parameter.
loglevel	The log level to set. Default: INFO • DEBUG • ERROR • FATAL • INFO • TRACE • WARN
node	The node on which to set the log level.
port	The MapR File System port. Default: 5660

Examples

CLI

```
maprcli setloglevel fileserver
    -classname FileServer
    -loglevel debug
    -node centos26.lab
```

REST

```
https://abc.sj.us:8443/rest/
setloglevel/fileserver?
classname=FileServer&loglevel=debug&no
de=centos26.lab
```

setloglevel hbmaster

Sets the log level on the HBase Master service. Permissions required: `fc` or a.

Syntax

CLI

```
maprcli setloglevel hbmaster
    -classname <class>
    -loglevel DEBUG|ERROR|FATAL|INFO|
TRACE|WARN
    -node <node>
    -port <port>
```

REST

```
http[s]://<host>:<port>/rest/
setloglevel/hbmaster?<parameters>
```


last-updated: Apr 11, 2022

Parameters

Parameter	Description
classname	The name of the class for which to set the log level. The class can be specified at the package level or for a specific class. Contact MapR Support for this parameter.
loglevel	The log level to set. Default: INFO • DEBUG • ERROR • FATAL • INFO • TRACE • WARN
node	The node on which to set the log level.
port	The HBase Master webserver port. Default: 16000 (16010 for the WebUI)

Examples

CLI

```
maprcli setloglevel hbmaster
    -classname
org.apache.hadoop.hbase.master
    -loglevel debug
    -node centos26.lab
    -port 16000
```

REST

```
https://centos26.lab:8443/rest/
setloglevel/hbmaster?
classname=org.apache.hadoop.hbase.mast
er&loglevel=debug&node=centos26.lab&po
rt=16000
```

setloglevel hbregionserver

Sets the log level on the HBase RegionServer service. Permissions required: `fc` or `a`.

Syntax

CLI

```
maprcli setloglevel hbregionserver
    -classname <class>
    -loglevel DEBUG|ERROR|FATAL|INFO|
TRACE|WARN
    -node <node>
    -port <port>
```

REST

```
http[s]://<host>:<port>/rest/
setloglevel/hbregionserver?
<parameters>
```

Parameters

Parameter	Description
classname	The name of the class for which to set the log level. The class can be specified at the package level or for a specific class. Contact MapR Support for this parameter.
loglevel	The log level to set. Default: INFO • DEBUG • ERROR • FATAL • INFO • TRACE • WARN
node	The node on which to set the log level.
port	The HBase Region Server webserver port. Default: 16020 (16030 for the WebUI)

Examples

CLI

```
maprcli setloglevel hbregionserver
    -classname
org.apache.hadoop.hbase.regionserver
    -loglevel debug
    -node centos26.lab
    -port 16020
```

REST

```
https://centos26.lab:8443/rest/
setloglevel/hbregionserver?
classname=org.apache.hadoop.hbase.regi
onserver&loglevel=debug&node=centos26.
lab&port=16020
```

setloglevel nfs

Sets the log level on the NFS service. Permissions required: `fc` or `a`.

Syntax

CLI

```
maprcli setloglevel nfs
    -classname <class>
    -loglevel DEBUG|ERROR|FATAL|INFO|
TRACE|WARN
```


last-updated: Apr 11, 2022

```
    -node <node>
    -port <port>
    -isusermode <TRUE|FALSE>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/setloglevel/nfs?<parameters>`

Parameters

Parameter	Description
classname	The name of the class for which to set the log level. The classname is listed under the `maprcli trace info` command. Contact MapR Support for this parameter.
loglevel	The log level to set. Default: INFO • DEBUG • ERROR • FATAL • INFO • TRACE • WARN
node	The node on which to set the log level.
port	The NFS port. Default: 9998
isusermode	Whether or not is the request is for user mode.

Examples

CLI

```
maprcli setloglevel nfs
    -classname NFSD
    -loglevel debug
    -node centos26.lab
```

REST

```
https://abc.sj.us:8443/rest/
setloglevel/nfs?
classname=NFSD&loglevel=debug&node=cen
tos26.lab
```

stream
Manages stream functionality.
stream assign list
For the given stream, lists consumers and the topics and partitions that the consumers are reading messages from.

Permissions Required
To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the path
- `adminperm`, `consumeperm`, `produceperm`, or `topicperm` permission on the stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream assign list -path <Stream Path > [ -consumergroup <Consumer Group ID> ] [ -topic <Topic Name> ] [ -partition <Partition ID> ] [ -detail <Detail Parameter takes no value>  ]`
REST	`http[s]://<host>:<port>/rest/stream/assign/list?path=<path>`

Parameters

Parameter	Description
`path`	The path and name of the stream.
`consumergroup`	Specifies the ID of a particular consumer group that you want to list the consumers for.
`topic`	The name of a topic to list the consumers for. If you also specify the `-partition` parameter, only the consumers that are reading from the indicated partition are listed.
`partition`	The ID of a specific partition. If you specify this ID, you must also use the `-topic` parameter.
`detail`	Includes the values of additional parameters in the output. These parameters are used internally.

Sample Output
With the `-detail` parameter:

```
# maprcli stream assign list -path /s1 -json -detail
{
    "timestamp":1441965109585,
    "timeofday":"2015-09-11 02:51:49.585 GMT-0700",
    "status":"OK",
    "total":1,
    "data":[
        {
            "consumergroup":"xyzt1",
            "topic":"topic1",
            "assignseqnum":1,
            "consumerguid":"F3693413-2600-0876-CC91-052FA4F25500",
            "consumer":"ravindra.perf",
            "consumerip":"10.10.30.200",
```


last-updated: Apr 11, 2022

```
            "consumerpid":"30768",
            "assignment":"0,1,2,3"
        }
    ]
}
```

Without the `-detail` parameter:

```
# maprcli stream assign list -path /s1 -json
{
    "timestamp":1441965116100,
    "timeofday":"2015-09-11 02:51:56.100 GMT-0700",
    "status":"OK",
    "total":1,
    "data":[
        {
            "consumergroup":"xyzt1",
            "topic":"topic1",
            "consumer":"ravindra.perf",
            "consumerip":"10.10.30.200",
            "consumerpid":"30768",
            "assignment":"0,1,2,3"
        }
    ]
}
```

Field Descriptions

consumergroup	The name of the consumer group that is reading messages from this topic partition.
topic	The name of the topic.
assignseqnum	The sequence number of the current assignment of this partition. This value is used internally.
consumerguid	The globally unique identifer for the consumer. This value is used internally.
consumer	The ID of the consumer. This value is set with the client.id configuration parameter.
consumerip	The IP address of the consumer.
consumerpid	The process ID of the consumer.
assignment	The index numbers of the partitions that are assigned to this consumer.

`stream create`

Creates a new stream.

After you create a stream, you can edit the values of its parameters with the command `maprcli stream edit`.

To see the value of a stream's parameters, use the command `maprcli stream info`.

To run this command, your user ID must have write permission on the directory in which you want to create a stream.

Permissions Required

To run this command, your user ID must have the following permissions:

- readAce and writeAce on the volume

- lookupdir on directories in the path

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream create -path <Stream Path> [-ttl <Time to live in second> default:604800] [-autocreate <Auto create topics> default:true] [-defaultpartitions <Default partitions per topic> default:1] [-compression off|lz4|lzf|zlib. default: inherit from parent directory] [-produceperm <Producer access control expression> default u:creator] [-consumeperm <Consumer access control expression> default u:creator] [-topicperm <Topic CRUD access control expression> default u:creator] [-copyperm <Stream copy access control expression> default u:creator] [-adminperm <Stream administration access control expression> default u:creator] [-copymetafrom <Stream to copy attributes from> default:none] [-ischangelog <true|false> default: false] [-defaulttimestamptype timestamp type: createtime | logappendtime. default: createtime] [-pidexpirysecs <Producer ID expiry time in seconds. Default: 6048000>] [-mincompactionlag <Set time in melliseconds for which a message remains uncompacted. Default: 0>] [-deleteretention <Set the time in milliseconds for which delete records are retained. Default: 86400000>]`
REST	`http[s]://<host>:<port>/rest/stream/create?path=<path>`

Parameters

Parameter	Description
`path`	The path and name of the stream to create. The path to the stream can include any character allowed by MapR. For example, `/my/path/with:/to/mystream` is valid, but `/my/path/with:/to/mystream:withcolon` is invalid. The name of the stream cannot include a colon (:) or a forward slash (/).

last-updated: Apr 11, 2022

Parameter	Description
`ttl`	Specifies the number of seconds to elapse between the publication of a message in a topic in this stream and the expiration of that message. Consumers do not see messages that have expired. Messages that have expired are deleted during the next purge process. See Time-to-Live for Messages for details. A value of 0 causes messages to be retained indefinitely.
`autocreate`	Specifies whether to create a topic automatically when a producer tries to write the first message to it. Values are `true` and `false`. The default is `true`.
`defaultpartitions`	Specifies the default number of partitions to allocate to new topics in the stream.
`compression`	Specifies the compression setting to use for the stream. Producer client libraries can bundle messages that are to be published on the same partition and compress them. The messages are sent to the server compressed, are stored compressed, are replicated to other containers compressed, and (if stream replication is configured) replicated to replica streams compressed. Consumer client libraries receive compressed data, decompresses it, and passes it to client applications. Valid options are `off`, `lzf`, `lz4`, and `zlib`. The default setting is the type of compression that is set for the directory in which the stream is located. For more information, see Compression.
`produceperm`	Specifies the access-control expression that controls who can publish messages to topics in the stream. See ACE Syntax.
`consumeperm`	Specifies the access-control expression that controls who can who can listen to topics in the stream. See ACE Syntax.
`topicperm`	Specifies the access-control expression that controls who can create, edit, or remove topics in the stream. See ACE Syntax.
`copyperm`	Specifies the access-control expression that controls who can use `mapr copystream` or `mapr diffstreams` on the stream. See ACE Syntax.
`adminperm`	Determines which users can modify ACEs for a stream, set up replication of a stream, and modify other attributes of a stream. **Important:** By default, only the stream owner can modify this setting; however, a patch is available that changes this behavior. After applying the patch, both the stream owner and the MAPR user can modify this setting. The patch works with MapR Core-6.1.0. To install patches, see Applying a Patch. See ACE Syntax. This permission includes the `topicperm` permission.

last-updated: Apr 11, 2022

Parameter	Description
`copymetafrom`	If you plan to replicate messages to this stream from another stream, specify the path to that other stream. The metadata from that stream will be copied to the new stream when the new stream is created.
`ischangelog`	Specifies whether the stream is for the Change Data Capture feature's changed data records. Value: true|false. Default: false. • If you want to use a non-default partition value (Default: 1) for the topic, use the `maprcli stream create -path </mypath/stream:topic> -ischangelog true -defaultpartitions < value other than 1 >` command to create the stream and then create the topic with the `maprcli streams topic create` command. • If you want to use the default partitions value (1) for the Change Data Capture feature, use the `maprcli stream create -path </mypath/stream> -ischangelog true` command to create the stream and then use the `maprcli table changelog add -path <mypath/stream:topic` command to set up the Change Data Capture feature and create the topic.
`defaulttimestamptype`	Specifies the type of timestamp stored in the topic's message. Value: createtime | logappendtime Default: createtime. The topic inherits the default value from the stream unless the topic sets the timestamp type to a different value. A `createtime` value is the time defined by the user or application (when creating the message). If user or application does not define this value (or passes null), the client uses the current system timestamp. A `logappendtime` value is the time when the message (log) was appended to the server.
`pidexpirysecs`	Specifies the expiration time for the Producer ID. This parameter fixes the lifetime for the Producer ID. Default: 604800
`mincompactionlag`	Sets the **minimum** delay (in milliseconds) before which messages are **not** compacted. It is the **minimum** time that the messages are available for consumption. Beyond this time period, the messages **may** be compacted. Default: 0 The lag is calculated from the time that a message was produced to the stream topic-partition. **Note:** Compaction is set only when you edit the stream. See stream edit on page 1765.
`deleteretention`	Sets the **minimum** time (in milliseconds) before which deleted records are removed. It is the **minimum** time that the deleted records are still available. Beyone this time period, the deleted messages **may** be removed. Default: 86400000 Used with log compaction.

last-updated: Apr 11, 2022

`stream cursor delete`

Deletes committed cursors that are in the partitions in a stream.

Note: Deleting the committed cursors for active consumers has no effect on the consumers. Consumers use read cursors to keep track of where they currently are in partitions.

For example, the consumer `consumer1` continues reading the messages in a partition from the position of the consumer's read cursor even after the consumer's committed cursor is deleted. However, if `consumer1` goes offline and the partition is reassigned to another consumer (`consumer2`) in the same consumer group before `consumer1` creates another committed cursor, `consumer2` starts reading the partition at the most recent message.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminperm` or `consumeperm` permission on the stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream cursor delete -path <Stream Path> [ -consumergroup <Consumer Group ID> ] [ -topic <Topic Name> ] [ -partition <Partition ID> ]`
REST	`http[s]://<host>:<port>/rest/stream/cursor/delete?path=<path>`

Parameters

Parameter	Description
`path`	The path and name of the stream in which the committed cursors are located.
`consumergroup`	Specifies the ID of a particular consumer group that you want to delete the committed cursors for.
`topic`	The name of a topic to delete committed cursors from. If you also specify the `-partition` parameter, only the committed cursors in the indicated partition are deleted.
`partition`	The ID of the partition where the committed cursors that you want to delete is located. If you specify this ID, you must also use the `-topic` parameter.

`stream cursor list`

Lists the cursors for the consumers of a stream.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the path
- `adminperm`, `consumeperm`, `produceperm`, or `topicperm` permission on the stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream cursor list -path <Stream Path> [ -consumergroup <Consumer Group ID> ] [ -topic <Topic Name> ] [ -partition <Partition ID> ]`
REST	`http[s]://<host>:<port>/rest/stream/cursor/list?path=<path>`

Parameters

Parameter	Description
`path`	The path and name of the stream in which the cursors are located.
`consumergroup`	Specifies the ID of a particular consumer group that you want to list the cursors for.
`topic`	The name of a topic to list committed cursors from. If you also specify the `-partition` parameter, only the committed cursors in the indicated partition are listed.
`partition`	The ID of the partition where the particular cursor that you want to list is located. If you specify this ID, you must also use the `-topic` parameter.

Sample Output

```
# maprcli stream cursor list -path /s1 -topic topic0 -json
{
    "timestamp":1441883091373,
    "timeofday":"2015-09-10 04:04:51.373 GMT-0700",
    "status":"OK",
    "total":4,
    "data":[
        {
            "consumergroup":"consume.full",
            "topic":"topic0",
            "partitionid":"0",
            "produceroffset":"249890625",
            "committedoffset":"249874696",
            "producertimestamp":"2015-09-10T03:48:14.080-0700",
            "committedtimestamp":"2015-09-10T03:48:14.080-0700",
            "consumerlagmillis":"0"
        },
        {
            "consumergroup":"consume.half",
            "topic":"topic0",
```


last-updated: Apr 11, 2022

```
            "partitionid":"0",
            "produceroffset":"249890625",
            "committedoffset":"113214511",
            "producertimestamp":"2015-09-10T03:48:14.080-0700",
            "consumertimestamp":"2015-09-10T03:48:07.768-0700",
            "consumerlagmillis":"6312"
        },
        {
            "consumergroup":"consume.full",
            "topic":"topic0",
            "partitionid":"1",
            "produceroffset":"249890625",
            "committedoffset":"239303323",
            "producertimestamp":"2015-09-10T03:48:14.082-0700",
            "consumertimestamp":"2015-09-10T03:48:13.581-0700",
            "consumerlagmillis":"501"
        },
        {
            "consumergroup":"consume.half",
            "topic":"topic0",
            "partitionid":"1",
            "produceroffset":"249890625",
            "committedoffset":"113214511",
            "producertimestamp":"2015-09-10T03:48:14.082-0700",
            "consumertimestamp":"2015-09-10T03:48:07.769-0700",
            "consumerlagmillis":"6313"
        },
    ]
}
```


Field Descriptions

consumergroup	The ID of the consumer group to which belongs the consumer that owns the committed cursor.
committedoffset	The last offset that was committed by the consumer that is reading from the listed partition and that belongs to the listed consumer group.
consumerlagmillis	The difference in milliseconds between the timestamp of the last published message and the timestamp of the last message consumed by the consumer.
consumertimestamp	The timestamp of the most recent message that the consumer has consumed.
partitionid	The index number of the partition within the topic. The first partition in a topic has an index of 0, the next partition an index of 1, and so on.
produceroffset	The maximum offset produced for this partition.
topic	The name of the topic that the cursor corresponds to.

stream delete

Deletes the specified stream. Deleted streams cannot be recovered unless they were previously replicated.

Producers are no longer able to publish messages to topics in the stream, and consumers are no longer able to read messages from topics in the stream.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume

last-updated: Apr 11, 2022

- `lookupdir` on directories in the path

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream delete -path <Stream Path>`
REST	`http[s]://<host>:<port>/rest/stream/delete?path=<path>`

Parameters

Parameter	Description
`path`	The path and name of the stream to delete.

`stream edit`

Edits the values of parameters for the specified stream.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminperm` permission on the stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

last-updated: Apr 11, 2022

Syntax

CLI	`maprcli stream edit -path Stream Path [-ttl <Time to live in seconds>] [-autocreate true|false] [-defaultpartitions <Default partitions per topic>] [-compression off|lz4|lzf|zlib] [-produceperm <Producer access control expression> default u:creator] [-consumeperm <Consumer access control expression> default u:creator] [-topicperm <Topic CRUD access control expression> default u:creator] [-copyperm <Stream copy access control expression> default u:creator] [-adminperm <Stream administration access control expression> default u:creator] [-defaulttimestamptype timestamp type: createtime | logappendtime. default: createtime] [-compact <Sets log compaction for a stream. Value: true | false default: false>] [-pidexpirysecs <Producer ID expiry time in seconds. Default: 604800>] [-mincompactionlag <Sets time in milliseconds for which a message remains uncompacted. default: 0>] [-deleteretention <Sets the time in milliseconds for which delete records are retained. Default: 86400000>] [-force <When used with -compact, forces enabling log compaction on a stream. Parameter takes no value.>]`
REST	`http[s]://<host>:<port>/rest/stream/edit?path=<path>`

Parameters

Parameter	Description
`path`	The path and name of the stream to create.
`ttl`	Specifies the number of seconds to elapse between the publication of a message in a topic in this stream and the expiration of that message. Consumers do not see messages that have expired. Messages that have expired are deleted during the next purge process. See Time-to-Live for Messages for details. A value of 0 causes messages to be retained indefinitely.
`autocreate`	Specifies whether to create a topic automatically when a producer tries to write the first message to it. Values are `true` and `false`. The default is `true`.
`defaultpartitions`	Specifies the default number of partitions to allocate to new topics in the stream.

Parameter	Description
`compression`	Specifies the compression setting to use for the stream. Producer client libraries can bundle messages that are to be published on the same partition and compress them. The messages are sent to the server compressed, are stored compressed, are replicated to other containers compressed, and (if stream replication is configured) replicated to replica streams compressed. Consumer client libraries receive compressed data, decompresses it, and passes it to client applications. Valid options are `off`, `lzf`, `lz4`, and `zlib`. The default setting is the type of compression that is set for the directory in which the stream is located. For more information, see Compression.
`produceperm`	Specifies the access-control expression that controls who can publish messages to topics in the stream. See ACE Syntax.
`consumeperm`	Specifies the access-control expression that controls who can who can listen to topics in the stream. See ACE Syntax.
`topicperm`	Specifies the access-control expression that controls who can create, edit, or remove topics in the stream. See ACE Syntax.
`copyperm`	Specifies the access-control expression that controls who can use `mapr copystream` or `mapr diffstreams` on the stream. See ACE Syntax.
`adminperm`	Determines which users can modify ACEs for a stream, set up replication of a stream, and modify other attributes of a stream. **Important:** By default, only the stream owner can modify this setting; however, a patch is available that changes this behavior. After applying the patch, both the stream owner and the MAPR user can modify this setting. The patch works with MapR Core-6.1.0. To install patches, see Applying a Patch. See ACE Syntax. This permission includes the `topicperm` permission.
`defaulttimestamptype`	Specifies the type of timestamp stored in the topic's message. Value: createtime | logappendtime Default: createtime. The topic inherits the default value from the stream unless the topic sets the timestamp type to a different value. A `createtime` value is the time defined by the user or application (when creating the message). If user or application does not define this value (or passes null), the client uses the current system timestamp. A `logappendtime` value is the time when the message (log) was appended to the server.
`pidexpirysecs`	Specifies the expiration time for the Producer ID. This parameter fixes the lifetime for the Producer ID. Default: 604800
`compact`	Sets log compaction for stream. When set to true, enables log compaction. When set to false, disables log compaction. Value: true|false Default: false **Note:** A license is required to run the `-compact` option; otherwise, the command hangs. See Adding a License on page 777.

Parameter	Description
`mincompactionlag`	Sets the **minimum** delay (in milliseconds) before which messages are **not** compacted. It is the **minimum** time that the messages are available for consumption. Beyond this time period, the messages **may** be compacted. Default: 0 The lag is calculated from the time that a message was produced to the stream topic-partition.
`deleteretention`	Sets the **minimum** time (in milliseconds) before which deleted records are removed. It is the **minimum** time that the deleted records are still available. Beyone this time period, the deleted messages **may** be removed. Default: 86400000
`force`	Used with the `-compact` parameter to force log compaction on a stream parameter. No values are passed. Used for backward compatibility.

`stream info`

Displays the values of the parameters of the specified stream.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the path
`adminperm`	When a user with this permission runs the command, the output includes the access-control expressions for the `adminperm` and `topicperm` permissions.
`produceperm`, `consumeperm`, or `topicperm`	When a user with one of these permissions runs the command, the output does not include any access-control expressions.

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream info -path <Stream Path>`
REST	`http[s]://<host>:<port>/rest/stream/info?path=<path>`

Parameters

Parameter	Description
`path`	The path and name of the stream that you want to see information about.

Sample Output

```
maprcli stream info -path /streamVol/stream1 -json
{
    "timestamp":1521233326943,
    "timeofday":"2018-03-16 01:48:46.943 GMT-0700 PM",
```

```
    "status":"OK",
    "total":1,
    "data":[
        {
            "path":"/streamVol/stream1",
            "physicalsize":57344,
            "logicalsize":32768,
            "numtopics":1,
            "defaultpartitions":1,
            "ttl":604800,
            "compression":"lz4",
            "autocreate":true,
            "produceperm":"u:root",
            "consumeperm":"u:root",
            "topicperm":"u:root",
            "copyperm":"u:root",
            "adminperm":"u:root",
            "ischangelog":false,
            "timestamptype":"createtime"
        }
    ]
}
```

`stream purge`
Runs the purge process, removing messages that are marked for deletion and reclaiming disk space.

For information about the purge process, see Time-to-Live for Messages.

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Permissions Required
To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminperm` permission on the stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream purge -path <Stream Path>`
REST	`http[s]://<host>:<port>/rest/stream/purge?path=<path>`

Parameters

Parameter	Description
`path`	The path and name of the stream to reclaim disk space from.

last-updated: Apr 11, 2022

`stream replica add`

Registers an existing stream as a replica of the specified stream.

Note: A license is required to run this command. Running this command without a license can cause the command to hang. See Adding a License on page 777.

Permissions Required at the Source Cluster

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminperm` and `copyperm` permissions on the source stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Permissions Required at the Target Cluster

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path

Syntax

CLI	`maprcli stream replica add -path <stream path> -replica <remote stream path> [-paused <start replication in paused state> default: false] [-throttle <throttle replication operations under load> default: false] [-networkencryption <enable on-wire encryption> default: false] [-synchronous <replicate to remote stream before acknowledging producers> default: false] [-networkcompression <on-wire compression type: off|lz4|lzf|zlib> default: compression setting on stream]`
REST	`http[s]://<host>:<port>/rest/stream/replica/add?path=<path>&replica=<name>`

Parameters

Parameter	Description
`path`	The path and name of the stream that you want to create a replica for.
`replica`	The path and name of the stream that you want to create as a replica of the stream that you specified with the `-path` parameter.

last-updated: Apr 11, 2022

Parameter	Description
`paused`	A boolean value that specifies whether to pause the replication so that it does not start immediately. The replication can be resumed using the replica resume command at a later time. The values are `true` or `false`. The default is `false`. Set `-paused` to `true` if you want to run `mapr copystream` to load the replica stream before starting replication. If it is not paused, replication starts immediately after you run the commands `maprcli stream replica add` and `maprcli stream upstream add`, in which case the replica stream starts empty and accumulates messages over time. If you are interested only in the messages that are published to the source stream after replication starts, then you do not need to pause replication initially. However, if you want the full set of messages from the source stream that have not yet been purged or marked for deletion, then pause replication initially.
`throttle`	A boolean value that specifies whether to throttle replication operations. Throttle the replication stream to minimize the impact of the replication process on incoming operations during periods of heavy load. The values are `true` or `false`. The default is `false`. Throttling has two effects, both of which allow MapR Event Store For Apache Kafka to use more system resources to process new messages: • Throttling slows down the rate at which changes to a stream are replicated. • Throttling slows down the rate at which messages to be replicated are read from disk.
`networkencryption`	A boolean value that specifies whether or not to enable on-wire encryption. The values are `true` or `false`. The default is `false`. If you set the value to `true`, the local cluster and any other cluster that is part of the replication process must be enabled for security.
`synchronous`	A boolean value that specifies whether replication is synchronous or asynchronous. The values are `true` or `false`. The default is `false` and specifies asynchronous replication.
`networkcompression`	Specifies the type of compression to use when replicating messages. For more information, see Managing Compression.

`stream replica autosetup`
Sets up and starts replication between a source stream and replica stream.

The `maprcli stream replica autosetup` command performs the following steps to set up replication:

1. Creates a stream in the destination cluster.
2. Declares the new stream to be a replica of the source stream and ensures that replication does not begin immediately after the next step.
3. Declares the source stream as the original of the replica stream.

last-updated: Apr 11, 2022

4. Runs the `mapr copystream` utility to load a copy of the source data into the replica.
5. For multi-master replication, it declares the source stream to be a replica of the new stream and then declares the new stream to be an upstream source for the source stream.
6. Clears the paused replication state to start replication.

For more information about the automatic setup process, see Replica Autosetup for Streams on page 659.

Note: Before you set up replication for a stream, verify that the cluster is setup for replication. For more information, see Preparing Clusters for Stream Replication on page 1130.

Permissions Required at the Source Cluster
To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminperm` and `copyperm` permissions on the source stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Permissions Required at the Target Cluster
To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path

Syntax

CLI	`maprcli stream replica autosetup -path <stream path> -replica <remote stream path> [-synchronous <replicate to remote stream before acknowledging producers> default: false] [-multimaster <set up bi-directional replication> default: false] [-throttle <throttle replication operations under load> default: false] [-networkencryption <enable on-wire encryption> default: false] [-networkcompression <on-wire compression type: off|lz4|lzf|zlib> default: compression setting on stream] [-directcopy enable directcopy. default: true] [-useexistingreplica use existing replica table if present. default: false]`
REST	`http[s]://<host>:<port>/rest/stream/replica/autosetup?path=<path>&replica=<name>`

Parameters

Parameter	Description
`path`	The path and name of the stream that you want to create a replica for.
`replica`	The path and name of the stream that you want to create as a replica of the stream that you specified with the `-path` parameter.
`synchronous`	A boolean value that specifies whether replication is synchronous or asynchronous. The values are `true` or `false`. The default is `false` and specifies asynchronous replication.
`multimaster`	A boolean value that specifies whether or not to set up a multi-master topology. The values are `true` or `false`. The default is `false` and specifies to use the basic primary-secondary topology, rather than the multi-master topology.
`throttle`	A boolean value that specifies whether to throttle replication operations. Throttle the replication stream to minimize the impact of the replication process on incoming operations during periods of heavy load. The values are `true` or `false`. The default is `false`. Throttling has two effects, both of which allow MapR Event Store For Apache Kafka to use more system resources to process new messages: • Throttling slows down the rate at which changes to a stream are replicated. • Throttling slows down the rate at which messages to be replicated are read from disk.
`networkencryption`	A boolean value that specifies whether or not to enable on-wire encryption. The values are `true` or `false`. The default is `false`. If you set the value to `true`, the local cluster and any other cluster that is part of the replication process must be enabled for security.
`networkcompression`	Specifies the type of compression to use when replicating messages. For more information, see Managing Compression.
directcopy	A Boolean value that specifies whether or not autosetup will use the directcopy option . The values are `true` or `false.` Autosetup with direct copy (`true`) is the default. If you set this parameter to `false`, the cluster will run autosetup without the directcopy option. For more information, see Replica Autosetup for Streams on page 659.
useexistingreplica	When the directcopy parameter is set to `true` (default), this Boolean value specifies whether or not an existing stream can be used as the replica stream. The values for this parameter are `true` or `false`. No reuse of existing tables (`false`) is the default. If a stream exists with the specified name, and this parameter is set to false, the create stream operation will fail.

`stream replica edit`

Modifies the way in which messages are replicated from one stream to another.

last-updated: Apr 11, 2022

Permissions Required at the Source Cluster
To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminperm` permission on the source stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Permissions Required at the Target Cluster
To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path

Syntax

CLI	`maprcli stream replica edit`
	`-path <stream path>`
	`-replica <remote stream path>`
	`[ -newreplica <renamed stream path> ]`
	`[ -throttle <throttle replication operations under load> ]`
	`[ -networkencryption <enable on-wire encryption> ]`
	`[ -synchronous <replicate to remote stream before acknowledging producers> ]`
	`[ -networkcompression <on-wire compression type: off|lz4|lzf|zlib> ]`

REST	`http[s]://<host>:<port>/rest/stream/replica/edit?path=<path>&replica=<name>`

Parameters

Parameter	Description
`path`	The path and name of the stream that you want to create a replica for.
`replica`	The path and name of the stream that you want to create as a replica of the stream that you specified with the `-path` parameter.
`newreplica`	Specifies a new name to give to the replica stream.

Parameter	Description
`throttle`	A boolean value that specifies whether to throttle replication operations. Throttle the replication stream to minimize the impact of the replication process on incoming operations during periods of heavy load. The values are `true` or `false`. The default is `false`. Throttling has two effects, both of which allow MapR Event Store For Apache Kafka to use more system resources to process new messages: • Throttling slows down the rate at which changes to a stream are replicated. • Throttling slows down the rate at which messages to be replicated are read from disk.
`networkencryption`	A boolean value that specifies whether or not to enable on-wire encryption. The values are `true` or `false`. The default is `false`. If you set the value to `true`, the local cluster and any other cluster that is part of the replication process must be enabled for security.
`synchronous`	A boolean value that specifies whether replication is synchronous or asynchronous. The values are `true` or `false`. The default is `false` and specifies asynchronous replication.
`networkcompression`	Specifies the type of compression to use when replicating messages. For more information, see Managing Compression.

`stream replica list`
Lists the replicas of a given stream.

Permissions Required on the Source Cluster
To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the path
- `adminperm` permission on the source stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream replica list`
	`-path <stream path>`
	`[ -refreshnow <immediately refresh replication statistics>`
	`default: false ]`

REST	`http[s]://<host>:<port>/rest/stream/replica/list?path=<path>`

last-updated: Apr 11, 2022

Parameters

Parameter	Description
`path`	The path and name of the stream that you want to list the replicas of.
`refreshnow`	A boolean value that specifies whether to trigger an immediate update of the replica statistics. The values are `true` or `false`. By default, the value is `false` and the command lists the current version of the replica statistics, which could be a maximum of five minutes old.

Sample with Output

```
# maprcli stream replica list -path /srcVol/srcStream -json
{
    "timestamp":1507758209755,
    "timeofday":"2017-10-11 02:43:29.755 GMT-0700",
    "status":"OK",
    "total":1,
    "data":[
        {
            "cluster":"my.cluster.com",
            "stream":"/destVol",
            "type":"MapRStream",
            "replicaPath":"/destVol",
            "replicaState":"REPLICA_STATE_CREATE_SCHEDULE",
            "paused":false,
            "throttle":false,
            "idx":1,
            "networkencryption":false,
            "synchronous":false,
            "networkcompression":"lz4",
            "propagateExistingData":false,
            "isUptodate":true,
            "minPendingTS":0,
            "maxPendingTS":0,
            "bytesPending":0,
            "bucketsPending":0,
            "copyTableCompletionPercentage":0,
        }
    ]
}
```

Data Fields

Data Fields	Description
cluster	The cluster on which the replica stream resides.
stream	The path of the replica stream.
type	Identifies the type of table: MapR Database table or MapR Event Store For Apache Kafka stream.
replicaPath	The replica location of the source stream.
replicaState	The replication state indicates when stream replication is in progress and it also displays the status of operations related to replica autosetup with directcopy.
paused	A Boolean values that specifies if replication is paused.
throttle	A Boolean value that specifies if replication is throttled.

Data Fields	Description
idx	The index number of the replica stream.
networkencryption	A Boolean value that specifies if replication is encrypted.
synchronous	A Boolean value that specifies whether replication is synchronous or asynchronous.
networkcompression	The type of on-wire compression.
propagateExistingData	Used to identify whether existing data in a CDC table is propagated to stream topic.
isUptodate	A Boolean value that specifies if the replica is up-to-date.
minPendingTS	The epoch time in milliseconds of the oldest message that has yet to be replicated.
maxPendingTS	The epoch time in milliseconds of the newest message that has yet to be replicated.
bytesPending	The number of bytes that have yet to be replicated.
bucketsPending	The number of buckets that have yet to be replicated.
copyTableCompletionPercentage	The percentage of data replication completed from the source stream to the destination stream. **Note:** When replicating MapR Database data, the copyTablePercentageCompletion data may re-adjust to a lower rate. This depends on table region (also referred to as tablets) splits and merges as well as the rate of incoming data to replicating data.

`stream replica pause`

Pauses replication from a *source* stream to a *replica* stream during autosetup and replication phases.

Permissions Required on the Source Cluster

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminperm` permission on the source stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream replica pause -path <stream path> -replica <remote stream path>`
REST	`http[s]://<host>:<port>/rest/stream/replica/pause?path=<path>&replica=<name>`

last-updated: Apr 11, 2022

Parameters

Parameter	Description
`path`	The path and name of the stream that is the source for the replica that you want to pause replication to.
`replica`	The path and name of the stream replica that you want to pause replication to.

`stream replica remove`

Unregisters a stream as the replica of another stream.

Permissions Required on the Source Cluster

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminperm` permission on the source stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream replica remove -path <stream path> -replica <remote stream path>`
REST	`http[s]://<host>:<port>/rest/stream/replica/remove?path=<path>&replica=<name>`

Parameters

Parameter	Description
`path`	The path and name of the stream that is the source for the replica that you want to remove.
`replica`	The path and name of the stream replica that you want to remove.

`stream replica resume`

Resumes replication from one stream to another stream Replication can be paused during autosetup and replication phases. When replication resumes, it continues from where it left off.

Permissions Required on the Source Cluster

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminperm` permission on the source stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream replica resume -path <stream path> -replica <remote stream path>`
REST	`http[s]://<host>:<port>/rest/stream/replica/resume?path=<path>&replica=<name>`

Parameters

Parameter	Description
`path`	The path and name of the stream that is the source for the replica that you want to resume replicating to.
`replica`	The path and name of the stream replica that you want to resume replicating to.

`stream upstream add`

Registers a stream as an upstream source for a given stream. For example, if you wanted to replicate messages from `Stream_A` to `Stream_B`, `Stream_A` would be the upstream source for `Stream_B`.

Permissions Required on the Target Cluster

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminperm` permission on the source stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream upstream add -path <stream path> -upstream <upstream stream path>`
REST	`http[s]://<host>:<port>/rest/stream/upstream/add?path=<path>&upstream=<name>`

Parameters

Parameter	Description
`path`	The path and name of the stream that you want to specify a source stream for.

last-updated: Apr 11, 2022

Parameter	Description
`upstream`	The path and name of the stream that you want to use as a source for the stream that you specified with the `-path` parameter.

`stream upstream list`
Lists all of the streams that are replicating to a given stream.

Permissions Required on the Target Cluster
To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the path
- `adminperm` permission on the source stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream upstream list -path <Stream Path>`
REST	`http[s]://<host>:<port>/rest/stream/upstream/list?path=<path>`

Parameters

Parameter	Description
`path`	The path and name of the stream that you want to list the source streams for.

Sample Output

```
# maprcli stream upstream list -path /dst -json
{
    "timestamp":1437992841303,
    "timeofday":"2015-07-27 03:27:21.303 GMT-0700",
    "status":"OK",
    "total":1,
    "data":[
        {
            "cluster":"my.cluster.com",
            "stream":"/src",
            "idx":1,
            "uuid":"3e98ee93-d88a-f3d6-bc80-001b02b65500"
        }
    ]
}
```


last-updated: Apr 11, 2022

Field Descriptions

`cluster`	The name of the MapR cluster in which the upstream stream is located.
`stream`	The name of the upstream stream.
`idx`	The index number of the upstream stream.
`uuid`	The upstream stream's universally unique identifier.

`stream upstream remove`
Unregisters a stream as an upstream source for a given stream.

Permissions Required on the Target Cluster
To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminperm` permission on the source stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream upstream remove -path <stream path> -upstream <upstream stream path>`
REST	`http[s]://<host>:<port>/rest/stream/upstream/remove?path=<path>&upstream=<name>`

Parameters

Parameter	Description
`path`	The path and name of the stream that you want to remove a source stream from.
`upstream`	The path and name of the stream that you want to remove as a source for the stream that you specified with the `-path` parameter.

`stream topic create`
Creates a topic in the specified stream.

Permissions Required
To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `topicperm` permission on the stream

last-updated: Apr 11, 2022

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream topic create -path <Stream Path> -topic <Topic Name> [ -partitions <Number of partitions> default: attribute defaultpartitions on the stream ] [ -timestamptype Timestamp type: createtime | logappendtime default: createtime ]`
REST	`http[s]://<host>:<port>/rest/stream/topic/create?path=<path>&topic=<name>`

Parameters

Parameter	Description
`path`	The path and name of the stream in which to create the topic.
`topic`	The name of the topic to create. A name can include alphanumeric characters and the period, underscore, and dash characters.
`partitions`	The number of partitions to use for the topic. After you create the topic, you can increase the number of partitions, but you cannot reduce the number. The default number of partitions for new topics is set by the defaultpartitions parameter in the commands maprcli stream create and maprcli stream edit. **Important:** A CDC changelog stream's default partitions can impact how many partitions a stream topic can have. This is because once you create a stream topic for a changelog stream, the number of topic partitions is *locked*. The number of topic partitions cannot change. • If the `stream topic create` command is used to create a stream topic, then the number of topic partitions can be set at creation time and then is *locked*. • If the table changelog add on page 1813 command is used to add a stream topic (as well as establish a relationship between the source table and the changelog stream), then the number of topic partitions is inherited from the changelog stream and is *locked*.

Parameter	Description
`timestamptype`	Specifies the type of timestamp stored in the topic's message. Value: createtime | logappendtime Default: createtime. The topic inherits the default value from the stream unless the topic sets the timestamp type to a different value. A `createtime` value is the time defined by the user or application (when creating the message). If user or application does not define this value (or passes null), the client uses the current system timestamp. A `logappendtime` value is the time when the message (log) was appended to the server.

`stream topic delete`

Deletes the specified topic from the specified stream.

Consumers do not have to stop consuming from a topic before the topic is deleted.

The deletion of the topic and the messages is immediate. However, the command also starts a background process for the purging the topic and messages to reclaim disk space.

If the parameter `-autocreate` for the stream is set to `true`, a topic with the same name is created if a producer writes a message to a topic of the same name. For example, if you delete the topic `Topic_A` and then a producer writes a message to the topic `Topic_A`, MapR Event Store For Apache Kafka creates a topic that is named `Topic_A`. Aside from the name, the new topic `Topic_A` shares nothing with the deleted topic `Topic_A`.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `topicperm` permission on the stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream topic delete`
	`-path <Stream Path>`
	`-topic <Topic Name>`

REST	`http[s]://<host>:<port>/rest/stream/topic/delete?path=<path>&topic=<name>`

Parameters

Parameter	Description
`path`	The path and name of the stream from which to delete the topic.
`topic`	The name of the topic to delete.

last-updated: Apr 11, 2022

`stream topic edit`

Allows you to increase the number of partitions that are in the specified topic.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `topicperm` permission on the stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream topic edit -path <Stream Path> -topic <Topic Name> [ -partitions <Number of partitions> ] [ -timestamptype Timestamp type: createtime | logappendtime default: createtime ]`
REST	`http[s]://<host>:<port>/rest/stream/topic/edit?path=<path>&topic=<name>&partitions=<number>`

Parameters

Parameter	Description
`path`	The path and name of the stream in which the topic is located.
`topic`	The name of the topic to edit.

Parameter	Description
`partitions`	The number of partitions to use for the topic. You cannot reduce the number of partitions. To find out how many partitions a topic is currently using, run the command `maprcli stream topic info.` **Important:** A CDC changelog stream's default partitions can impact how many partitions a stream topic can have. This is because once you create a stream topic for a changelog stream, the number of topic partitions is *locked*. The number of topic partitions cannot change and the `stream topic edit` command can not modify the topic's partition number. • If the `stream topic create` command is used to create a stream topic, then the number of topic partitions can be set at creation time and then is *locked*. • If the table changelog add on page 1813 command is used to add a stream topic (as well as establish a relationship between the source table and the changelog stream), then the number of topic partitions is inherited from the changelog stream and is *locked*.
`timestamptype`	Specifies the type of timestamp stored in the topic's message. Value: createtime | logappendtime Default: createtime. The topic inherits the default value from the stream unless the topic sets the timestamp type to a different value. A `createtime` value is the time defined by the user or application (when creating the message). If user or application does not define this value (or passes null), the client uses the current system timestamp. A `logappendtime` value is the time when the message (log) was appended to the server.

`stream topic info`
Lists information about a stream's topic, grouped by partition ID.

Permissions Required
To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the path
- `adminperm`, `consumeperm`, `produceperm`, or `topicperm` permission on the stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

last-updated: Apr 11, 2022

Syntax

CLI	`maprcli stream topic info -path <Stream Path> -topic <Topic Name>`
REST	`http[s]://<host>:<port>/rest/stream/topic/info?path=<path>&topic=<name>`

Parameters

Parameter	Description
`path`	The path and name of the stream for which you want to display information about topics.
`topic`	The name of the topic for which you want to display information.

Sample Output

```
maprcli stream topic info -path /streamVol/stream1 -topic topic1 -json
{
    "timestamp":1521232252550,
    "timeofday":"2018-03-16 01:30:52.550 GMT-0700 PM",
    "status":"OK",
    "total":1,
    "data":[
        {
            "partitionid":0,
            "physicalsize":0,
            "logicalsize":0,
            "maxoffset":-1,
            "minoffsetacrossconsumers":0,
            "mintimestamp":"1969-12-31T04:00:00.000-0800 PM",
            "maxtimestamp":"1969-12-31T04:00:00.000-0800 PM",
            "mintimestampacrossconsumers":"1969-12-31T04:00:00.000-0800 PM",
            "fid":"2113.32.131400",
            "master":"doc24.lab:5660",
            "servers":"doc24.lab:5660",
            "timestamptype":"createtime",

"logcompactionlaststarted":"1969-12-31T04:00:00.000-0800 PM",

"logcompactionlastcompleted":"1969-12-31T04:00:00.000-0800 PM",
                    "logcompactionstatus":"not started"
        }
    ]
}
```

Field Descriptions

`partitionid`	The index number of the partition within the topic. The first partition in a topic has an index of 0, the next partition an index of 1, and so on.
`physicalsize`	The physical size (in bytes) of the stream topic with data compression.
`logicalsize`	The logical size (in bytes) of the stream topic without data compression.

maxoffset	The maximum offset for this partition.
minoffsetacrossconsumers	All known consumers for this partition have consumed messages at least up to this offset.
mintimestamp	The timestamp of oldest message in the partition.
maxtimestamp	The timestamp of newest message in the partition.
mintimestampacrossconsumers	All known consumers for this partition have consumed messages older than this timestamp.
fid	The inode hosting the head of the partition.
master	*For use by MapR support:* The master server that is hosting the head of the partition.
servers	*For use by MapR support:* Lists all of the servers that are hosting the head of the partition.
timestamptype	The type of timestamp stored in the topic's message. Possible values: createtime (default) and logappendtime.
logcompactionlaststarted	Displays the last time log compaction was started if log compaction was enabled. The value is displayed in epoch time. This field displays when there is a change in the value for this field.
logcompactionlastcompleted	Displays the last time log compaction completed if log compaction was enabled. The value is displayed in epoch time. This field displays when there is a change in the value for this field
logcompactionstatus	Displays whether log compaction was started or completed.

stream topic list
Lists the topics that are in a stream, as well as the number of partitions in each topic.

Permissions Required
To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the path
- `adminperm`, `consumeperm`, `produceperm`, or `topicperm` permission on the stream

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI	`maprcli stream topic list -path <Stream Path>`
REST	`http[s]://<host>:<port>/rest/stream/topic/list?path=<path>`

last-updated: Apr 11, 2022

Parameters

Parameter	Description
`path`	The path and name of the stream for which you want to list the topics.

Sample Output

```
# maprcli stream topic list -path /s1 -json
{
    "timestamp":1441882201851,
    "timeofday":"2015-09-10 03:50:01.851 GMT-0700",
    "status":"OK",
    "total":2,
    "data":[
        {
            "topic":"topic0",
            "partitions":4,
            "consumers":8,
            "physicalsize":148373504,
            "logicalsize":1009713152,
            "maxlag":6314
        },
        {
            "topic":"topic1",
            "partitions":4,
            "consumers":8,
            "physicalsize":148373504,
            "logicalsize":1009713152,
            "maxlag":6385
        }
    ]
}
```

`table`
Performs functions related to MapR Database tables.
`table create`
Creates a MapR binary or JSON table.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
/opt/mapr/bin/maprcli table create
  -path <path>
  [ -copymetafrom <path to source
table> ]
  [ -copymetatype all|cfs|aces|splits|
attrs ]
  [ -regionsizemb <region size in MB> ]
  [ -autosplit true|false ]
  [ -bulkload true|false ]
  [ -audit true|false ]
  [ -tabletype <Table Type - json or
binary> default: binary ]
  [ -packperm <Pack Permission
settings> ]
  [ -bulkloadperm <Bulk load Permission
settings> ]
  [ -splitmergeperm <Split and Merge
Permission settings> ]
  [ -createrenamefamilyperm <Add/Rename
Family Permission settings>]
  [ -deletefamilyperm <Delete Family
Permission settings> ]
  [ -adminaccessperm <ACE Admin
Permission settings> ]
  [ -replperm <Replication Admin
Permission settings> ]
  [ -indexperm <ACE Admin Permission
settings> ]
  [ -defaultversionperm <CF Versions
Default Permission setting>]
  [ -defaultcompressionperm <CF
Compression Default Permission
setting> ]
  [ -defaultmemoryperm <CF Memory
Default Permission setting>]
  [ -defaultreadperm <CF Read Default
Permission setting> ]
  [ -defaultwriteperm <CF Write Default
Permission setting> ]
  [ -defaulttraverseperm CF Traverse
Default Permission ]
  [ -defaultappendperm <CF Append
Default Permission setting>]

  [ -metricsinterval <metric interval
setting> ]
```

REST

```
curl -k -X POST
  'http[s]://<host>:<port>/rest/table/
create?path=<path>&<parameters>'
  -u <username>:<password>
```


last-updated: Apr 11, 2022

Parameters

Parameter	Description

path	The path to the new MapR table.
	• For a path on the local cluster, start the path at the volume mount point. For example, for a table named `test` under a volume with a mount point at /volume1, specify the following path: `/volume1/test`
	• For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customer` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`
	  **Note:** You cannot use the following characters in the table name:
	`< > ? % \`
	To use the following characters in the table name, enclose them either in single or double quotes:
	`; | ( ) /`
	For example:
	`maprcli table create -path "/♦=#; {}&()/" (or) maprcli table create -path '/♦=#; {}&()/'`
	To use either the **'** or the **"** character in the table name, enclose:
	• the **'** character within double quotes (")
	• the **"** character within single quote (')
	For example:
	`maprcli table create -path "/'♦=#; {}&()/" (or) maprcli table create -path '/"♦=#; {}&()/'`

Parameter	Description
copymetafrom	The path to a table that contains the metadata that should be used to create the table. • For a path on the local cluster, start the path at the volume mount point. For example, if you want to copymetadata from a table named test under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, if you want to copymetadata from a table named test under volume1 in the sanfrancisco cluster, specify the following path: `/mapr/sanfrancisco/volume1/customer`
copymetatype	The type of metadata to copy from the table identified in the `copymetafrom` parameter. You can specify one or more of the following options in a comma separated list: • `all`. Copy all metadata. This is the default. • `cfs`. Copy column family metadata. • `aces`. Copy ACE permissions. • `splits`. Copy split keys. • `attrs`. Copy table attributes.
regionsizemb	The average size of the regions into which MapR tries to split the table as the table grows. The default is 4096 MB. This value is ignored if `autosplit` is set to `false`. If `autosplit` is set to `true`, MapR splits a region when the size of the region exceeds 150% of the average value. For example, if the average value is 4096 MB, MapR splits a region that is larger than 6144 MB. Although splits are automatic, merges are not. For example, if the value of `regionsizemb` is changed from 8 GB to 4 GB, all regions that are eligible are split automatically, if `autosplit` is set to `true`. However, if the value of `regionsizemb` is changed from 2 GB to 4 GB, regions smaller than 4 GB are not automatically merged. **Note:** When a table has less than 4 regions, MapR ignores the `regionsizemb` parameter and splits regions at a lower threshold.
autosplit	A Boolean value that specifies whether to split the table into regions automatically as the table grows. The average size of each region is determined by the `regionsizemb` parameter. The default value is `true`. If you set the value to `false`, you can manually split tables into regions by using the `table region split` command.

last-updated: Apr 11, 2022

Parameter	Description
bulkload	A Boolean value that specifies whether to allow a full bulk load of the table. The default is `false`. For more information, see Loading Data into Binary Tables on page 1040 and Loading Documents into JSON Tables on page 1036.
audit	Specifies whether to turn auditing on for the table. If auditing is also enabled at the cluster level with the `maprcli audit data` command and enabled for the current volume, setting this value to true causes auditing to start for the table.
tabletype	Specifies whether the table will be a binary table or a JSON table. The values are `binary` and `json`. The default is `binary`.
packperm	The *ACE* that controls who can pack table regions. By default, permission is given to the user ID that is used to create the table.
bulkloadperm	The *ACE* that controls who can load this table with bulk loads if the table was created with bulk load support. By default, permission is given to the user ID that is used to create the table.
splitmergeperm	The *ACE* that controls who can take the following actions: • Run the `table region split` and `table region merge` commands to split the table into regions or to merge regions of the table together. • Change the value of `regionsizemb`. By default, permission is given to the user ID that is used to create the table.
createrenamefamilyperm	The *ACE* that controls who can create column families for this table or rename existing column families. By default, permission is given to the user ID that is used to create the table.
deletefamilyperm	The *ACE* that defines access to delete column families for this table. Delimit the expression with single-quotation marks. By default, permission is given to the user ID that is used to create the table.
adminaccessperm	The *ACE* that controls who can view and edit the permissions for this table. By default, permission is given to the user ID that is used to create the table.
replperm	The *ACE* that controls who can set up replication either to or from a table. By default, permission is given to the user ID that is used to create the table
indexperm	The secondary index Admin permissions setting that controls who can create an index associated with this table. By default, permission is given to the user ID that is used to create the table.

Parameter	Description
defaultversionperm	The default *ACE* for the version permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `versionperm` in the `table cf create` and `table cf edit` commands overrides this value. **Note:** This permission is not applicable to JSON tables. Versioning is not supported for JSON documents.
defaultcompressionperm	**Applies to binary tables only:** The default *ACE* for the compression permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `compressionperm` in the `table cf create` and `table cf edit` commands overrides this value.
defaultmemoryperm	The default *ACE* for the memory permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `memoryperm` in the `table cf create` and `table cf edit` commands overrides this value.
defaultreadperm	The default *ACE* for the read permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `readperm` in the `table cf create` and `table cf edit` commands overrides this value. See table cf create on page 1799 and table cf edit on page 1806
defaultwriteperm	The default *ACE* for the write permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `writeperm` in the `table cf create` and `table cf edit` commands overrides this value. See table cf create on page 1799 and table cf edit on page 1806
defaulttraverseperm	**Applies to JSON tables only:** The default Access Control Expression for the traverse permission on new column families. For more information about this permission, see Permission Types for Fields and Column Families in JSON Tables on page 1462.
defaultappendperm	**Applies to binary tables only:** The default *ACE* for the append permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `appendperm` in the `table cf create` and `table cf edit` commands overrides this value.

Parameter	Description
metricsinterval	The metrics collection interval, in seconds, for the table. Possible values: 10, 60, 600 Default: 60 seconds When configured to 10 seconds, under normal workloads, the metrics are available in OpenTSDB in about 30 seconds. At an interval of 60 seconds, the metrics are available in about 90 seconds. **Note:** You cannot disable metrics collection for a table by setting the interval to 0.

Example

Creates a MapR table named `newtable` in `volume1`:

CLI	`/opt/mapr/bin/maprcli table create -path /volume1/newtable`
REST	`curl -k -X POST \ 'https://r1n1.sj.us:8443/rest/table/create?path=%2Fvolume1%2Fnewtable' \ -u mapr:mapr`

`table cf`
Manages column families for MapR Database tables.
`table cf colperm get`
Lists the Access Control Expressions (ACEs) for a specified column.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the path
- `adminaccessperm` on the table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table cf colperm get
  -path <path>
  -cfname <column-family name>
  [ -name <column name> ]
  [ -json | -long ]
```


last-updated: Apr 11, 2022

REST

```
curl -k -X GET
  'http[s]://<host>:<port>/rest/
table/cf/colperm/get?
path=<path>&cfname=<name>&name=<name>'

  -u <username>:<password>
```

Parameters

Parameter	Description
path	The path to the table. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named test under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named test under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`
cfname	The name of the column family in which the column is located.
name	The name of the column that you want to list the ACEs for. If you do not specify the column name, the ACEs for all of the columns in the family are listed.
json	This command returns multiple levels of data. You must specify to display the output either in JSON or the "long" format to see the full set of information.
long	This command returns multiple levels of data. You must specify to display the output either in JSON or the "long" format to see the full set of information.

Example

Lists ACEs for column `col1` in table `mytable` and column family `cf1`:

CLI

```
maprcli table cf colperm get -path /
mytable -cfname cf1 -name col1 -long
```

REST

```
curl -k -X GET \
  'https://r1n1.sj.us:8443/
rest/table/cf/colperm/get?
path=%2Fmytable&cfname=cf1&name=col1'
\
  -u mapr:mapr
```

`table cf colperm set`
Sets access control expressions (ACEs) for a specified column.

last-updated: Apr 11, 2022

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminaccessperm` on the table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
/opt/mapr/bin/maprcli table cf
colperm set
  -path <path>
  -cfname <column-family name>
  -name <column name>
  [ -appendperm <Access Control
Expression for column appends> ]
  [ -readperm <Access Control
Expression for column reads> ]
  [ -writeperm <Access Control
Expression for column writes> ]
  [ -traverseperm <Access Control
Expression for column traversals in
JSON tables> ]
```

REST

```
curl -k -X POST
  'http[s]://<host>:<port>/rest/
table/cf/colperm/set?
path=<path>&cfname=<name>&name=<name>&
<parameters>'
  -u <username>:<password>
```

Parameters

Parameter	Description
path	The path to the table. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named test under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named test under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`

last-updated: Apr 11, 2022

Parameter	Description
cfname	The name of the column family in which the column is located.
name	**For binary tables:** The name of the column for which you want to set the *ACE*. **For JSON tables:** The fieldpath of the field on which you want to set permissions. For example, if you wanted to grant `readperm` to a user on field `b` in the following document. the fieldpath would be `a.b`. `{ "a" : { "b" : "value_b" } }`
appendperm	**Applies to binary tables only:** The *ACE* for column appends. Use single quotation marks around the *ACE*. Column appends require permission both at the column-family level and at the column level.
readperm	The *ACE* for column reads. Use single quotation marks around the *ACE*. Reads require permission both at the column-family level and at the column level (for binary tables) or field level (for JSON tables). In JSON tables, this permission is inherited by fields within the column family.
writeperm	The *ACE* for column writes (puts and deletes). Use single quotation marks around the *ACE*. Writes require permission both at the column-family level and at the column level (for binary tables) or field level (for JSON tables). In JSON tables, this permission is inherited by fields within the column family.
traverseperm	**Applies to JSON tables only:** The Access Control Expressions that specifies who has permission to pass over fields in JSON documents. For example, suppose that a JSON table contains documents of this general structure: `{ "_id" : "ID", "a" : { "b" : "value", "c" : "value" } }` Suppose further that the user `sjohnson` has read permission on `a.b`, but not on `a`. For `sjohnson` to read `a.b`, the user needs the traverse permission on `a`. The user can then pass over field `a` to `a.b`. This permission is inherited by fields within the column family. By default, this permission is given to the value of defaulttraverseperm for the JSON table.

last-updated: Apr 11, 2022

Example

Sets `readperm` ACE for column `col1` in table `mytable` and column family `cf1` :

CLI

```
/opt/mapr/bin/maprcli table cf
colperm set -path /mytable -cfname
cf1 -name col1 -readperm 'g:group1'
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/
table/cf/colperm/set?
path=%2Fmytable&cfname=cf1&name=col1&r
eadperm="g:group1"' \
  -u mapr:mapr
```

`table cf colperm delete`
Deletes the Access Control Expressions (ACEs) for a specified column. Deletion cannot be undone.

Note: When a user, group, or role requests to read data from, write data to, or append data to a column, MapR Database checks whether that user, group, or role has read or write permission for the column family AND read or write permission for the column. For example, suppose user `i_montoya` tries to write data to columns `col1` and `col2` in column family `cf1`. MapR Database checks whether `i_montoya` has write permission on `cf1` AND `col1` AND `col2`. If `i_montoya` does not have all three permissions, MapR Database returns an error that says access for the write is denied.

If this user were to try to read from the same two columns, MapR Database would simply not return the data. If the user tried to read from those two columns and additional columns on which he had read permissions, the results would contain the data for those additional columns but exclude the data for `col1` and `col2`.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminaccessperm` on the table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table cf colperm delete
  -path <path>
  -cfname <column-family name>
  -name <column name>
```

REST

```
curl -k -X POST
  'http[s]://<host>:<port>/
```

```
rest/table/cf/colperm/delete?
path=<path>&cfname=<name>&name=<name>'
  -u <username>:<password>
```

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Parameters

Parameter	Description
path	The path to the table. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named test under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named test under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`
cfname	The name of the column family in which the column is located.
name	The name of the column that you want to delete the ACEs for.

Example

Deletes ACEs for column `col1` in table `mytable` and column family `cf1`:

CLI

```
maprcli table cf
colperm delete -path /mytable -cfname
cf1 -name col1
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/
rest/table/cf/colperm/delete?
path=%2Fmytable&cfname=cf1&name=col1'
\
  -u mapr:mapr
```

`table cf create`
Creates a column family for a HPE Ezmeral Data Fabric binary or JSON table.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path

last-updated: Apr 11, 2022

- `createrenamefamilyperm` on the table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
/opt/mapr/bin/maprcli table cf create
  -path <Table path >
  -cfname <Column family name >
  [ -minversions <Min versions to
keep> Default: 0 ]
  [ -maxversions <Max versions to
keep> Default: 1 ]
  [ -ttl <Time to live> Enter 0 for
forever, otherwise, enter time in
seconds. Default: 0 ]
  [ -inmemory <In-memory> Default:
false ]
  [ -compression <off|lzf|lz4|zlib>
Default: table's compression setting
is applied. ]
  [ -versionperm <Version
Permissions> ]
  [ -compressionperm <Compression
Permissions> ]
  [ -memoryperm <Memory Permissions> ]
  [ -readperm <Read Permissions> ]
  [ -writeperm <Write Permissions> ]
  [ -appendperm <Append
Permissions> ]

  [ -jsonpath Json <Family Path -
needed for JSON column family, like
a.b.c> ]
  [ -force <Force create non-default
column family for json tabletype>
Default: false ]
  [ -traverseperm <Traverse
Permissions> ]
```

REST

```
curl -k -X POST
  'http[s]://<host>:<port>/rest/
table/cf/create?
path=<path>&cfname=<name>&<parameters>
'
  -u <username>:<password>
```

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

last-updated: Apr 11, 2022

Parameters

Parameter	Description
path	The path to the table. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named test under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named test under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`
cfname	The name of the column family to create.
minversions	**Applies to binary tables only:** Minimum number of versions of column values to keep. The default is zero.
maxversions	**Applies to binary tables only:** Maximum number of versions of column values to keep. The default is one.
ttl	Time to live in seconds. When the age of the data in this column family exceeds the value of the `ttl` parameter, the data is purged. Setting the value of `ttl` to 0 is equivalent to allowing data to remain indefinitely. Default: 0  **Note:** If the value of `-ttl` for an existing column family in a JSON table is not 0, you cannot add another column family. You also cannot set the TTL for a JSON table if it has secondary indexes. See Setting TTL for Data.

last-updated: Apr 11, 2022

Parameter	Description
inmemory	Boolean. Determines whether preference is given to values of this column family for storage with row keys. Because row keys are cached in memory in preference to row data, column-family data that is stored inline with the row keys is also cached in memory. For all column families in a table together, up to 200 bytes of row data will be stored inline with each row key. Storing data inline with a row key might speed retrieval of the data from a column family because disk access can often be avoided. For each column family, up to 32 bytes can be stored inline with each row key even if its `inmemory` parameter is set to `false`, but preference will be given to column families where this parameter is set to `true`. A column family can have more than 32 bytes stored inline if its `inmemory` parameter is set to `true`. If the total number of bytes for all column families together exceeds 200 for a row, then preference for inclusion within the inline storage for that row is given to column families that have the `inmemory` parameter set to `true`. **Note:** All of the data for a column family are either stored in-line with the row key, or not stored at all. If the contents in a column family for a particular row are larger than the maximum number of bytes that are allowed to be stored for that column family, then data is not stored in-line for that column family. The default value for the `inmemory` parameter is `false`.
compression	The compression setting to use for the column family. Valid options are `off`, `lzf`, `lz4`, and `zlib`. The default setting is equal to the compression setting for the directory in which the table is located. To find out whether a directory is compressed and the type of compression, see Turning Compression On or Off on Directories Using the CLI on page 990.
versionperm	**Applies to binary tables only:** *ACE* for changing the value of the `maxversions` and `minversions` parameters. By default, permission is given to the value of `defaultversionperm` for the table.
compressionperm	**Applies to binary tables only:** *ACE* for changing the value of the `compression` parameter. By default, permission is given to the value of `defaultcompressionperm` for the table.
memoryperm	The *ACE* for changing the value of the `inmemory` parameter. Use single quotation marks around the *ACE*. By default, permission is given to the value of `defaultmemoryperm` for the table.

Parameter	Description
readperm	The *ACE* for column reads. Use single quotation marks around the *ACE*. Reads require permission both at the column-family level and at the column level (for binary tables) or field level (for JSON tables). In JSON tables, this permission is inherited by fields within the column family. By default, permission is given to the value of `defaultreadperm` for the table.
writeperm	The *ACE* for column writes (puts and deletes). Use single quotation marks around the *ACE*. Writes require permission both at the column-family level and at the column level (for binary tables) or field level (for JSON tables). In JSON tables, this permission is inherited by fields within the column family. By default, permission is given to the value of `defaultwriteperm` for the table.
appendperm	**Applies to binary tables only:** The *ACE* for column appends. Use single quotation marks around the *ACE*. Column appends require permission both at the column-family level and at the column level. By default, permission is given to the value of `defaultappendperm` for the table.
jsonpath	**Applies to JSON tables only:** Specifies the path to the column family. The path is in dotted notation. For example, suppose the table contained JSON documents that were of this general structure: `{ "_id" : "ID", "a" : { "b" : { "c" : "value", }, "e" : "value" } }` You want to create a column family at the field `d` in the new path `a.b.d` because you plan to store image files in fields in that column family. **Important:** Ensure that the field at which you want to create the column family does not yet exist. Also ensure that there are no secondary indexes defined on the field. If the field does exist or is a field in an index, the data in the field could become inaccessible after you create the column family. **Restriction:** As of MapR 6.0, a column family cannot be deleted from a JSON table.

last-updated: Apr 11, 2022

Parameter	Description
force	**Applies to JSON tables only:** By default, every time you try to create a non-default column family in a JSON table, this command fails and returns a warning message that you should ensure there is no existing data at the specified path. Set this parameter to true if you want to override this warning mechanism and create a column family.
traverseperm	**Applies to JSON tables only:** The Access Control Expressions that specifies who has permission to pass over fields in JSON documents. For example, suppose that a JSON table contains documents of this general structure: `{ "_id" : "ID", "a" : { "b" : "value", "c" : "value" } }` Suppose further that the user `sjohnson` has read permission on `a.b`, but not on `a`. For `sjohnson` to read `a.b`, the user needs the traverse permission on `a`. The user can then pass over field `a` to `a.b`. This permission is inherited by fields within the column family. By default, this permission is given to the value of defaulttraverseperm for the JSON table.

Note: If a field is specified as a column family JSON path name, that field cannot be defined as either an indexed or included field when creating an index. For example, suppose you have the following JSON table:

```
{
    "_id" : "ID",
    "a" :
        {
            "b" :
                {
                    "c" : "value",
                    "d" : "value"
                },
            "e" : "value"
        }
}
```

If you created a column family at field `c` in the JSON path `a.b.c`, when creating an index, field `a.b.c` cannot be defined as an indexed or included field. However, you can define, as either an indexed or included field, fields `a`, `a.b`, `a.b.d`.

Example

Creates a new column family `mynewcf` for table `mytable`, keeping four versions in memory:

last-updated: Apr 11, 2022

CLI

```
/opt/mapr/bin/maprcli table cf
create -path /volume1/mytable -cfname
mynewcf \
  -maxversions 4 -inmemory true
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/
table/cf/create?
path=%2Fvolume1%2Fmytable&cfname=mynew
cf&maxversions=4&inmemory=true' \
  -u mapr:mapr
```

`table cf delete`
Deletes a column family from a MapR Database binary table or JSON table. Deletion cannot be undone.

Important: As of MapR 6.0, a column family cannot be deleted from a JSON table.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `deletefamilyperm` on the table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table cf delete
  -path <path>
  -cfname <name>
```

REST

```
curl -k -X POST
  'http[s]://
<host>:<port>/rest/table/cf/delete?
path=<path>&cfname=<name>'
  -u <username>:<password>
```

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

last-updated: Apr 11, 2022

Parameters

Parameter	Description
path	The path to the table. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named test under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named test under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`
cfname	The name of the column family to delete. **Note:** In JSON tables, it is not possible to delete column families in addition to the default column family.

Example

Deletes a column family `mycf` from table `thetable`:

CLI

```
maprcli table cf delete -path /
volume1/thetable -cfname mycf
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/
table/cf/delete?
path=%2Fvolume1%2Fthetable&cfname=mycf
' \
  -u mapr:mapr
```

`table cf edit`
Edits a column family in a binary table or JSON table.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `createrenamefamilyperm` on the table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
/opt/mapr/bin/maprcli table cf edit
  -path <Table path >
  -cfname <Column family name>
  [ -newcfname <New column family
name> ]
  [ -minversions <Min versions to
keep> ]
  [ -maxversions <Max versions to
keep> ]
  [ -ttl <Time to live> Enter 0 for
forever, otherwise, enter time in
seconds. Default: 0 ]
  [ -inmemory <In-memory> ]
  [ -compression <off|lzf|lz4|zlib> ]
  [ -versionperm <Version
Permissions> ]
  [ -compressionperm <Compression
Permissions> ]
  [ -memoryperm <Memory Permissions> ]
  [ -readperm <Read Permissions> ]
  [ -writeperm <Write Permissions> ]
  [ -appendperm <Append Permissions> ]
  [ -traverseperm <Traverse
Permissions> ]
```

REST

```
curl -k -X POST
  'http[s]://<host>:<port>/rest/
table/cf/edit?
path=<path>&cfname=<name>&<parameters>
'
  -u mapr:mapr
```

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Parameters

Parameter	Description
path	The path to the table. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named test under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named test under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`
cfname	The name of the column family to edit.

last-updated: Apr 11, 2022

Parameter	Description
newcfname	The new name to give to the column family.
minversions	**Applies to binary tables only:** Minimum number of versions of column values to keep. The default is zero.
maxversions	**Applies to binary tables only:** Maximum number of versions of column values to keep. The default is one.
ttl	Time to live in seconds. When the age of the data in this column family exceeds the value of the `ttl` parameter, the data is purged. Setting the value of `ttl` to 0 is equivalent to allowing data to remain indefinitely. Default: 0 **Note:** If the value of `-ttl` for an existing column family in a JSON table is not 0, you cannot add another column family. You also cannot set the TTL for a JSON table if it has secondary indexes. See Setting TTL for Data.
inmemory	Boolean. Determines whether preference is given to values of this column family for storage with row keys. Because row keys are cached in memory in preference to row data, column-family data that is stored inline with the row keys is also cached in memory. For all column families in a table together, up to 200 bytes of row data will be stored inline with each row key. Storing data inline with a row key might speed retrieval of the data from a column family because disk access can often be avoided. For each column family, up to 32 bytes can be stored inline with each row key even if its `inmemory` parameter is set to `false`, but preference will be given to column families where this parameter is set to `true`. A column family can have more than 32 bytes stored inline if its `inmemory` parameter is set to `true`. If the total number of bytes for all column families together exceeds 200 for a row, then preference for inclusion within the inline storage for that row is given to column families that have the `inmemory` parameter set to `true`. **Note:** All of the data for a column family will be stored in-line with the row key, or none will be. If the contents in a column family for a particular row are larger than the maximum number of bytes that are allowed to be stored for that column family, no data at all will be stored in-line for that column family. The default value for the `inmemory` parameter is `false`.
compression	The compression setting to use for the column family. Valid options are `off`, `lzf`, `lz4`, and `zlib`. The default setting is equal to the compression setting for the directory in which the table is located. To find out whether a directory is compressed and the type of compression, see Turning Compression On or Off on Directories Using the CLI on page 990.
versionperm	**Applies to binary tables only:** *ACE* for changing the value of the `maxversions` and `minversions` parameters. By default, permission is given to the value of `defaultversionperm` for the table.

last-updated: Apr 11, 2022

Parameter	Description
compressionperm	**Applies to binary tables only:** *ACE* for changing the value of the `compression` parameter. By default, permission is given to the value of `defaultcompressionperm` for the table.
memoryperm	*ACE* for changing the value of the `inmemory` parameter. Use single quotation marks around the *ACE*. By default, permission is given to the value of `defaultmemoryperm` for the table.
readperm	The *ACE* for column reads. Use single quotation marks around the *ACE*. Reads require permission both at the column-family level and at the column level (for binary tables) or field level (for JSON tables). In JSON tables, this permission is inherited by fields within the column family. By default, permission is given to the value of `defaultreadperm` for the table.
writeperm	The *ACE* for column writes (puts and deletes). Use single quotation marks around the *ACE*. Writes require permission both at the column-family level and at the column level (for binary tables) or field level (for JSON tables). In JSON tables, this permission is inherited by fields within the column family. By default, permission is given to the value of `defaultwriteperm` for the table.
appendperm	**Applies to binary tables only:** *ACE* for column appends. Use single quotation marks around the *ACE*. Column appends require permission both at the column-family level and at the column level. By default, permission is given to the value of `defaultappendperm` for the table.
traverseperm	**Applies to JSON tables only:** The Access Control Expressions that specifies who has permission to pass over fields in JSON documents. For example, suppose that a JSON table contains documents of this general structure: `{ "_id" : "ID", "a" : { "b" : "value", "c" : "value" } }` Suppose further that the user `sjohnson` has read permission on `a.b`, but not on `a`. For `sjohnson` to read `a.b`, the user needs the traverse permission on `a`. The user can then pass over field `a` to `a.b`. This permission is inherited by fields within the column family. By default, this permission is given to the value of defaulttraverseperm for the JSON table.

last-updated: Apr 11, 2022

Note: If a field is specified as a column family JSON path name, that field cannot be defined as either an indexed or included field when creating an index. For example, suppose you have the following JSON table:

```
{
     "_id" : "ID",
     "a" :
          {
               "b" :
                   {
                       "c" : "value",
                       "d" : "value"
                   },
               "e" : "value"
          }
}
```

If you created a column family at field c in the JSON path `a.b.c`, when creating an index, field `a.b.c` cannot be defined as an indexed or included field. However, you can define, as either an indexed or included field, fields `a`, `a.b`, `a.b.d`.

Example

Changes the name of a column family in table `mytable` from `mycf` to `mynewcfname`. Also changes the time to live setting.

CLI

```
/opt/mapr/bin/maprcli table
cf edit -path /my.cluster.com/volume1/
mytable -cfname mycf \
  -newcfname mynewcfname -ttl 86400
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/
table/cf/edit?
path=%2Fmy.cluster.com%2Fvolume1%2Fmyt
able&cfname=mycf&newcfname=mynewcfname
&ttl=86400' \
  -u mapr:mapr
```

`table cf list`
Lists the column families for a MapR table.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the path
- `adminaccessperm` on the table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

last-updated: Apr 11, 2022

Syntax

CLI

```
maprcli table cf list
  -path <path>
  [ -cfname <name>]
```

REST

```
curl -k -X GET \
  'http[s]://<host>:<port>/rest/
table/cf/list? \
   path=<path>&cfname=<name>' \
  -u <username>:<password>
```

Parameters

Parameter	Description
path	The path to the table. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named test under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named test under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`
cfname	The name of the column family.
showcol	Set to `false` by default. If set to `true`, then all column-level attributes for this column family are displayed.

Output Fields

Verbose Field Name	Terse Field Name	Field Value
inmemory	inmem	Whether or not this column value resides in memory
cfname	n	The column family name
maxversions	vmax	Maximum number of versions for this column family
minversions	vmin	Minimum number of versions for this column family
compression	comp	Compression scheme used for this column family
ttl	ttl	Time to live for this column family

last-updated: Apr 11, 2022

Verbose Field Name	Terse Field Name	Field Value
compressionperm	pcomp	**Applies to binary tables only:** *ACE* for changing the value of the `compression` parameter. By default, permission is given to the value of `defaultcompressionperm` for the table.
memoryperm	pmem	*ACE* for changing the value of the `inmemory` parameter. Use single quotation marks around the *ACE*. By default, permission is given to the value of `defaultmemoryperm` for the table.
readperm	pread	The *ACE* for column reads. Use single quotation marks around the *ACE*. Reads require permission both at the column-family level and at the column level (for binary tables) or field level (for JSON tables). In JSON tables, this permission is inherited by fields within the column family. By default, permission is given to the value of `defaultreadperm` for the table.
traverseperm	ptraverse	**Applies to JSON tables only:** The Access Control Expressions that specifies who has permission to pass over fields in JSON documents. For example, suppose that a JSON table contains documents of this general structure: `{ "_id" : "ID", "a" : { "b" : "value", "c" : "value" } }` Suppose further that the user `sjohnson` has read permission on `a.b`, but not on `a`. For `sjohnson` to read `a.b`, the user needs the traverse permission on `a`. The user can then pass over field `a` to `a.b`. This permission is inherited by fields within the column family. By default, this permission is given to the value of defaulttraverseperm for the JSON table.

Verbose Field Name	Terse Field Name	Field Value
writeperm	pwrite	The *ACE* for column writes (puts and deletes). Use single quotation marks around the *ACE*. Writes require permission both at the column-family level and at the column level (for binary tables) or field level (for JSON tables). In JSON tables, this permission is inherited by fields within the column family. By default, permission is given to the value of `defaultwriteperm` for the table.

Example

This example lists all column families for the table `newtable`.

CLI

```
maprcli table cf list -path /
my.cluster.com/volume1/newtable
```

REST

```
curl -k -X GET \
  'https://r1n1.sj.us:8443/rest/
table/cf/list?
path=%2Fmy.cluster.com%2Fvolume1%2Fnew
table' \
  -u mapr:mapr
```

Example Output

```
[user@node]# maprcli table cf list -path /mapr/default/user/user/newtable
comp  inmem  vmax  n           ttl         vmin
lz4   false  3     dine        2147483647  0
lz4   false  3     nahashchid  2147483647  0
lz4   false  3     wollachee   2147483647  0
```

table changelog

These maprcli commands are used to create and manage Change Data Capture (CDC) changelogs. A changelog is used to establish a relationship between a MapR Database source table (JSON or binary) and a MapR Event Store For Apache Kafka stream topic and to manage the propagation process.
table changelog add

Description

Creates a changelog and creates a stream topic if one does not already exist. A changelog establishes a relationship between a MapR Database source table (JSON or binary) and a MapR Event Store For Apache Kafka stream topic.

Syntax

Table

CLI	`maprcli table changelog add -path <source table path> -changelog <destination stream path>:<topic name>`

last-updated: Apr 11, 2022

Table (Continued)

REST	`http://<ipaddress>:8443/rest/table/changelog/add?path=<source-table-path>&changelog=<destination stream path>:<topic name>`

Parameters

Table

Parameter	Description
path	(Required) Path of the source table.
changelog	(Required) Target of the change log, specified as <stream_path>:<topic_name>, to which all change data records will be published. The stream must exist, otherwise, the command fails. If the topic does not already exist, `maprcli table changelog add` creates it. To propagate to an existing topic, specify `-useexistingtopic`. **Note:** The `maprcli stream create` command is used to create the changelog stream. **Important:** A CDC changelog stream's default partitions can impact how many partitions a stream topic can have. This is because once you create a stream topic for a changelog stream, the number of topic partitions is *locked*. The number of topic partitions cannot change. When using the `table changelog add` command to add a stream topic (as well as establish a relationship between the source table and the changelog stream), then the number of topic partitions is inherited from the changelog stream and is *locked*.
useexistingtopic	If true, allows propagation to an existing topic. Default: false.
propagateexistingdata	If true, initiates propagation of the existing data to the stream topic, otherwise, only new changes are propagated. Default: true
columns	For MapR Database JSON, a comma separated list of field paths to be propagated. For MapR Database Binary, a comma separated list of column family names, for example, <family>[:<column>] to be propagated. Default: All fields are propagated.
throttle	If true, data transfers to the specified sink are throttled. Default: false
pause	If true, pauses propagation after the changlog is created. Default: false
synchronous	If true, acknowledges the client writes to the table before the internal CDC gateway receives the data. Default: false
networkencryption	Specifies whether the data transfer between MapR filesystem and the internal gateway is encrypted. If true, data propagation is encrypted on-wire. Default: false
networkcompression	Specifies the compression scheme (off|lzf|lz4|zlib) of the data transfer on-wire. Default: lz4

Example

```
maprcli table changelog add -path /tableVolume/cdcTable -changelog /
streamVolume/changelogStream:cdcTopic1
```

```
https://ip.address:8443/rest/table/changelog/add?path=/tableVolume/
cdcTable&changelog=/streamVolume/changelogStream:cdcTopic1
```

table changelog edit

Description

Changes changelog specifications.

Syntax

Table

CLI	`maprcli table changelog edit -path <source table path> -changelog <destination stream path>:<topic name>`
REST	`http://<ipaddress>:8443/rest/table/changelog/edit?path=<source-table-path>&changelog=<destination-stream-path>:<topic-name>`

Parameters

Table

Parameters	Description
path	(Required) Path of the MapR Database source table
changelog	(Required) Target of this changelog.
throttle	If true, data propation is throttled. Default: false
synchronous	If true, acknowledges the client writes to the table before the internal CDC gateway receives the data. Default: false
networkencryption	If true, data propagation is encrypted on-wire. Default: false
networkcompression	Specifies the compression scheme (off|lzf|lz4|zlib) of the data propagation on-wire. Default: lz4

Example

```
maprcli table changelog edit -path /tableVolume/cdcTable -changelog /
streamVolume/changelogStream:cdcTopic1
```

```
https://10.10.100.17:8443/rest/table/changelog/edit?path=/tableVolume/
cdcTable&changelog=/streamVolume/changelogStream:cdcTopic1
```

table changelog info

Description

Displays changelog source table information.

Syntax

Table

CLI	`maprcli table changelog info -path <destination stream path>:<topic name>`
REST	`http://<ipaddress>:8443/rest/table/changelog/info?path=<destination stream path>:<topic name>`

Parameters

Table

Parameters	Description
path	(Required) Path to the stream with topic. Specified in the format: `pathToStream:streamTopic`

Example

```
maprcli table changelog info -changelog  /streamVolume/
changelogStream:cdcTopic1 -json
https://10.10.100.17:8443/rest/table/changelog/info?changelog=/streamVolume/
changelogStream:cdcTopic1
```

Output

```
{
"timestamp":1498526974087,
"timeofday":"2017-06-26 06:29:34.087 GMT-0700",
"status":"OK",
"total":1,
"data":[
{
"cluster":"clst185",
"changelog":"/streamVolume/changelogStream/cdcTopic1",
"idx":0,
"uuid":"59bf0064-97a1-c417-b6d7-0a6681515900"
}
]
}
```

table changelog list

Description

Lists changelog information.

Syntax

Table

CLI	`maprcli table changelog list -path <source table path>`
REST	`http://<ipaddress>:8443/rest/table/changelog/list?path=<source-table-path>`

Parameters

Table

Parameters	Description
path	(Required) Path of the source table in the MapR Database cluster.
refreshnow	Specifies if the user wants to trigger an immediate update of the sink statistics.

Example

```
// CLI example
maprcli table changelog list -path /tableVolume/cdcTable -json

// REST example
https://10.10.100.17:8443/rest/table/changelog/list?path=/tableVolume/
cdcTable
```

Output

```
{
    "timestamp":1505779365019,
    "timeofday":"2017-09-18 05:02:45.019 GMT-0700",
    "status":"OK",
    "total":1,
    "data":[
        {
            "cluster":"my.cluster.com",
            "changelog":"/streamVolume/changelogStream:cdcTopic1",
            "changelogStream":"/streamVolume/changelogStream",
            "replicaState":"REPLICA_STATE_REPLICATING",
            "paused":false,
            "throttle":false,
            "idx":1,
            "networkencryption":false,
            "synchronous":false,
            "networkcompression":"lz4",
            "propagateExistingData":true,
            "isUptodate":true,
            "minPendingTS":0,
            "maxPendingTS":0,
            "bytesPending":0,
            "putsPending":0,
            "bucketsPending":0,
            "uuid":"76a3efd3-6357-8cd6-092f-0fca5dc05900",
            "copyTableCompletionPercentage":100
        }
```


last-updated: Apr 11, 2022

```
    ]
}
```

Output Data Fields

The following fields display for each replica.

Field	Description
cluster	The cluster on which the replica resides.
changelog	Identifies the destination stream topic for the changelog.
changelogstream	Identifies the destination stream for the changelog.
replicaState	The replication state. For information about the replication states, see Table Replication States on page 625.
paused	A Boolean values that specifies if replication is paused.
throttle	A Boolean value that specifies if replication is throttled.
idx	The internal index value.
networkencryption	A Boolean value that specifies if replication is encrypted.
synchronous	A Boolean value that specifies whether replication is synchronous or asynchronous.
networkcompression	The type of on-wire compression.
propagateExistingData	Identifies whether existing data in the source table is propagated to the destination stream topic.
isUptodate	A Boolean value that specifies if the replica is up-to-date.
minPendingTS	The epoch time in milliseconds of the oldest operation that has yet to be replicated to the replica.
maxPendingTS	The epoch time in milliseconds of the newest operation that has yet to be replicated to the replica.
bytesPending	The number of bytes that have yet to be replicated to the replica.
putsPending	The number of puts that have yet to be replicated to the replica.
bucketsPending	The number of buckets that have yet to be replicated to the replica.
uuid	The table UUID.
copyTableCompletionPercentage	When propagation of existing data is in progress, this value is the percentage of data from the source table that has been propagated to the destination stream topic. **Note:** When replicating MapR Database data, the copyTablePercentageCompletion data may re-adjust to a lower rate. This depends on table region (also referred to as tablets) splits and merges as well as the rate of incoming data to replicating data.
errors	If applicable, an error is displayed.

table changelog pause

last-updated: Apr 11, 2022

Description

Pauses the propagation of changed data records.

Syntax

Table

CLI	`maprcli table changelog pause -path <source table path> -changelog <destination stream path>:<topic name>`
REST	`http://<ipaddress>:8443/rest/table/changelog/pause?path=<source-table-path>&changelog=<destination stream path>:<topic_name>`

Parameters

Table

Parameters	Description
path	(Required) Path of the MapR Database source table
changelog	(Required) Target of this change log.

Example

```
maprcli table changelog pause -path /tableVolume/cdcTable -changelog /
streamVolume/changelogStream:cdcTopic1
```

```
https://10.10.100.17:8443/rest/table/changelog/pause?path=/tableVolume/
cdcTable&changelog=/streamVolume/changelogStream:cdcTopic1
```

table changelog remove

Description

Removes the changelog connection.

Syntax

Table

CLI	`maprcli table changelog remove -path <source table path> -changelog <destination stream path>:<topic name>`
REST	`http://<ipaddress>:8443/rest/table/changelog/remove?path=<source-table-path>&changelog=<destination stream path>:<topic_name>`

last-updated: Apr 11, 2022

Parameters

Table

Parameters	Description
path	(Required) Path of the source table in the MapR Database cluster.
changelog	(Required) Target of the change log.

Example

```
maprcli table changelog remove -path /tableVolume/cdcTable -changelog /
streamVolume/changelogStream:cdcTopic1
```

```
https://10.10.100.17:8443/rest/table/changelog/remove?path=/tableVolume/
cdcTable&changelog=/streamVolume/changelogStream:cdcTopic1
```

table changelog resume

Description

Resumes the propagation of changed data records after a pause.

Syntax

Table

CLI	`maprcli table changelog resume -path <source table path> -changelog <destination stream path>:<topic name>`
REST	`http://<ipaddress>:8443/rest/table/changelog/resume?path=<source-table-path>&changelog=<destination stream path>:<topic_name>`

Parameters

Table

Parameters	Description
path	(Required) Path of the MapR Database source table
changelog	(Required) Target of this change log.

Example

```
maprcli table changelog resume -path /tableVolume/cdcTable -changelog /
streamVolume/changelogStream:cdcTopic1
```

```
https://10.10.100.17:8443/rest/table/changelog/resume?path=/tableVolume/
cdcTable&changelog=/streamVolume/changelogStream:cdcTopic1
```

`table delete`
Deletes a MapR Database binary or JSON table.

last-updated: Apr 11, 2022

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminaccessperm` on the table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table delete -path <path>
```

REST

```
curl -k -X POST
  'http[s]://<host>:<port>/rest/table/
delete?path=<path>'
  -u <username>:<password>
```

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Parameters

Parameter	Description
`path`	Path to the MapR table to delete. • For a path on the local cluster, start the path at the volume mount point. For example, if you want to delete a table named `test` under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, if you want to delete a table named `customer` under volume1 in the sanfrancisco cluster, specify the following path: `/mapr/sanfrancisco/volume1/customer`

Example

Deletes the table `table`:

CLI

```
maprcli table delete -path /mapr/
mycluster/volume1/table
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/table/
delete?
```


last-updated: Apr 11, 2022

```
path=%2Fmapr%2Fmycluster%2Fvolume1%2Ft
able' \
  -u mapr:mapr
```

`table edit`

Edits the attributes of a MapR Database binary or JSON table.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `adminaccessperm` on the table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table edit
  -path <path >
  [ -audit true|false ]
  [ -autosplit <Auto Split table> ]
  [ -regionsizemb <Region Size in
MB> ]
  [ -bulkload <Bulk load> ]
  [ -deletettl <delete TTL in secs> ]
  [ -packperm <Pack Permission
settings> ]
  [ -bulkloadperm <Bulk load
Permission settings> ]
  [ -splitmergeperm <Split and Merge
Permission settings>]
  [ -createrenamefamilyperm <Add/
Rename Family Permission settings> ]
  [ -deletefamilyperm <Delete Family
Permission settings> ]
  [ -adminaccessperm <Secondary Index
Admin Permission settings> ]
  [ -replperm <Replication Admin
Permission settings> ]
  [ -indexperm <Ace Admin Permission
settings> ]
  [ -defaultversionperm <CF Versions
Default Permission> ]
  [ -defaultcompressionperm <CF
Compression Default Permission> ]
  [ -defaultmemoryperm <CF Memory
Default Permission> ]
  [ -defaultreadperm <CF Read Default
Permission> ]
  [ -defaultwriteperm <CF Write
Default Permission> ]
  [ -defaulttraverseperm <CF Traverse
Default Permission> ]
```

```
  [ -defaultappendperm <CF Append
Default Permission> ]

  [ -metricsinterval <Metrics
collection interval, in seconds> ]
```

REST

```
curl -k -X POST \
  'http[s]://<host>:<port>/rest/table/
edit?path=<path>&<parameters>'
  -u <username>:<password>
```

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Parameter	Description
path	The path to the table. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named test under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named test under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`
audit	Specifies whether to turn auditing on for the table. If auditing is also enabled at the cluster level with the `maprcli audit data` command and enabled for the current volume, setting this value to true causes auditing to start for the table. The possible values are `true` and `false`. By default, the value is `false`.
autosplit	A Boolean value that specifies whether to split the table into regions automatically as the table grows. The average size of each region is determined by the `regionsizemb` parameter. The default value is `true`. If you set the value to `false`, you can manually split tables into regions by using the `table region split` command.

last-updated: Apr 11, 2022

Parameter	Description
regionsizemb	The average size of the regions into which MapR Database tries to split the table as the table grows. The default is 4096 MB. This value is ignored if `autosplit` is set to `false`. If `autosplit` is set to `true`, MapR Database splits a region when the size of the region exceeds **150%** of the average value. For example, if the average value is 4096 MB, MapR Database splits a region that is larger than 6144 MB. Although splits are automatic, merges are not. For example, if the value of `regionsizemb` is changed from 8 GB to 4 GB, all regions that are eligible are split automatically, if `autosplit` is set to `true`. However, if the value of `regionsizemb` is changed from 2 GB to 4 GB, regions smaller than 4 GB are not automatically merged. **Note:** When a table has less than 4 regions, MapR Database ignores the `regionsizemb` parameter and splits regions at a lower threshold.
bulkload	A Boolean value that specifies whether to allow a full bulk load of the table. The default is `false`. For more information, see Loading Data into Binary Tables on page 1040 and Loading Documents into JSON Tables on page 1036.
deletettl	The number of seconds to wait before purging the delete operations. The time-to-live for deletes should be greater than the amount of time that it takes replicated operations to reach replicas. By default, the value is 24 hours for tables configured for replication. If the table is not configured for replication, the default is 0 hours.
packperm	The Access Control Expression that controls who can pack table regions. By default, permission is given to the user ID that was used to create the table.
bulkloadperm	The Access Control Expression that controls who can load this table with bulk loads if the table was created with bulk load support. By default, permission is given to the user ID that was used to create the table.
splitmergeperm	The Access Control Expression that controls who can take the following actions: • Run the `table region split` and `table region merge` commands to split the table into regions or to merge regions of the table together. • Change the value of `regionsizemb`. By default, permission is given to the user ID that was used to create the table.
createrenamefamilyperm	The Access Control Expression that controls who can create column families for this table or rename existing column families. By default, permission is given to the user ID that was used to create the table.

Parameter	Description
deletefamilyperm	The Access Control Expression that defines access to delete column families for this table. Delimit the expression with single-quotation marks. By default, permission is given to the user ID that was used to create the table.
adminaccessperm	The Access Control Expression that controls who can view and edit the permissions for this table. By default, permission is given to the user ID that was used to create the table.
replperm	The Access Control Expression that controls who can set up replication either to or from a table. By default, permission is given to the user ID that is used to create the table.
indexperm	The secondary index admin permission setting that controls who can create an index associated with this table. By default, permission is given to the user ID that is used to create the table.
defaultversionperm	The default Access Control Expression for the version permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `versionperm` in the `table cf create` and `table cf edit` commands overrides this value. **Note:** This permission is not applicable to JSON tables. Versioning is not supported for JSON documents.
defaultcompressionperm	**Applies to binary tables only:** The default Access Control Expression for the compression permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `compressionperm` in the `table cf create` and `table cf edit` commands overrides this value.
defaultmemoryperm	The default Access Control Expression for the memory permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `memoryperm` in the `table cf create` and `table cf edit` commands overrides this value.
defaultreadperm	The default Access Control Expression for the read permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `readperm` in the `table cf create` and `table cf edit` commands overrides this value. See table cf create on page 1799 and table cf edit on page 1806

last-updated: Apr 11, 2022

Parameter	Description
defaultwriteperm	The default Access Control Expression for the write permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `writeperm` in the `table cf create` and `table cf edit` commands overrides this value. See table cf create on page 1799 and table cf edit on page 1806
defaulttraverseperm	**Applies to JSON tables only:** The default Access Control Expression for the traverse permission on new column families. For more information about this permission, see Permission Types for Fields and Column Families in JSON Tables on page 1462.
defaultappendperm	**Applies to binary tables only:** The default Access Control Expression for the append permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `appendperm` in the `table cf create` and `table cf edit` commands overrides this value.
metricsinterval	The metrics collection interval, in seconds, for the table. Possible values: 10, 60, 600 Default: 60 seconds When configured to 10 seconds, under normal workloads, the metrics are available in OpenTSDB in about 30 seconds. At an interval of 60 seconds, the metrics are available in about 90 seconds. **Note:** You cannot disable metrics collection for a table by setting the interval to 0.

Example

Changes the value of `regionsizemb` for the table `mytable`:

CLI

```
maprcli table edit -path /volume1/
mytable -regionsizemb 8192
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/table/
edit?
path=%2Fvolume1%2Fmytable&regionsizemb
=8192' \
  -u mapr:mapr
```

table index

Manages indexes for MapR Database JSON tables.

Permissions Required

To run this command, your user ID must have the following permissions:

- readAce on the volume

- lookupdir on directories in the table path

Note: The mapr user is not treated as a superuser. MapR Database does not allow the mapr user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

table index add
This topic describes how to add secondary indexes on MapR Database JSON tables.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the table path
- `indexperm` permission on the table

If you created the table in version 6.0 or later, you automatically have `indexperm` permission. For tables created before 6.0, even if you are the owner of the table, you must explicitly add `indexperm` permission.

Note: The mapr user is not treated as a superuser. MapR Database does not allow the mapr user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table index add
  -path <path>
  -index <index name>
  -indexedfields < indexed field
names >
  [ -includedfields < included field
names > ]
  [ -hashed [ enable hashed index:
true | false> ]
  [ -numhashpartitions < number of
hash index partitions when hashed
index is enabled > ]
```

REST

```
curl -k -X POST \
  'http[s]://<host>:<port>/rest/table/
index/add?path=<path>&index=<index
name>&indexedfields=<indexed field
names>&<parameters>' \
  -u <username>:<password>
```

Parameters

Parameter	Description
path	(Required) Path to where the parent JSON table resides.
index	(Required) Name of the index.

last-updated: Apr 11, 2022

Parameter	Description

indexedfields	(Required) Names of the indexed fields. This is a comma separated list of the fields from the JSON table that are indexed and used for ordering the index. The sort ordering of each field can be specified separately. The syntax is as follows:
	`-indexedfields <fieldname>:<sort_order>,<fieldname>:<sort_order>,...`
	Important: Do not place a space between the commas and the field names.
	A sort_order of `asc`, `ASC`, or `1` denotes an ascending sort order. This is the default.
	A sort _order of `desc`, `DESC`, or `-1` denotes a descending sort order.
	The following example specifies two indexed fields. `fieldName1` has an ascending sort, while `fieldName2` is descending.
	`-indexedfields fieldName1:asc,fieldName2:desc`
	If an indexed field contains a colon (:) in the name, you need to escape the last colon in the name. In the example below, the indexed field names are the following:
	1. `field1`
	2. `field2`
	3. `colonField:X:Y`
	The following shows how to escape the last colon in the third indexed field.
	`-indexedfields field1,field2,colonField:X\\:Y`
	Note: The CAST function can be applied on indexed fields. You must enclose each CAST function call in single quotes. See the next section for details.

includedfields	(Optional) Names of the included fields. This is a comma separated list of the fields from the JSON table that are part of the index, but not used for ordering. The syntax is as follows: `-includedfields <fieldname>,<fieldname>,...` **Important:** Do not place a space between the commas and the included field names.
hashed	(Optional) True | False. Default: false

Parameter	Description
numhashpartitions	(Optional) Number of hash index partitions when the hashed index option is enabled. This parameter determines the number of logical partitions MapR Database distributes keys across. Incoming keys are hashed to 2 byte partition IDs. Default: 10

Applying CAST on Indexed Fields

Indexes can be defined with the CAST function applied to an indexed field.

The following statement queries a table named `lineitem` and casts the `L_LINENUMBER` and `L_ORDERKEY` fields to the `int` data type.

```
SELECT L_LINESTATUS, L_QUANTITY FROM lineitem WHERE CAST(L_LINENUMBER as
int) = 1 AND CAST(L_ORDERKEY as int) = 550;
```

To optimize the previous statement, you can create an index on the `L_LINENUMBER` and `L_ORDERKEY` fields, and use the CAST function to map each field to a specific data type, as shown below:

```
maprcli table index add \
  -path /drill/testdata/qa/sf1/maprdb/json/lineitem \
  -index l_cast_comp_1 -indexedfields
'$CAST(L_LINENUMBER@INT)','$CAST(L_ORDERKEY@INT)' \
  -includedfields L_LINESTATUS,L_QUANTITY
```

The index stores the values of the `L_LINENUMBER` and `L_ORDERKEY` fields as the `int` data type. MapR Database can use the index for any subsequent queries that cast these fields to `int` instead of accessing data in the primary table and converting the values to `int`.

See Using Casts in Secondary Indexes on page 557 for more information.

Restrictions

The following restrictions apply to creating indexes.

Name Restrictions

You cannot use the following characters in the index name and in the indexed fields:

```
< > ? % \
```

To use the following characters in the index name and in the indexed fields, enclose them either in single or double quotes:

```
; | ( ) /
```

For example:

```
maprcli table index
add -path /volume1/MYTABLE -index
"MYTABLE1_ANALYSIS_1♦=#;{}&()/" \
  -indexedfields "●timestamp":desc,"●
♥,"LOTNo" -includedfields \
  "♥,"♦=#;{}&()/" (or)

maprcli table index
add -path /volume1/MYTABLE -index
'MYTABLE1_ANALYSIS_1♦=#;{}&()/' \
  -indexedfields "●timestamp":desc,"●
```


last-updated: Apr 11, 2022

```
●,"LOTNo" -includedfields \
  '●,'●=#;{}&()/'
```

To use either the ' or the " character in the index name and in the indexed fields, enclose:

- the ' character within double quotes (")
- the " character within single quote (')

For example:

```
maprcli table index
add -path /volume1/MYTABLE -index
"'MYTABLE1_ANALYSIS_1●=#;{}&()/" \
  -indexedfields "_●timestamp":desc,"●●
●,"LOTNo" -includedfields \
  "●","●=#;{}&()/" (or)

maprcli table index
add -path /volume1/MYTABLE -index
'"MYTABLE1_ANALYSIS_1●=#;{}&()/' \
  -indexedfields "'_●timestamp":desc,"●
●,"LOTNo" -includedfields \
  '●','●=#;{}&()/"
```

Type Restrictions

- If a composite index includes the same subfield in multiple indexed fields, the implied types of the subfields must be consistent.

  For example, you cannot create an index with the following indexed fields:

  ```
  a.b[].c, a.b.d
  ```

  Although subfield b appears in both indexed fields, in the first, it is an array and in the second, it is a nested document.

  See Composite Indexes and Container Field Paths on page 554 for more details.

Size Restrictions

- The maximum size of all indexed fields in an index is 32 KB.

  If the collective size exceeds 32 KB, then an insert of the corresponding document results in an encoding error (`INDEX_ROW_KEY_ENCODER_ERROR_ENCODING_IS_TOO_LONG`).

- The maximum number of indexes that you can create on a JSON table is 32.

Field Definition Restrictions

- You cannot specify individual array elements as indexed fields.
- You cannot specify a table's `_id` field as an indexed field.

- If a field contains an array of nested documents and you want to index on subfields in the nested documents, then you must define the indexed field using a container field path.
- You can include a specific field only once as either an indexed or included field, with the following two exceptions:
  - The indexed field is a container field path:

    ```
    maprcli table index add -path /
    people \
       -index phoneNumberIdx \
       -indexedfields
    Phones[].Number \
       -includedfields
    Phones[].Number
    ```

  - The field specifies a cast to another type.

    You can create an index in which the `score` field is an indexed field cast as a `double` type, and `score` is also an included field. The included field retains the original data type of the `score` field:

    ```
    maprcli table index add -path /
    castTable \
       -index castIdx1 \
       -indexedfields
    '$CAST(score@DOUBLE)' \
       -includedFields score
    ```

    You can create an index in which the `score` field is an indexed field, cast as a `double` type, and the `score` field is also another indexed field, cast as a `long` type:

    ```
    maprcli table index add -path /
    castTable \
       -index castIdx2 \
       -indexedfields
    '$CAST(score@DOUBLE)','$CAST(scor
    e@LONG)'
    ```

- You cannot use casts with included fields.

- You cannot specify a field as either an indexed or included field if the field is also specified as a column family JSON path name.

  For example, suppose you have the following JSON table:

```
{
     "_id" : "ID",
     "a" :
          {
               "b" :
                    {
                         "c" :
"value",
                         "d" :
"value"
                    },
               "e" : "value"
          }
}
```

  If you create a column family at field `c` in the JSON path `a.b.c`, you cannot define field `a.b.c` as either an indexed or included field. You can define the fields `a`, `a.b`, and `a.b.d` as either indexed or included fields.

last-updated: Apr 11, 2022

- You cannot specify an included field in which the data in the field spans more than one column family.

  In the following example, the included field `sl1.sl2` spans column families, `cf2` and `cf3`:

```
maprcli table cf list -path /cftab
compressionperm  readperm
traverseperm  jsonfamilypath
writeperm  minversions
maxversions  compression
ttl         inmemory  cfname
memoryperm
u:root           u:root
u:root
u:root     0
1            lz4
2147483647  false     default
u:root
u:root           u:root
u:root         sl1
u:root     0
1            lz4
2147483647  false     cf1
u:root
u:root           u:root
u:root         sl1.sl2.sl3
u:root     0
1            lz4
2147483647  false     cf2
u:root
u:root           u:root
u:root         sl1.sl2.sl3.sl4
u:root     0
1            lz4
2147483647  false     cf3
u:root

maprcli table index add -path /
cftab -index i1 -indexedfields
sl1.sl2.sl3.sl4.l4a,
sl1.l1a -includedfields
sl1.sl2,sl1.sl2.sl3.sl4.sl5.l5b -js
on
{
        "timestamp":1507419777919,
        "timeofday":"2017-10-07
04:42:57.919 GMT-0700 PM",
        "status":"ERROR",
        "errors":[
                {
                        "id":22,

"desc":"Data for included field
sl1.sl2 may not span more than one
column family."
                }
         ]
}
```


last-updated: Apr 11, 2022

- You cannot specify a composite index with more than one container field path as your indexed fields, unless the prefixes of the container field paths are the same.

  See Composite Indexes and Container Field Paths on page 554 for more details.

- You cannot specify a composite index with an indexed field that is a subfield of another indexed field.

  For example, you cannot create an index with the following indexed fields:

  ```
  a, a.b
  ```

  The indexed field `a.b` is a subfield of the indexed field `a`.

Option Restrictions

- As indexes are automatically split, you cannot disable splits when you create your index.

Index Use Restrictions

- Indexes do not optimize non-existence filter conditions.

Example

CLI

```
maprcli table index
add -path /demo/business -index
newIndex -indexedfields fieldName
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/table/
index/add?
path=%2Fdemo%2Fbusiness&index=newIndex
&indexedfields=fieldName' \
  -u mapr:mapr
```

table index list

This topic describes how to list information about the secondary indexes created on MapR Database JSON tables.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the table path

Note: The mapr user is not treated as a superuser. MapR Database does not allow the mapr user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

last-updated: Apr 11, 2022

Syntax

CLI

```
maprcli table index list
  -path <path>
  [ -indexname <index name> ]
  [ -refreshnow < true | false > ]
```

REST

```
curl -k -X GET
  'http[s]://<host>:<port>/rest/table/
index/list?path=<path>&<parameters>'
  -u <username>:<password>
```

Parameters

Parameter	Description
path	(Required) Path to where the parent JSON table resides
indexname	(Optional) Name of the index for which to display information. If omitted, the output includes all indexes created on the table.
refreshnow	(Optional) Whether to fetch the current status of the index Default: False

Example

CLI

```
maprcli table index list -path /
my.cluster.com/volume1/table1
```

```
maprcli table index list -path /demo/
business -json
```

REST

```
curl -k -X GET \
  'https://r1n1.sj.us:8443/rest/table/
index/list?
path=%my.cluster.com%2Fvolume1%2Ftable
1' \
  -u mapr:mapr
```

```
curl -k -X GET \
  'https://r1n1.sj.us:8443/rest/table/
index/list?path=%2Fdemo%2Fbusiness' \
  -u mapr:mapr
```

Output Fields

Output Field	Description
cluster	The cluster on which the index resides
type	For indexes, this is always `maprdb.si`

last-updated: Apr 11, 2022

Output Field	Description
indexFid	A unique id used to identify the index in MapR File System
indexName	Name of the index
hashed	A boolean value that specifies whether the index is hashed
indexState	The replication state of the index. For information about the replication states, see Table Replication States on page 625.
idx	The index id. Unique per table.
indexedFields	The list of indexed fields with the sort order of each key
includedFields	The list of included fields in the index. Missing from output if there are no included fields.
isUptodate	A boolean value that specifies if the index is up-to-date
minPendingTS	The epoch time in milliseconds of the oldest operation that has yet to be replicated to the index
maxPendingTS	The epoch time in milliseconds of the newest operation that has yet to be replicated to the index
bytesPending	The number of bytes that have yet to be replicated to the index
putsPending	The number of puts that have yet to be replicated to the index
bucketsPending	The number of buckets that have yet to be replicated to the index
copyTableCompletionPercentage	The percentage of data from the source that has been copied to the index during the setup phase of replication. After replication setup completes, the value remains at 100. **Note:** When replicating data to the index, the `copyTableCompletionPercentage` value may decrease. This happens when splits or merges occur in the JSON table's regions, or the table receives new data.
numTablets	The number of tablets the index occupies
numRows	The number of rows in the index
totalSize	The total size of the index

Example Output

```
maprcli table index list -path /demo/business -json -indexname i1
{
        "timestamp":1506617667735,
        "timeofday":"2017-09-28 04:54:27.735 GMT+0000 PM",
        "status":"OK",
        "total":1,
        "data":[
                {
                        "cluster":"my.cluster.com",
                        "type":"maprdb.si",
                        "indexFid":"2049.93.10257820",
```


last-updated: Apr 11, 2022

```
                        "indexName":"i1",
                        "hashed":false,
                        "indexState":"REPLICA_STATE_REPLICATING",
                        "idx":1,
                        "indexedFields":"a.b:ASC",
                        "isUptodate":true,
                        "minPendingTS":0,
                        "maxPendingTS":0,
                        "bytesPending":0,
                        "putsPending":0,
                        "bucketsPending":0,
                        "copyTableCompletionPercentage":100,
                        "numTablets":1,
                        "numRows":4,
                        "totalSize":24576
                }
        ]
}
```

Troubleshooting Use Cases

Situations where you can use this command include the following:

- Examine the properties of an index.
- Determine if there is a lag in updates in an index.

See Troubleshooting Secondary Indexes on page 1092 for more information on these use cases.

table index remove
This topic describe how to remove secondary indexes that are no longer needed.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the table path
- `indexperm` permission on the table, if you did not create the table

Note: The mapr user is not treated as a superuser. MapR Database does not allow the mapr user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table index remove
  -path <path>
  -index <index name>
```

REST

```
curl -k -X POST \
  'http[s]://<host>:<port>/rest/table/
index/remove?path=<path>&index=<index
name>'
  -u <username>:<password>
```


last-updated: Apr 11, 2022

Parameters

Parameter	Description
path	(Required) Path to where the parent JSON table resides
index	(Required) Name of the index

Example

CLI

```
maprcli table
index remove -path /my.cluster.com/
volume1/newtable -index testIndex
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/table/
index/remove?
path=%2Fmy.cluster.com%2Fvolume1%2Fnew
table&index=testIndex' \
  -u mapr:mapr
```

`table info`

Displays information about a MapR Database binary or JSON table, or an index on a JSON table.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the path

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table info
  -path <path>
  [ -index <index name> ]
```

REST

```
curl -k -X GET \
  'http[s]://<host>:<port>/rest/table/
info?path=<path>&index=<index name>'
  -u <username>:<password>
```

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

last-updated: Apr 11, 2022

Parameters

Parameter	Description
path	The path to the table. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `test` under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customer` under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`
index	The name of the index for which to display information.

Example

Lists the information for a table named `mytable` in the JSON format, as described in Common Options on page 1525:

CLI

```
maprcli table info -path /mapr/
my.cluster.com/volume1/mytable -json
```

REST

```
curl -k -X GET \
  'https://r1n1.sj.us:8443/rest/table/
info?
path=%2Fmapr%2Fmy.cluster.com%2Fvolume
1%2Fmytable' \
  -u mapr:mapr
```

Sample Output

```
# maprcli table info -path /mydirectory/mytable -json
{
    "timestamp":1444671479053,
    "timeofday":"2015-10-12 10:37:59.053 GMT-0700",
    "status":"OK",
    "total":1,
    "data":[
        {
        "path":"/mydirectory/mytable",
        "numregions":1,
        "totallogicalsize":0,
        "totalphysicalsize":0,
        "totalcopypendingsize":0,
        "totalrows":0,
        "autosplit":true,
        "bulkload":false,
        "tabletype":"json",
        "regionsizemb":4096,
        "audit":false,
        "metricsinterval":60,
        "maxvalueszinmemindex":100,
        "adminaccessperm":"u:root",
        "createrenamefamilyperm":"u:root",
```


last-updated: Apr 11, 2022

```
        "bulkloadperm":"u:root",
        "indexperm":"u:root",
        "packperm":"u:root",
        "deletefamilyperm":"u:root",
        "replperm":"u:root",
        "splitmergeperm":"u:root",
        "defaultappendperm":"u:root",
        "defaultcompressionperm":"u:root",
        "defaultmemoryperm":"u:root",
        "defaultreadperm":"u:root",
        "defaulttraverseperm":"u:root",
        "defaultversionperm":"u:root",
        "defaultwriteperm":"u:root",
        "uuid":"3b4ef43c-83d4-06eb-99ba-08c8ef1b5600"
        }
    ]
}
```

Output Fields

```
maprcli table info -path /mapr/my.cluster.com/volume1/mytable -json

{
        "timestamp":1540362830403,
        "timeofday":"2018-10-23 11:33:50.403 GMT-0700 PM",
        "status":"OK",
        "total":1,
        "data":[
                {
                        "path":"/mapr/my.cluster.com/volume1/mytable",
                        "numregions":1,
                        "totallogicalsize":0,
                        "totalphysicalsize":0,
                        "totalcopypendingsize":0,
                        "totalrows":0,
                        "totalnumberofspills":0,
                        "totalnumberofsegments":0,
                        "autosplit":true,
                        "bulkload":false,
                        "wireencryptionfrompolicies":false,
                        "tabletype":"json",
                        "securitypolicy":"[Credit_Card_Data,Confidential]",
                        "regionsizemb":4096,
                        "audit":false,
                        "metricsinterval":10,
                        "maxvalueszinmemindex":100,
                        "adminaccessperm":"u:root",
                        "createrenamefamilyperm":"u:root",
                        "bulkloadperm":"u:root",
                        "indexperm":"u:root",
                        "packperm":"u:root",
                        "deletefamilyperm":"u:root",
                        "replperm":"u:root",
                        "splitmergeperm":"u:root",
                        "defaultcompressionperm":"u:root",
                        "defaultmemoryperm":"u:root",
                        "defaultreadperm":"u:root",
                        "defaulttraverseperm":"u:root",
                        "defaultwriteperm":"u:root",

                        "uuid":"8fea24dc-6e56-6d56-6336-0e0408e15e00"
                }
```

```
        ]
}
```

Output Field	Description
path	The path to the MapR Database table • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `test` under a volume with a mount point at /volume1, specify the following path: `/volume1/test` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customer` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`
numregions	Number of regions in the table
totallogicalsize	Estimated size (in bytes) of uncompressed data stored in table (excluding replication)
totalphysicalsize	Estimated size (in bytes) of actual data stored in table (excluding replication). Includes internal metadata and reflects compressed data size when compression is enabled.
totalcopypendingsize	Total size (in bytes) of pending data for replication
totalrows	Estimated number of rows in a table
autosplit	A Boolean value that specifies whether to split the table into regions automatically as the table grows. The average size of each region is determined by the `regionsizemb` parameter. The default value is `true`. If value is set to `false`, you can manually split tables into regions by using the `table region split` command.
bulkload	A Boolean value that specifies whether to allow a full bulk load of the table. The default is `false`. For more information, see Loading Data into Binary Tables on page 1040 and Loading Documents into JSON Tables on page 1036.
tabletype	Specifies whether the table will be a binary table or a JSON table. The values are `binary` and `json`. The default is `binary`.

Output Field	Description
regionsizemb	The average size of the regions into which MapR Database tries to split the table as the table grows. The default is 4096 MB. This value is ignored if `autosplit` is set to `false`. If `autosplit` is set to `true`, MapR Database splits a region when the size of the region exceeds 150% of the average value. For example, if the average value is 4096 MB, MapR Database splits a region that is larger than 6144 MB. Although splits are automatic, merges are not. For example, if the value of `regionsizemb` is changed from 8 GB to 4 GB, all regions that are eligible are split automatically, if `autosplit` is set to `true`. However, if the value of `regionsizemb` is changed from 2 GB to 4 GB, regions smaller than 4 GB are not automatically merged. **Note:** When a table has less than 4 regions, MapR Database ignores the `regionsizemb` parameter and splits regions at a lower threshold.
audit	Specifies whether to turn auditing on for the table. If auditing is also enabled at the cluster level with the `maprcli audit data` command and enabled for the current volume, setting this value to true causes auditing to start for the table.
metricsinterval	The table metrics collection interval, in seconds
maxvalueszinmemindex	The maximum value size to save in an in-memory index
adminaccessperm	The Access Control Expression that controls who can view and edit the permissions for this table. By default, permission is given to the user ID that is used to create the table.
createrenamefamilyperm	The Access Control Expression that controls who can create column families for this table or rename existing column families. By default, permission is given to the user ID that is used to create the table.
bulkloadperm	The Access Control Expression that controls who can load this table with bulk loads if the table was created with bulk load support. By default, permission is given to the user ID that is used to create the table.
indexperm	The secondary index Admin permissions setting that controls who can create an index associated with this table. By default, permission is given to the user ID that is used to create the table.
packperm	The Access Control Expression that controls who can pack table regions. By default, permission is given to the user ID that is used to create the table.
deletefamilyperm	The Access Control Expression that defines access to delete column families for this table. Delimit the expression with single-quotation marks. By default, permission is given to the user ID that is used to create the table.

last-updated: Apr 11, 2022

Output Field	Description
replperm	The Access Control Expression that controls who can set up replication either to or from a table. By default, permission is given to the user ID that is used to create the table
splitmergeperm	The Access Control Expression that controls who can take the following actions: • Run the `table region split` and `table region merge` commands to split the table into regions or to merge regions of the table together. • Change the value of `regionsizemb`. By default, permission is given to the user ID that is used to create the table.
defaultappendperm	**Applies to binary tables only:** The default Access Control Expression for the append permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `appendperm` in the `table cf create` and `table cf edit` commands overrides this value.
defaultcompressionperm	**Applies to binary tables only:** The default Access Control Expression for the compression permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `compressionperm` in the `table cf create` and `table cf edit` commands overrides this value.
defaultmemoryperm	The default Access Control Expression for the memory permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `memoryperm` in the `table cf create` and `table cf edit` commands overrides this value.
defaultreadperm	The default Access Control Expression for the read permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `readperm` in the `table cf create` and `table cf edit` commands overrides this value. See table cf create on page 1799 and table cf edit on page 1806
defaulttraverseperm	**Applies to JSON tables only:** The default Access Control Expression for the traverse permission on new column families. For more information about this permission, see Permission Types for Fields and Column Families in JSON Tables on page 1462.
defaultwriteperm	The default Access Control Expression for the write permission on new column families that are created in this table. If no value is specified, the default is `u:<username of the table creator>`. This value of the parameter `writeperm` in the `table cf create` and `table cf edit` commands overrides this value. See table cf create on page 1799 and table cf edit on page 1806.

last-updated: Apr 11, 2022

Output Field	Description
wireencryptionfrompolicies	The system automatically sets this field to true if at least one security policy has wire-level encryption enabled, false otherwise.

`table region`
Manages table regions for MapR Database binary and JSON tables.
`table region list`
Lists the regions that make up a specified table or index.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the path

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table region list
  -path <path>
  [ -start <offset from starting
region> ]
  [ -limit <number of regions to
return> ]
  [ -index <index name> ]
  [ -output terse | verbose ]
```

REST

```
curl -k -X GET \
  'http[s]://<host>:<port>/rest/table/
region/list?path=<path>&<parameters>'
  -u <username>:<password>
```

Parameters

Parameter	Description
path	Path to the table. • For a path on the local cluster, start the path at the volume mount point. For example, if you want to list regions for a table named `test` under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, if you want to list regions for a table named `test` under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`

Parameter	Description
start	The offset from the starting region. The default value is 0.
limit	The number of regions to return, counting from the starting region. The default value is 2147483647.
index	The name of the index for which to list region information.
output	Specifies whether the output should be `terse` or `verbose`. Default: `verbose`

Output Fields

Verbose Field Name	Terse Field Name	Field Value
primarymfs	pn	Host name and port of the primary node for this region
secondarymfs	sn	Host names and ports of the secondary nodes where this region is replicated
startkey	sk	Value of the start key for this region For the first region in a table, this value is exclusive. For all other regions, it is inclusive. See the example output.
endkey	ek	Value of the end key for this region This value is always exclusive. See the example output.
lastheartbeat	lhb	Time since last heartbeat from the region's primary node
fid	fid	The region's FID.
logicalsize	ls	The logical size (in bytes) of the region without data compression (excluding replication).
physicalsize	ps	The physical size (in bytes) of the region with data compression (excluding replication).
copypendingsize	cps	Amount of data remaining to be replicated
numberofrows	nr	Number of rows in the region
numberofrowswithdelete	nrd	Number of rows in the region, counting deleted rows
numberofspills	nsp	Number of spills for the region
numberofsegments	nsg	Number of segments in the region

Examples

Lists the Region Information for a Table

This example lists the region information for the table `newtable`.

CLI

```
maprcli table region list -path /
my.cluster.com/volume1/newtable
```

REST

```
curl -k -X GET \
  'https://r1n1.sj.us:8443/rest/table/
region/list?
path=%2Fmy.cluster.com%2Fvolume1%2Fnew
table' \
  -u mapr:mapr
```

Example Output Using the -json Option

This example shows two table regions. The value of `endkey` for the first region is the value of `startkey` for the second region. The value of `endkey` is always exclusive. So, for the first region, `endkey` shows that the first region was split with the addition of the record with the key `5190414F2E44DB732547630A9A81452539749000`; for the second region, `startkey` shows that the region begins with that record.


```
{
    "timestamp":1452554659812,
    "timeofday":"2016-01-11 03:24:19.812 GMT-0800",
    "status":"OK",
    "total":2,
    "data":[
        {
            "primarymfs":"test150.qa.lab:5660",
            "secondarymfs":"test156.qa.lab:5661, test151.qa.lab:5660",
            "startkey":"-INFINITY",
            "endkey":"5190414F2E44DB732547630A9A81452539749000",
            "lastheartbeat":0,
            "fid":"2068.100.131676",
            "logicalsize":794624,
            "physicalsize":794624,
            "copypendingsize":0,
            "numberofrows":0,
            "numberofrowswithdelete":0,
            "numberofspills":0,
            "numberofsegments":0
        },
        {

            "primarymfs":"test161.qa.lab:5660",
            "secondarymfs":"test157.qa.lab:5661, test162.qa.lab:5660",
            "startkey":"5190414F2E44DB732547630A9A81452539749000",
            "endkey":"INFINITY",
            "lastheartbeat":0,
            "fid":"2069.181.131578",
            "logicalsize":745472,
            "physicalsize":745472,
            "copypendingsize":0,
            "numberofrows":0,
            "numberofrowswithdelete":0,
            "numberofspills":0,
            "numberofsegments":0
        }
    ]
}
```

table region merge
Merges regions of a table together to reduce the number of regions that a table occupies.

This command merges the region that you specify with the region that contains the row keys that immediately follow the row keys of the specified region.

Note: Consider the table configuration when you decide to merge regions because it is possible that MapR Database might immediately split the regions after they are merged. If `autosplit` is set to true, MapR Database splits a region when the size of the region exceeds 150% of the average value (`regionsizemb`). For example, if the average value is 4096 MB, MapR Database splits a region that is larger than 6144 MB.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `splitmergeperm` permission on the table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table region merge
  -fid <regionFID>
  -path <table path>
```

REST

```
curl -k -X POST
  'http[s]://<host>:<port>/rest/
table/region/merge?fid=<region
FID>&path=<path>'
  -u <username>:<password>
```

Parameters

Parameter	Description
fid	The FID for the table region that you want to merge. The output of `maprcli table region list` lists the FIDs for the table.

Parameter	Description
path	The path to the table whose regions are being merged. • For a path on the local cluster, start the path at the volume mount point. For example, if you want to merge regions for table named `test` under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on a remote, you must also specify the cluster name in the path. For example, if you want to merge regions for table named `test` under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`

Example

Merges the specified region:

CLI

```
maprcli table region merge -path /
user/test5 -fid 2086.32.131296
```

REST

```
curl -k -X POST \
  'https://myhost:8443/rest/table/
region/merge?
path=%2Fuser%2Ftest5&fid=2086.32.13129
6' \
  -u mapr:mapr
```

`table region pack`
Manually triggers the packing of regions.

MapR Database automatically compacts or packs regions and reclaims space when 25% of the data contained in the partitions (max of 3 per tablet) has expired; however, for a time series table, you **must** run this command to reclaim space used by expired rows and to avoid read amplification, if the old rows are never accessed.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `packperm` permission on the table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table region pack
  -path <table path>
```

```
  -fid <fid>|all
  [ -nthreads <number of threads> ]
```

REST

```
curl -k -X POST
  'http[s]://
<host>:<port>/rest/table/region/pack?
path=<path>&fid=<fid>&<parameters>'
  -u <username>:<password>
```

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Parameters

Parameter	Description
path	Specifies the path to the table. • For a path on the local cluster, start the path at the volume mount point. For example, if you want to pack a table named `test` under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on another cluster, you must also specify the cluster name in the path. For example, if you want to pack a table named `customer` under volume1 in the sanfrancisco cluster, specify the following path: `/mapr/sanfrancisco/volume1/customer`
fid	Specifies that you want to pack all table regions or a single table region that you identify with a FID. The output of maprcli table region list lists the FIDs for the table.
nthreads	Specifies the number of threads allocated to process the packing of table regions. Default:16

Example

Packs the specified region:

CLI

```
maprcli table region pack -path /user/
test5 -fid 2086.32.131296
```

REST

```
curl -k -X POST \
  'https://myhost:8443/rest/table/
region/pack?
path=%2Fuser%2Ftest5&fid=2086.32.13129
6' \
  -u mapr:mapr
```

`table region split`
Splits a region in a table.

last-updated: Apr 11, 2022

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `splitmergeperm` permission on the table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table region split
  -path <path>
  -fid <fid>
```

REST

```
curl -k -X POST
  'http[s]://<host>:<port>/rest/table/
region/split?path=<path>&fid=<fid>'
  -u <username>:<password>
```

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Parameters

Parameter	Description
path	Path to the table. • For a path on the local cluster, start the path at the volume mount point. For example, if you want to split regions in a table named `test` under volume1 which has a mount point at /volume1, specify the following path: `/volume1/test` • For a path on another cluster, you must also specify the cluster name in the path. For example, if you want to split regions in a table named `customer` under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customer`
fid	The FID of the region to split. The output of `maprcli table region list` lists the FIDs for the table's regions.

Example

This example splits a region in the table `newtable`.

last-updated: Apr 11, 2022

CLI

```
maprcli table region split -path /
my.cluster.com/volume1/newtable -fid
2086.32.131296
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/table/
region/split?
path=%2Fmy.cluster.com%2Fvolume1%2Fnew
table&fid=2086.32.131296' \
  -u mapr:mapr
```

`table replica`
Performs functions related to replication of MapR Database binary and JSON tables. Replication occurs for binary-to-binary tables and JSON-to-JSON tables.
`table replica add`
Registers a table as a replica of another MapR Database binary or JSON table.

Note: You do not need to use this command if you use the `table replica autosetup` command.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on both the source volume and the target volume
- `lookupdir` on directories in the paths of both tables
- `readperm` and `replperm` permissions on the source table

Note: The **mapr user** is not treated as a superuser. MapR Database does not allow the **mapr user** to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
/opt/mapr/bin/maprcli table replica
add
  -path <table path>
  -replica <replica table path>
  [ -columns <comma separated list of
<family>[:<column>]>  ]
  [ -paused <is replication paused>
default: false ]
  [ -throttle <throttle replication
ops> default: false> ]
  [ -networkencryption <enable
on-wire encryption> default: false> ]
  [ -synchronous <is synchronous
replication> default: false ]
  [ -networkcompression <on-wire
compression type: off|on|lzf|lz4|
zlib> default: on ]
```

REST

```
curl -k -X POST
  'http[s]://<host>:<port>/rest/table/
```


last-updated: Apr 11, 2022

```
replica/add?
path=<path>&replica=<name>&<parameters
>
  -u <username>:<password>
```

Note: The **mapr user** is not treated as a superuser. MapR Database does not allow the **mapr user** to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Parameters

Parameter	Description
path	The path to the source table that you want to replicate. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `testsrc` under a volume with a mount point at /volume1, specify the following path: `/volume1/testsrc` • For a path on another cluster, you must also specify the cluster name in the path. For example, for a table named `customersrc` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customersrc`
replica	The path to the replica. • For a table on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under a volume with a mount point at `/volume1`, specify the following path: `/volume1/testdst` • For a table on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customerdst` under `volume1` in the `sanfrancisco` cluster, specify the following path: `/mapr/sanfrancisco/volume1/customerdst` **Note:** For replication to a table, the command will fail if the table in the replica path does not exist.

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>columns</td><td>By default, all columns in the source table are replicated.

If you do not want to replicate all columns in the table, you can specify specific columns to replicate:

For binary tables

Provide a comma-separated list of column families or columns from a certain column family (column family:qualifier). For example, use the following syntax to replicate the column family purchases and the column stars in the reviews column family: <code>-columns purchases,reviews:stars</code>

Note: While the column families that you specify must already exist in the source table, the columns that you specify do not have to exist in the destination table for replication to succeed. If the column is added at a later date, replication for that column will start at that time.

For JSON tables

Provide a comma-delimited list of fields to replicate. Include the full field path for each field.

Example

Suppose your table contains documents that contain this general structure:

<pre>{
     "_id" : "ID",
     "a" :
          {
"b" :
  {
       "c" :
"value",
  },
"e" : "value"
          }
}</pre>
To replicate fields a, c, and e, you would specify these field paths:</td></tr>
</table>

Parameter	Description
paused	A Boolean value that specifies whether to pause the replication so that it does not start immediately. The replication can be resumed using the replica resume command at a later time. The values are `true` or false. Default: Not paused (`false`)
throttle	A Boolean value that specifies whether to throttle replication operations. Throttle the replication stream to minimize the impact of the replication process on incoming operations during periods of heavy load. The values are true or false. Default: No throttle (`false`)
networkencryption	A Boolean value that specifies whether or not to enable on-wire encryption. The values are `true` or `false`. If you set this to `true`, the local cluster and any other cluster that is part of the replication process must be enabled for security. Default: No encryption (false)
synchronous	A Boolean value that specifies whether replication is synchronous or asynchronous. The values are `true` or `false`. Default: Asynchronous (`false`)
networkcompression	The type of on-wire compression. Default: on The types are: • `off` • on (default) • `lzf` • `lz4` • `zlib` The default compression is lz4, which can be set by specifying on or `lz4` as value.

Example

Registers a table on the local cluster as a replica of another table on the local cluster:

CLI

```
/opt/mapr/bin/maprcli table
replica add -path /volume1/
custA -replica /volume2/custA
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/table/
replica/add?
path=%2Fvolume1%2FcustA&replica=%2Fvol
ume2%2FcustA' \
  -u mapr:mapr
```

`table replica autosetup`
Sets up and starts replication between a *source* MapR Database binary or JSON table to a *replica* MapR Database binary or JSON table.

The `maprcli table replica autosetup` command performs the following steps to set up replication:

1. Creates a new table with metadata from the source table in the destination cluster.
2. Declares the new table to be a replica of the source table and ensures that replication does not begin immediately after the next step.
3. Declares the source table as an upstream source for the replica.
4. For multi-master replication, replica autosetup declares the source table to be a replica of the new table and then declares the new table to be an upstream source for the source table.
5. Loads a copy of the source data into the replica(s).
6. Clears the paused replication state to start the replication stream.

For more information about the automatic setup process, see Replica Autosetup for MapR Database Tables on page 624.

Before you set up replication for a table, verify that the cluster is setup for replication. For more information, see Preparing Clusters for Table Replication on page 1066.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on both the source volume and the target volume
- `lookupdir` on directories in the paths of both tables
- `readperm` and `replperm` permissions on the source table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with *ACE*.

Syntax

CLI

```
/opt/mapr/bin/maprcli table replica
autosetup
  -path <table path>
  -replica <replica table path>
  [ -columns <comma separated list of
<family>[:<column>]> ]
  [ -synchronous <is synchronous
replication> default: false ]
  [ -multimaster <is multi master
replication> default: false ]
  [ -throttle <throttle replication
ops> default: false ]
  [ -networkencryption <enable
on-wire encryption> default: false ]
  [ -networkcompression <on-wire
compression type: off|on|lzf|lz4|
zlib> default: on ]
  [ -directcopy <enable directcopy>
default: true ]
  [ -useexistingreplica <use existing
replica table if present> default:
false ]
```


last-updated: Apr 11, 2022

REST

```
curl -k -X POST
  'http[s]://<host>:<port>/rest/table/
replica/autosetup?
path=<path>&replica=<path>&<parameters
>'
  -u <username>:<password>
```

Parameters

path	The path to the source table that you want to replicate. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `testsrc` under a volume with a mount point at `/volume1`, specify the following path: `/volume1/testsrc` • For a path on another cluster, you must also specify the cluster name in the path. For example, for a table named `customersrc` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customersrc`
replica	The path to the replica. • For a table on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under a volume with a mount point at `/volume1`, specify the following path: `/volume1/testdst` • For a table on another cluster, you must also specify the cluster name in the path. For example, for a table named `customerdst` under`volume1` in the `sanfrancisco` cluster, specify the following path: `/mapr/sanfrancisco/volume1/customerdst`  **Note:** For replication to a table, the command will fail if the replica path you specify points to table that already exists.

last-updated: Apr 11, 2022

columns

By default, all columns in the source table are replicated.

If you do not want to replicate all columns in the table, you can specify specific columns to replicate:

For binary tables

Provide a comma-separated list of column families or columns from a certain column family (column family:qualifier). For example, use the following syntax to replicate the column family purchases and the column stars in the reviews column family: `-columns purchases,reviews:stars`

Note: While the column families that you specify must already exist in the source table, the columns that you specify do not have to exist in the destination table for replication to succeed. If the column is added at a later date, replication for that column will start at that time.

For JSON tables

Provide a comma-delimited list of fields to replicate. Include the full field path for each field.

Example

Suppose your table contains documents that contain this general structure:

```
{
     "_id" : "ID",
     "a" :
          {
"b" :
  {
       "c" :
"value",
  },
"e" : "value"
          }
}
```

To replicate fields a, c, and e, you would specify these field paths:

```
a,a.b.c,a.e
```


last-updated: Apr 11, 2022

synchronous	A Boolean value that specifies whether replication is synchronous or asynchronous. The value is either `true` or `false`. Asynchronous (`false`) is the default.
multimaster	A Boolean value that specifies whether or not to set up a multi-master topology. The value is either `true` or `false`. Basic primary-secondary topology (`false`) is the default.
throttle	A Boolean value that specifies whether or not to throttle replication operations. Throttle the replication stream to minimize the impact of the replication process on incoming operations during periods of heavy load. The value is either `true` or `false`. No throttle (`false`) is the default.
networkencryption	A Boolean value that specifies whether or not to enable on-wire encryption. The value is either `true` or `false`. No encryption (`false`) is the default. If you set this to `true`, the local cluster and any other cluster that is part of the replication process must be enabled for security.
networkcompression	The type of on-wire compression. The types are: • off • on (default) • `lzf` • `lz4` • `zlib` lz4 is the default compression which it set by parameter values `on` or `lz4`.
directcopy	A Boolean value that specifies whether or not autosetup will use the directcopy option . The value is either `true` or `false`. Autosetup with direct copy (`true`) is the default. If you set this parameter to `false`, the cluster will run autosetup without the directcopy option. For more information, see Replica Autosetup for MapR Database Tables on page 624. **Note:** If a table was originally created in MapR 5.x and the `maprcli table replica autosetup` command is specified with `directcopy=false`, then an error, “Copy Table failed for tables”, occurs. This is due to the introduction of new table meta information in 6.0. It is recommended that replication be setup using `directcopy=true` (which is the default). If the default method is not desired, then replication should be setup manually.
useexistingreplica	When the directcopy parameter is set to `true` (default), this Boolean value specifies whether or not an existing table can be used as the replica table. The value is either `true` or `false`. No reuse of existing tables (`false`) is the default. If a table exists with the specified name, and this parameter is set to false, the create table operation will fail.

last-updated: Apr 11, 2022

Example

CLI

```
/opt/mapr/bin/maprcli table
replica autosetup -path /volume1/
custBsrc -replica /volume2/custBdst
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/table/
replica/autosetup?
path=%2Fvolume2%2FcustBsrc&replica=%2F
volume2%2FcustBdst' \
  -u mapr:mapr
```

`table replica edit`
Edits the properties of a replica of a MapR Database binary or JSON table.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on both the source volume and the target volume
- `lookupdir` on directories in the paths of both tables
- `readperm` and `replperm` permissions on the source table

Note: The **mapr user** is not treated as a superuser. MapR Database does not allow the **mapr user** to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table replica edit
  -path <table path>
  -replica <replica table path>
  [ -newreplica <renamed table path> ]
  [ -columns <comma separated list of
<family>[:<column>]> ]
  [ -throttle <throttle replication
ops> ]
  [ -networkencryption <enable
on-wire encryption> ]
  [ -synchronous <is synchronous
replication> ]
  [ -networkcompression <on-wire
compression type: off|on|lzf|lz4|
zlib> ]
```

REST

```
curl -k -X POST
  'http[s]://<host>:<port>/rest/table/
replica/edit?
path=<path>&replica=<path>&<parameters
>'
  -u <username>:<password>
```

Note: The **mapr user** is not treated as a superuser. MapR Database does not allow the **mapr user** to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Parameters

Parameter	Description
path	The path to the source table that you want to replicate. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `testsrc` under a volume with a mount point at /volume1, specify the following path: `/volume1/testsrc` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customersrc` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customersrc`
replica	The path to the replica. • For a table on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under a volume with a mount point at `/volume1`, specify the following path: `/volume1/testdst` • For a table on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customerdst` under `volume1` in the `sanfrancisco` cluster, specify the following path: `/mapr/sanfrancisco/volume1/customerdst`
newreplica	The updated replica path due to a renamed replica table, renamed cluster, or changed table path. The table specified in the `replica` parameter and the table specified in the `newreplica` parameter must have the same UUID.

last-updated: Apr 11, 2022

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>columns</td><td>By default, all columns in the source table are replicated.

If you do not want to replicate all columns in the table, you can specify specific columns to replicate:

For binary tables

Provide a comma-separated list of column families or columns from a certain column family (column family:qualifier). For example, use the following syntax to replicate the column family purchases and the column stars in the reviews column family: <code>-columns purchases,reviews:stars</code>

Note: While the column families that you specify must already exist in the source table, the columns that you specify do not have to exist in the destination table for replication to succeed. If the column is added at a later date, replication for that column will start at that time.

For JSON tables

Provide a comma-delimited list of fields to replicate. Include the full field path for each field.

Example

Suppose your table contains documents that contain this general structure:

<pre>{
     "_id" : "ID",
     "a" :
          {
"b" :
  {
       "c" :
"value",
  },
"e" : "value"
          }
}</pre>
To replicate fields a, c, and e, you would specify these field paths:</td></tr>
</table>

Parameter	Description
throttle	A Boolean value that specifies whether or not to throttle replication operations. Throttle the replication stream to minimize the impact of the replication process on incoming operations during periods of heavy load. The values are `true` or `false`. No throttle (`false`) is the default.
networkencryption	A Boolean value that specifies whether or not to enable on-wire encryption. The values are `true` or `false`. No encryption (`false`) is the default. If you set this to true, the local cluster and any other cluster that is part of the replication process must be enabled for security.
synchronous	A Boolean value that specifies whether replication is synchronous or asynchronous. The values are `true` or `false`. Asynchronous (`false`) is the default.
networkcompression	The type of on-wire compression. The types are: • off • on (default) • `lzf` • `lz4`. This is the default • `zlib` lz4 is the default compression which it set by parameter values on or lz4.

Examples

Changes the replica path to reflect that replica `t2dst` is renamed to `t2dst_new`:

CLI

```
maprcli table replica edit -path /
volume1/t1src -replica /volume1/t2dst
\
  -newreplica /volume1/t2dst_new
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/table/
replica/edit?
path=%2Fvolume1%2Ft1&replica=%2Fvolume
1%2Ft2&newreplica=%2Fvolume1%2Ft2_new'
 \
  -u mapr:mapr
```

Changes the column families to replicate:

CLI

```
maprcli table
replica edit -path /volume1/
custAsrc -replica /volume2/custAdst \
  -columns purchases, reviews, returns
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/table/
replica/edit?
path=%2Fvolume1%2FcustAsrc&replica=%2F
volume2%2FcustAdst&columns=purchases,r
eviews,returns' \
  -u mapr:mapr
```

table replica list
Lists replicas and the associated replica statistics for a specified MapR Database binary or JSON table. By default, replica statistics are updated every five minutes.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` on the volume
- `lookupdir` on directories in the path

Note: The **mapr user** is not treated as a superuser. MapR Database does not allow the **mapr user** to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
/opt/mapr/bin/maprcli table replica
list
  -path <table path>
  [ -refreshnow true|false ]
```

REST

```
curl -k -X GET
  'http[s]://
<host>:<port>/rest/table/replica/list?
path=<path>&refreshnow=false'
  -u <username>:<password>
```

Parameters

Parameter	Description
path	The path to the table that you want to list replicas for. • For a table on the local cluster, start the path at the volume mount point. For example, for a table named `test` under a volume with a mount point at `/volume1`, specify the following path: `/volume1/test` • For a table on another cluster, you must also specify the cluster name in the path. For example, for a table named `customer` under `volume1` in the `sanfranciscocluster`, specify the following path: `/mapr/sanfrancisco/volume1/customer`

last-updated: Apr 11, 2022

Parameter	Description
refreshnow	A Boolean value that specifies if you want to trigger an immediate update of the replica statistics. The values are `true` or `false`. By default, the value is `false`; the command lists the current version of the replica statistics, which could be a maximum of five minutes old.

Output

Lists information about each replica for the specified table.

Output Data Fields

The following fields display for each replica.

Field	Description
cluster	The cluster on which the replica resides.
table	The table name for the replica.
type	The table type.
paused	A Boolean values that specifies if replication is paused.
replicaPath	The table replica path.
replicaState	The replication state. For information about the replication states, see Table Replication States on page 625.
throttle	A Boolean value that specifies if replication is throttled.
idx	The internal index value.
networkencryption	A Boolean value that specifies if replication is encrypted.
synchronous	A Boolean value that specifies whether replication is synchronous or asynchronous.
networkcompression	The type of on-wire compression.
isUptodate	A Boolean value that specifies if the replica is up-to-date.
minPendingTS	The epoch time in milliseconds of the oldest operation that has yet to be replicated to the replica.
maxPendingTS	The epoch time in milliseconds of the newest operation that has yet to be replicated to the replica.
bytesPending	The number of bytes that have yet to be replicated to the replica.
putsPending	The number of puts that have yet to be replicated to the replica.
bucketsPending	The number of buckets that have yet to be replicated to the replica.
uuid	The table UUID.

Field	Description
copyTableCompletionPercentage	When replica autosetup with directcopy is in progress, this value is the percentage of data from the source that has been copied to the replica. After replication is setup, the value remains at 100. **Note:** When replicating MapR Database data, the copyTablePercentageCompletion data may re-adjust to a lower rate. This depends on table region (also referred to as tablets) splits and merges as well as the rate of incoming data to replicating data.
errors	If applicable, an error is displayed.

Sample Output

```
{
  "timestamp":1485555420019,
  "timeofday":"2017-01-27 10:17:00.019 GMT+0000",
  "status":"OK",
  "total":1,
  "data":[
    {
      "cluster":"cluster",
      "table":"/dst",
      "type":"MapRDB",
      "replicaPath":"/dst",
      "replicaState":"REPLICA_STATE_REPLICATING",
      "paused":false,
      "throttle":false,
      "idx":1,
      "networkencryption":false,
      "synchronous":false,
      "networkcompression":"lz4",
      "isUptodate":true,
      "minPendingTS":0,
      "maxPendingTS":0,
      "bytesPending":0,
      "putsPending":0,
      "bucketsPending":0,
      "uuid":"4164f38a-b4ed-0302-f929-0d8bc68b5800",
      "copyTableCompletionPercentage":100
    }
    ]
}
```

Example

Lists replicas for the `custA` table:

CLI

```
/opt/mapr/bin/maprcli table replica
list -path /volume1/custA
```

REST

```
curl -k -X GET \
  'https://r1n1.sj.us:8443/rest/table/
replica/list?path=%2Fvolume1%2FcustA'
\
  -u mapr:mapr
```


last-updated: Apr 11, 2022

table replica pause
Pauses the replication of data from a *source* MapR Database binary or JSON table to a *replica* MapR Database binary or JSON table during autosetup and replication phases.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on both the source volume and the target volume
- `lookupdir` on directories in the paths of both tables
- `replperm` permissions on the source table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table replica pause
  -path <table path>
  -replica <replica table path>
```

REST

```
curl -k -X POST
  'http[s]://
<host>:<port>/rest/table/replica/
pause?path=<path>&replica=<path>'
  -u <username>:<password>
```

Parameters

Parameter	Description
path	The path to the source table. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `testsrc` under a volume with a mount point at /volume1, specify the following path: `/volume1/testsrc` • For a path on another cluster, you must also specify the cluster name in the path. For example, for a table named `customersrc` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customersrc`

last-updated: Apr 11, 2022

Parameter	Description
replica	The path to the replica that will receive updates from the source. • For a table on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under a volume with a mount point at `/volume1`, specify the following path: `/volume1/testdst` • For a table on another cluster, you must also specify the cluster name in the path. For example, for a table named `customerdst` under `volume1` in the `sanfrancisco` cluster, specify the following path: `/mapr/sanfrancisco/volume1/customerdst`

Example

Sets the replication state to paused:

CLI

```
maprcli table
replica pause -path /volume1/
custAsrc -replica /volume2/custAdst
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/table/
replica/pause?
path=%2Fvolume1%2FcustAsrc&replica=%2F
volume2%2FcustAdst' \
  -u mapr:mapr
```

`table replica remove`
De-registers the specified MapR Database binary or JSON table as a replica.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on both the source volume and the target volume
- `lookupdir` on directories in the paths of both tables
- `replperm` permissions on the source table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

After running this command, the specified table or index is no longer a replica of the source table and will no longer receive updates from the source table.

In addition, run the `table upstream remove` command to remove the association between the source table and the replica table.

last-updated: Apr 11, 2022

Syntax

CLI

```
/opt/mapr/bin/maprcli table replica
remove
  -path <table path>
  -replica <replica table path>
```

REST

```
curl -k -X POST
  'http[s]://
<host>:<port>/rest/table/replica/
remove?path=<path>&replica=<path>'
  -u <username>:<password>
```

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Parameters

Parameter	Description
path	The path to the source table that is being replicated. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `testsrc` under a volume with a mount point at /volume1, specify the following path: `/volume1/testsrc` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customersrc` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customersrc`
replica	The path to the replica. • For a table on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under a volume with a mount point at `/volume1`, specify the following path: `/volume1/testdst` • For a table on another cluster, you must also specify the cluster name in the path. For example, for a table named `customerdst` under `volume1` in the `sanfrancisco` cluster, specify the following path: `/mapr/sanfrancisco/volume1/customerdst`

Example

De-registers table `custAdst` as a replica of table `custAsrc`:

CLI

```
/opt/mapr/bin/maprcli table
replica remove -path /volume1/
custAsrc -replica /volume2/custAdst
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/table/
replica/remove?
path=%2Fvolume1%2FcustAsrc&replica=%2F
volume2%2FcustAdst' \
  -u mapr:mapr
```

`table replica resume`
Resumes replication between a *source* MapR Database binary or JSON table and a *replica* of that table. Replication can be paused during autosetup and replication phases. When replication resumes, it continues from where it left off.

Permissions Required

To run this command, your user ID must have the following permissions:

- `readAce` and `writeAce` on both the source volume and the target volume
- `lookupdir` on directories in the paths of both tables
- `replperm` permissions on the source table

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

CLI

```
maprcli table replica resume
  -path <table path>
  -replica <replica table path>
```

REST

```
curl -k -X POST
  'http[s]://
<host>:<port>/rest/table/replica/
resume?path=<path>&replica=<path>'
  -u <username>:<password>
```

Parameters

Parameter	Description
path	The path to the table that will be replicated. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `testsrc` under a volume with a mount point at /volume1, specify the following path: `/volume1/testsrc` • For a path on another cluster, you must also specify the cluster name in the path. For example, for a table named `customersrc` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customersrc`

last-updated: Apr 11, 2022

Parameter	Description
replica	The path to the replica. • For a table on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under a volume with a mount point at `/volume1`, specify the following path: `/volume1/testdst` • For a table on another cluster, you must also specify the cluster name in the path. For example, for a table named `customerdst` under `volume1` in the `sanfrancisco` cluster, specify the following path: `/mapr/sanfrancisco/volume1/customerdst`

Example

CLI

```
maprcli table
replica resume -path /volume1/
custAsrc -replica /volume2/custAdst
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/table/
replica/resume?
path=%2Fvolume1%2FcustAsrc&replica=%2F
volume2%2FcustAdst' \
  -u mapr:mapr
```

`table upstream`
Performs functions related to upstream sources for table replication.
`table upstream add`
Adds a binary table as upstream source for a replica.

Note: You do not need to use this command if you use the `table replica autosetup` command.

Syntax

CLI

```
maprcli table upstream add
  -path <table path>
  -upstream <upstream table
path>
```

REST

```
curl -k -X POST
  'http[s]://
<host>:<port>/rest/table/upstream/add?
path=<path>&upstream=<name>'
  -u <username>:<password>
```

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

last-updated: Apr 11, 2022

Parameters

Parameter	Description
path	The path to the replica. • For a path to a table on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under volume1 which has a mount point at /volume1, specify the following path: `/volume1/testdst` • For a path to a table on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customerdst` under volume1 in the sanfrancisco cluster, specify the following path: `/mapr/sanfrancisco/volume1/customerdst`
upstream	The path to the source table. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `testsrc` under volume1 which has a mount point at /volume1, specify the following path: `/volume1/testsrc` • For a path on another cluster, you must also specify the cluster name in the path. For example, for a table named `customersrc` under volume1 in the sanfrancisco cluster, specify the following path: `/mapr/sanfrancisco/volume1/customersrc`

Example

Adds `company1src` as the upstream source for replica `company1dst`:

CLI

```
maprcli table
upstream add -path /volume2/
company1dst -upstream /volume1/
company1src
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/table/
upstream/add?
path=%2Fvolume2%2Fcompany1dst&upstream
=%2Fvolume1%2Fcompany1src' \
  -u mapr:mapr
```

`table upstream list`
Lists the binary tables that replicate data to the specified replica binary table.

Syntax

CLI

```
maprcli table upstream list
  -path <table path>
```


last-updated: Apr 11, 2022

REST

```
curl -k -X GET
  'http[s]://<host>:<port>/rest/table/
upstream/list?path=<path>'
  -u <username>:<password>
```

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Parameters

Parameter	Description
path	The path to the replica. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under `volume1` which has a mount point at `/volume1`, specify the following path: `/volume1/testdst` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named customerdst under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customerdst`

Sample Output

```
maprcli table upstream list -path /volume2/company1 -json
{
        "timestamp":1423162601288,
        "timeofday":"2015-02-05 10:56:41.288 GMT-0800",
        "status":"OK",
        "total":1,
        "data":[
                {
                        "cluster":"mycluster",
                        "table":"/volume1/company1",
                        "idx":1,
                        "uuid":"P?\\x18\\xCC\\x17\\xB1&\\xA7i,\\x04\\xBB\
\xB8\\xD3T\\x00"
                }
        ]
}
```

Example

Lists sources that replicate data to the replica `/volume2/company1`:

CLI

```
maprcli table upstream list -path /
volume2/company1 -json
```

REST

```
curl -k -X GET \
  'https://
r1n1.sj.us:8443/rest/table/upstream/
```

```
list?path=%2Fvolume2%2Fcompany1' \
  -u mapr:mapr
```

`table upstream remove`
Un-registers a binary table as an upstream source for a replica.

Note: This step is separate from the `table replica remove` command, which stops replication updates to a replica.

Syntax

CLI

```
maprcli table upstream remove
  -path <table path>
  -upstream <upstream table path>
```

REST

```
curl -k -X POST
  'http[s]://
<host>:<port>/rest/table/upstream/
remove?path=<path>&upstream=<path>'
  -u <username>:<password>
```

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this command unless that user is given the relevant permission or permissions with access-control expressions.

Parameters

Parameter	Description
path	The path to the replica. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under `volume1` which has a mount point at `/volume1`, specify the following path: `/volume1/testdst` • For a path on another cluster, you must also specify the cluster name in the path. For example, for a table named `customerdst` under `volume1` in the `sanfrancisco` cluster, specify the following path: `/mapr/sanfrancisco/volume1/customerdst`
upstream	The path to the source table. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `testsrc` under `volume1` which has a mount point at `/volume1`, specify the following path: `/volume1/testsrc` • For a path on another cluster, you must also specify the cluster name in the path. For example, for a table named `customersrc` under `volume1` in the `sanfrancisco` cluster, specify the following path: `/mapr/sanfrancisco/volume1/customersrc`

last-updated: Apr 11, 2022

Example

Removes `company1src` as the upstream source for replica `company1dst`:

CLI

```
maprcli table upstream remove -path /
volume2/company1dst -upstream /
volume1/company1src
```

REST

```
curl -k -X POST \
  'https://r1n1.sj.us:8443/rest/table/
upstream/remove?
path=%2Fvolume2%2Fcompany1dst&upstream
=%2Fvolume1%2Fcompany1src' \
  -u mapr:mapr
```

tier
Lets you create, modify, remove, and retrieve list of tiers and tiering rules.

tier create
Creates a new tier.

Syntax

CLI

```
maprcli tier create
       -name <tier_name>
       -type cold|ectier
       [ -url <tier_url> ]
       [ -credential
<credentials_file_path> ]
       [ -tag <object_store_type> ]
       [ -credential_str
<tier_credentials> ]
       [ -dbtopology
<metadata_volume_path> ]
       [ -cluster <cluster_name> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host:port>/rest/tier/create?<parameters>`

Usage

To create a warm tier:

```
maprcli tier create
          [ -cluster <cluster_name> ]
          -name <tier_name>
          -type ectier
          [ -dbtopology <path> ]
```


last-updated: Apr 11, 2022

To create a cold tier:

```
maprcli tier create
          [ -cluster <cluster_name> ]
          -name <tier_name>
          -type cold -url <tier_URL>
          -credential|credential_str <credential>
          [ -dbtopology <path> ]
          [ -tag S3-AWS|S3-GCS|S3-HDS|S3-IBM|Azure-Blobs|S3-Others ]
```

Note: The `-tag` parameter is required for Azure.

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`credential`	(For tier of type cold only) The path to the credentials file to use for accessing the tier. The credentials file must already exist on the node from where the tier is being created. For more information, see Setting up a Credentials File for Connecting to a Cold Tier Using the CLI or REST API on page 962. **Note:** Either this or `-credential_str` is required for creating a cold tier.
`credential_str`	(For tier of type `cold` only) The credentials, access key and secret key, bucket name, and region in JSON format. Either this or `-credential` is required for creating a cold tier.
`dbtopology`	The rack path to the volume where metadata is stored in DB tables. The default value is `/data`.
`name`	The name of the tier.
`tag`	(For tier of type `cold` only) The object store to connect to. Value can be one of the following: • S3-GCS (for Google Cloud Platform) • S3-HDS (for Hitachi HCP) • S3-IBM (for IBM Cloud Object Storage) • S3-AWS (for Amazon AWS) • Azure-Blobs (for Microsoft Azure) • S3-Others (for other all vendors) The MAST Gateway uses this to determine the connector library (such as libcurl, etc.) to use. See Specifying the Vendor/Object Store for a Cold Tier on page 964 for more information on the object store. **Note:** This parameter is required for Azure.

last-updated: Apr 11, 2022

Parameter	Description
`type`	The type of tier to create. Value can be: • `cold` — to offload to low-cost storage alternative on the cloud • `ectier` — to offload to low-cost storage alternative on the MapR cluster
`url`	(For tier of type `cold` only) The URL (or endpoint) of the tier in the following format: `<protocol>://<IP|hostname>.<domain>`. Fore more information, see Specifying the Vendor/Object Store for a Cold Tier on page 964. When specifying the URL (for S3), use double quotes. If the protocol is `https`, the MAST Gateway uses HTTPS to upload data to the cold-tier. If the cold-tier storage does not support HTTPS, all tier related operations will fail. If the cold tier does not support HTTPS, set the protocol to `http`, which is the default.

Examples

Create a cold tier for offloading to S3:

CLI

```
# /opt/mapr/bin/maprcli
tier create -name
ksTestCold -type cold -url
"s3.amazonaws.com" -credential
credentials.txt -json
{
    "timestamp":1519669953410,
    "timeofday":"2018-02-26
10:32:33.410 GMT-0800 AM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created tier:
'ksTestCold'"
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/create?
name=ksTestCold&type=cold&url=s3.amazo
naws.com&credential=/root/
credentials.txt' --user mapr:mapr
{"timestamp":1519679457859,"timeofday"
:"2018-02-26 01:10:57.859 GMT-0800
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
tier: 'ksTestCold'"]}
```

Create a EC tier for offloading to a erasure coded volume on the MapR cluster:

last-updated: Apr 11, 2022

CLI

```
# /opt/mapr/bin/maprcli tier
create -name ksTestEC -type
ectier -json
{
    "timestamp":1519664750448,
    "timeofday":"2018-02-26
09:05:50.448 GMT-0800 AM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created tier:
'ksTestEC'"
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/create?
name=ksTestEC&type=ectier' --user
mapr:mapr
{"timestamp":1519679884411,"timeofday"
:"2018-02-26 01:18:04.411 GMT-0800
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
tier: 'ksTestEC'"]}
```

Create a cold tier by sending the credentials as a string:

CLI

```
# maprcli tier create -name
testCold -type cold -url
"s3.amazon.com" -credential_str
'{"bucketName":"testbucket","credentia
ls":
{"accessKey":"ABCDEFGHIJKLM","secretKe
y":"OPQRSTUVWXYZ"}}' -json
{
    "timestamp":1526406945863,
    "timeofday":"2018-05-15
10:55:45.863 GMT-0700 AM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created tier:
'testCold'"
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/create?
name=testCold&type=cold&url=s3.amazon.
com&credential_str=%7B%22bucketName%22
%3A%22testbucket%22%2C%22credentials%2
2%3A%7B%22accessKey%22%3A%ABCDEFGHIJKL
```

```
%22%2C%22secretKey%22%3A%22OPQRSTUVWXY
Z%22%7D%7D' --user mapr:mapr
{"timestamp":1526483636503,"timeofday"
:"2018-05-16 08:13:56.503 GMT-0700
AM","status":"OK","total":0,"data":
[],"messages":["Successfully created
tier: 'testCold'"]}
```

tier info
Retrieves information about a tier.

Syntax

CLI

```
$ maprcli tier info
    -name <tier_name>
    [ -cluster <cluster_name> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host:port>/rest/tier/info?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`name`	The name of the tier.

Output

The command returns the following:

`volume`	The name of the volume associated with the tier.
`tiertype`	The type of tier. Value can be one of the following: • cold • ectier
`tierid`	The ID of the tier.
`dbtopology`	The topology of the volume associated with the tier.
`dbvolumeid`	The ID of the volume associated with the tier.
`tiername`	The name of the tier.
`bucketname`	The name of the bucket. The value is displayed for cold tiers only.
`region`	The region. The value is displayed for cold tiers only.

objectstoretype	The type of object store (for cold tiers only). Value can be one of the following: • S3-GCS • S3-HDS • S3-IBM • S3-Others • Azure-Blobs
url	The tier URL. The value is displayed for cold tiers only.

Examples

Retrieve information about a warm tier:

CLI

```
# maprcli tier info -name testWarm
volume
tiertype  dbtopology  dbvolumeid
tierid    tiername
mapr.internal.tier.testWarm
ectier    /data       201186661
74117928  testWarm
```

REST

```
# curl -k -X GET 'https://
abc.sj.us:8443/rest/tier/info?
name=testWarm' --user mapr:mapr
{"timestamp":1530987914127,"timeofday"
:"2018-07-07 11:25:14.127 GMT-0700
AM","status":"OK","total":1,"data":
[{"tierid":"74117928","tiername":"test
Warm","tiertype":"ectier","volume":"ma
pr.internal.tier.testWarm","dbtopology
":"/data","dbvolumeid":201186661}]}
```

Retrieve information about a cold tier:

CLI

```
# maprcli tier info -name testCold
volume
tiertype  dbtopology  dbvolumeid
tierid    tiername  bucketname
region     objectstoretype
url
mapr.internal.tier.testCold
cold      /data       13372843
49971858  testCold  testbucket
us-east-1  S3-AWS           http://
s3.amazon.com
```

REST

```
# curl -k -X GET 'https://
abc.sj.us:8443/rest/tier/info?
name=testCold' --user mapr:mapr
{"timestamp":1530987683808,"timeofday"
:"2018-07-07 11:21:23.808 GMT-0700
AM","status":"OK","total":1,"data":
```

```
[{"tierid":"49971858","tiername":"test
Cold","tiertype":"cold","url":"http://
s3.amazon.com","bucketname":"testbucke
t","region":"us-east-1","volume":"mapr
.internal.tier.testCold","dbtopology":
"/
data","dbvolumeid":13372843,"objectsto
retype":"S3-AWS"}]}
```

tier list
Lists the tiers on the cluster.

Syntax

CLI

```
maprcli tier list
    [ -cluster <cluster_name> ]
    [ -sortby <attribute> ]
    [ -sortorder asc|desc ]
```

REST

Request Type	GET
Request URL	`http[s]://<host:port>/rest/tier/list?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`sortby`	Specifies one of the following attributes to sort the list of tiers by: `tierid`, `tiername`, `tiertype`, `url`, `throttling`, `bucketname`, `region`, `objectstoretype`, `volume`, `topology`
`sortorder`	The order to sort the results by. Value can be: • `asc` - for ascending order • `desc` - for descending order

Output

The command returns the following:

`volume`	The name of the tiered volume.
`tiertype`	The type of tier. Value can be one of the following: • cold • ectier
`dbtopology`	The topology of the metadata volume associated with the tier.
`dbvolumeid`	The ID of the metadata volume associated with the tier.

`tierid`	The ID of the tier.
`tiername`	The name of the tier.
`bucketname`	The name of the bucket. The value is displayed for cold tiers only.
`region`	The region. The value is displayed for cold tiers only.
`objectstoretype`	The type of object store (for cold tiers only). Value can be one of the following: • S3-AWS • S3-GCS • S3-HDS • S3-IBM • S3-Others • Azure-Blobs
`url`	The tier URL. The value is displayed for cold tiers only.

Example

Get the list of tiers:

CLI

```
# maprcli tier list
volume
tiertype  dbtopology  dbvolumeid
tierid     tiername    bucketname
region     objectstoretype
url
mapr.internal.tier.ksTestCold
cold      /data       135415553
30712925   ksTestCold  ksekhar-test
us-east-1  S3-AWS           http://
s3.amazonaws.com
mapr.internal.tier.testCold
cold      /data       192997092
189158428  testCold    testbucket
us-east-1  S3-AWS           http://
s3.amazon.com
mapr.internal.tier.ksTestEC
ectier    /data       87658196
198680137  ksTestEC
```

REST

```
# curl -k -X GET 'https://
abc.sj.us:8443/rest/tier/list' --user
mapr:mapr
{"timestamp":1533055528861,"timeofday"
:"2018-07-31 09:45:28.861 GMT-0700
AM","status":"OK","total":0,"data":
[{"tierid":"30712925","tiername":"ksTe
stCold","tiertype":"cold","url":"http:
//
s3.amazonaws.com","bucketname":"ksekha
r-test","region":"us-east-1","volume":
"mapr.internal.tier.ksTestCold","dbtop
ology":"/
data","dbvolumeid":135415553,"objectst
```


last-updated: Apr 11, 2022

```
oretype":"S3-AWS"},
{"tierid":"189158428","tiername":"test
Cold","tiertype":"cold","url":"http://
s3.amazon.com","bucketname":"testbucke
t","region":"us-east-1","volume":"mapr
.internal.tier.testCold","dbtopology":
"/
data","dbvolumeid":192997092,"objectst
oretype":"S3-AWS"},
{"tierid":"198680137","tiername":"ksTe
stEC","tiertype":"ectier","volume":"ma
pr.internal.tier.ksTestEC","dbtopology
":"/data","dbvolumeid":87658196}
```


tier modify
Modifies the credentials used to access tier.

Syntax

CLI

```
maprcli tier modify
     -name <tier_name>
     [ -credential
<path_to_credentials_file> ]
     [ -credential_str
<tier_credentials> ]
     [ -cluster <cluster_name> ]
     [ -force true|false ]
     [ -tag <object_store_type> ]
     [ -url <tier_url> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host:port>/rest/tier/modify?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`credential`	(For cold tier only) The path to the credentials file to use to access the tier. **Note:** You cannot modify the bucket name in the credentials file after the tier is created; only the accesskey and the secretkey can be modified. For more information, see Setting up a Credentials File for Connecting to a Cold Tier Using the CLI or REST API on page 962.
`credential_str`	(For cold tier only) The region, bucket, and credentials, access key and secret key, specified in JSON format. Either this or `-credential` is required to connect to a cold tier.

Parameter	Description
`force`	Required to force a change of any of the following: • Bucket on the tier where data is offloaded. • Region where the bucket resides. • URL (or endpoint) of the tier. Value can be one of the following: • `true` — to force a change • `false` — to not change The default value is `false`.
`name`	The name of the tier.
`tag`	(For cold tier only) The tier to connect to. Value can be one of the following: • S3-GCS • S3-HDS • S3-IBM • S3-AWS • S3-Others • Azure-Blobs The MAST Gateway uses this to determine the connector library (such as libcurl, etc.) to use. **Note:** You must specify this parameter to connect to Azure.
`url`	(For cold tier only) The URL (or endpoint) of the tier. This can be modified only with the `-force` option. See Specifying the Vendor/Object Store for a Cold Tier on page 964 for information on the tier endpoints and supported authentication protocols. **Note:** If the credentials for the new URL are different, specify the new credentials through the credentials file or using the `credential_str` parameter.

Examples

Modify the credentials (credential file) used to access the tier:

CLI

```
# /opt/mapr/bin/maprcli tier
modify -name testCold -credential
credentials.txt -json
{
    "timestamp":1519670281090,
    "timeofday":"2018-02-26
10:38:01.090 GMT-0800 AM",
    "status":"OK",
    "total":0,
```


last-updated: Apr 11, 2022

```
    "data":[

    ],
    "messages":[
        "Successfully updated tier:
'ksTestCold'"
    ]
}
```

REST

```
# curl -k -X POST 'https://
10.10.82.24:8443/rest/tier/modify?
name=testCold&credential=credentials.t
xt' --user mapr:mapr
{"timestamp":1526485277061,"timeofday"
:"2018-05-16 08:41:17.061 GMT-0700
AM","status":"OK","total":0,"data":
[],"messages":["Successfully updated
tier: 'testCold'"]}
```

Modify the tier by passing the credentials as a string:

CLI

```
# maprcli tier modify -name
testCold -credential_str
'{"bucketName":"testbucket","credentia
ls":
{"accessKey":"ABCDEFGHIJKL","secretKey
":"MNOPQRSTUVWXYZ"}}' -json
{
    "timestamp":1526484682668,
    "timeofday":"2018-05-16
08:31:22.668 GMT-0700 AM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully updated tier:
'testCold'"
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/modify?
name=testCold&credential_str=%7B%22buc
k3A%22testbucket%22%2C%22credentials%2
2%3A%7B%22accessKey%22%3A%22ABCDEFGHIJ
KLMN%22%2C%22secretKey%22%3A%22OPQRSTU
VWXYZ%22%7D%7D' --user mapr:mapr
{"timestamp":1526485116177,"timeofday"
:"2018-05-16 08:38:36.177 GMT-0700
AM","status":"OK","total":0,"data":
[],"messages":["Successfully updated
tier: 'testCold'"]}
```

tier remove
Removes a tier.

Note: You cannot remove a tier currently associated with a volume.

Syntax

CLI

```
$ maprcli tier remove
    -name <tier_name>
    [ -cluster <cluster_name> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host:port>/rest/tier/remove?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`name`	The name of the tier to remove.

Examples

Remove a tier (specified by name):

CLI

```
# /opt/mapr/bin/maprcli tier
remove -name testCold -json
{
    "timestamp":1521064355911,
    "timeofday":"2018-03-14
02:52:35.911 GMT-0700 PM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully deleted tier:
'testCold'"
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/remove?
name=testCold' --user mapr:mapr
{"timestamp":1526485963448,"timeofday"
:"2018-05-16 08:52:43.448 GMT-0700
AM","status":"OK","total":0,"data":
[],"messages":["Successfully deleted
tier: 'testCold'"]}
```

tier rule create
Creates a rule for offloading data to a tier.

last-updated: Apr 11, 2022

Syntax

CLI

```
$ maprcli tier rule create
    -name <rule_name>
    -expr <regular_expression>
   [ -cluster <cluster_name> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host:port>/rest/tier/rule/create?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.

last-updated: Apr 11, 2022

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td>expr</td><td colspan="2">The criteria for offloading data. The criteria can be defined using a combination of the following:</td></tr>
<tr><td></td><td>u</td><td>Username or user ID, as configured in the OS registry (such as /etc/passwd file, LDAP, etc.), of a specific user.
Usage: u:<username or user ID></td></tr>
<tr><td></td><td>g</td><td>Group name or group ID, as configured in the OS registry (such as /etc/group file, LDAP, etc.), of a specific group.
Usage: g:<groupname or group ID></td></tr>
<tr><td></td><td>m</td><td>(mtime) Time (in seconds or days) since the files were last modified. The number of seconds can be specified by appending s to value and the number of days can be specified by appending d to the value.
Usage:
• "m:<value>s" — specifies mtime in seconds
• "m:<value>d" — specifies mtime in days
All files that are not modified since the specified amount of time, are offloaded.
Note: If the system time on CLDB and file server nodes are different, the mtime rule for offloading data may not work as intended.</td></tr>
<tr><td></td><td>s</td><td>The size of the file in bytes, kilobytes, megabytes, or gigabytes. The size of the file can be specified by appending one of the following to the value: b for bytes, k for kilobytes, m for megabytes, or g for gigabytes.
Usage
• "s:<value>b" — specifies file size in bytes
• "s:<value>k" — specifies file size in KB
• "s:<value>m"</td></tr>
</table>

Parameter	Description
name	The name of the rule.

Examples

Create a rule to offload files older than a year:

CLI

```
/opt/mapr/bin/maprcli tier
rule create -name rule1 -expr
"m:365d" -json
{
    "timestamp":1519681290079,
    "timeofday":"2018-02-26
01:41:30.079 GMT-0800 PM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created rule:
'rule1'"
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/rule/create?
name=rule1&expr=m:365d' --user
mapr:mapr
{"timestamp":1519681475025,"timeofday"
:"2018-02-26 01:44:35.025 GMT-0800
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
rule: 'rule1'"]}
```

Create a rule to offload files larger than 5 GB:

CLI

```
# /opt/mapr/bin/maprcli tier rule
create -name rule2 -expr "s:5g" -json
{
    "timestamp":1519681586774,
    "timeofday":"2018-02-26
01:46:26.774 GMT-0800 PM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created rule:
'rule2'"
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/rule/create?
```

```
name=rule2&expr=s:5g' --user mapr:mapr
{"timestamp":1519681667766,"timeofday"
:"2018-02-26 01:47:47.766 GMT-0800
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
rule: 'rule2'"]}
```

Create rule to offload files whose owner is m7user1:

CLI

```
# /opt/mapr/bin/maprcli tier
rule create -name rule3 -expr
"u:m7user1" -json
{
    "timestamp":1519682014521,
    "timeofday":"2018-02-26
01:53:34.521 GMT-0800 PM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created rule:
'rule3'"
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/rule/create?
name=rule3&expr=u:m7user1' --user
mapr:mapr
{"timestamp":1519682095080,"timeofday"
:"2018-02-26 01:54:55.080 GMT-0800
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
rule: 'rule3'"]}
```

Create rule to offload all files:

CLI

```
# /opt/mapr/bin/maprcli tier rule
create -name rule4 -expr "p" -json
{
    "timestamp":1519682694183,
    "timeofday":"2018-02-26
02:04:54.183 GMT-0800 PM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created rule:
'rule4'"
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/rule/create?
name=rule4&expr=p' --user mapr:mapr
{"timestamp":1519682828031,"timeofday"
:"2018-02-26 02:07:08.031 GMT-0800
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
rule: 'rule4'"]}
```

Create rule to not offload any files:

```
# /opt/mapr/bin/maprcli tier rule create -name rule5 -expr "" -json
{
    "timestamp":1519682947271,
    "timeofday":"2018-02-26 02:09:07.271 GMT-0800 PM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created rule: 'rule5'"
    ]
}
```

Create a rule, called testRule, for offloading all files owned by user m7user1 or for offloading files owned by user mapr and whose size is greater than 5 GB or whose file modification timestamp is greater than 365 (days):

CLI

```
# /opt/mapr/bin/maprcli tier
rule create -name testRule -expr
"u:m7user1 | (u:mapr & (s:5g |
m:365d))" -json
{
    "timestamp":1519683138305,
    "timeofday":"2018-02-26
02:12:18.305 GMT-0800 PM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created rule:
'testRule'"
    ]
}
```

REST

```
# curl -k -X POST 'https://
10.10.82.24:8443/rest/tier/rule/
create?
name=testRule&expr=u%3Am7user1%7C%28u%
3Amapr%26%28s%3A5g%20%7C%20m%3A365d%29
%29' --user mapr:mapr
{"timestamp":1526488621687,"timeofday"
:"2018-05-16 09:37:01.687 GMT-0700
AM","status":"OK","total":0,"data":
```

```
[],"messages":["Successfully created
rule: 'testRule'"]}
```

tier rule info
Retrieves information on a rule (specified by name).

Syntax

CLI

```
maprcli tier rule info
    -name <rule_name>
    [ -output verbose ]
    [ -cluster <cluster_name> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host:port>/rest/tier/rule/info?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`name`	The name of the rule.
`output`	The type of output. The default value is verbose.

Output

The command returns the following:

`expression`	The rules defined using a combination of expressions.
`inuse`	Whether (`true`) or not (`false`) the rule is associated with a volume.
`rulename`	The name of the rule.
`ruleid`	The ID of the rule.

Example

Retrieve information on the rule named testRule:

CLI

```
# maprcli tier rule info -name
testRule
expression
  inuse  rulename  ruleid
u:m7user1 | (u:mapr & (s:5g |
m:365d))  true   testRule  2
```

REST

```
# curl -k -X GET 'https://
abc.sj.us:8443/rest/tier/rule/info?
```

```
name=testRule' --user mapr:mapr
{"timestamp":1528147823598,"timeofday"
:"2018-06-04 02:30:23.598 GMT-0700
PM","status":"OK","total":1,"data":
[{"ruleid":"2","rulename":"testRule","
expression":"u:m7user1 | (u:mapr &
(s:5g | m:365d))","inuse":"true"}]}
```

tier rule list
Retrieves the list of rules for offloading data.

Syntax

CLI

```
$ maprcli tier rule list
    [ -output terse|verbose ]
    [ -cluster cluster_name ]
    [ -sortby <attribute> ]
    [ -sortorder <asc|desc> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host:port>/rest/tier/rule/list?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`output`	Specifies whether the output should be `terse` or `verbose`. Default: `verbose`.
`sortby`	The attributes by which to sort the list of tiers. Value can be one of the following: `ruleid`, `rulename`, `expression`
`sortorder`	The order to sort the results by. Value can be: • `asc` - for ascending order • `desc` - for descending order

Output

The command returns the following:

`expression`	The rules defined using a combination of expressions.
`rulename`	The name of the rule.
`ruleid`	The ID of the rule.

Example

Retrieve the list of tier rules:

CLI

```
# /opt/mapr/bin/maprcli tier rule list
expression
  rulename  ruleid
m:365d
  rule1     1
s:5g
  rule2     2
u:m7user1
  rule3     3
p
  rule4     4

  rule5     5
u:m7user1 | (u:mapr & (s:5g |
m:365d))  testRule  6
```

REST

```
# curl -k -X GET 'https://
abc.sj.us:8443/rest/tier/rule/
list' --user mapr:mapr
{"timestamp":1519840839491,"timeofday"
:"2018-02-28 10:00:39.491 GMT-0800
AM","status":"OK","total":6,"data":
[{"ruleid":"1","rulename":"rule1","exp
ression":"m:365d"},
{"ruleid":"2","rulename":"rule2","expr
ession":"s:5g"},
{"ruleid":"3","rulename":"rule3","expr
ession":"u:m7user1"},
{"ruleid":"4","rulename":"rule4","expr
ession":"p"},
{"ruleid":"5","rulename":"rule5","expr
ession":""},
{"ruleid":"6","rulename":"testRule","e
xpression":"u:m7user1 | (u:mapr &
(s:5g | m:365d))"}]}
```

tier rule modify

Modifies the criteria in a tiering rule (specified by name).

Syntax

CLI

```
$ maprcli tier rule modify
    -name <rule_name>
    -expr <expression>
    [ -cluster <cluster_name> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host:port>/rest/tier/rule/modify?<parameters>`

last-updated: Apr 11, 2022

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`expr`	The criteria for offloading data. The criteria can be defined using a combination of the following: • `u` — the user who owns the file(s) to offload • `g` — the group that owns the file(s) to offload • `m` — the time since the file was modified • `s` — the size of the file to offload. Use `b` for bytes, `k` for kilobytes, `m` for megabytes, or `g` for gigabytes. Or, use: • `p` — to offload all the files • `""` — empty string to not offload any files Use the following to string multiple criteria for offload: • `&` — to indicate all specified criteria must be met for offload • `|` — to indicate any of the specified criteria is adequate for offload • `( )` — to specify sub-expressions
`name`	The name of the rule.

Examples

Modify the criteria in the tiering rule, ksTestRule, to offload all files in the volume:

CLI

```
# /opt/mapr/bin/maprcli tier rule
modify -name ksTestRule -expr
"p" -json
{
    "timestamp":1516225073780,
    "timeofday":"2018-01-17
09:37:53.780 GMT+0000",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully updated rule:
'ksTestRule'"
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/rule/modify?
name=ksTestRule&expr=p' --user
mapr:mapr
```


last-updated: Apr 11, 2022

```
{"timestamp":1526489124827,"timeofday"
:"2018-05-16 09:45:24.827 GMT-0700
AM","status":"OK","total":0,"data":
[],"messages":["Successfully updated
rule: 'ksTestRule'"]}
```

tier rule remove
Removes the rule for offloading data.

Note: You cannot remove a rule that is currently associated with a volume.

Syntax

CLI

```
$ maprcli tier rule remove
    -name <rule_name>
    [ -cluster <cluster_name> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host:port>/rest/tier/rule/remove?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`name`	The name of the rule to remove.

Examples

Remove the rule named testRule:

CLI

```
# /opt/mapr/bin/maprcli tier rule
remove -name testRule -json
{
    "timestamp":1516225222172,
    "timeofday":"2018-01-17
09:40:22.172 GMT+0000",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully deleted rule:
'testRule'"
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/tier/rule/remove?
```

```
name=testRule' --user mapr:mapr
{"timestamp":1526488467571,"timeofday"
:"2018-05-16 09:34:27.571 GMT-0700
AM","status":"OK","total":0,"data":
[],"messages":["Successfully deleted
rule: 'testRule'"]}
```

trace
Lets you view and modify the trace buffer, and the trace levels for the system modules.

The valid trace levels are:

- DEBUG
- INFO
- ERROR
- WARN
- FATAL

trace dump
Dumps the contents of the trace buffer into the MapR filesystem log.

Syntax

CLI

```
maprcli trace dump
    [ -host <host> ]
    [ -port <port> ]
```

REST

None.

Parameters

Parameter	Description
host	The IP address of the node from which to dump the trace buffer. Default: `localhost`
port	The port to use when dumping the trace buffer. Default: `5660`

Examples

Dump the trace buffer to the MapR filesystem log:

CLI

```
maprcli trace dump
```

trace info
Displays the trace level of each module.

Syntax

CLI

```
maprcli trace info
    [ -host <host> ]
    [ -port <port> ]
```

REST

None.

Parameters

Parameter	Description
host	The IP address of the node on which to display the trace level of each module. Default: `localhost`
port	The port to use. Default: `5660`

Output

A list of all modules and their trace levels.

Sample Output

```
RPC Client Initialize
**Trace is in DEFAULT mode.
**Allowed Trace Levels are:
FATAL
ERROR
WARN
INFO
DEBUG
**Trace buffer size: 2097152
**Modules and levels:
Global : INFO
RPC : ERROR
MessageQueue : ERROR
CacheMgr : INFO
IOMgr : INFO
Transaction : ERROR
Log : INFO
Cleaner : ERROR
Allocator : ERROR
BTreeMgr : ERROR
BTree : ERROR
BTreeDelete : ERROR
BTreeOwnership : INFO
MapServerFile : ERROR
MapServerDir : INFO
Container : INFO
Snapshot : INFO
Util : ERROR
Replication : INFO
PunchHole : ERROR
KvStore : ERROR
Truncate : ERROR
Orphanage : INFO
FileServer : INFO
Defer : ERROR
ServerCommand : INFO
NFSD : INFO
```


last-updated: Apr 11, 2022

```
Cidcache : ERROR
Client : ERROR
Fidcache : ERROR
Fidmap : ERROR
Inode : ERROR
JniCommon : ERROR
Shmem : ERROR
Table : ERROR
Fctest : ERROR
DONE
```

Examples

Display trace info:

CLI

```
maprcli trace info
```

trace print
Manually dumps the trace buffer to stdout.

Syntax

CLI

```
maprcli trace print
    [ -host <host> ]
    [ -port <port> ]
    -size <size>
```

REST

None.

Parameters

Parameter	Description
host	The IP address of the node from which to dump the trace buffer to stdout. Default: `localhost`
port	The port to use. Default: `5660`
size	The number of kilobytes of the trace buffer to print. Maximum: 64

Output

The most recent `<size>` bytes of the trace buffer.

```
-----------------------------------------------------
2010-10-04 13:59:31,0000 Program: mfs on Host: fakehost IP: 0.0.0.0, Port:
0, PID: 0

-----------------------------------------------------
DONE
```

Examples

Display the trace buffer:

last-updated: Apr 11, 2022

CLI

```
maprcli trace print
```

trace reset
Resets the in-memory trace buffer.

Syntax

CLI

```
maprcli trace reset
    [ -host <host> ]
    [ -port <port> ]
```

REST

None.

Parameters

Parameter	Description
host	The IP address of the node on which to reset the trace parameters. Default: `localhost`
port	The port to use. Default: `5660`

Examples

Reset trace parameters:

CLI

```
maprcli trace reset
```

trace resize
Resizes the trace buffer.

Syntax

CLI

```
maprcli trace resize
    [ -host <host> ]
    [ -port <port> ]
    -size <size>
```

REST

None.

Parameters

Parameter	Description
host	The IP address of the node on which to resize the trace buffer. Default: `localhost`
port	The port to use. Default: `5660`
size	The size of the trace buffer, in kilobytes. Default: `2097152` Minimum: `1`

last-updated: Apr 11, 2022

Examples

Resize the trace buffer to 1000

CLI

```
maprcli trace resize -size 1000
```

trace setlevel

Sets the trace level on one or more modules.

Syntax

CLI

```
/opt/mapr/bin/maprcli trace setlevel
    [ -host <host> ]
    -level <trace level>
    -module <module name>
    [ -port <port> ]
```

REST

None.

Parameters

Parameter	Description
host	The node on which to set the trace level. Default: `localhost`
module	The module on which to set the trace level. If set to `all`, sets each module to the specified trace level.

Parameter	Description
level	The new trace level. Set the level to `default`, to set the trace level of the specified module(s) to its default. If you do not set the level, then INFO is set as the level. You can find the existing trace level of each module, using the command: `/opt/mapr/bin/maprcli trace info` The current modules along with their default trace level are: • Global : INFO • RPC : ERROR • MessageQueue : ERROR • CacheMgr : INFO • IOMgr : INFO • Transaction : ERROR • Log : ERROR • Cleaner : INFO • Allocator : ERROR • BTreeMgr : ERROR • BTree : ERROR • BTreeDelete : ERROR • BTreeOwnership : INFO • MapServerFile : ERROR • MapServerDir : INFO • MFSReadAhead : INFO • Container : INFO • Snapshot : INFO • Util : ERROR • Replication : INFO • PunchHole : INFO • KvStore : INFO • Truncate : ERROR • Orphanage : INFO • FileServer : INFO • Heartbeat : ERROR • Defer : ERROR • ServerCommand : INFO • Write : ERROR • DB : INFO

last-updated: Apr 11, 2022

Parameter	Description
port	The port to use. Default: `5660`

Examples

Set the trace level of the Log module to INFO:

CLI

```
/opt/mapr/bin/maprcli trace
setlevel -module Log -level info
```

Set the trace level of the BTreeMgr module to FATAL:

CLI

```
/opt/mapr/bin/maprcli trace
setlevel -module BTreeMgr -level FATAL
```

Set the trace levels of all modules to their defaults:

CLI

```
/opt/mapr/bin/maprcli trace
setlevel -module all -level default
```

Set the trace levels of all modules to INFO:

CLI

```
/opt/mapr/bin/maprcli trace
setlevel -module all -level INFO
```

or equivalently:

```
/opt/mapr/bin/maprcli trace
setlevel -module all
```

trace setmode
Sets the trace mode.

There are two modes:

- Default
- Continuous

In default mode, all trace messages are saved in a memory buffer. If there is an error, the buffer is dumped to stdout. In continuous mode, every allowed trace message is dumped to stdout in real time.

Syntax

CLI

```
maprcli trace setmode
    [ -host <host> ]
    -mode default|continuous
    [ -port <port> ]
```

REST

None.

Parameters

Parameter	Description
host	The IP address of the host on which to set the trace mode
mode	The trace mode.
port	The port to use.

Examples

Set the trace mode to continuous:

CLI

```
maprcli trace setmode -mode continuous
```

urls
Displays the status page URL for the specified service.

Syntax

CLI

```
/opt/mapr/bin/maprcli urls
                      [ -cluster
<cluster name> ]
                      [ -zkconnect
<ZooKeeper Connect String:
'host:port,host:port,host:port,...'> ]
                      -name <name of
the service link is required for>
                      [ -validate
<Validate if URL is reachable or not.
default: true> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/urls/statuspage?<parameters>` If you submit the request without `statuspage` in the request URL, the return value contains a 404 error because the API requires a subcommand (empty or otherwise) to be present even though the subcommand is ignored.

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to save the configuration.

Parameter	Description
`name`	The name of the service for which to get the status page: • cldb
`validate`	Enables (`true`) or disables (`false`) validating whether the URL is reachable.
`zkconnect`	ZooKeeper Connect String

Examples

Display the URL of the status page for the CLDB service:

CLI

```
# /opt/mapr/bin/maprcli urls -name
cldb
url
https://abc.sj.us:7443/cldb.jsp
```

REST

```
curl -k -X GET 'https://
abc.sj.us:8443/rest/urls/statuspage?
name=cldb' --user mapr:mapr
{"timestamp":1544561148186,"timeofday"
:"2018-12-11 12:45:48.186 GMT-0800
PM","status":"OK","total":1,"data":
[{"url":"https://abc.sj.us:7443/
cldb.jsp"}]}
```

virtualip
Manages virtual IP addresses for NFS nodes.

Table

Field	Description
macaddress	The MAC address of the virtual IP.
netmask	The netmask of the virtual IP.
virtualipend	The virtual IP range end.

virtualip add
Adds a virtual IP address.

Permissions Required

`fc` or `a` on the cluster.

Syntax

CLI

```
maprcli virtualip add
    [ -cluster <cluster> ]
    [ -gateway <gateway> ]
    [ -macs <MAC address> ]
    -netmask <netmask>
    -virtualip <virtualip>
```

```
    [ -virtualipend <virtual IP range
end> ]
    [ -preferredmac <MAC address> ]
    [ -service nfs3|nfs4 ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/virtualip/add?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`gateway`	The NFS gateway IP or address
`macs`	A list of the MAC addresses that represent the NICs on the nodes that the VIPs in the VIP range can be associated with. Use this list to limit VIP assignment to NICs on a particular subnet when your NFS server is part of multiple subnets.
`netmask`	The netmask of the virtual IP.
`preferredmac`	The preferred MAC for this virtual IP. When an NFS server restarts, the MapR system attempts to move all of the virtual IP addresses that list a MAC address on this node as a preferred MAC to this node. If the new value is null, this parameter resets the preferred MAC value.
`service`	The service to assign VIPs to. Value can be one of the following: • `nfs3` — for NFSv3 • `nfs4` — for NFSv4 The default value is `nfs3`, which is used if this option is not specified. You must specify the MAC addresses (`macs`) with this option.
`virtualip`	The virtual IP, or the start of the virtual IP range.
`virtualipend`	The end of the virtual IP range.

Example

Add VIP for NFSv3 node:

CLI

```
maprcli virtualip add
        -cluster
mycluster.402.source
        -macs "09:0C:29:3C:47:AB
03:3C:34:76:CF:21 02:0E:22:71:AD:34"
        -netmask 255.255.255.0
        -virtualip  10.1.1.5
```


last-updated: Apr 11, 2022

```
        -preferredmac
"02:0E:22:71:AD:34"
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/virtualid/add?
cluster=mycluster.402.source&macs=%220
9%3A0C%3A29%3A3C%3A47%3AAB%2003%3A3C%3
A34%3A76%3ACF%3A21%2002%3A0E%3A22%3A71
%3AAD%3A34%22&netmask=255.255.255.0&vi
rtualid=10.1.1.5&preferredmac=%2202%3A
0E%3A22%3A71%3AAD%3A34%22' --user
mapr:mapr
```

Add VIP range for NFSv3 nodes:

CLI

```
maprcli virtualip add
        -cluster
mycluster.402.source
        -service nfs3
        -macs "09:0C:29:3C:47:AB
03:3C:34:76:CF:21 02:0E:22:71:AD:34"
        -netmask 255.255.255.0
        -virtualip  10.1.1.5
        -virtualipend 10.1.1.7
        -preferredmac
"02:0E:22:71:AD:34"
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/virtualid/add?
cluster=mycluster.402.source&service=n
fs3&macs=%2209%3A0C%3A29%3A3C%3A47%3AA
B%2003%3A3C%3A34%3A76%3ACF%3A21%2002%3
A0E%3A22%3A71%3AAD%3A34%22&netmask=255
.255.255.0&virtualid=10.1.1.5&virtuali
pend=10.1.1.7&preferredmac=%2202%3A0E%
3A22%3A71%3AAD%3A34%22' --user
mapr:mapr
```

Add VIP for NFSv4 node:

CLI

```
maprcli virtualip add
        -cluster
mycluster.402.source
        -service nfs4
        -macs "09:0C:29:3C:47:AB
03:3C:34:76:CF:21 02:0E:22:71:AD:37"
        -netmask 255.255.255.0
        -virtualip  10.1.2.7
        -preferredmac
"02:0E:22:71:AD:37"
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/virtualid/add?
cluster=mycluster.402.source&service=n
fs4&macs=%2209%3A0C%3A29%3A3C%3A47%3AA
```

```
B%2003%3A3C%3A37%3A76%3ACF%3A21%2002%3
A0E%3A22%3A71%3AAD%3A34%22&netmask=255
.255.255.0&virtualid=10.1.2.7&preferre
dmac=%2202%3A0E%3A22%3A71%3AAD%3A37%22
' --user mapr:mapr
```

virtualip edit
Edits a virtual IP (VIP) range. Permissions required: `fc` or `a`.

Syntax

CLI

```
maprcli virtualip edit
    [ -cluster <cluster> ]
    [ -macs <MAC addresses> ]
    -netmask <netmask>
    -virtualip <virtualip>
    [ -virtualipend <virtual IP range
end> ]
    [ -preferredmac <MAC address> ]
    [ -service nfs3|nfs4 ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/virtualip/edit?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`macs`	A list of the MAC addresses that represent the NICs on the nodes to which the VIPs in the VIP range can be associated. Use this list to limit VIP assignment to NICs on a particular subnet when your NFS server is part of multiple subnets.
`netmask`	The netmask of the virtual IP.
`preferredmac`	The preferred MAC for this virtual IP. When a NFS server restarts, the MapR system attempts to move all of the virtual IP addresses that list a MAC address on this node as a preferred MAC to this node. If the new value is null, this parameter resets the preferred MAC value.
`service`	The service to which the VIPs need to be assigned. The Value can be one of the following: • `nfs3` — for NFSv3 • `nfs4` — for NFSv4 The default value is `nfs3`. You must specify the MAC addresses (`macs`) with this option.
`virtualip`	The virtual IP, or the start of the virtual IP range.

last-updated: Apr 11, 2022

Parameter	Description
`virtualipend`	The end of the virtual IP range.

Examples

Single virtual IP

CLI

```
maprcli virtualip edit
             -cluster
mycluster.402.source
             -macs
"09:0C:29:3C:47:AB
00:0c:29:9e:96:15"
             -netmask 255.255.255.0
             -virtualip 10.1.1.5
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/virtualip/remove?
cluster=mycluster.402.source&macs=%220
9%3A0C%3A29%3A3C%3A47%3AAB%2000%3A0c%3
A29%3A9e%3A96%3A15%22&netmask=255.255.
255.0&virtualip=10.1.1.5' --user
mapr:mapr
```

Virtual IP range

CLI

```
maprcli virtualip edit
                     -cluster
mycluster.402.source
                     -macs
"09:0C:29:3C:47:AB
00:0c:29:9e:96:15"
                     -netmask
255.255.255.0
                     -virtualip
10.1.1.5
                     -virtualipend
10.1.1.8
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/virtualip/remove?
cluster=mycluster.402.source&macs=%220
9%3A0C%3A29%3A3C%3A47%3AAB%2000%3A0c%3
A29%3A9e%3A96%3A15%22&netmask=255.255.
255.0&virtualip=10.1.1.5&virtualipend=
10.1.1.8' --user mapr:mapr
```

virtualip list

Lists the virtual IP addresses in the cluster.

Syntax

CLI

```
maprcli virtualip list
    [ -cluster <cluster> ]
    [ -columns <columns> ]
    [ -filter <filter> ]
    [ -limit <limit> ]
    [ -nfsmacs <NFS macs> ]
```

```
    [ -output <output> ]
    [ -range <range> ]
    [ -sortby <attribute> ]
    [ -start <start> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/virtualip/list[?<parameters>]`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`columns`	The columns to display.  **Note:** • The `hostname` and `ip` fields are always returned in the query. • The value(s) for `assignables` are not returned as a column.
`filter`	A filter specifying VIPs to list. See Filters for more information.
`limit`	The number of records to return.
`nfsmacs`	Specifies whether (`1`) or not (`0`) to return the MAC addresses of servers running NFS. If value is `1`, the command returns the MAC addresses of the NFS servers.
`output`	Whether the output should be `terse` or `verbose`.
`range`	Specifies whether (`1`) or not (`0`) to return the VIP ranges. The default value is `0`, which returns all VIPs individually. If: • The value is `0`, the command returns the assignment of VIPs to hosts (specified by `hn`, `mac`, and `ip` in the output). • The value is `1`, the command returns the VIP ranges and the assignables, which is the group of nodes amongst which the VIP range (specified by `vip` and `vipe` in the output) must be distributed. If assignables is empty in the output, the range of VIPs (specified by `vip` and `vipe` in the output) can be assigned to any NFSv3 server.  **Note:** The `assignables` contains the list of MAC addresses only if the VIP assignment is restricted to a group of nodes. You must specify `-json` with the command to view the `assignables`.

last-updated: Apr 11, 2022

Parameter	Description
`sortby`	Specifies one of the following attributes to sort the list of virtual IP addresses by: `vipip`, `vipendip`, `vipnetmask`, `vipgateway`, `vipnumdevices`, `viphealth`, `vipassigneddevname`, `vipassigneddevip`, `vipassigneddevmac`, `vippreferredhostname`, `vippreferredip`, `vippreferredmac`.
`start`	The index of the first record to return.

Output

Field	Description
`assignables`	The group of nodes to assign the VIP range to. If empty, the range of VIPs can be assigned to any NFSv3 server. **Note:** You must specify `-json` with the command to view the `assignables`.
`hn`	The hostname.
`ip`	The IP address.
`mac`	The MAC address.
`nm`	The netmask.
`vip`	The virtual IP. If output contains VIP range, this is the start of the VIP range.
`vipe`	The end of the VIP range.

Examples

Return the list of VIPs:

CLI

```
# maprcli virtualip list
hn                      ip
vip          mac
nm
atsqa4-164.nfs4ad.com  10.10.88.164
10.10.88.10  0c:c4:7a:1f:91:a5
255.255.255.0
atsqa4-161             10.10.88.161
10.10.88.11  0c:c4:7a:1f:91:0a
255.255.255.0
atsqa4-161             10.10.88.161
10.10.88.12  0c:c4:7a:1f:91:0a
255.255.255.0
atsqa4-162             10.10.88.162
10.10.88.13  0c:c4:7a:1f:92:12
255.255.255.0
atsqa4-164.nfs4ad.com  10.10.88.164
10.10.88.14  0c:c4:7a:1f:91:a5
255.255.255.0
atsqa4-162             10.10.88.162
10.10.88.15  0c:c4:7a:1f:92:12
255.255.255.0
atsqa4-161             10.10.88.161
```


last-updated: Apr 11, 2022

```
10.10.88.17  0c:c4:7a:1f:91:0a
255.255.255.0
atsqa4-164.nfs4ad.com  10.10.88.164
10.10.88.18  0c:c4:7a:1f:91:a5
255.255.255.0
```

REST

```
curl -k -X
GET 'https://abc.sj.us:8443/rest/
virtualip/list' --user mapr:mapr
```

Return 2 virtual IPs:

CLI

```
# maprcli virtualip list -limit 2
hn                    ip
vip          mac
nm
atsqa4-164.nfs4ad.com  10.10.88.164
10.10.88.10  0c:c4:7a:1f:91:a5
255.255.255.0
atsqa4-161            10.10.88.161
10.10.88.11  0c:c4:7a:1f:91:0a
255.255.255.0
```

REST

```
curl -k -X
GET 'https://abc.sj.us:8443/rest/
virtualip/list?limit=2' --user
mapr:mapr
```

Return a list of VIP ranges:

CLI

```
# maprcli virtualip list -range 1
assignables  vip
vipe         nm
...          10.10.88.10
10.10.88.13  255.255.255.0
...          10.10.88.14
10.10.88.15  255.255.255.0
...          10.10.88.17
10.10.88.18  255.255.255.0
```

REST

```
curl -k -X
GET 'https://abc.sj.us:8443/rest/
virtualip/list?range=1' --user
mapr:mapr
```

virtualip move

Reassigns a virtual IP or a range of virtual IP addresses to a specified Media Access Control (MAC) address.

Syntax

CLI

```
maprcli virtualip move
    [ -cluster <cluster name> ]
```

```
    -virtualip <virtualip>
    [ -virtualipend <virtualip end
range>
    -tomac <mac>
    [ -service nfs3|nfs4 ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/virtualip/move?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster where the virtual IP addresses are being moved.
`service`	The service to assign the VIPs to. Value can be one of the following: • `nfs3` — for NFSv3 • `nfs4` — for NFSv4 The default value is `nfs3`, which is used if this option is not specified.
`tomac`	The MAC address that the virtual IP addresses are being assigned.
`virtualip`	A virtual IP address. If you provide a value for `-virtualipend`, this virtual IP address defines the beginning of the range.
`virtualipend`	A virtual IP address that defines the end of a virtual IP address range.

Examples

Move a range of three virtual IP addresses to a MAC address for the cluster my.cluster.com:

CLI

```
maprcli virtualip move -cluster
my.cluster.com -virtualip
192.168.0.8 -virtualipend
192.168.0.10 -tomac 00:FE:ED:CA:FE:99
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/virtualip/move?
cluster=my.cluster.com&virtualip=192.1
68.0.8&virtualipend=192.168.0.10&tomac
=00%3AFE%3AED%3ACA%3AFE%3A99' --user
mapr:mapr
```

virtualip remove

Removes a virtual IP (VIP) or a VIP range. Permissions required: `fc` or `a`.

Syntax

CLI

```
maprcli virtualip remove
    [ -cluster <cluster> ]
    -virtualip <virtual IP>
    [ -virtualipend <Virtual IP Range
End> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/virtualip/remove?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`virtualip`	The virtual IP or the start of the VIP range to remove.
`virtualipend`	The end of the VIP range to remove.

Examples

CLI

```
maprcli virtualip remove -virtualip
10.1.1.5
```

REST

```
https://abc.sj.us:8443/rest/virtualip/
remove?virtualip=10.1.1.5
```

volume

Manages volumes, snapshots and mirrors.

Fields

The following table lists the data fields that provide information about each volume. Each field has two names:

- Field name - displayed in the output of the `volume list` command and used to specify the columns displayed using the `columns` parameter
- Short name - used to specify the columns displayed using the `columns` parameter

The short name is also used when specifying rows with a filter, for example when specifying a set of volumes about which to get information.

Field Name	Short Name	Description
accesstime	va	A value that can be used to determine when this volume was accessed.

last-updated: Apr 11, 2022

actualreplication	arf	The actual current replication factor by percentage of the volume, as a zero-based array of integers from 0 to 100. For each position in the array, the value is the percentage of the volume that is replicated index number of times. Example: arf=5,10,85 means that 5% is not replicated, 10% is replicated once, 85% is replicated twice.
advisoryquota	aqt	The advisory quota for the volume, in MB. A value of `0` indicates there are no soft or advisory quotas for this volume.
AdvisoryQuotaExceededAlarm	aqa	Alarm raised if the volume size is more than the value configured for the advisory quota.
aename	aen	The *accounting entity (AE)* name.
aetype	aet	The type of *accounting entity (AE)*. Value can be: • `0` - user • `1` - group
allowGrant	ag	Specifies whether a parent volume grants permission for a child volume to inherit its properties. Value can be `true` or `false`.
AlmostFullTopologyAlarm	afta	Timestamp when Topology Almost Full on page 2244 alarm was raised.
audited	ea	Indicates whether `1` or not (`0`) auditing is enabled for the volume. See Enabling Auditing for the steps to enable auditing on a volume and on directories, files, and tables in that volume.
auditVolume	av	Indicates whether (`1`) or not volume accommodates audit logs.
BecomeMasterStuckAlarm	bms	Timestamp when the Volume Become Master Stuck on page 2245 alarm has been raised for the volume.
CannotMirrorAlarm	cma	Timestamp when the "Cannot Mirror" alarm was raised.
coalesceInterval	ci	The interval of time in minutes during which only the first instance of an operation on a node is recorded in audit logs, if auditing is enabled. Subsequent identical operations performed on the same node are ignored during the interval. Setting this field to a larger number helps prevent audit logs from growing quickly. **Note:** The default value is 60 minutes.

last-updated: Apr 11, 2022

CompactionFailureAlarm	cfa	Timestamp when the Compaction Failed on page 2240 alarm has been raised for the volume.
containerAllocationFactor	caf	Indicates the number of containers created for the volume.
ContainersNonLocalAlarm	cnla	Timestamp when the Local Volume containers non-local alarm was raised.
creationTime	ct	The volume creation time (epoch time in seconds). This is only available on volumes created using MapR v6.0.0 or later. For volumes created using older MapR versions, the `volume info` on page 1965 and `volume list` on page 1979 commands will return empty value for this field.
creator	on	Name of the user that created the volume.
creatorcontainerid	ccid	ID for the container.
creatorvolumeuuid	cvid	ID that supports the container chain identification for mirroring. The creatorcontainerid and creatorvolumeuuid fields combined form a unique identifier for the container chain.
CriticallyDegradedEcStripesAlarm	cea	Timestamp when the Data Under-Encoded on page 2242 alarm has been raised for the volume.
criticalReReplTimeOutSec	crto	The timeout (in seconds) before re-replicating critically under-replicated containers only.
cvtotalused		The total space used by the associated cache volume.
dareEnabled	de	Indicates whether (`1`) or not (`0`) data-at-rest encryption is enabled for the volume.
data-size-mirrored-mb	dsm	Indicates the amount of data (in MB) that has been mirrored to the volume.
data-size-to-mirror-mb	dstm	Indicates the amount of data (in MB) that is yet to be mirrored to the volume.
DataUnavailableAlarm	dua	Timestamp when the Data Unavailable on page 2240 alarm was raised.
DataUnderReplicatedAlarm	rfa	Timestamp when the Data Under-Replicated on page 2241alarm was raised.
dbindexlagsecalarmthresh	dilsat	Defines the lag time in seconds for updating secondary indexes after which the Secondary Index Encoding Error on page 2240 alarm is raised.

last-updated: Apr 11, 2022

dbrepllagsecalarmthresh	dlsat	Defines the lag time in seconds for replication after which the Table Replication Lag High on page 2238 alarm is raised.
DegradedEcStripesAlarm	dea	Timestamp when the Warm-Tier Data Node Down on page 2241 alarm has been raised for the volume.
disableddataauditoperations	ddao	The list of operations excluded from auditing. For more information, see Auditing Data Access Operations on page 698 and Selective Auditing of MapR File System, MapR Database Table, and MapR Event Store For Apache Kafka Operations Using the CLI on page 761.
ecscheme	ecs	The erasure coding scheme for the volume if the volume is enabled for warm tiering.
ecstorevolume	ecstore	The name of the backend volume or erasure coded volume associated with the tiering enabled front-end volume.
ecstripedepthmb	ecstripedepthmb	The stripe depth of the erasure coded volume. The default value is 4 MB.
ectopology	ectopo	The rack path to the erasure coded volume.
ectotalused	ecused	The total space, after compression, used by the erasure-coded volume. This includes the disk space used by the parity fragments.
enableddataauditoperations	edao	The list of operations selected for auditing. For more information, see Auditing Data Access Operations on page 698 and Selective Auditing of MapR File System, MapR Database Table, and MapR Event Store For Apache Kafka Operations Using the CLI on page 761.
enforceMinReplicationForIO	esmr	Indicates whether (`true`) or not (`false`) to enforce minimum number of replicas for the volume.
fixCreatorId	fcid	An internal flag for MapR volumes to fix the creator container ID.
forceAudit	fa	Indicates whether (`1`) or not (`0`) to force audit of operations on all files, tables, and streams in the volume.
FullTopologyAlarm	fta	Timestamp when the Topology Full Alarm on page 2244 was raised.
gateway	gwips	The hostname or IP address and port of the MAST Gateway associated with the volume.
InodesExceededAlarm	ia	Timestamp when the Inodes Exceeded alarm was raised.

LargeRowWarning	lrwarning	Timestamp when the Large Row on page 2243 alarm was raised.
lastSuccessfulMirrorTime	lmt	Last time when the mirror completed successfully.
limitspread	ls	An internal flag for MapR volumes to control the growth of volumes in terms of the number of containers. When this flag is set, CLDB tried to limit the number of new containers created depending on the present size of the volume. If the volume size (the data in the volume) is small, the CLDB tries to reuse space in existing containers thus avoiding the creation of new containers.
localpath	lp	Topology of the volume.
logicalUsed	dlu	Logical size of disk used by this volume in MB.
maxinodesalarmthreshold	miath	The threshold of inodes in use that set off the Inodes Limit Exceeded on page 2242 alarm.
maxnssizembalarmthreshold	mnsszath	The namespace container size, which when exceeded raises the INODES_EXCEEDED alarm.
metricsEnabled	me	Indicates whether (`1`) or not (`0`) metrics collection is enabled for the volume.
minreplicas	mrf	Minimum number of replicas before re-replication starts.
mirror-percent-complete	mpc	Percentage complete for the most recent or current mirror operation.
mirrorDataSrcCluster	mdc	Name of the cluster of the originator volume.
mirrorDataSrcVolume	mds	Name of the originator volume. This is used to identify the mirror family in cascaded mirroring.
mirrorDataSrcVolumeId	mdi	ID of the originator volume.
MirrorFailureAlarm	mfa	Timestamp when the Mirror Failure on page 2243 alarm was raised.
mirrorId	mid	Current mirror ID of the volume.
mirrorscheduleid	msid	ID of the schedule that determines when the volume needs to be mirrored.
mirrorSrcCluster	msc	Name of the source cluster from which the current mirroring will happen.
mirrorSrcVolume	src	Name of the source volume from which the current mirrroring will happen.
mirrorSrcVolumeId	msi	ID of the source volume from which the current mirroring will happen.

last-updated: Apr 11, 2022

mirrorstatus	mst	Status of the last mirror attempt.
mirrorthrottle	dt	Flag to determine if the throttling need to be done on mirroring. Value can be: • `0` - disabled • `1` - enabled
mirrortype	mrt	Determines the type of volume: • `0` - Read-write Volume • `1` - Mirror Volume • `2` - Mirror than can be converted to read-write • `3` - Read-write volume that can be converted to mirror
mountdir	p	The path the volume is mounted on.
mounted	mt	A value of `1` indicates the volume is mounted
nameContainerDataThresholdMB	ncdt	Maximum amount of data allowed in the name container.
nameContainerSizeMB	ncsmb	Size of the name container for this volume in MB.
nsMinReplicas	nsmr	Minimum replication level for the namespace container.
nsNumReplicas	nsnr	Replication level for the namespace container.
needsGfsck	nfsck	Indicates whether (`true`) or not (`false`) this volume requires a filesystem check.
nextMirrorId	nmid	Mirror ID that is assigned if the mirroring successfully completes the next time.
NoNodesInTopologyAlarm	nna	Timestamp when the No Nodes in Topology on page 2243 alarm was raised.
nsMinReplicas	nsmr	Indicates the minimum number of name space replicas configured for the volume.
nsNumRelicas	nsnr	Indicates the desired number of name space replicas configured for the volume.
numactivecgcontainers	numactivecgcntrs	The number of active CG containers for the volume.
numcontainers	nc	Number of containers that the volume has.
numreplicas	drf	Desired number of replicas. Containers with this amount of replicas are not re-replicated.

last-updated: Apr 11, 2022

OffloadRecallFailureAlarm	ora	Timestamp when the Offload/Recall Failed on page 2242 alarm has been raised for the volume.
partlyOutOfTopology	poot	A value of `1` indicates this volume is partly out of its topology
quota	qta	Quota for limiting disk size in MB. A value of `0` indicates there are no hard quotas for this volume
QuotaExceededAlarm	qa	Timestamp when the Volume Quota Alarm on page 2245 was raised.
rackpath	rp	The rack path for this volume.
readonly	ro	A value of `1` indicates the volume is read-only.
replicatedlogicalused	replused	Replicated logical used data of this volume
replicatedtotalused	replusedtotal	Replicated total used data of this volume
replicationtype	dcr	Replication type. Value can be `low_latency` (star replication) or `high_throughput` (chain replication).
ReplTypeConversionInProgress	rtip	Indicates whether (`1`) replication type conversion is currently happening.
reReplTimeOutSec	rto	Timeout (in seconds) before attempting re-replication of replica containers. This volume property defines the timeout period until CLDB starts re-replicating the containers on the node of the volume when CLDB stops receiving a heartbeat from the node.
scheduleid	sid	The ID of the schedule, if any, used by this volume.
schedulename	sn	The name of the schedule, if any, used by this volume.
skipWireSecuityForTierInternalOps	swsfti	Indicates whether the skip wire security tier internal ops got enabled for this volume.
snapshotcount	sc	The number of snapshots for this volume.
SnapshotFailureAlarm	sfa	Timestamp when the Snapshot Failure on page 2244 alarm was raised.
snapshotused	ssu	Total space used (in MB) by the data owned only by the snapshot and not present in the RW volume. Data shared between the snapshot and RW volume is not counted in this field.
TableIndexEncodingErrorAlarm	vatinee	Timestamp when the the VOLUME_ALARM_TABLE_INDEX_ENCODING_ERROR alarm has been raised for the volume.

last-updated: Apr 11, 2022

TableIndexErrorAlarm	vatinde	Timestamp when the VOLUME_ALARM_TABLE_INDEX_ERROR alarm has been raised for the volume.
TableIndexLagHighAlarm	vatindlh	Timestamp when the VOLUME_ALARM_TABLE_INDEX_LAG_HIGH alarm has been raised for the volume.
TableReplicationAsyncAlarm	vatrepa	Timestamp when the Table Replication Asynchronous on page 2238 alarm was raised.
TableReplicationErrorAlarm	vatrepe	Timestamp when the Table Replication Errors on page 2237 alarm was raised.
TableReplicationLagHighAlarm	vatreplh	Timestamp when the Table Replication Lag High on page 2238 alarm was raised.
tiercompactionoverheadthreshold	tcovr	The percentage of offloaded data that must have been deleted on the MapR cluster to qualify the data for compaction (or deletion from the tier).
tiercompactionscheduleid	tcsid	The ID of the schedule to use for running the compactor.
tierenable	tenb	Indicates whether (1) or not (0) staorage efficiency through tiering is enabled for the volume.
tierencryption	tenc	Indicates whether (`true`) or not (`false`) encryption of data on the tier is enabled.
tierid	tid	The ID of the tier.
tierjobendtime	tjetime	The date and time when the last tiering operation was completed.
tierjoboffloadavgthroughputmbps	tjospeed	The average throughput (MB per second) for offloaded data.
tierjobrecallavgthroughputmbps	tjrspeed	The average throughput (MB per second) for recalled data.
tierjobprogress	tjprog	The completion percentage of the currently running or last tiering operation.
tierjobstarttime	tjstime	The date and time when the currently running or last tiering operation was started.

tierjobstate	tjstatus	The status of the currently running or last tiering operation. Value can be one of the following: • Scheduled • Running • FailureFatal • FailureRetriable • Success • Aborted • AbortInProgress • AbortedInternal Fore more information on these, see Statuses on page 2040.
tierjobtotaloffloadsize	tjosize	The total amount of data offloaded to the tier during the last offload operation.
tierjobtype	tjtype	The type of tiering operation currently running or last performed on the volume.
tierLocal	tloc	The amount of (in MB) physical user data (including recalled data) on the volume in the cluster.
tiername	tname	The name of the tier.
tieroffloadscheduleid	tsid	The ID of the schedule for offloading data to the tier.
tierPurged	tpur	The amount (in MB) of physical user data that is offloaded to the tier.
tierRecall	trec	The amount of (in MB) physical user data that is recalled to the cluster.
tierrecallexpirytime	ret	The amount of time to keep recalled data on the MapR cluster before offloading (if there are changes) or purging (if there are no changes) the data.
tierruleid	rid	The ID of the rule or storage policy.
tiertype	ttype	The type of tier. Value can be either: • cold • ectier
totalused	tsu	Total space used for volume and snapshots, in MB.
used	dsu	Disk space used (in MB), not including snapshots.
volumeAces	vace	Displays the ACE values set for this volume.

last-updated: Apr 11, 2022

volumeid	id	The volume ID.
volumename	n	The name of the volume.
volumetype	t	The volume type (for backward-compatibility): • `0` - Read-write Volume • `1` - Mirror Volume
wireSecurity	ws	Indicates whether (`1`) or not (`0`) wire-level security is enabled.

Related concepts

node on page 1694
Manages nodes in the cluster

Related reference

disk add on page 1602
Adds one or more disks to the specified node. Permissions required: `fc` or `a`.

volume create on page 1931
Creates a volume.

node list on page 1705
Lists nodes in the cluster.

dump volumeinfo on page 1637
Returns information about volumes and the associated containers. For JSON formatted output, use the -json option from the command line.

configure.sh on page 2053
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

`volume audit`

Enables or disables auditing on the specified volume.

You must have the `fc` permission on the cluster to use this command. See `acl` for details about this permission.

To learn how to determine whether auditing is enabled for a volume, see Checking Whether Auditing is Enabled for a Directory, File, or MapR Database Table.

Syntax

CLI

```
maprcli volume audit
   [ -cluster <cluster name> ]
   -name <volume name>
   [-dataauditops <+|-operations>]
   [ -enabled <true|false> ]
   [ -forceenable true|false ]
   [ -coalesce <interval in
minutes> ]
```

REST

Request Type	POST

Request URL	`http[s]://<host>:<port>/rest/volume/audit?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which the volume is located. This parameter is required if the volume is on a remote cluster. The remote cluster must be listed in the `mapr-clusters.conf` file for the cluster where you run the command.
coalesce	The interval of time during which READ, WRITE, or GETATTR operations on one file from one IP address or UID are logged only once for a particular operation, if auditing is enabled. For example, suppose that a client application reads a single file three times in 6 minutes, so that there is one read at 0 minutes, another at 3 minutes, and a final read at 6 minutes. If the coalesce interval is at least 6 minutes, then only the first read operation is logged. However, if the interval is between 4 minutes, then only the first and third read operations are logged. If the interval is 2 minutes, all three read operations are logged. Now however, if the client was also writing to the file, irrespective of the coalesce interval for the read operation in the example stated previously, the write operation is logged, as it is a different operation from reading. The default value is 60 minutes. Setting this field to a larger number helps prevent audit logs from growing quickly.

last-updated: Apr 11, 2022

Parameter	Description
`dataauditops`	The comma separated list of filesystem operations to include (specified with a preceding plus sign (+)) and/or exclude (specified with a preceding minus sign (-)) from auditing. **Note:** If the first operation in the list is to be excluded from auditing, it must be preceded by two minus (`--`) signs. Subsequent operations to exclude must be preceded by only a single minus (`-`) sign, whether or not the first operation was included (using a plus (+) sign) or excluded (using two minus (`--`) signs). If neither sign is specified, the given operation is included for auditing. The operations that can be included (+) and/or excluded (-) from auditing are listed here. You can, alternatively, group all the filesystem and table operations using the keyword `all`, which: • If included (+), cannot be specified with a list of other included operations. • If excluded (-), cannot be specified with a list of other excluded operations. **Note:** You can specify a mixed list of included and excluded operations. There are no changes to operations that are not specified with the command.
`enabled`	Enables or disables the auditing of operations within the volume. You must use either this parameter, the `-coalesce` parameter, or both. See Enabling Auditing for the steps to enable auditing on directories, files, and tables in a volume. When you set the value to false, auditing of operations within the volume ceases. None of the auditing settings are changed on the directories, files, and MapR Database tables within the volume. If you later run the `maprcli volume audit` command with `-enabled` set to `true`, auditing begins again on the objects that were already enabled for auditing.
`forceenable`	Enables or disables auditing of all directories, files, tables, and streams in the volume whether or not auditing is enabled at the individual file, table, and/or stream level.
`name`	The name of the volume.

Examples

Enable Auditing for a volume

The following example shows how to enable auditing for the volume “auditVolume”:

CLI

```
maprcli volume audit -name
auditVolume -enabled true
```


last-updated: Apr 11, 2022

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/audit?
name=auditVolume&enabled=true' --user
mapr:mapr
```

Modify the list of operations to audit

The following example shows how to specify the operations to audit. Here, `create` operation is included for auditing and `lookup` operation is excluded from auditing. There are no changes to operations that are not specified.

CLI

```
maprcli volume audit -name
sampleAuditVolume -dataauditops
+create,-lookup
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/audit?
name=sampleAuditVolume&dataauditops=%2
Bcreate%2C%2Dlookup' --user mapr:mapr
```

volume balancecontainers
Balances the containers, or stops the balancing of containers associated with the volume.

Syntax

CLI

```
maprcli volume balancecontainers
    [ -cluster cluster_name ]
      -name <volume name>
    [ -cancel <true|false>. default:
false ]
```

REST API N/A

Parameters

Parameter	Description
`cancel`	Stop the balancing of containers associated with the volume. Set this parameter to `true` to cancel balancing a volume.
`cluster`	The name of the cluster on which to run the command.
`name`	The name of the volume.

Examples

Start balancing the containers associated with a volume:

```
maprcli volume balancecontainers -name sampleVol
```

Stop balancing the containers associated with a volume:

```
maprcli volume balancecontainers -name sampleVol -cancel true
```

volume balancinginfo
Fetch currently running or scheduled balancer information for one or more volumes.

Note: For best results, use the `-json` option when running the command.

Syntax

CLI

```
maprcli volume balancinginfo
    [ -cluster cluster_name ]
    [ -name <volume name> ]
```

REST API N/A

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`name`	The name of the volume.

Output

The command returns the following fields:

Field	Description
`spId`	The ID of the storage pool.
`capacity`	The capacity/size of the storage pool in MB.
`usedSize`	The size of the storage pool that is consumed.
`desiredSize`	The total size of the containers of a volume that should be allocated on this storage pool.
`isUnderweight`	Value is `true` if the storage pool contains less than 50% of the `desiredSize` for this volume.
`isOverweight`	Value is `true` if the storage pool contains more than 1.5 times the `desiredSize` for this volume.

Examples

Fetch the list of volumes whose balancing is currently in progress or scheduled:

```
maprcli volume balancinginfo -json
```

Fetch the balancing information for a volume:

```
maprcli volume balancinginfo -name snapshotVolume1 -json
```

Output:

```
{
  "timestamp":1502529117881,
  "timeofday":"2017-08-12 09:11:57.881 GMT+0000",
  "status":"OK",
  "total":5,
  "data":[
    {
```

```
      "volumeName":"snapshotVolume1"
    },
    {
      "isBalancingInProgress":false
    },
    {
      "numContainers":15
    },
    {
      "volumeSize":384
    },
    {
      "spInfo":[
        {
          "spId":"f891ae9e6663fa2000598ec48808155c",
          "capacity":152969,
          "usedSize":96,
          "desiredSize":95,
          "isUnderweight":false,
          "isOverweight":false
        },
        {
          "spId":"bed92c0ecfaefc8b00598ec48b01cdfe",
          "capacity":152969,
          "usedSize":96,
          "desiredSize":95,
          "isUnderweight":false,
          "isOverweight":false
        },
        {
          "spId":"b61aa1b814fd8bbc00598ec48d0af1d2",
          "capacity":157065,
          "usedSize":96,
          "desiredSize":97,
          "isUnderweight":false,
          "isOverweight":false
        },
        {
          "spId":"7af11d5b9d223baa00598ec4850efb57",
          "capacity":152969,
          "usedSize":96,
          "desiredSize":95,
          "isUnderweight":false,
          "isOverweight":false
        }
      ]
    }
  ]
}
```


volume compact
Runs the compactor to remove recalled data on the MapR cluster or stale data on the tier.

Permissions Required

The user running the command must have one of the following:

- Full control (fc) on the cluster or volume
- Volume edit permissions

last-updated: Apr 11, 2022

Syntax

CLI

```
maprcli volume compact
    [ -cluster <cluster_name> ]
     -name <vol_name>
     [ -forcerecallexpiry true|false ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/compact?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`forcerecallexpiry`	Specifies whether (`true`) or not (`false`) to purge recalled data on the MapR cluster. If the command is run with the value for this set to `true`, the compactor purges recalled data on the MapR cluster whether or not the expiry time for recalled data has been reached. If this is not specified or if the value for this is `false`, the compactor purges stale data on the tier and recalled data on the MapR cluster if the expiry time for recalled data has been reached or has passed. The default value is `false`.
`name`	The name of the volume.

Examples

Remove stale data on the tier for the volume named sampleVol:

CLI

```
# maprcli volume compact -name
sampleVol -json
{
    "timestamp":1528299575917,
    "timeofday":"2018-06-06
08:39:35.917 GMT-0700 AM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully started
compaction."
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/compact?
name=sampleVol' --user mapr:mapr
```


last-updated: Apr 11, 2022

```
{"timestamp":1528299575917,"timeofday"
:"2018-06-06 08:39:35.917 GMT-0700
AM","status":"OK","total":0,"data":
[],"messages":["Successfully started
compaction."]}
```

Remove recalled data immediately on the volume named sampleVol:

CLI

```
# maprcli volume compact -name
sampleVol -forcerecallexpiry
true -json
{
    "timestamp":1528299765110,
    "timeofday":"2018-06-06
08:42:45.110 GMT-0700 AM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully started
compaction."
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/compact?
name=sampleVol&forcerecallexpiry=true'
 --user mapr:mapr
{"timestamp":1528299765110,"timeofday"
:"2018-06-06 08:42:45.110 GMT-0700
AM","status":"OK","total":0,"data":
[],"messages":["Successfully started
compaction."]}
```

volume container move

Moves a container. Permissions required: `fc` or `m` on the volume.

The volume container move command moves a specified container (`cid`) from a source file server (`fromfileserver`) to a destination file server (`tofileserver`). If the `tofileserver` parameter is not specified, a destination file server is chosen by the CLDB. If the `tofileserver` is specified but does not exist, the command fails with an error. If the `fromfileserver` does not exist or is down, the container move occurs once the source file server comes back up.

Syntax

CLI

```
volume container move
    -cid <cid>
    -fromfileserverid
<fromfileserverid>
    [ -tofileserverid
<tofileserverid> ]
```

REST

Request Type	POST

last-updated: Apr 11, 2022

Request URL	`http[s]://<host>:<port>/rest/volume/container/move?<parameters>`

Parameters

Parameter	Description
`cid`	The container ID.
`fromfileserverid`	The ID of the file server on which the container to be moved currently resides. The ID is available from the `maprcli node list` command.
`tofileserverid`	The ID of the file server to which to move the container. If not specified, a file server is chosen by the CLDB. The ID is available from the `maprcli node list` command.

Examples

CLI

```
maprcli volume container
move -cid 2316 -fromfileserverid
5227152973904547710 -tofileserverid
875290643748357753
```

REST

```
curl -k -X POST 'https://abc.sj.us:8443/
rest/volume/container/move?
cid=2316&fromfileserverid=52271529739045
47710&tofileservicerid=87529064374835775
3' --user mapr:mapr
```

volume container switchmaster

Switches the master replica for a specified container to another replica in the replica chain.

This command fails if there is only one up-to-date replica for the container.

Note: Only the `root` and the MAPR_USER (user name under which MapR services run) user have permissions to run this command.

Syntax

CLI

```
maprcli volume container switchmaster
   [ -cluster <cluster_name> ]
   -cid <cid>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/container/switchmaster?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command. If this parameter is omitted, the command is run on the same cluster where it is issued. In multi-cluster contexts, you can use this parameter to specify a different cluster on which to run the command.
`cid`	The unique ID number for the container that you want to run the command on.

Example
Switches the master container for a specified container:

CLI

```
maprcli volume container
switchmaster -cid 2049
```

REST

```
https://abc.sj.us:8443/rest/volume/
container/switchmaster?cid=2049
```

volume create
Creates a volume.

Permissions Required

`cv` or `fc` on the cluster.

Note: See acl on page 1528 for more information.

Syntax

CLI

```
/opt/mapr/bin/maprcli volume create
   -name <volume name>
   [ -advisoryquota <advisory quota> ]
   [ -ae <accounting entity> ]
   [ -aetype <accounting entity
type> ]
   [ -allowgrant true|false ]
   [ -allowinherit true|false ]
   [ -allowreadforexecute Enable
reads for files with execute
permission. <true|false> ]

   [ -auditenabled true|false ]
   [ -autooffloadthresholdgb <offload
size threshold> ]
   [ -cluster <cluster> ]
   [ -coalesce <interval in mins> ]
   [ -compactionoverheadthreshold
<compaction_overhead> ]
   [ -compactionschedule
<compaction_schedule_ID> ]
   [ -containerallocationfactor
<positive integer> ]
   [ -createparent 0|1 ]
```

```
[ -criticalrereplicationtimeoutsec ]
   [ -dare true|false ]
   [ -dataauditops <+|- operations> ]
   [ -dbindexlagsecalarmthresh
<threshold> ]
   [ -dbrepllagsecalarmthresh
<threshold> ]
   [ -ecenable true|false ]

   [ -ecscheme <ec scheme> ]
   [ -ectopology <path to ec volume> ]

   [ -enforceminreplicationforio true|
false ]

   [ -forceauditenable true|false ]
   [ -group <list of
group:allowMask> ]
   [ -inherit <volume name> ]

   [ -localvolumehost
<localvolumehost> ]
   [ -localvolumeport
<localvolumeport> ]
   [ -maxinodesalarmthreshold
<maxinodesalarmthreshold> ]
   [ -maxnssizembalarmthreshold
<maxnssizembalarmthreshold> ]
   [ -metricsenabled true|false ]
   [ -minreplication <minimum
replication factor> ]
   [ -mirrorschedule <mirror schedule
ID> ] (4.0.2 only)
   [ -mirrorthrottle 0|1 ]
   [ -mount 0|1 ]
   [ -namecontainerdatathreshold
<size> ]

   [ -nsminreplication <minimum
replication factor> ]
   [ -nsreplication <replication
factor> ]

   [ -offloadschedule <schedule ID> ]
   [ -path <mount path> ]
   [ -quota <quota> ]
   [ -readAce <access control
expression> ]
   [ -readonly <read-only status> ]
   [ -recallexpirytime <expiry time> ]
   [ -replication <replication
factor> ]
   [ -replicationtype <type> ]
   [ -rereplicationtimeoutsec
<seconds> ]
   [ -rootdirgroup <root directory
group> ]
   [ -rootdirperms <root directory
permissions> ]
   [ -rootdiruser <root directory
user> ]
```

```
   [ -schedule <ID>

   [ -skipinherit schedule|tiername ]

   [ -source <source> ]
   [ -tenantuser <tenant name> ]
   [ -tierencryption true|false ]
   [ -tieringenable true|false ]
   [ -tieringrule <rulename> ]
   [ -tierkey <tier encryption key> ]
   [ -tiername <tiername> ]
   [ -topology <topology> ]
   [ -type rw|mirror ]
   [ -user <list of user:allowMask> ]
   [ -wiresecurityenabled true|false ]
   [ -writeAce <access control
expression> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/create?<parameters>`

Parameters

Parameter: `advisoryquota`

Default Value: No default value

Possible Values: 0 or any other integer value.

Description: The advisory quota for the volume as `integer` plus `unit. Example: quota=500G; Units: B`, `K`, `M`, `G`, `T`, `P`

Parameter: `ae`

Default Value: No default value

Possible Values: Name of the entity that owns the volume.

Description: The accounting entity that owns the volume.

Parameter: `aetype`

Default Value: No default value

Possible Values:

- `0=user`
- `1=group`

Description: Type of accounting entity.

Parameter: `allowgrant`

Default Value: false

Possible Values:

- `true`
- `false`

Description: Specifies whether the volume as a parent, grants permission for a child volume to inherit its properties.

Parameter: `allowinherit`

Default Value: true

last-updated: Apr 11, 2022

Possible Values:

- `true`
- `false`

Description: Specifies whether a new volume inherits properties from the parent mount point volume.

Parameter: `allowreadforexecute`

Default Value: false

Possible Values:

- `true`
- `false`

Description: Allows execution of SUID binaries with only their executable bit set, on a FUSE filesystem. This parameter works in conjunction with the `fuse.mount.setuid` FUSE option. For more information, see Configuring the MapR FUSE-Based POSIX Client on page 1240.

Parameter: `auditenabled`

Default Value: true

Possible Values:

- `true`
- `false`

Description: Specifies whether to turn on auditing for the volume. If you enable auditing at the cluster level with the `audit data on page 1553` command, setting this value to `true` causes auditing to start for any directories, files, tables, or streams that are already enabled for auditing. If none are yet enabled, enabling auditing on any of them causes auditing of them to start.

Set `auditenabled` to `true` to enable auditing on directories, files, tables, and streams in the volume.

You must have the `fc` permission on the cluster to use this parameter. See `acl` for details about this permission.

Parameter: `autooffloadthresholdgb`

Default Value: 1024 GB

Possible Values: Any positive integer.

Description: The size of the volume in GB (threshold). When this threshold is reached or exceeded, volume data is automatically offloaded by the Automatic Tiering Scheduler. To use the global size threshold (of 1024 GB), set the value to `0`.

Parameter: `cluster`

Default Value: No default value

Possible Values: Any valid cluster.

Description: The cluster on which to create the volume.

Parameter: *`coalesce`*

Default Value: 60 minutes

Possible Values: Set this parameter to a large number of minutes to prevent audit logs from growing quickly.

Description: The interval of time (in minutes) during which READ, WRITE, or GETATTR operations on one

last-updated: Apr 11, 2022

file from one IP address or UID are logged only once for a particular operation, if auditing is enabled.

For example, suppose that a client application reads a single file three times in 6 minutes, so that there is one read at 0 minutes, another at 3 minutes, and a final read at 6 minutes. If the coalesce interval is at least 6 minutes, then only the first read operation is logged. However, if the interval is 4 minutes, then only the first and third read operations are logged. If the interval is 2 minutes, all three read operations are logged.

Now however, if the client was also writing to the file, irrespective of the coalesce interval for the read operation in the example stated previously, the write operation is logged, as it is a different operation from reading.

Parameter: `compactionoverheadthreshold`

Default Value: 30%

Possible Values: 0-100%

Description: Specifies the percentage of offloaded data that must have been deleted on the MapR cluster to qualify the data for compaction (or deletion from the tier).

Parameter: `compactionschedule`

Default Value: Automatic Internal Schedule

Possible Values: Any valid schedule ID.

Set this parameter to `0` to disable the compactor.

Description: Specifies the schedule to use for running the compactor.

Parameter: `containerallocationfactor`

Default Value: 5

Recommended Value: 2* SP count in the volume topology.

Description: Specifies the number of containers to create when the first write from a remote client is sent to the volume. The pre-created containers are distributed equally across topologies, servers, MapR File System instances, and storage pools. CLDB also takes into consideration the load (IO/Space) when selecting target storage pools for containers. The value must be a positive integer.

Parameter: `createparent`

Default Value: 1

Possible Values:

- 0 - Do not create a parent directory
- 1 - Create a parent directory

Description: Specifies whether or not to create a parent directory to hold the volume link.

Parameter: `criticalrereplicationtimeoutsec`

Default Value: 0 (No timeout)

Possible Values: Any integer between 300 and 3600 (seconds)

Description: Timeout (in seconds) before re-replicating only the critically under-replicated containers . If you set both `rereplicationtimeoutsec` and `criticalrereplicationtimeoutsec`, and if the value of:

- `rereplicationtimeoutsec` is less than `criticalrereplicationtimeoutsec`, `rereplicationtimeoutsec` overrides the `criticalrereplicationtimeoutsec` setting for both under-replicated and critically under-replicated containers.
- `rereplicationtimeoutsec` is greater than `criticalrereplicationtimeoutsec`, `criticalrereplicationtimeoutsec` overrides the `rereplicationtimeoutsec` setting only for critically under-replicated containers; `rereplicationtimeoutsec` setting is still applicable for under-replicated containers.

Parameter: `dare`

Default Value: false

Possible Values:

- true
- false

Description: Specifies whether or not to enable data-at-rest encryption for volume. This setting takes effect only if the data-at-rest encryption feature is also enabled at the cluster level. Once enabled, this feature cannot be disabled.

Parameter: `dataauditops`

Default Value: Default enabled audit ops:
`setattr,chown,chperm,chgrp,getxattr,listxattr,setxattr,removexattr,read,write,create,delete,mkdir,readdir,rmdir,createsym,lookup,rename,createdev,truncate,tablecfcreate,tablecfdelete,tablecfmodify,tablecfScan,tableget,tableput,tablescan,tablecreate,tableinfo,tablemodify,getperm,getpathforfid,hardlink,filescan,fileoffload,filerecall,filetierjobstatus,filetierjobabort`

Possible Values: Any audit operations that you want to enable.

Description: The comma separated list of filesystem operations to include (specified with a preceding plus sign (+)) or exclude (specified with a preceding minus sign (-)) from auditing.

To exclude the first operation in the list (of operations) from auditing, precede it by two minus (`--`) signs. To exclude subsequent operations, precede them by only a single minus (`-`) sign, irrespective of whether the first operation was included (using a plus (+) sign) or excluded (using two minus (`--`) signs). If neither sign is specified, the given operation is included for auditing.

The operations that can be included (+) or excluded (-) from auditing are listed here. You can, alternatively, group all the operations using the keyword `all`, which:

- If included (+), cannot be specified with a list of other included operations.

- If excluded (`-`), cannot be specified with a list of other excluded operations.

You can specify a mixed list of included and excluded operations. There is no change to operations that are not specified with the command.

Note: If you specify only `setattr` as the audit operation to enable, `chown`, `chperm`, and `chgrp` are automatically added and enabled.

For more information, see Selective Auditing of Filesystem and Table Operations.

Parameter: `dbindexlagsecalarmthresh`

Default Value: 300 seconds

Possible Values: Any integer value.

Description: Specifies the threshold (in seconds) to raise an alarm for index update lag.

Parameter: `dbrepllagsecalarmthresh`

Default Value: 900 seconds

Possible Values: Any integer value.

Description: Specifies the threshold (in seconds) to raise an alarm for DB replication lag.

Parameter: `ecenable`

Default Value: false

Possible Values:

- `true`
- `false`

Description: Enable (`true`) or disable (`false`) warm tiering for the volume. Either this parameter or `tieringenable` is required to enable warm tiering. If you specify this parameter, you cannot specify `tiername`; when the command runs, a new tier is created for the volume. If you do not specify any rule, the default rule, which is all files (p), is associated with the volume.

Parameter: `ecscheme`

Default Value: 4+2

Possible Values: Any valid EC scheme.

Description:

Note: This parameter is applicable only for EC volumes, and only when you set the `ecenable` parameter to `true`.

The number of data chunks and the number of parity chunks separated by a plus (+) sign.

For schemes with local parity, the scheme is of the form `x+y+z`, where x is the number of data chunks, y is the number of local parity chunks, and z is the number of global parity chunks.

For information on the supported schemes, see Erasure Coding Scheme for Data Protection and Recovery on page 926.

Parameter: `ectopology`

Default Value: `/data/default-rack`

Possible Values: Any topology that exists in your environment.

last-updated: Apr 11, 2022

Description: Sets the topology to store the erasure coded volume. Once set, you cannot change the topology of an erasure coded volume using this command. To change the topology of an erasure coded volume, use `volume move` on page 2021.

Note:

1. This parameter is applicable only for EC volumes.
2. The specified EC topology needs to have sufficient nodes for the selected EC Scheme. For example, 6 nodes for 4+2, 5 nodes for 3+2 etc.

Parameter: `enforceminreplicationforio`

Default Value: false

Possible Values:

- `true`
- `false`

Description: Specifies whether (`true`) or not (`false`) to enforce minimum number of replicas for the (read-write) volume during IO. This flag ensures that further updates (writes) to volume are successful only when the minimum number of copies of the container are available. Setting this parameter to `true` ensures that if writes succeed, then it has been applied to at least the minimum number of copies; if writes fail, it may have been applied to zero or more copies.

Enabling this parameter, may stall `volume dump` and `volume snapshot create` operations, if the minimum number of copies of the container are not available.

If you do not set this parameter on a volume, or if you modified this parameter from `false` to `true`, then you need to restart all the nodes where the containers associated with the volume exist, for the changes to take effect.

This flag is ignored on mirror volumes. If there are more than five cluster nodes, this flag is set to `true`, by default, on the tier volume. If the number of cluster nodes is less than five, this flag is set to `false`, by default, on the tier volume.

Parameter: `forceauditenable`

Default Value: false

Possible Values:

- `true` - force audit of all content
- `false` - do not force audit

Description: Specifies whether (`true`) or not (`false`) to force audit of operations on all files, tables, and streams in the volume if auditing is enabled at the cluster and volume levels, irrespective of the audit setting on the individual directory, file, table, and stream.

Parameter: `group`

Default Value: No default value

last-updated: Apr 11, 2022

Possible Values: Any user with `Create Volume` privileges.

Description: Space-separated list of `group:permission` pairs. Use commas to separate permissions. For example: `group:permission,permission,...`

Parameter: `inherit`

Default Value: No default value

Possible Values: Any existing volume name

Description: Specifies the name of the volume from which the new volume inherits properties. When you specify `inherit`, you do not need to specify `allowgrant`. See the following section on Inheritance for more information.

Parameter: `localvolumehost`

Default Value: No default value

Possible Values: Any existing volume name

Description: Specifies the name of the local volume host.

Parameter: `localvolumeport`

Default Value: 5660

Possible Values: Any valid port number

Description: Specifies the port number of the local volume host.

Parameter: `maxinodesalarmthreshold`

Default Value: 50000000

Possible Values: Any positive integer.

Description: The number of inodes, which when exceeded raises the INODES_EXCEEDED alarm.

Parameter: `maxnssizembalarmthreshold`

Default Value: 500 GB

Possible Values: Any positive integer.

Description: The namespace container size, which when exceeded raises the INODES_EXCEEDED alarm.

Parameter: `metricsenabled`

Default Value: false

Possible Values:

- `true`
- `false`

Description: Specifies whether (`true`) or not (`false`) to enable metrics collection for a volume.

Parameter: `minreplication`

Default Value: 2

Possible Values: Can be any value that you desire based on the replication you need.

Description: The minimum replication level. When the replication factor falls below this minimum, re-replication occurs as aggressively as possible to restore the replication level. If any containers in the CLDB volume fall below the minimum replication factor, writes are disabled until aggressive re-replication restores the minimum level of replication.

> **Tip:** For more information, see Understanding Replication on page 454.

Parameter: `mirrorschedule`

Default Value: 0

last-updated: Apr 11, 2022

Possible Values: Any valid schedule ID.

Description: The schedule ID corresponding to the schedule to be used for mirroring. If you specify a mirror schedule ID, then the mirror volume automatically syncs with its source volume on the specified schedule. Pre-assigned IDs include 1 for critical data, 2 for important data, and 3 for normal data. Custom schedules are assigned ID numbers in sequence. To determine the ID number, use the `schedule list` command.

Parameter: `mirrorthrottle`

Default Value: true

Possible Values:

- `true`
- `false`

Description: Specifies whether mirror throttling is enabled (`true`) or disabled (`false`). Throttling is set on the source volume and applies to all its mirrors.

Parameter: `mount`

Default Value: true

Possible Values:

- `true`
- `false`

Description: Specifies whether to mount the volume (`true`) or not (`false`) after creating the volume.

Parameter: `name`

Default Value: No default value

Possible Values: Any valid name

Description: Specifies the name for the volume.

The name should contain only the following characters:

```
A-Z a-z 0-9 _ - .
```

For tiering-enabled volumes, the volume name cannot exceed 98 characters. For regular volumes, the volume name should be a maximum of 128 characters.

Parameter: `namecontainerdatathreshold`

Default Value: 524288 MB

Possible Values: Any integer value. The value is interpreted as being in MB.

If you set this parameter to `0`, there is no limit on the size of user data that can be stored in the name container.

If chunk size is `0`, by default, all data is stored in the name container. However, if this property is set, all data is stored in a second container and only the meta data is stored in the name container once the threshold is reached. The size of the second container is not limited by this setting; you must ensure that the size does not grow too large, by limiting the amount of data in the volume.

Description: Limits the size of user data that can be placed in the name container. The value is interpreted as being in MB. If the user data size limit:

- Has not yet been reached, the first 64 KB of data is stored in name container, and the rest of the data is stored in data containers.
- Has already been reached, only meta data is stored in the name container, and the data is stored in data containers. For example, if you set the current name container size to 200GB and the limit to 100GB, then all new user data is stored in data containers.

Parameter: `nsminreplication`

Default Value: 2

Possible Values: Any integer value.

Description: A replication factor of the namespace container. When the replication factor falls below this value, re-replication occurs as aggressively as possible to restore the replication level. If any containers in the CLDB volume fall below the minimum replication factor, writes are disabled until aggressive re-replication restores the minimum level of replication.

When enabled, the CLDB manages the namespace container replication separate from the data container replication. You use this capability when you have low volume replication but want to have higher namespace replication.

Set the value to be the same or larger than the value of the equivalent data replication parameter, `minreplication`.

See also: Understanding Replication on page 454.

Parameter: `nsreplication`

Default Value: 3

Possible Values: Any integer value.

Description: A replication factor of the namespace container. When the number of copies fall below the desired replication factor, but remains equal to or above the minimum replication factor, re-replication occurs after the timeout specified in the `cldb.fs.mark.rereplicate.sec` parameter. This timeout is the time given for a node that is offline to come back online. After this timeout period, the CLDB takes action to restore the replication factor. When enabled, the CLDB manages the namespace container replication separate from the data container replication. This capability is used when you have low volume replication but want to have higher namespace replication. By default, the value of this parameter is the same or larger than the value of the equivalent data replication parameter. However, to set the value of this parameter lower than the replication value, first set `engg.manual.override` to `true` in `cldb.conf`. *See also: Understanding Replication on page 454*.

Parameter: `offloadschedule`

Default Value: No default value

Possible Values: Any valid schedule ID. To disable schedule-based offload, set this value to `0`.

Description: The ID of the schedule to associate with the volume for offloading volume data to the tier.

Note: This parameter is required only for Cold/EC tiered volumes.

last-updated: Apr 11, 2022

Parameter: `path`

Default Value: No default value

Possible Values: Any valid path.

Description: The path at which to mount the volume. The path must be relative to `/` and cannot be in the form of a global namespace path (for example, `/mapr/<cluster-name>/`).

Parameter: `quota`

Default Value: 0

Possible Values: Any integer value along with a unit.

Description: The quota for the volume as `integer` plus `unit. Example: quota=500G; Units: B`, `K`, `M`, `G`, `T`, `P`

Do not use two-letter abbreviations for quota units, such as `GB` and `MB`.

When you set a quota for a tiering-enabled volume, the quota is the total space allocated for the volume irrespective of the location (cluster or tier) where the volume data is stored. For example, if you allocate 1GB of hard quota for a tiering-enabled volume, writes fail after you write 1GB of data whether or not the volume data is local (on the cluster) or offloaded (to the tier).

Note that quotas for source and mirror volumes must match.

Parameter: `readAce`

Default Value: `p` (grant access to all users)

Possible Values: Any valid permissions.

Description: Specifies Access Control Expressions(ACEs) that grant permissions at the volume level to read files and tables in the volume. The default value is `p`, which grants access to all users.

See *ACE*s.

Parameter: `readonly`

Default Value: No default value

Possible Values:

- `0`
- `1`

Description: Specifies whether the volume is read-only.

- `0` - read/write
- `1` - read-only

Parameter: `recallexpirytime`

Default Value: 1 day

Possible Values: Any integer between `1` and `7500`.

Description: The amount of time (in days) to keep the recalled data before purging or offloading it.

Parameter: `replication`

Default Value: 3

Possible Values: Any integer starting at `0`.

Description: The desired replication level. When the number of copies falls below the desired replication factor, but remains equal to or above the minimum replication factor, re-replication occurs after the timeout specified in the

`cldb.fs.mark.rereplicate.sec` parameter. Note that this timeout is the time given for a node that is offline to come back online. After this timeout period, the CLDB takes action to restore the replication factor.

> **Tip:** For more information, see Understanding Replication on page 454.

Parameter: `replicationtype`

Default Value: `high_throughput`

Possible Values:

- `low_latency` (star replication)
- `high_throughput` (chain replication)

Description: The desired replication type. The default setting is `high_throughput`.

Parameter: `rereplicationtimeoutsec`

Default Value: 3600 seconds (1 hour)

Possible Values: Any positive integer.

Description: Timeout (in seconds) before attempting re-replication of replica containers. This volume property defines the timeout period until CLDB starts re-replicating the containers on the node of the volume after CLDB stops receiving a heartbeat from the node.

When a node is down, CLDB gives the node an hour to come back online before it takes any action for the containers on this node. You can set this parameter on volumes to reduce the default value to a shorter time period. This option is provided mainly for local volumes, so that when the MapR File System is down, CLDB can give up quickly and decide that the container has no master. This forces the TT to give up on local containers, and take the appropriate recovery action of deleting the mapred volume and creating another one.

Parameter: `rootdirgroup`

Default Value: User who is running the command

Possible Values: Any valid group

Description: Group that owns the root directory

Parameter: `rootdirperms`

Default Value: `rwxr-xr-x`

Possible Values: Any valid permission

Description: Permissions on the volume root directory.

Parameter: `rootdiruser`

Default Value: User who is running the command

Possible Values: Any valid user

Description: User that owns the root directory.

Parameter: `schedule`

Default Value: 0

Possible Values: `0` or a valid schedule ID.

Description: The ID of a schedule. Use the `schedule list` command to find the ID of the named schedule that you want to apply to the volume.

To disable the schedule, set this parameter to `0`.

Parameter: `skipinherit`

Default Value: No default value

Possible Values:

last-updated: Apr 11, 2022

- `schedule`
- `tiername`

Description:Specifies not to inherit given properties associated with the:

- Parent volume (for other volumes)
- Source volume (for mirror volumes)

Value must be either or both:

- `schedule` to not inherit snapshot schedule settings
- `tiername` to not:
  - Inherit tiering properties like `tierid`, `tieroffloadscheduleid`, `ecshceme`, `ectopology`
  - Set default values for `compactionscheduleid` and `compactionoverheadthreshold`

Use comma to separate multiple values.

Parameter: `source`

Default Value: No default value

Possible Values: Any volume.

Description: The source volume from which a mirror volume receives updates, specified in the format `<volume>@<cluster>`.

Parameter: `tenant`

Default Value: No default value

Possible Values: Any valid tenant user.

Description: The tenant is the entity for which resources such as volumes are created. The tenant can be an organization, a department within an organization, or an individual.

This parameter indicates the tenant for whom the volume is being created. All resources within the created volume are owned by the specified tenant.

Parameter: `tierencryption`

Default Value: false

Possible Values:

- `true`
- `false`

Description: Specifies whether to enable (`true`) or disable (`false`) encryption of data on the object store. This parameter is applicable only for cold-tier volumes. If you enable this parameter, user data is encrypted before being written to the object, and the HTTPS protocol is used for communication with the object store to ensure that data is encrypted both on the wire and on the tier.

You can set this parameter only if you specify a tier name (see the `tiername` parameter) as well. You cannot modify this parameter after you set it.

If you set the value to `true`, you can also specify a custom key using the `tierkey` parameter. Once set to `true`, the MAST Gateway uses HTTPS to upload data to the cold-tier. If the cold tier does not support HTTPS, all tier related operations fail. If the cold-tier does not support HTTPS, you must explicitly set the value for this to `false` at the time of associating a tier with the volume because the default value for this parameter is `true`.

> **Tip:** For warm tier, use `-dare` option on the front-end volume to enable or disable encryption of data-at-rest.

Parameter: `tieringenable`

Default Value: No default value

Possible Values:

- true
- false

Description: Enable (`true`) or disable (`false`) tiering for the volume. When you specify this parameter, you must also specify the `tiername`. For creating a tiering-enabled mirror volume, specify this parameter if the source volume is enabled for cold-tier; specify either this parameter or `ecenable`, if the source volume is enabled for warm-tier.

Parameter: `tieringrule`

Default Value: p (all files)

Possible Values: Name of any valid rule

Description: The name of the rule (referred to as storage policy in the Control System) to use for offloading data to the tier. If you do not specify a rule, the default rule, which is all files (p), is associated with the volume. See Creating a Rule in Creating a Storage Tier Policy on page 972 for more information.

Parameter: `tierkey`

Default Value: Auto generated

Possible Values:Any 32-character HEX string, or let CLDB auto-generate this string

Description: The 32-character HEX string to use for encryption only for cold tier volumes. If you do not specify a string, CLDB generates a 32 character HEX string to use for encrypting the data to offload to the tier.

Parameter: `tiername`

Default Value: No default value

Possible Values: Any

Description: The name of the tier to use for offloading data. You can set this name only once and cannot modify it.

For warm tiering, you cannot specify this parameter if `ecenable` is set to `true`.

Parameter: `topology`

Default Value: `/data`

Possible Values: Any

Description: The rack path to the volume.

To create a volume in a specific topology, you must have the Converged Enterprise Edition installed

last-updated: Apr 11, 2022

on your system. Without the Converged Enterprise Edition, when you run the `maprcli volume create` command with the `-topology` option, the following error message is returned:

```
ERROR (10010) -  Volume Creation
Failed: Setting topology on
                        a volume
requires data placement feature.
License not found for data placement.
```

Parameter: `type`

Default Value: 0

Possible Values:

- `mirror`
- `rw`
- `0`
- `1`

Description: The type of volume to create.

The following values are accepted:

- `mirror` - standard mirror (read-only) volume (promotable to standard read-write volume)
- `rw` - standard (read-write) volume (convertible to standard mirror volume)
- `0` - standard (read-write) volume (for backward compatibility)
- `1` - non-convertible mirror (read-only) volume (for backward compatibility)

Parameter: `user`

Default Value: All permissions (`dump, restore, m, a, d, fc`) for the administrator who created the volume

Possible Values: Any valid permissions

Description: Space-separated list of `user:permission` pairs.

Use comma to separate permissions. For example: `user:permission,permission,...`

Parameter: `wiresecurityenabled`

Default Value: true

Possible Values:

- `true`
- `false`

Description: Enables (`true`) or disables (`false`) on-wire encryption for all files, tables, and streams in the volume for secure clusters. This parameter is not supported on insecure clusters.

If `true`, this setting overrides all file, table, and stream level encryption settings (set using the `hadoop mfs` command) and enables on-wire encryption for all

files, tables, and streams. If you disable (`false`) this parameter at the volume level, but enable it at the file, table, or stream level, the file, table, or stream level encryption setting overrides this setting on those files, tables, and streams where it is enabled; for all other files, tables, and streams where encryption is not enabled at the file, table, or stream level, the on-wire encryption is disabled.

Parameter: `writeAce`

Default Value: p (grants access to all users)

Possible Values: Any valid permissions

Description: Specifies Access Control Expressions (ACEs) that grant permission at the volume level to write to files and tables in the volume. The default value is p, which grants access to all users.

See *ACE*s.

Inheritance

The following table shows the list of inheritable parameters that are (Yes) and are not (No) inherited by a:

- Mirror volume from the source volume on the same cluster
- Mirror volume from the source volume on a different cluster

Note: All (non-mirror) volumes inherit all the inheritable properties from the parent volume. For more information on the properties, refer to `volume create` parameters.

Inheritable Properties (which are inherited by non-mirror volumes by default)	Inherited by Mirror Volume on the same cluster as the source volume?	Inherited by Mirror Volume on a different cluster from the source volume?
`advisoryquota`	Yes	Yes
`ae`	Yes	No
`aetype`	Yes	No
`allowgrant`	Yes	Yes
`allowinherit`	Yes	Yes
`auditenabled`	Yes	Yes
`coalesce`	Yes	Yes
`dare`	Yes	Yes[1], No[2]
`dataauditops`	Yes	Yes
`dbindexlagsecalarmthresh`	Yes	Yes
`dbrepllagsecalarmthresh`	Yes	Yes
`ecscheme`	Yes	No
`ectopology`	Yes	No
`group`	Yes	Yes
`inherit`	Yes	Yes
`localvolumehost`	No	No
`localvolumeport`	No	No

last-updated: Apr 11, 2022

Inheritable Properties (which are inherited by non-mirror volumes by default)	Inherited by Mirror Volume on the same cluster as the source volume?	Inherited by Mirror Volume on a different cluster from the source volume?
`maxinodesalarmthreshold`	Yes	Yes
`minreplication`	Yes	Yes
`mirrorschedule`	Yes	No
`mirrorthrottle`	Yes	Yes
`nsminreplication`	Yes	Yes
`nsreplication`	Yes	Yes
`ofloadschedule`	Yes	No
`quota`	Yes	Yes
`readonly`	Yes	Yes
`recallexpirytime`	Yes	No
`replication`	Yes	Yes
`replicationtype`	Yes	Yes
`rereplicationtimeoutsec`	Yes	Yes
`rootdirperms`	Yes	Yes
`schedule`	Yes[3]	No
`source`	Yes	Yes
`tierencryption`	Yes	No
`tieringenable`	Yes	No
`tieringrule`	Yes	No
`tierkey`	Yes	No
`tiername`[4]	Yes	No
`topology`	Yes	No
`type`	Yes	Yes
`user`	Yes	Yes
`wiresecurityenabled`	Yes	Yes

- [1] If destination cluster is also enabled for data-at-rest encryption, `dare` setting is inherited by the mirror volume on the destination cluster.
- [2] If destination cluster is not enabled for data-at-rest encryption, `dare` setting is not inherited by the mirror volume on the destination cluster.
- [3] If `schedule` keyword is specified with the `skipinherit` parameter, schedule(s) are not inherited while inheriting volume properties from the source volume.
- [4] If `tiername` keyword is specified with the `skipinherit` parameter:
  - The tiering properties are not inherited by the mirror volume while inheriting volume properties from the tiering-enabled source volume.
  - For volumes enabled for warm-tier, the backend erasure-coded volume is not created.

last-updated: Apr 11, 2022

Examples

Create the volume "test-volume" mounted at "/test/test-volume"

CLI

```
/opt/mapr/bin/maprcli volume
create -name test-volume -path /test/
test-volume -type rw  -json
{
    "timestamp":1526522204072,
    "timeofday":"2020-05-16
06:56:44.072 GMT-0700 PM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created volume:
'test-volume'"
    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=test-volume&path=/test/
test-volume&type=rw' --user mapr:mapr
{"timestamp":1526522305703,"timeofday"
:"2020-05-16 06:58:25.703 GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'test-volume'"]}
```

Create Volume with a Quota and an Advisory Quota

This example creates a volume with the following parameters:

- advisoryquota: 100M
- name: volumename
- path: /volumepath
- quota: 500M
- replication: 3
- schedule: 2
- topology: /East Coast
- type: rw

CLI

```
/opt/mapr/bin/maprcli volume
create -name volumename -path /
volumepath -advisoryquota 100M -quota
500M -replication 3 -schedule
2 -topology "/East Coast" -type
rw -json
{
```


last-updated: Apr 11, 2022

```
    "timestamp":1526522474660,
    "timeofday":"2018-05-16
07:01:14.660 GMT-0700 PM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created volume:
'volumename'"
    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=volumename&path=/
volumepath&advisoryquota=100M&quota=50
0M&replication=3&schedule=2&type=rw'
 --user mapr:mapr
{"timestamp":1526522622494,"timeofday"
:"2018-05-16 07:03:42.494 GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'volumename'"]}
```

Create the mirror volume "test-volume.mirror" from source volume "test-volume" and mount at "/test/test-volume-mirror"

CLI

```
/opt/mapr/bin/maprcli
volume create -name
test-volume.mirror -source
test-volume@ksTest -path /test/
test-volume-mirror -type mirror -json
{
    "timestamp":1526524458615,
    "timeofday":"2018-05-16
07:34:18.615 GMT-0700 PM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created
volume: 'test-volume.mirror'"
    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=test-volume.mirror&path=/test/
test-volume-mirror&type=mirror&source=
test-volume@ksTest' --user mapr:mapr
{"timestamp":1526524637534,"timeofday"
:"2018-05-16 07:37:17.534 GMT-0700
PM","status":"OK","total":0,"data":
```

```
[],"messages":["Successfully created
volume: 'test-volume.mirror'"]}
```

Create volumes that inherit from a parent volume

When creating and mounting a volume, the location of the mount path is specified by the path parameter. Volumes can be mounted via the web console, the `maprcli` commands, or the REST commands. The `maprcli` commands include `volume create -path` command and the `maprcli volume mount -path` command if the volume was previously created. Sub-volumes (children) can inherit properties from their parent volume.

In the following example, a parent volume and two (2) child volumes are create where the child volume inherit properties from the parent. When the `inherit` flag is explicitly used, the `allowgrant` parameter for the parent volume is not required.

- For child volumes, c1 and c2, inheritance is explicit because the `inherit` option is specified. Thus, p1.c1 and p1.c2 volumes will inherit all properties from volume p1 (note that p1 is not a parent of p1.c1) regardless of whether the `allowgrant` option is set on p1 or not. In this case, there is an explicit inheritance ant the `allowgrant` flag is ignored and volume properties are inherited.
- For the child volume, c3, inheritance is implicit. Meaning, the child volume, p1.c3, inherits all properties from the parent volume, p1, only if the `allowgrant` option is set on p1.

CLI

```
/opt/mapr/bin/maprcli volume
create -name p1 -path /p1
/opt/mapr/bin/maprcli volume
create -name p1.c1 -inherit p1
/opt/mapr/bin/maprcli
volume create -name
p1.c2 -path /p1/c2 -inherit p1
/opt/mapr/bin/maprcli volume
create -name p1.c3 -path /p1/c3
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=p1&path=%2Fp1' --user mapr:mapr
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=p1.c1&inherit=p1' --user
mapr:mapr
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=p1.c2&path=%2Fp1%2Fc2&inherit=p1'
 --user mapr:mapr
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=p1.c3&path=%2Fp1%2Fc3' --user
mapr:mapr
```

In the following example, the p1.child volume normally inherits from the p1 parent volume properties because p1.child is mounted under p1 and `allowgrant` option is set to `true` on the parent volume. However, if the child volume doesn't want to inherit properties, then set the `allowinherit` option to `false` (default: `true`).

CLI

```
/opt/mapr/bin/maprcli volume
create -name p1 -path /p1 -allowgrant
```

```
true
/opt/mapr/bin/maprcli volume
create -name p1.child -path /p1/
p1.child -allowinherit false
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=p1&path=%2Fp1&allowgrant=true'
 --user mapr:mapr
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=p1.child&path=%2Fp1%2Fp1.child&al
lowinherit=false' --user mapr:mapr
```

Create a volume with namespace container replicas

CLI

```
/opt/mapr/bin/maprcli
volume create -name
testVol -nsminreplication
2 -nsreplication 3 -json
{
    "timestamp":1526525132522,
    "timeofday":"2018-05-16
07:45:32.522 GMT-0700 PM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created volume:
'testVol'"
    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=testVol&path=/
testVol&nsminreplication=2&nsreplicati
on=3' --user mapr:mapr
{"timestamp":1526525257461,"timeofday"
:"2018-05-16 07:47:37.461 GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'testVol'"]}
```

Create a volume and set ACEs

CLI

```
/opt/mapr/bin/maprcli volume
create -name testVol -readAce
p -writeAce 'g:group1&!u:user1' -json
{
    "timestamp":1526525429326,
    "timeofday":"2018-05-16
07:50:29.326 GMT-0700 PM",
    "status":"OK",
    "total":0,
```

```
    "data":[

    ],
    "messages":[
        "Successfully created volume:
'testVol'"
    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=testVol&readAce=p&writeAce=g%3Agr
oup1%26%21u%3Auser1' --user mapr:mapr
{"timestamp":1526525572035,"timeofday"
:"2018-05-16 07:52:52.035 GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'testVol'"]}
```

Create a volume with auditing disabled for specific operations

CLI

```
/opt/mapr/bin/maprcli volume
create -name
test-volume -auditenabled
true -dataauditops --lookup,-read,-wri
te -json
{
    "timestamp":1526525720308,
    "timeofday":"2018-05-16
07:55:20.308 GMT-0700 PM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created volume:
'test-volume'"
    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=test-volume&path=/test/
test-volume&auditenabled=true&dataaudi
tops=%2D%2Dlookup%2C%2Dread%2C%2Dwrite
' --user mapr:mapr
{"timestamp":1526525795017,"timeofday"
:"2018-05-16 07:56:35.017 GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'test-volume'"]}
```

Create a volume and grant user permissions on the volume:

CLI Example 1

```
/opt/mapr/bin/maprcli volume
create -name testVol1 -path /
```

```
testVol1 -user user1:dump
user2:fc -json
{
    "timestamp":1521162402826,
    "timeofday":"2018-03-15
06:06:42.826 GMT-0700 PM",
    "status":"OK",
    "total":0,
    "data":[
    ],
    "messages":[
        "Successfully created volume:
'testVol1'"
    ]
}
```

REST

```
curl -k -X POST 'https://
10.10.82.24:8443/rest/volume/create?
name=testVol&path=/
testVol&user=user1%3Adump%20user2%3Afc
' --user mapr:mapr
{"timestamp":1526526072608,"timeofday"
:"2018-05-16 08:01:12.608 GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'testVol'"]}
```

CLI Example 2

```
/opt/mapr/bin/maprcli volume
create -name testVol -path /
testVol2 -user user1:dump,restore
user2:a,fc -json
{
    "timestamp":1521162467485,
    "timeofday":"2018-03-15
06:07:47.485 GMT-0700 PM",
    "status":"OK",
    "total":0,
    "data":[
    ],
    "messages":[
        "Successfully created volume:
'testVol2'"
    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=testVol2&path=/
testVol2&user=user1%3Adump%2Crestore%2
0user2%3Aa%2Cfc' --user mapr:mapr
{"timestamp":1526526256845,"timeofday"
:"2018-05-16 08:04:16.845 GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'testVol2'"]}
```

Create a volume for a tenant

This example creates a volume for a tenant with the following parameters:

- `name`: volumename
- `path`: /volumepath
- `advisoryquota`: 500MB
- `quota`: 1GB
- `replication`: 3

CLI

```
/opt/mapr/bin/maprcli volume
create -name tenantVol -cluster
ksTest -path /egTenant -tenantuser
egTenant -advisoryquota 500M -quota
1G -replication 3 -json
{
    "timestamp":1526526462865,
    "timeofday":"2018-05-16
08:07:42.865 GMT-0700 PM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created volume:
'tenantVol'"
    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=tenantVol&cluster=ksTest&path=/
egTenant&advisoryquota=500M&quota=1G&r
eplication=3' --user mapr:mapr
{"timestamp":1526526615167,"timeofday"
:"2018-05-16 08:10:15.167 GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'tenantVol'"]}
```

Create a volume with on-wire encryption enabled for all files and tables in the volume:

CLI

```
/opt/mapr/bin/maprcli
volume create -name
test-Volume -path /testvolume -type
rw -wiresecurityenabled true -json
{
    "timestamp":1526526686905,
    "timeofday":"2018-05-16
08:11:26.905 GMT-0700 PM",
    "status":"OK",
    "total":0,
    "data":[

    ],
```

```
    "messages":[
        "Successfully created volume:
'test-Volume'"
    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=test-volume&path=/
testVolume&type=rw&wiresecurityenabled
=true' --user mapr:mapr
{"timestamp":1526526748723,"timeofday"
:"2018-05-16 08:12:28.723 GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'test-volume'"]}
```

Create a sub-volume and do not inherit schedules from the parent volume:

CLI

```
/opt/mapr/bin/maprcli volume
create -name p1.c2 -path /p1/
p1.c2 -skipinherit schedule -json
{
        "timestamp":1505196021575,
        "timeofday":"2017-09-11
11:00:21.575 GMT-0700",
        "status":"OK",
        "total":0,
        "data":[

        ],
        "messages":[
                "Successfully created
volume: 'p1.c2'"
        ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=p1.c2&path=/p1/
p1.c2&skipinherit=schedule' --user
mapr:mapr
{"timestamp":1526526980643,"timeofday"
:"2018-05-16 08:16:20.643 GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'p1.c2'"]}
```

Create a mirror volume and do not inherit the schedule(s) from the source volume:

CLI

```
/opt/mapr/bin/maprcli volume
create -name p1.m2 -path /p1/
p1.m2 -type mirror -source
p1@ksTest -skipinherit schedule -json
{
        "timestamp":1505196450141,
        "timeofday":"2017-09-11
```

```
11:07:30.141 GMT-0700",
        "status":"OK",
        "total":0,
        "data":[

        ],
        "messages":[
                "Successfully created
volume: 'p1.m2'"
        ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=p1.m2&path=/p1/
p1.m2&type=mirror&source=p1@ksTest&ski
pinherit=schedule' --user mapr:mapr
{"timestamp":1526527151925,"timeofday"
:"2018-05-16 08:19:11.925 GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'p1.m2'"]}
```

Create a volume and enable tiering, but do not specify the tier type and do not associate an offload rule or schedule:

CLI

```
/opt/mapr/bin/maprcli volume
create -name sampleVol -path /
sampleVol -tieringenable true -json
{
    "timestamp":1519922099117,
    "timeofday":"2018-03-01
08:34:59.117 GMT-0800 AM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created volume:
'sampleVol'"
    ]
}
```

REST

```
curl -k -X POST 'https://
10.10.82.24:8443/rest/volume/create?
name=sampleVol&path=/
sampleVol&tieringenable=true' --user
mapr:mapr
{"timestamp":1519922181381,"timeofday"
:"2018-03-01 08:36:21.381 GMT-0800
AM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'sampleVol'"]}
```

Create a volume, enable cold tiering, associate a rule and schedule for offloading data, and set the number of days to keep recalled data:

CLI

```
/opt/mapr/bin/maprcli volume
create -name sampleVol -path /
sampleVol -tieringenable
true -tiername
ksTestCold -tieringrule
rule1 -offloadschedule
2 -recallexpirytime 2 -json
{
    "timestamp":1519922642632,
    "timeofday":"2018-03-01
08:44:02.632 GMT-0800 AM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created volume:
'sampleVol'"
    ]
}
```

REST

```
curl -k -X POST 'https://
10.10.82.24:8443/rest/volume/create?
name=sampleVol&path=/
sampleVol&tieringenable=true&tiername=
ksTestCold&tieringrule=rule1&offloadsc
hedule=2&recallexpirytime=2' --user
mapr:mapr
{"timestamp":1519922784818,"timeofday"
:"2018-03-01 08:46:24.818 GMT-0800
AM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'sampleVol'"]}
```

Create a volume, enable warm tiering, associate a rule and schedule for offloading data, and set the number of days to keep recalled data:

CLI

```
/opt/mapr/bin/maprcli volume
create -name sampleVol -path /
sampleVol -tieringenable
true -tiername ksTestEC -tieringrule
testRule -ecscheme 6+3 -ectopology /
ecdata -offloadschedule
2 -recallexpirytime 2 -json
{
    "timestamp":1516336193635,
    "timeofday":"2018-01-19
04:29:53.635 GMT+0000",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created volume:
'sampleVol'"
```

```
    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=sampleVol&path=/
sampleVol&tieringenable=true&tiername=
testWarm&tieringrule=testRule&ecscheme
=6%2B3&ectopology=/
ecdata&offloadschedule=2&recallexpiryt
ime=2' --user mapr:mapr
{"timestamp":1526521538688,"timeofday"
:"2018-05-16 06:45:38.688 GMT-0700
PM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'sampleVol'"]}
```

Create a volume and enable it for warm tiering, but do not specify tier name:

CLI

```
/opt/mapr/bin/maprcli volume
create -name sampleVol3 -path /
sampleVol3 -ecenable true -json
{
    "timestamp":1527690187540,
    "timeofday":"2018-05-30
07:23:07.540 GMT-0700 AM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully created volume:
'sampleVol3'"
    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/create?
name=sampleVol3&path=/
sampleVol3&ecenable=true'  --user
mapr:mapr
{"timestamp":1527690187540,"timeofday"
:"2018-05-30 07:23:07.540 GMT-0700
AM","status":"OK","total":0,"data":
[],"messages":["Successfully created
volume: 'sampleVol3'"]}
```

Related concepts

node on page 1694
Manages nodes in the cluster

Related reference

disk add on page 1602
Adds one or more disks to the specified node. Permissions required: `fc` or `a`.

node list on page 1705
Lists nodes in the cluster.

configure.sh on page 2053
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

volume dump create
Creates a volume *dump file* containing data from a volume for distribution or restoration.

Permissions Required

`dump` or `fc` on the volume.

Note: In a secure cluster, you must use the MapR user ID. Using `root` or any other user ID results in the system hanging.

Syntax

CLI

```
/opt/mapr/bin/maprcli volume dump
create
     [ -cluster cluster_name
     [-s startvolumepointname]
     [-e endvolumepointname]
     [-o (for dumpfile on stdout)]
     [-dumpfile dumpfilename (ignored
if -o is used)]
     -name volumename
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/dump/create?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`dumpfile`	The name of the dump file (ignored if `-o` is used).
`e`	The name of the state file to create for the end point of the dump.
`name`	A volume name.
`o`	This option dumps the volume to stdout instead of to a file.
`s`	The start point for an incremental dump.

Note: The data is not encrypted in the dump file created for a volume enabled for data at rest encryption.

Examples

Create a full dump:

CLI

```
/opt/mapr/bin/maprcli volume dump
create -e statefile1 -dumpfile
fulldump1 -name volume -n
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/dump/
create?
e=statefile1&dumpfile=fulldump1&name=v
olume&n' --user mapr:mapr
```

Create an incremental dump:

CLI

```
/opt/mapr/bin/maprcli volume
dump create -s statefile1 -e
statefile2 -name volume -dumpfile
incrdump1
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/dump/
create?
s=statefile1&e=statefile2&name=volume&
dumpfile=incrdump1' --user mapr:mapr
```

Create and Maintain Volume Dump File
Describes how to create full dump files and add incremental volume dump files.

You can use `volume dump create` to create two types of files:

- *full* dump files containing all data in a volume
- *incremental* dump files that contain changes to a volume between two points in time

A full dump file is useful for restoring a volume from scratch. An incremental dump file contains the changes necessary to take an existing (or restored) volume from one point in time to another. Along with the dump file, a full or incremental dump operation can produce a *state* file (specified by the ?-e parameter) that contains a table of the version number of every container in the volume at the time the dump file was created. This represents the *end point* of the dump file, which is used as the *start point* of the next incremental dump. The main difference between creating a full dump and creating an incremental dump is whether the -s parameter is specified; if -s is not specified, the volume create command includes all volume data and creates a full dump file. If you create a full dump followed by a series of incremental dumps, the result is a sequence of dump files and their accompanying state files:

dumpfile1 statefile1

dumpfile2 statefile2

dumpfile3 statefile3

Note: You can restore the volume from scratch, using the volume dump restore command with the full dump file, followed by each dump file in sequence.

When you create a dump file for a volume enabled for data at rest encryption, data in the dump file is not encrypted.

Note: In a secure cluster, you must use the MapR user ID. Using root or any other user ID results in the system hanging.

last-updated: Apr 11, 2022

To create and maintain an up-to-date dump of a volume:

1. Create a full dump file.
   Example:

```
maprcli volume dump create -name cli-created -dumpfile fulldump1 -e
statefile1
```

2. Periodically, add an incremental dump file.
   Examples:

```
maprcli volume dump create -s statefile1 -e statefile2 -name
cli-created -dumpfile incrdump1
maprcli volume dump create -s statefile2 -e statefile3 -name
cli-created -dumpfile incrdump2
maprcli volume dump create -s statefile3 -e statefile4 -name
cli-created -dumpfile incrdump3
```

   ...and so on.

volume dump restore
Restores or updates a volume from a dump file. Permissions required: `fc` or `restore` on the volume.

Syntax

CLI

```
maprcli volume dump restore
     [ -cluster cluster_name ]
     [-i (read dump from stdin)]
     [-n (create new volume if it
doesn't exist)]
     [-dumpfile dumpfilename (ignored
if -i is used)]
     [-full <true|false> (perform
full volume restore, default:false ]
     -name volumename
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/dump/restore?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`dumpfile`	The name of the dumpfile (ignored if `-i` is used).
`i`	This option reads the dump file from `stdin`.
`n`	This option creates a new volume if it doesn't exist.
`full`	Perform either a full volume restore or an incremental volume restore. The default restore is incremental.

Parameter	Description
`name`	A volume name, in the form `volumename`

Note: In a secure cluster, you must use the MapR user ID. Using `root` or any other user ID results in the system hanging.

Examples
Restore a volume from a full dump file:

CLI

```
maprcli volume dump restore -name
volume -dumpfile fulldump1 -full true
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/dump/
restore?
name=volume&dumpfile=fulldump1&full=tr
ue' --user mapr:mapr
```

Apply an incremental dump file to a volume:

CLI

```
maprcli volume dump restore -name
volume -dumpfile incrdump1
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/dump/
restore?
name=volume&dumpfile=incrdump1' --user
 mapr:mapr
```

Restore Volume From a Dump
Describes how to restore from full and incremental dump files.

There are two ways to use volume dump restore:

- With a full dump file, `volume dump restore` recreates a volume from scratch from volume data stored in the dump file.
- With an incremental dump file, `volume dump restore` updates a volume using incremental changes stored in the dump file.

The volume that results from a `volume dump restore` operation is a mirror volume whose source is the volume from which the dump was created. After the operation, this volume can perform mirroring from the source volume.

When you are updating a volume from an incremental dump file, you must specify an existing volume and an incremental dump file. To restore from a sequence of previous dump files would involve first restoring from the volume's full dump file, then applying each subsequent incremental dump file.

Note: In a secure cluster, you must use the MapR user ID. Using root or any other user ID results in the system hanging.

A restored volume may contain mount points that represent volumes that were mounted under the original source volume from which the dump was created. In the restored volume, these mount points have no meaning and do not provide access to any volumes that were mounted under the source volume. If the

last-updated: Apr 11, 2022

source volume still exists, then the mount points in the restored volume will work if the restored volume is associated with the source volume as a mirror.

To restore from a full dump plus a sequence of incremental dumps:

1. Restore from the full dump file, using the `-n` option to create a new mirror volume and the `-name` option to specify the name.
   Example:

   ```
   maprcli volume dump restore -dumpfile fulldump1 -name restore1 -n
   ```

2. Restore from each incremental dump file in order, specifying the same volume name.
   Examples:

   ```
   maprcli volume dump restore -dumpfile incrdump1 -name restore1 maprcli
                           volume dump restore -dumpfile incrdump2 -name
   restore1 maprcli volume dump restore
                           -dumpfile incrdump3 -name restore1
   ```

   ...and so on.

volume fixmountpath
Corrects the mount path of a volume. Permissions required: `fc` or `m` on the volume.

The CLDB maintains information about the mount path of every volume. If a directory in a volume's path is renamed (by a `hadoop fs` command, for example) the information in the CLDB will be out of date. The `volume fixmountpath` command does a reverse path walk from the volume and corrects the mount path information in the CLDB.

Syntax

CLI

```
maprcli volume fixmountpath
-name <name>
[ -cluster <clustername> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/fixmountpath?<parameters>`

Parameters

Parameter	Description
`name`	The volume name.
`cluster`	The cluster name

Examples

Fix the mount path of volume v1:

CLI

```
maprcli volume fixmountpath -name v1
```

REST

```
https://abc.sj.us:8443/rest/volume/
fixmountpath?name=v1
```

volume info
Displays information about the specified volume. For JSON formatted output, use the `-json` option when running the command.

Syntax

CLI

```
maprcli volume info
    [ -cluster <cluster name> ]
    [ -output verbose. default:
verbose ]
    [ -path <mount directory> ]
    [ -name <volume name> ]
    [ -columns comma separated list
of column names. default: all ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/volume/info?<parameters>`

Parameters

You must specify either name or path, but not both.

Parameter	Description
`cluster`	The cluster on which to run the command.
`columns`	A comma-separated list of fields to return in the query. See the Fields table on the `volume` command page. Default: `all`
`name`	The volume name for which to retrieve information. When issuing the `maprcli volume info -columns` and `marcli volume list -columns` commands, the column for the volume name is `volumename`.
`output`	Indicates whether the output should be terse or verbose. Default: verbose
`path`	The mount path of the volume for which to retrieve information.

Output

For definitions of the output fields, and short names for use with filters, see the Fields table on the `volume` command page.

Examples

Return information on standard volume named `test_vol`:

CLI

```
/opt/mapr/bin/maprcli volume
info -name test_vol -json
{
    "timestamp":1520275171388,
    "timeofday":"2018-03-05
06:39:31.388 GMT+0000",
    "status":"OK",
    "total":1,
    "data":[
        {
            "acl":{
                "Principal":"User
root",
                "Allowed actions":[
                    "dump",
                    "restore",
                    "m",
                    "a",
                    "d",
                    "fc"
                ]
            },
            "creator":"root",
            "aename":"root",
            "aetype":0,


            "numreplicas":"3",
            "minreplicas":"2",
            "nsNumReplicas":"3",
            "nsMinReplicas":"2",

"enforceMinReplicationForIO":"false",

"containerAllocationFactor":"0",
            "allowGrant":"false",
            "reReplTimeOutSec":"0",

"criticalReReplTimeOutSec":"0",

"replicationtype":"high_throughput",
            "rackpath":"/data/
default-rack/atsqa4-197.qa.lab",
            "mirrorthrottle":"1",
            "accesstime":"March 5,
2018",
            "readonly":"0",
            "mountdir":"/
VOL3_TST_1517344011351",

"volumename":"VOL3_TST_1517344011351",
            "mounted":1,
            "quota":"0",
            "advisoryquota":"0",
            "snapshotcount":"0",
            "logicalUsed":"0",

"replicatedlogicalused":"0",
```

```
            "used":"0",
            "snapshotused":"0",
            "totalused":"0",
            "replicatedtotalused":"0",
            "scheduleid":0,
            "schedulename":"",
            "mirrorscheduleid":0,
            "volumetype":0,
            "mirrortype":3,
            "creatorcontainerid":2825,

"creatorvolumeuuid":"-6688596250779888
467:-14709419608161716 92",
            "volumeid":28803390,
            "actualreplication":[
                0,
                100,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0
            ],
            "nameContainerSizeMB":0,
            "nameContainerId":2825,

"nameContainerDataThresholdMB":524288,
            "needsGfsck":false,

"maxinodesalarmthreshold":"0",

"maxnssizembalarmthreshold":"0",

"dbrepllagsecalarmthresh":"0",

"dbindexlagsecalarmthresh":"0",
            "limitspread":"true",
            "partlyOutOfTopology":0,
            "wireSecurity":0,
            "auditVolume":0,
            "audited":0,
            "coalesceInterval":60,

"enableddataauditoperations":"getattr,
setattr,chown,chperm,chgrp,
getxattr,listxattr,setxattr,removexatt
r,read,write,create,delete,mkdir,
readdir,rmdir,createsym,lookup,rename,
createdev,truncate,tablecfcreate, \

tablecfdelete,tablecfmodify,tablecfSca
n,tableget,tableput,tablescan,

tablecreate,tableinfo,tablemodify,getp
erm,getpathforfid,hardlink,cdscan,

fileoffload,filerecall,filetierjobstat
us,filetierjobabort",
```

```
"disableddataauditoperations":"getattr
,filetieroffloadevent,

filetierrecallevent",
            "volumeAces":{
                "readAce":"p",
                "writeAce":"p"
            },
            "fixCreatorId":"false",

"ReplTypeConversionInProgress":"0",

"creationTime":1524067828279,
            "metricsEnabled":0,
            "dareEnabled":1,

              "tierenable":"false"
        }
    ]
}
```

REST

```
curl -k -X GET 'https://
abc.sj.us:8443/rest/volume/info?
name=test_vol' --user mapr:mapr
{"timestamp":1529545951212,"timeofday"
:"2018-06-20 06:52:31.212 GMT-0700
PM","status":"OK",
 "total":1,"data":[{"acl":
{"Principal":"User mapr",
 "Allowed actions":"[dump, restore,
m, a, d,
fc]"},"creator":"mapr","aename":"mapr"
,"aetype":"0",
"numreplicas":"3","minreplicas":"2","n
sNumReplicas":"3","nsMinReplicas":"2",

"enforceMinReplicationForIO":"true","c
ontainerAllocationFactor":"0","reReplT
imeOutSec":"0",

"criticalReReplTimeOutSec":"0","replic
ationtype":"high_throughput","rackpath
":"/data",

"mirrorthrottle":"1","accesstime":"Jun
e 20,
2018","readonly":"0","mountdir":"/
test_vol",

"volumename":"test_vol","mounted":1,"q
uota":"0","advisoryquota":"0","snapsho
tcount":0,

"logicalUsed":"0","replicatedlogicalus
ed":"0","used":"0","snapshotused":"0",
"totalused":"0",

"replicatedtotalused":"0","scheduleid"
:"0","schedulename":"","mirrorschedule
id":"0",
```

```
"volumetype":"0","mirrortype":3,"creat
orcontainerid":2117,

"creatorvolumeuuid":"-8953141547368591
763:-5762925753444373354","volumeid":8
4378231,
 "actualreplication":
[0,100,0,0,0,0,0,0,0,0,0],"nameContain
erSizeMB":0,"nameContainerId":2117,

"nameContainerDataThresholdMB":524288,
"needsGfsck":"false","maxinodesalarmth
reshold":"0",

"maxnssizembalarmthreshold":"0","dbrep
llagsecalarmthresh":"0","dbindexlagsec
alarmthresh":"0",

"limitspread":"true","partlyOutOfTopol
ogy":0,"wireSecurity":1,"auditVolume":
0,"audited":0,

"forceAudit":0,"coalesceInterval":60,"
enableddataauditoperations":"setattr,c
hown,chperm,chgrp,

getxattr,listxattr,setxattr,removexatt
r,read,write,create,delete,mkdir,readd
ir,rmdir,createsym,

lookup,rename,createdev,truncate,table
cfcreate,tablecfdelete,tablecfmodify,t
ablecfScan,tableget,

tableput,tablescan,tablecreate,tablein
fo,tablemodify,getperm,getpathforfid,h
ardlink,filescan,

fileoffload,filerecall,filetierjobstat
us,filetierjobabort",

"disableddataauditoperations":"getattr
,filetieroffloadevent,filetierrecallev
ent",
 "volumeAces":
{"writeAce":"p","readAce":"p"},"fixCre
atorId":"false",

"ReplTypeConversionInProgress":0,"crea
tionTime":1529545198145,"metricsEnable
d":1,

"dareEnabled":1,"tierenable":"false"}]
}
```


Return information on volume named `volt_warm` enabled for warm-tier:

CLI


```
/opt/mapr/bin/maprcli volume
info -name volt_warm -json
{
```

```
    "timestamp":1529546449530,
    "timeofday":"2018-06-20
07:00:49.530 GMT-0700 PM",
    "status":"OK",
    "total":1,
    "data":[
        {
            "acl":{
                "Principal":"User
mapr",
                "Allowed
actions":"[dump, restore, m, a, d,
fc]"
            },
            "creator":"mapr",
            "aename":"mapr",
            "aetype":"0",


            "numreplicas":"3",
            "minreplicas":"2",
            "nsNumReplicas":"3",
            "nsMinReplicas":"2",

"enforceMinReplicationForIO":"false",

"containerAllocationFactor":"0",
            "reReplTimeOutSec":"0",

"criticalReReplTimeOutSec":"0",

"replicationtype":"high_throughput",
            "rackpath":"/data",
            "mirrorthrottle":"1",
            "accesstime":"June 18,
2018",
            "readonly":"0",
            "mountdir":"/volt_warm",
            "volumename":"volt_warm",
            "mounted":1,
            "quota":"0",
            "advisoryquota":"0",
            "snapshotcount":0,
            "logicalUsed":"0",

"replicatedlogicalused":"0",
            "used":"0",
            "snapshotused":"0",
            "totalused":"0",
            "replicatedtotalused":"0",
            "scheduleid":"0",
            "schedulename":"",
            "mirrorscheduleid":"0",
            "volumetype":"0",
            "mirrortype":3,
            "creatorcontainerid":2070,

"creatorvolumeuuid":"-8953141547368591
763:5326977893687028655",
            "volumeid":236703387,
            "actualreplication":[
                0,
```

```
                100,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0
            ],
            "nameContainerSizeMB":0,
            "nameContainerId":2070,

"nameContainerDataThresholdMB":524288,
            "needsGfsck":"false",

"maxinodesalarmthreshold":"0",

"maxnssizembalarmthreshold":"0",

"dbrepllagsecalarmthresh":"0",

"dbindexlagsecalarmthresh":"0",
            "limitspread":"true",
            "partlyOutOfTopology":0,
            "wireSecurity":1,
            "auditVolume":0,
            "audited":0,
            "forceAudit":0,
            "coalesceInterval":60,

"enableddataauditoperations":"setattr,
chown,chperm,chgrp,getxattr,listxattr,

setxattr,removexattr,read,write,create
,delete,mkdir,readdir,rmdir,createsym,

lookup,rename,createdev,truncate,table
cfcreate,tablecfdelete,tablecfmodify,

tablecfScan,tableget,tableput,tablesca
n,tablecreate,tableinfo,tablemodify,

getperm,getpathforfid,hardlink,filesca
n,fileoffload,filerecall,

filetierjobstatus,filetierjobabort",

"disableddataauditoperations":"getattr
,filetieroffloadevent,

filetierrecallevent",
            "volumeAces":{
                "readAce":"p",
                "writeAce":"p"
            },
            "fixCreatorId":"false",

"ReplTypeConversionInProgress":0,

"creationTime":1529342213327,
```

```
            "metricsEnabled":0,
            "dareEnabled":0,
            "tierlocal":"0",
            "tierpurged":"0",
            "tierrecall":"0",
            "tierenable":"true",
            "tierid":"136140692",
            "tierruleid":"1",

"tieroffloadscheduleid":"4",
            "tierencryption":"false",

"tierrecallexpirytime":"1",

"tiercompactionscheduleid":"4",

"tiercompactionoverheadthresh":"30",
            "gateway":"Currently
down",
            "ecscheme":"4+2",
            "ecstripedepthmb":"4",

"ecstorevolume":"mapr.internal.ec.volt
_warm.236703387",
            "ectopology":"/data",

"eclabel":"anywhere",
            "ectotalused":0
        }
    ]
}
```

REST

```
curl -k -X GET 'https://
abc.sj.us:8443/rest/volume/info?
name=volt_warm' --user mapr:mapr
{"timestamp":1529546479334,"timeofday"
:"2018-06-20 07:01:19.334 GMT-0700
PM",
"status":"OK","total":1,"data":
[{"acl":{"Principal":"User
mapr","Allowed actions":"[dump,
restore,
m, a, d,
fc]"},"creator":"mapr","aename":"mapr"
,"aetype":"0","numreplicas":"3","minre
plicas":"2",
"nsNumReplicas":"3","nsMinReplicas":"2
","enforceMinReplicationForIO":"false"
,
"containerAllocationFactor":"0","reRep
lTimeOutSec":"0","criticalReReplTimeOu
tSec":"0",
"replicationtype":"high_throughput","r
ackpath":"/data","mirrorthrottle":"1",
"accesstime":"June 18,
2018","readonly":"0","mountdir":"/
volt_warm","volumename":"volt_warm",
"mounted":1,"quota":"0","advisoryquota
":"0","snapshotcount":0,"logicalUsed":
"0",
"replicatedlogicalused":"0","used":"0"
```

```
,"snapshotused":"0","totalused":"0",
"replicatedtotalused":"0","scheduleid"
:"0","schedulename":"","mirrorschedule
id":"0",
"volumetype":"0","mirrortype":3,"creat
orcontainerid":2070,
"creatorvolumeuuid":"-8953141547368591
763:5326977893687028655","volumeid":23
6703387,
"actualreplication":
[0,100,0,0,0,0,0,0,0,0,0],"nameContain
erSizeMB":0,"nameContainerId":2070,
"nameContainerDataThresholdMB":524288,
"needsGfsck":"false","maxinodesalarmth
reshold":"0",
"maxnssizembalarmthreshold":"0","dbrep
llagsecalarmthresh":"0","dbindexlagsec
alarmthresh":"0",
"limitspread":"true","partlyOutOfTopol
ogy":0,"wireSecurity":1,"auditVolume":
0,"audited":0,
"forceAudit":0,"coalesceInterval":60,"
enableddataauditoperations":"setattr,c
hown,chperm,chgrp,
getxattr,listxattr,setxattr,removexatt
r,read,write,create,delete,mkdir,readd
ir,rmdir,createsym,
lookup,rename,createdev,truncate,table
cfcreate,tablecfdelete,tablecfmodify,t
ablecfScan,tableget,
tableput,tablescan,tablecreate,tablein
fo,tablemodify,getperm,getpathforfid,h
ardlink,filescan,
fileoffload,filerecall,filetierjobstat
us,filetierjobabort",
"disableddataauditoperations":"getattr
,filetieroffloadevent,filetierrecallev
ent",
"volumeAces":
{"readAce":"p","writeAce":"p"},"fixCre
atorId":"false",
"ReplTypeConversionInProgress":0,"crea
tionTime":1529342213327,"metricsEnable
d":0,
"dareEnabled":0,"tierlocal":"0","tierp
urged":"0","tierrecall":"0","tierenabl
e":"true",
"tierid":"136140692","tierruleid":"1",
"tieroffloadscheduleid":"4","tierrecal
lexpirytime":"1",
"tiercompactionscheduleid":"4","tierco
mpactionoverheadthresh":"30",
"gateway":"Currently
down","ecscheme":"4+2","ecstripedepthm
b":"4",
"ecstorevolume":"mapr.internal.ec.volt
_warm.236703387","ectopology":"/
data","eclabel":"anywhere","ectotaluse
d":0}]}
```


Return information on volume named `volt_cold` enabled for cold-tier:

last-updated: Apr 11, 2022

CLI

```
maprcli volume info -name
volt_cold -json
{
    "timestamp":1529545651807,
    "timeofday":"2018-06-20
06:47:31.807 GMT-0700 PM",
    "status":"OK",
    "total":1,
    "data":[
        {
            "acl":{
                "Principal":"User
mapr",
                "Allowed
actions":"[dump, restore, m, a, d,
fc]"
            },
            "creator":"mapr",
            "aename":"mapr",
            "aetype":"0",


            "numreplicas":"3",
            "minreplicas":"2",
            "nsNumReplicas":"3",
            "nsMinReplicas":"2",

"enforceMinReplicationForIO":"false",

"containerAllocationFactor":"0",
            "reReplTimeOutSec":"0",

"criticalReReplTimeOutSec":"0",

"replicationtype":"high_throughput",
            "rackpath":"/data",
            "mirrorthrottle":"1",
            "accesstime":"June 18,
2018",
                "readonly":"0",
            "mountdir":"/volt_cold",
            "volumename":"volt_cold",
            "mounted":1,
            "quota":"0",
            "advisoryquota":"0",
            "snapshotcount":0,
            "logicalUsed":"0",

"replicatedlogicalused":"0",
            "used":"0",
            "snapshotused":"0",
            "totalused":"0",
            "replicatedtotalused":"0",
            "scheduleid":"0",
            "schedulename":"",
            "mirrorscheduleid":"0",
            "volumetype":"0",
            "mirrortype":3,
            "creatorcontainerid":2073,

"creatorvolumeuuid":"-8953141547368591
763:8600373021905500606",
```

```
            "volumeid":20110455,
            "actualreplication":[
                0,
                100,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0
            ],
            "nameContainerSizeMB":0,
            "nameContainerId":2073,

"nameContainerDataThresholdMB":524288,
            "needsGfsck":"false",

"maxinodesalarmthreshold":"0",

"maxnssizembalarmthreshold":"0",

"dbrepllagsecalarmthresh":"0",

"dbindexlagsecalarmthresh":"0",
            "limitspread":"true",
            "partlyOutOfTopology":0,
            "wireSecurity":1,
            "auditVolume":0,
            "audited":0,
            "forceAudit":0,
            "coalesceInterval":60,

"enableddataauditoperations":"setattr,
chown,chperm,chgrp,getxattr,

listxattr,setxattr,removexattr,read,wr
ite,create,delete,mkdir,readdir,rmdir,
createsym,

lookup,rename,createdev,truncate,table
cfcreate,tablecfdelete,tablecfmodify,

tablecfScan,tableget,tableput,tablesca
n,tablecreate,tableinfo,tablemodify,

getperm,getpathforfid,hardlink,filesca
n,fileoffload,filerecall,filetierjobst
atus,filetierjobabort",

"disableddataauditoperations":"getattr
,filetieroffloadevent,filetierrecallev
ent",
            "volumeAces":{
                "readAce":"p",
                "writeAce":"p"
            },
            "fixCreatorId":"false",

"ReplTypeConversionInProgress":0,
```

```
"creationTime":1529342278943,
            "metricsEnabled":0,
            "dareEnabled":0,
            "tierlocal":"0",
            "tierpurged":"0",
            "tierrecall":"0",
            "tierenable":"true",
            "tierid":"222693986",
            "tierruleid":"1",

"tieroffloadscheduleid":"0",
            "tierencryption":"false",

"tierrecallexpirytime":"1",

"tiercompactionscheduleid":"4",

"tiercompactionoverheadthresh":"30",
            "gateway":"Currently down"
        }
    ]
}
```

REST

```
curl -k -X GET 'https://
abc.sj.us:8443/rest/volume/info?
name=volt_cold' --user mapr:mapr
{"timestamp":1529546321584,"timeofday"
:"2018-06-20 06:58:41.584 GMT-0700
PM",
"status":"OK","total":1,"data":
[{"acl":{"Principal":"User mapr",
"Allowed actions":"[dump, restore, m,
a, d, fc]"},
"creator":"mapr","aename":"mapr","aety
pe":"0","numreplicas":"3","minreplicas
":"2",
"nsNumReplicas":"3","nsMinReplicas":"2
","enforceMinReplicationForIO":"false"
,
"containerAllocationFactor":"0","reRepl
TimeOutSec":"0",
"criticalReReplTimeOutSec":"0","replic
ationtype":"high_throughput","rackpath
":"/data",
"mirrorthrottle":"1","accesstime":"Jun
e 18,
2018","readonly":"0","mountdir":"/
volt_cold",
"volumename":"volt_cold","mounted":1,"
quota":"0","advisoryquota":"0","snapsh
otcount":0,
"logicalUsed":"0","replicatedlogicalus
ed":"0","used":"0","snapshotused":"0",
"totalused":"0",
"replicatedtotalused":"0","scheduleid"
:"0","schedulename":"","mirrorschedule
id":"0",
"volumetype":"0","mirrortype":3,"creat
orcontainerid":2073,
"creatorvolumeuuid":"-8953141547368591
```

```
763:8600373021905500606","volumeid":20
110455,
"actualreplication":
[0,100,0,0,0,0,0,0,0,0,0],"nameContain
erSizeMB":0,"nameContainerId":2073,
"nameContainerDataThresholdMB":524288,
"needsGfsck":"false","maxinodesalarmth
reshold":"0",
"maxnssizembalarmthreshold":"0","dbrep
llagsecalarmthresh":"0","dbindexlagsec
alarmthresh":"0",
"limitspread":"true","partlyOutOfTopol
ogy":0,"wireSecurity":1,"auditVolume":
0,"audited":0,
"forceAudit":0,"coalesceInterval":60,"
enableddataauditoperations":"setattr,c
hown,chperm,chgrp,
getxattr,listxattr,setxattr,removexatt
r,read,write,create,delete,mkdir,readd
ir,rmdir,createsym,
lookup,rename,createdev,truncate,table
cfcreate,tablecfdelete,tablecfmodify,t
ablecfScan,tableget,
tableput,tablescan,tablecreate,tablein
fo,tablemodify,getperm,getpathforfid,h
ardlink,filescan,
fileoffload,filerecall,filetierjobstat
us,filetierjobabort","disableddataaudi
toperations":"getattr,
filetieroffloadevent,filetierrecalleve
nt","volumeAces":
{"readAce":"p","writeAce":"p"},
"fixCreatorId":"false","ReplTypeConver
sionInProgress":0,"creationTime":15293
42278943,
"metricsEnabled":0,"dareEnabled":0,"ti
erlocal":"0","tierpurged":"0","tierrec
all":"0",
"tierenable":"true","tierid":"22269398
6","tierruleid":"1","tieroffloadschedu
leid":"0",
"tierencryption":"false","tierrecallex
pirytime":"1","tiercompactionschedulei
d":"4",
"tiercompactionoverheadthresh":"30","g
ateway":"Currently down"}]}
```


Related concepts

node on page 1694
Manages nodes in the cluster

Related reference

disk add on page 1602
Adds one or more disks to the specified node. Permissions required: `fc` or `a`.

volume create on page 1931
Creates a volume.

node list on page 1705
Lists nodes in the cluster.

dump volumeinfo on page 1637

Returns information about volumes and the associated containers. For JSON formatted output, use the -json option from the command line.

configure.sh on page 2053
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

volume link create
Creates a link to a volume. Permissions required: `fc` or `m` on the volume.

Syntax

CLI

```
maprcli volume link create
    [ -cluster <clustername> ]
    -path <link path>
    -type <type>
    -volume <volume>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/link/create?<parameters>`

Parameters

Parameter	Description
`path`	The `-path` parameter specifies the link path: `/link` Example: `/home/abc/.rw`
`type`	The volume type: `writeable` or `mirror`.
`volume`	The volume name.
`cluster`	The cluster name.

Examples

Create a link to v1 at the path v1. mirror:

CLI

```
maprcli volume link create -volume
v1 -type mirror -path /v1.mirror
```

REST

```
https://abc.sj.us:8443/rest/
volume/link/create?path=/
v1.mirror&type=mirror&volume=v1
```

volume link remove
Removes the specified symbolic link. Permissions required: `fc` or `m` on the volume.

Syntax

CLI

```
maprcli volume link remove
    -path <path>
    [ -cluster <clustername> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/link/remove?<parameters>`

Parameters

Parameter	Description
`path`	The symbolic link to remove. The path parameter specifies the link path and other information about the symbolic link, using the following syntax: `/link/[maprfs::][volume::]<volume type>::<volume name>` • `link` - the symbolic link path *`maprfs` - a keyword to indicate a special MapR filesystem link • `volume` - a keyword to indicate a link to a volume • `volume type` - `writeable` or `mirror` • `volume name` - the name of the volume Example: `/abc/maprfs::mirror::abc`
`cluster`	The cluster name.

Examples

Remove the link /abc:

CLI

```
maprcli volume link remove -path /abc/
maprfs::mirror::abc
```

REST

```
https://abc.sj.us:8443/rest/
volume/link/remove?path=/abc/
maprfs::mirror::abc
```

volume list
Lists information about volumes specified by name, path, or filter.

See the Fields table on the volume on page 1913 page for the fields available to filter. See the Filters on page 1526 for more information.

last-updated: Apr 11, 2022

Syntax

CLI

```
/opt/mapr/bin/maprcli volume list
    [ -alarmedvolumes 0|1 ]

    [ -cluster <cluster> ]
    [ -columns <columns> ]

    [ -filter <filter> ]
    [ -limit <limit> ]
    [ -nodes <nodes> ]
    [ -output terse | verbose ]

    [ -sortby <attribute> ]
    [ -start <offset> ]
```

Note: For JSON formatted output, use the `-json` option when running `volume list` from the command line.

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/volume/list[?<parameters>]`

Parameters

Parameter	Description
`alarmedvolumes`	Specifies whether (1) or not (0) to list alarmed volumes only. Default: 0
`cluster`	The cluster name on which to run the command.
`columns`	A comma-separated list of fields to return in the query. See the volume on page 1913 table on the volume page. When issuing the `maprcli volume info -columns` and `marcli volume list -columns` commands, the column for the volume name is `volumename`.
`filter`	A filter specifying volumes to list. See Filters on page 1526 for more information. Default: none
`limit`	The number of rows to return, beginning at start. Default: 2147483647
`nodes`	A list of nodes. If specified, `volume list` only lists volumes on the specified nodes.
`output`	Specifies whether the output should be `terse` or `verbose`. Default: verbose
`sortby`	Specifies one of the following attributes to sort the list of volumes by: `volumeowner`, `volumenumreplicas`, `volumeminreplicas`, `volumerackpath`, `volumemountdir`, `volumename`, `volumequota`, `volumeused`, `volumequotaadvisory`, `volumeaename`, `volumeaetype`, `volumeschedule`, `volumetype`, `volumemirrorpercentcomplete`, `volumesnapshotcount`, `volumeid`, `volumenamecontainersize`, `volumelocalpath`, `volumesnapshotused`, `volumetotalused`, `volumelogicalused`, `volumecontainercount`, `volumemirrorschedule`, `volumeaccesstime`, `volumenamespacecontainernumreplicas`, `volumenamespacecontainerminreplicas`, `volumerereplicationtimeoutsec`, `volumecriticalrereplicationtimeoutsec`, `volumecreatetime`, `volumedareenabled`

Parameter	Description
`start`	The offset from the starting row according to sort. Default: 0

Output

Information about the specified volumes.

For standard and mirror volumes (not enabled for tiering), the output looks similar to the following:


```
{
    "timestamp":1435363624712,
    "timeofday":"2015-06-26 05:07:04.712 GMT-0700",
    "status":"OK",
    "total":14,
    "data":[
        {
            "creator":"mapr",
            "aename":"mapr",
            "aetype":0,

            "numreplicas":"3",
            "minreplicas":"2",
            "nsNumReplicas":"3",
            "nsMinReplicas":"2",
            "enforceMinReplicationForIO":"false",
            "containerAllocationFactor":"0",
            "allowGrant":"true",
            "reReplTimeOutSec":"0",
            "criticalReReplTimeOutSec":"0",
            "replicationtype":"high_throughput",
            "rackpath":"/data",
            "mirrorthrottle":"1",
            "accesstime":"June 25, 2018",
            "readonly":"0",
            "mountdir":"",
            "volumename":"sampleVol",
            "mounted":0,
            "quota":"0",
            "advisoryquota":"0",
            "snapshotcount":"0",
            "logicalUsed":"1",
            "replicatedlogicalused":"1",
            "used":"1",
            "snapshotused":"0",
            "totalused":"1",
            "replicatedtotalused":"0",
            "scheduleid":2,
            "schedulename":"Important data",
            "mirrorscheduleid":0,
            "volumetype":0,
            "mirrortype":3,
            "creatorcontainerid":0,
            "creatorvolumeuuid":"",
            "volumeid":172948486,
            "actualreplication":[
                0,
                100,
                0,
                0,
                0,
                0,
                0,
```


last-updated: Apr 11, 2022

```
                0,
                0,
                0,
                0
            ],
            "nameContainerSizeMB":0,
            "nameContainerDataThresholdMB":524288,
            "needsGfsck":false,
            "maxinodesalarmthreshold":"0",
            "maxnssizembalarmthreshold":"0",
            "dbrepllagsecalarmthresh":"0",
            "dbindexlagsecalarmthresh":"0",
            "limitspread":"true",
            "partlyOutOfTopology":0,
            "wireSecurity":0,
            "auditVolume":0,
            "audited":0,
            "forceAudit":0,
            "coalesceInterval":60,

"enableddataauditoperations":"setattr,chown,chperm,chgrp,getxattr,listxattr,
setxattr,removexattr,read,write,create,delete,mkdir,readdir,rmdir,createsym,
lookup,rename,createdev,truncate,tablecfcreate,tablecfdelete,tablecfmodify,t
ablecfScan,tableget,tableput,tablescan,tablecreate,tableinfo,tablemodify,get
perm,getpathforfid,hardlinkfilescan,fileoffload,filerecall,filetierjobstatus
,filetierjobabort",

"disableddataauditoperations":"getattr,filetieroffloadevent,filetierrecallev
ent",
            "mirrorSrcVolume":"",
            "mirrorSrcVolumeId":0,
            "mirrorSrcCluster":"",
            "mirrorDataSrcVolume":"",
            "mirrorDataSrcVolumeId":0,
            "mirrorDataSrcCluster":"",
            "lastSuccessfulMirrorTime":0,
            "mirror-percent-complete":0,
            "mirrorId":0,
            "nextMirrorId":0,
            "mirrorstatus":1,
            "numcontainers":"0",
            "fixCreatorId":"false",
            "ReplTypeConversionInProgress":0,
            "creationTime":1524064440329,
            "metricsEnabled":0,
            "dareEnabled":1,
            "tierenable":"false",
            "SnapshotFailureAlarm":0,
            "MirrorFailureAlarm":0,
            "DataUnderReplicatedAlarm":1435351165700,
            "DataUnavailableAlarm":0,
            "AdvisoryQuotaExceededAlarm":0,
            "QuotaExceededAlarm":0,
            "NoNodesInTopologyAlarm":0,
            "AlmostFullTopologyAlarm":0,
            "FullTopologyAlarm":0,
            "InodesExceededAlarm":0,
            "BecomeMasterStuckAlarm":0,
            "ContainersNonLocalAlarm":0,
            "CannotMirrorAlarm":0,
            "TableIndexLagHighAlarm":0,
            "LargeRowWarning":0,
            "TableIndexEncodingErrorAlarm":0,
            "TableReplicationErrorAlarm":0,
```

```
            "TableReplicationLagHighAlarm":0,
            "TableReplicationAsyncAlarm":0,
            "TableIndexErrorAlarm":0
        }
    ]
}
```


For standard and mirror volumes enabled for warm-tier, the output looks similar to the following:


```
{
    "creator":"mapr",
    "aename":"mapr",
    "aetype":"0",

    "numreplicas":"3",
    "minreplicas":"2",
    "nsNumReplicas":"3",
    "nsMinReplicas":"2",
    "enforceMinReplicationForIO":"false",
    "containerAllocationFactor":"0",
    "allowGrant":"false",
    "reReplTimeOutSec":"0",
    "criticalReReplTimeOutSec":"0",
    "replicationtype":"high_throughput",
    "rackpath":"/data",
    "mirrorthrottle":"1",
    "accesstime":"June 18, 2018",
    "readonly":"0",
    "mountdir":"/volt_warm",
    "volumename":"volt_warm",
    "mounted":1,
    "quota":"0",
    "advisoryquota":"0",
    "snapshotcount":0,
    "logicalUsed":"0",
    "replicatedlogicalused":"0",
    "used":"0",
    "snapshotused":"0",
    "totalused":"0",
    "replicatedtotalused":"0",
    "scheduleid":"0",
    "schedulename":"",
    "mirrorscheduleid":"0",
    "volumetype":"0",
    "mirrortype":3,
    "creatorcontainerid":0,
    "creatorvolumeuuid":"",
    "volumeid":236703387,
    "actualreplication":[
                0,
                100,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0
                 ],
    "nameContainerSizeMB":0,
    "nameContainerDataThresholdMB":524288,
```

```
    "needsGfsck":"false",
    "maxinodesalarmthreshold":"0",
    "maxnssizembalarmthreshold":"0",
    "dbrepllagsecalarmthresh":"0",
    "dbindexlagsecalarmthresh":"0",
    "limitspread":"true",
    "partlyOutOfTopology":0,
    "wireSecurity":1,
    "auditVolume":0,
    "audited":0,
    "forceAudit":0,
    "coalesceInterval":60,

"enableddataauditoperations":"setattr,chown,chperm,chgrp,getxattr,listxattr,
setxattr,removexattr,read,write,create,delete,mkdir,readdir,rmdir,createsym,
lookup,rename,createdev,truncate,tablecfcreate,tablecfdelete,tablecfmodify,t
ablecfScan,tableget,tableput,tablescan,tablecreate,tableinfo,tablemodify,get
perm,getpathforfid,hardlink,filescan,fileoffload,filerecall,filetierjobstatu
s,filetierjobabort",

"disableddataauditoperations":"getattr,filetieroffloadevent,filetierrecallev
ent",
    "mirrorSrcVolume":"",
    "mirrorSrcVolumeId":0,
    "mirrorSrcCluster":"",
    "mirrorDataSrcVolume":"",
    "mirrorDataSrcVolumeId":0,
    "mirrorDataSrcCluster":"",
    "lastSuccessfulMirrorTime":0,
    "mirror-percent-complete":0,
    "mirrorId":0,
    "nextMirrorId":0,
    "mirrorstatus":1,
    "numcontainers":"1",
    "fixCreatorId":"false",
    "ReplTypeConversionInProgress":0,
    "creationTime":1529342213327,
    "metricsEnabled":0,
    "dareEnabled":0,
    "tierlocal":"0",
    "tierpurged":"0",
    "tierrecall":"0",
    "tierenable":"true",
    "tierid":"136140692",
    "tierruleid":"1",
    "tieroffloadscheduleid":"4",
    "tierrecallexpirytime":"0",
    "tiercompactionscheduleid":"0",
    "tiercompactionoverheadthresh":"None",
    "tierjobtype":"offload",
    "tierjobstate":"FailureFatal",
    "tierjobstarttime":"2018-06-20 10:18:10.285 GMT-0700",
    "tierjobendtime":"2018-06-20 10:18:18.805 GMT-0700",
    "tierjobprogress":"0",
    "tierjobtotaloffloadsize":"0",
    "tierjoboffloadavgthroughputmbps":"0",
    "tierjobrecallavgthroughputmbps":"0",
    "gateway":"Currently down",
    "tiername":"autoec.volt_warm.1529342212",
    "tiertype":"ectier",
    "ecscheme":"4+2",
    "ecstripedepthmb":"4",
    "ecstorevolume":"mapr.internal.ec.volt_warm.236703387",
    "ectopology":"/data",
```


last-updated: Apr 11, 2022

```
    "ectotalused":0,
    "SnapshotFailureAlarm":0,
    "MirrorFailureAlarm":0,
    "DataUnderReplicatedAlarm":1529384390911,
    "DataUnavailableAlarm":0,
    "AdvisoryQuotaExceededAlarm":0,
    "QuotaExceededAlarm":0,
    "NoNodesInTopologyAlarm":0,
    "AlmostFullTopologyAlarm":0,
    "FullTopologyAlarm":0,
    "InodesExceededAlarm":0,
    "BecomeMasterStuckAlarm":0,
    "ContainersNonLocalAlarm":0,
    "CannotMirrorAlarm":0,
    "TableIndexLagHighAlarm":0,
    "LargeRowWarning":0,
    "TableIndexEncodingErrorAlarm":0,
    "TableReplicationErrorAlarm":0,
    "TableReplicationLagHighAlarm":0,
    "TableReplicationAsyncAlarm":0,
    "TableIndexErrorAlarm":0
}
```


For standard and mirror volumes enabled for cold-tier, the output looks similar to the following:


```
{
       "creator":"mapr",
    "aename":"mapr",
    "aetype":"0",

    "numreplicas":"3",
    "minreplicas":"2",
    "nsNumReplicas":"3",
    "nsMinReplicas":"2",
    "enforceMinReplicationForIO":"false",
    "containerAllocationFactor":"0",
    "allowGrant":"false",
    "reReplTimeOutSec":"0",
    "criticalReReplTimeOutSec":"0",
    "replicationtype":"high_throughput",
    "rackpath":"/data",
    "mirrorthrottle":"1",
    "accesstime":"June 18, 2018",
    "readonly":"0",
    "mountdir":"/volt_only",
    "volumename":"volt_only",
    "mounted":1,
    "quota":"0",
    "advisoryquota":"0",
    "snapshotcount":0,
    "logicalUsed":"0",
    "replicatedlogicalused":"0",
    "used":"0",
    "snapshotused":"0",
    "totalused":"0",
    "replicatedtotalused":"0",
    "scheduleid":"0",
    "schedulename":"",
    "mirrorscheduleid":"0",
    "volumetype":"0",
    "mirrortype":3,
    "creatorcontainerid":0,
    "creatorvolumeuuid":"",
```

```
    "volumeid":162353415,
    "actualreplication":[
                0,
                100,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0
               ],
    "nameContainerSizeMB":0,
    "nameContainerDataThresholdMB":524288,
    "needsGfsck":"false",
    "maxinodesalarmthreshold":"0",
    "maxnssizembalarmthreshold":"0",
    "dbrepllagsecalarmthresh":"0",
    "dbindexlagsecalarmthresh":"0",
    "limitspread":"true",
    "partlyOutOfTopology":0,
    "wireSecurity":1,
    "auditVolume":0,
    "audited":0,
    "forceAudit":0,
    "coalesceInterval":60,

"enableddataauditoperations":"setattr,chown,chperm,chgrp,getxattr,listxattr,
setxattr,removexattr,read,write,create,delete,mkdir,readdir,rmdir,createsym,
lookup,rename,createdev,truncate,tablecfcreate,tablecfdelete,tablecfmodify,t
ablecfScan,tableget,tableput,tablescan,tablecreate,tableinfo,tablemodify,get
perm,getpathforfid,hardlink,filescan,fileoffload,filerecall,filetierjobstatu
s,filetierjobabort",

"disableddataauditoperations":"getattr,filetieroffloadevent,filetierrecallev
ent",
    "mirrorSrcVolume":"",
    "mirrorSrcVolumeId":0,
    "mirrorSrcCluster":"",
    "mirrorDataSrcVolume":"",
    "mirrorDataSrcVolumeId":0,
    "mirrorDataSrcCluster":"",
    "lastSuccessfulMirrorTime":0,
    "mirror-percent-complete":0,
    "mirrorId":0,
    "nextMirrorId":0,
    "mirrorstatus":1,
    "numcontainers":"1",
    "fixCreatorId":"false",
    "ReplTypeConversionInProgress":0,
    "creationTime":1529342295570,
    "metricsEnabled":0,
    "dareEnabled":0,
    "tierlocal":"0",
    "tierpurged":"0",
    "tierrecall":"0",
    "tierenable":"true",
    "tieroffloadscheduleid":"0",
    "tierrecallexpirytime":"0",
    "tiercompactionscheduleid":"0",
    "tiercompactionoverheadthresh":"None",
    "gateway":"Currently down",
```


last-updated: Apr 11, 2022

```
    "SnapshotFailureAlarm":0,
    "MirrorFailureAlarm":0,
    "DataUnderReplicatedAlarm":1529384390923,
    "DataUnavailableAlarm":0,
    "AdvisoryQuotaExceededAlarm":0,
    "QuotaExceededAlarm":0,
    "NoNodesInTopologyAlarm":0,
    "AlmostFullTopologyAlarm":0,
    "FullTopologyAlarm":0,
    "InodesExceededAlarm":0,
    "BecomeMasterStuckAlarm":0,
    "ContainersNonLocalAlarm":0,
    "CannotMirrorAlarm":0,
    "TableIndexLagHighAlarm":0,
    "LargeRowWarning":0,
    "TableIndexEncodingErrorAlarm":0,
    "TableReplicationErrorAlarm":0,
    "TableReplicationLagHighAlarm":0,
    "TableReplicationAsyncAlarm":0,
    "TableIndexErrorAlarm":0
}
```


Fields

For definitions of the output fields, and short names for use with filters, see the Fields table on the `volume` command page.

Examples

List all volumes

CLI

```
/opt/mapr/bin/maprcli volume
list -json
```

REST

```
https://10.10.82.23:8443/rest/volume/
list
```

List the first ten volumes

The `start` and `limit` parameters are useful for windowing the results. You can list the first ten volumes, then the next ten, and so on.

CLI

```
/opt/mapr/bin/maprcli volume
list -start 0 -limit 10 -json
```

REST

```
curl -k -X
GET 'https://abc.sj.us:8443/rest/
node/list?start=0&limit=10' --user
mapr:mapr
```

Filter by aename

CLI

```
maprcli volume list -filter
'[aename==mapr]' -columns volumeid
volumeid
```


last-updated: Apr 11, 2022

```
243256560
139026416
192452723
1
237238261
185847104
83335307
97256251
248672744
206179696
59269298
23746740
155195506
204064014
243050615
175781739
109532950
86259431
161806152
111826621
161383512
42835142
91977453
40523868
246476194
26484100
157091944
184799162
141342643
29373265
153950841
193510550
17691624
```

REST

```
curl -k -X GET 'https://
abc.sj.us:8443/rest/volume/list?
filter=%5Baename%3D%3Dmapr%5D&columns=
volumeid' --user mapr:mapr
{"timestamp":1528315774026,"timeofday"
:"2018-06-06 01:09:34.026 GMT-0700
PM","status":"OK","total":33,"data":
[{"volumeid":243256560},
{"volumeid":139026416},
{"volumeid":192452723},{"volumeid":1},
{"volumeid":237238261},
{"volumeid":185847104},
{"volumeid":83335307},
{"volumeid":97256251},
{"volumeid":248672744},
{"volumeid":206179696},
{"volumeid":59269298},
{"volumeid":23746740},
{"volumeid":155195506},
{"volumeid":204064014},
{"volumeid":243050615},
{"volumeid":175781739},
{"volumeid":109532950},
{"volumeid":86259431},
{"volumeid":161806152},
{"volumeid":111826621},
{"volumeid":161383512},
```

```
{"volumeid":42835142},
{"volumeid":91977453},
{"volumeid":40523868},
{"volumeid":246476194},
{"volumeid":26484100},
{"volumeid":157091944},
{"volumeid":184799162},
{"volumeid":141342643},
{"volumeid":29373265},
{"volumeid":153950841},
{"volumeid":193510550},
{"volumeid":17691624}]}
```

Filter by tiertype

CLI

```
/opt/mapr/bin/maprcli
volume list -filter
'[tiertype==ectier]' -columns
volumename
volumename
egWarmVol
sampleECmirrorVol
sampleECvol
sampleVol
sampleVol3
sampleVol3Mirror
warmTierMirroVol
```

REST

```
curl -k -X GET 'https://
abc.sj.us:8443/rest/volume/list?
filter=%5Btiertype%3D%3Dectier%5D&colu
mns=volumename' --user mapr:mapr
{"timestamp":1528315936495,"timeofday"
:"2018-06-06 01:12:16.495 GMT-0700
PM","status":"OK","total":7,"data":
[{"volumename":"egWarmVol"},
{"volumename":"sampleECmirrorVol"},
{"volumename":"sampleECvol"},
{"volumename":"sampleVol"},
{"volumename":"sampleVol3"},
{"volumename":"sampleVol3Mirror"},
{"volumename":"warmTierMirroVol"}]}
```

volume mirror push

Pushes the changes in a volume to all of its mirror volumes in the same cluster, and waits for each mirroring operation to complete.

Use this command when you need to push recent changes.

Syntax

CLI

```
maprcli volume mirror push
    [ -cluster <cluster> ]
    -name <volume name>
    [ -verbose true|false ]
```

REST

None.

last-updated: Apr 11, 2022

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`name`	The volume to push.
`verbose`	Specifies whether the command output should be verbose. Default: true

Output

Sample Output

```
Starting mirroring of volume mirror1
Mirroring complete for volume mirror1
Successfully completed mirror push to all local mirrors of volume volume1
```

Examples

Push changes from the volume "volume1" to its local mirror volumes:

CLI

```
maprcli volume mirror push -name
volume1 -cluster mycluster
```

volume mirror start
Starts mirroring on the specified volume from its source volume.

- License required: Enterprise Edition
- Permissions required: `fc` or `restore` on the volume

When a mirror is started, the mirror volume is synchronized from a hidden internal snapshot so that the mirroring process is not affected by any concurrent changes to the source volume. The `volume mirror start` command does not wait for mirror completion, but returns immediately. The changes to the mirror volume occur atomically at the end of the mirroring process; deltas transmitted from the source volume do not appear until mirroring is complete.

To provide rollback capability for the mirror volume, the mirroring process creates a snapshot of the mirror volume before starting the mirror, with the following naming format: `<volume>.mirrorsnap.<date>.<time>`.

Normally, the mirroring operation transfers only deltas from the last successful mirror. Under certain conditions (mirroring a volume repaired by `fsck`, for example), the source and mirror volumes can become out of sync. In such cases, it is impossible to transfer deltas, because the state is not the same for both volumes. Use the `-full` option to force the mirroring operation to transfer all data to bring the volumes back in sync.

Syntax

CLI

```
maprcli volume mirror start
    [ -cluster <cluster> ]
    [ -full true|false ]
    -name <volume name>
```

REST

Request Type	POST

last-updated: Apr 11, 2022

Request URL	`http[s]://<host>:<port>/rest/volume/mirror/start?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`full`	Specifies whether to perform a full copy of all data. If false, only the deltas are copied.
`name`	The volume for which to start the mirror.

Output

Sample Output

```
messages
Started mirror operation for volumes 'testMirror'
```

Examples

Start mirroring the mirror volume "testMirror":

CLI

```
maprcli volume mirror start -name
testMirror
```

REST

```
https://abc.sj.us:8443/rest/volume/
mirror/start?name=testMirror
```

volume mirror status
Displays the status of the mirroring operation in progress. Use this command to examine the progress of mirroring.

- License required: Enterprise Edition
- Permissions required: `fc` or `restore` on the volume

The `status` command displays the statstics of the mirroring operation including the total number of container IDs to resync, the current status of the mirroring, the number of indoes in use for the mirror, the container ID information for the source and destination volumes, and the error code if any, to name a few.

Syntax

CLI

```
maprcli volume mirror status
    [ -cluster cluster_name ]
    -name name
    [ -start start. default: 1 ]
    [ -limit limit. default:
2147483647 ]
    [ -verbose <true/false> if true,
```

```
will displayed detailed container
information. default: true ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/mirror/status?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`name`	The volume for which to determine the mirror status.
`start`	The container to start with.
`limit`	The number of containers for which to display status.
`verbose`	Whether to display a terse output or display full statistics.

Examples

Check the status of the mirroring on mirror volume "testvol":

CLI

```
maprcli volume mirror status -name
testvol -json

**** Displays mirroring statistics.
****
***  Here, it is resyncing
destination containers ****
   {
        "timestamp":1622443868513,
        "timeofday":"2021-05-31
06:51:08.513 GMT+0000 AM",
        "status":"OK",
        "total":1,
        "data":[
                {

"SourceVolumeName":"test",

"SourceClusterName":"my.cluster.com",

"MirroringStarted":"2021-05-31
06:51:02.532 GMT+0000",

"MirrorState":"ResyncDestinationContai
ners",

"TotalResyncInProgressCids":6,

"ResyncInProgressCids":[
                                {
```

```
  "ErrorCode":0,
  "Progress":0,
  "ResyncStartedTime":"2021-05-31
06:51:06.985 GMT+0000",
  "DestinationCid":{
          "ContainerId":2147,
          "Epoch":3,
"Master":"10.163.167.214:5660--3-VALID
",
          "ActiveServers":{
"IP":"10.163.167.214:5660--3-VALID"
          },
          "InactiveServers":{
          },
          "UnusedServers":{
          },
          "OwnedSizeMB":"0 MB",
          "SharedSizeMB":"0 MB",
          "LogicalSizeMB":"0 MB",
          "TotalSizeMB":"0 MB",
          "NumInodesInUse":256,
          "Mtime":"May 31, 2021",
          "NameContainer":"false",
          "CreatorContainerId":2138,
"CreatorVolumeUuid":"-8872774736600751
871:7950895803961029577",
          "UseActualCreatorId":false
  },
  "SourceSnapCid":{
          "ContainerId":256000069,
```

```
          "Epoch":3,

"Master":"10.163.167.214:5660--3-VALID
",
          "ActiveServers":{

"IP":"10.163.167.214:5660--3-VALID"
          },
          "InactiveServers":{

          },
          "UnusedServers":{

          },
          "OwnedSizeMB":"0 MB",
          "SharedSizeMB":"0 MB",
          "LogicalSizeMB":"0 MB",
          "TotalSizeMB":"0 MB",
          "NameContainer":"false",
          "RW ContainerId":2138,
          "RW VolumeId":217081367,
          "CreatorContainerId":2138,

"CreatorVolumeUuid":"-8872774736600751
871:7950895803961029577",
          "UseActualCreatorId":false
  }
                                  },
                                  {
  "ErrorCode":0,
  "Progress":0,
  "ResyncStartedTime":"2021-05-31
06:51:06.985 GMT+0000",
  "DestinationCid":{
          "ContainerId":2144,
          "Epoch":3,
```

```
"Master":"10.163.167.214:5660--3-VALID
",
          "ActiveServers":{

"IP":"10.163.167.214:5660--3-VALID"
          },
          "InactiveServers":{

          },
          "UnusedServers":{

          },
          "OwnedSizeMB":"0 MB",
          "SharedSizeMB":"32 MB",
          "LogicalSizeMB":"32 MB",
          "TotalSizeMB":"32 MB",
          "NumInodesInUse":33,
          "Mtime":"May 31, 2021",
          "NameContainer":"false",
          "CreatorContainerId":2142,

"CreatorVolumeUuid":"-8872774736600751
871:7950895803961029577",
          "UseActualCreatorId":false
  },
  "SourceSnapCid":{
          "ContainerId":256000073,
          "Epoch":3,

"Master":"10.163.167.214:5660--3-VALID
",
          "ActiveServers":{

"IP":"10.163.167.214:5660--3-VALID"
          },
```

```
          "InactiveServers":{

          },
          "UnusedServers":{

          },
          "OwnedSizeMB":"0 MB",
          "SharedSizeMB":"0 MB",
          "LogicalSizeMB":"0 MB",
          "TotalSizeMB":"0 MB",
          "NameContainer":"false",
          "RW ContainerId":2142,
          "RW VolumeId":217081367,
          "CreatorContainerId":2142,
"CreatorVolumeUuid":"-8872774736600751
871:7950895803961029577",
          "UseActualCreatorId":false
  }
                                  },
                                  {
  "ErrorCode":0,
  "Progress":0,
  "ResyncStartedTime":"2021-05-31
06:51:06.985 GMT+0000",
  "DestinationCid":{
          "ContainerId":2137,
          "Epoch":3,
"Master":"10.163.167.214:5660--3-VALID
",
          "ActiveServers":{
"IP":"10.163.167.214:5660--3-VALID"
          },
          "InactiveServers":{
```

```
          },
          "UnusedServers":{

          },
          "OwnedSizeMB":"0 MB",
          "SharedSizeMB":"0 MB",
          "LogicalSizeMB":"0 MB",
          "TotalSizeMB":"0 MB",
          "NumInodesInUse":34,
          "Mtime":"May 31, 2021",
          "NameContainer":"true",
          "CreatorContainerId":2136,
"CreatorVolumeUuid":"-8872774736600751
871:7950895803961029577",
          "UseActualCreatorId":false
  },
  "SourceSnapCid":{
          "ContainerId":256000068,
          "Epoch":3,
"Master":"10.163.167.214:5660--3-VALID
",
          "ActiveServers":{
"IP":"10.163.167.214:5660--3-VALID"
          },
          "InactiveServers":{

          },
          "UnusedServers":{

          },
          "OwnedSizeMB":"0 MB",
          "SharedSizeMB":"0 MB",
```

```
          "LogicalSizeMB":"0 MB",
          "TotalSizeMB":"0 MB",
          "NameContainer":"true",
          "RW ContainerId":2136,
          "RW VolumeId":217081367,
          "CreatorContainerId":2136,
"CreatorVolumeUuid":"-8872774736600751
871:7950895803961029577",
          "UseActualCreatorId":false
  }
                                  },
                                  {
  "ErrorCode":0,
  "Progress":0,
  "ResyncStartedTime":"2021-05-31
06:51:06.985 GMT+0000",
  "DestinationCid":{
          "ContainerId":2145,
          "Epoch":3,
"Master":"10.163.167.214:5660--3-VALID
",
          "ActiveServers":{
"IP":"10.163.167.214:5660--3-VALID"
          },
          "InactiveServers":{
          },
          "UnusedServers":{
          },
          "OwnedSizeMB":"0 MB",
          "SharedSizeMB":"0 MB",
          "LogicalSizeMB":"0 MB",
```

```
          "TotalSizeMB":"0 MB",
          "NumInodesInUse":256,
          "Mtime":"May 31, 2021",
          "NameContainer":"false",
          "CreatorContainerId":2140,
"CreatorVolumeUuid":"-8872774736600751
871:7950895803961029577",
          "UseActualCreatorId":false
  },
  "SourceSnapCid":{
          "ContainerId":256000071,
          "Epoch":3,
"Master":"10.163.167.214:5660--3-VALID
",
          "ActiveServers":{
"IP":"10.163.167.214:5660--3-VALID"
          },
          "InactiveServers":{
          },
          "UnusedServers":{
          },
          "OwnedSizeMB":"0 MB",
          "SharedSizeMB":"0 MB",
          "LogicalSizeMB":"0 MB",
          "TotalSizeMB":"0 MB",
          "NameContainer":"false",
          "RW ContainerId":2140,
          "RW VolumeId":217081367,
          "CreatorContainerId":2140,
```

```
"CreatorVolumeUuid":"-8872774736600751
871:7950895803961029577",

          "UseActualCreatorId":false

  }
                                  },
                                  {

  "ErrorCode":0,

  "Progress":0,

  "ResyncStartedTime":"2021-05-31
06:51:06.985 GMT+0000",

  "DestinationCid":{

          "ContainerId":2143,

          "Epoch":3,


"Master":"10.163.167.214:5660--3-VALID
",

          "ActiveServers":{


"IP":"10.163.167.214:5660--3-VALID"

          },

          "InactiveServers":{


          },

          "UnusedServers":{


          },

          "OwnedSizeMB":"0 MB",

          "SharedSizeMB":"0 MB",

          "LogicalSizeMB":"0 MB",

          "TotalSizeMB":"32 MB",

          "NumInodesInUse":32,

          "Mtime":"May 31, 2021",

          "NameContainer":"false",

          "CreatorContainerId":2141,


"CreatorVolumeUuid":"-8872774736600751
```

```
871:7950895803961029577",
          "UseActualCreatorId":false
  },
  "SourceSnapCid":{
          "ContainerId":256000072,
          "Epoch":3,
"Master":"10.163.167.214:5660--3-VALID
",
          "ActiveServers":{
"IP":"10.163.167.214:5660--3-VALID"
          },
          "InactiveServers":{
          },
          "UnusedServers":{
          },
          "OwnedSizeMB":"0 MB",
          "SharedSizeMB":"0 MB",
          "LogicalSizeMB":"0 MB",
          "TotalSizeMB":"0 MB",
          "NameContainer":"false",
          "RW ContainerId":2141,
          "RW VolumeId":217081367,
          "CreatorContainerId":2141,
"CreatorVolumeUuid":"-8872774736600751
871:7950895803961029577",
          "UseActualCreatorId":false
  }
                                },
                                {
  "ErrorCode":0,
  "Progress":0,
```

```
  "ResyncStartedTime":"2021-05-31
06:51:06.985 GMT+0000",
  "DestinationCid":{
          "ContainerId":2146,
          "Epoch":3,
"Master":"10.163.167.214:5660--3-VALID
",
          "ActiveServers":{
"IP":"10.163.167.214:5660--3-VALID"
          },
          "InactiveServers":{
          },
          "UnusedServers":{
          },
          "OwnedSizeMB":"0 MB",
          "SharedSizeMB":"0 MB",
          "LogicalSizeMB":"0 MB",
          "TotalSizeMB":"0 MB",
          "NumInodesInUse":256,
          "Mtime":"May 31, 2021",
          "NameContainer":"false",
          "CreatorContainerId":2139,
"CreatorVolumeUuid":"-8872774736600751
871:7950895803961029577",
          "UseActualCreatorId":false
  },
  "SourceSnapCid":{
          "ContainerId":256000070,
          "Epoch":3,
```

```
"Master":"10.163.167.214:5660--3-VALID
",
          "ActiveServers":{

"IP":"10.163.167.214:5660--3-VALID"
          },
          "InactiveServers":{

          },
          "UnusedServers":{

          },
          "OwnedSizeMB":"0 MB",
          "SharedSizeMB":"0 MB",
          "LogicalSizeMB":"0 MB",
          "TotalSizeMB":"0 MB",
          "NameContainer":"false",
          "RW ContainerId":2139,
          "RW VolumeId":217081367,
          "CreatorContainerId":2139,

"CreatorVolumeUuid":"-8872774736600751
871:7950895803961029577",
          "UseActualCreatorId":false
  }
                                  }
                          ]
                  }
        ]
   }
```


maprcli volume mirror status -name testvol -json


```
*** Now the resync is done and the
source snapshot is deleted. ***
   {
        "timestamp":1622443878136,
        "timeofday":"2021-05-31
06:51:18.136 GMT+0000 AM",
        "status":"OK",
        "total":1,
        "data":[
```

```
                {

"SourceVolumeName":"test",

"SourceClusterName":"my.cluster.com",

"MirroringStarted":"2021-05-31
06:51:02.532 GMT+0000",

"MirrorState":"DeleteSourceSnapshot"
                }
        ]
}
```

maprcli volume mirror status -name testvol -json

```
*** Mirroring is now complete ***
   {
        "timestamp":1622443883402,
        "timeofday":"2021-05-31
06:51:23.402 GMT+0000 AM",
        "status":"ERROR",
        "errors":[
                {
                        "id":0,
                        "desc":"No
mirror jobs are in progress for
volume testvol"
                }
        ]
   }
```

REST

```
https://abc.sj.us:8443/rest/volume/
mirror/status?name=testvol
```

volume mirror stop
Stops mirroring on the specified volume.

- License required: Enterprise Edition
- Permissions required: `fc` or `restore` on the volume

The `volume mirror stop` command lets you stop mirroring (for example, during a network outage). You can use the `volume mirror start` command to resume mirroring.

Syntax

CLI

```
maprcli volume mirror stop
    [ -cluster <cluster> ]
    -name <volume name>
```

REST

Request Type	POST

Request URL	`http[s]://<host>:<port>/rest/volume/mirror/stop?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`name`	The volume for which to stop the mirror.

Output

Sample Output

```
messages
Stopped mirror operation for volumes 'testMirror'
```

Examples

Stop mirroring the mirror volume "testMirror":

CLI

```
maprcli volume mirror stop -name
testMirror
```

REST

```
https://abc.sj.us:8443/rest/volume/
mirror/stop?name=testMirror
```

volume modify

Modifies an existing volume. Permissions required: `m` or `fc` on the volume.

An error occurs if the name or path refers to a non-existent volume, or cannot be resolved.

Syntax

CLI

```
/opt/mapr/bin/maprcli volume modify
    [ -cluster <cluster name> ]
    -name <volume name>
    [ -advisoryquota <advisory
quota> ]
    [ -ae <accounting entity> ]
    [ -aetype <aetype> ]
    [ -allowgrant true|false ]
    [ -allowreadforexecute Enable
reads for files with execute
permission. <true|false> ]

    [ -auditenabled true|false ]
    [ -autooffloadthresholdgb
<offload size threshold> ]
    [ -coalesce <interval in mins> ]
    [ -compactionoverheadthreshold
<compaction_overhead> ]
```

```
    [ -compactionschedule
<compaction_schedule_ID> ]
    [ -containerallocationfactor
<positive integer> ]

[ -criticalrereplicationtimeoutsec ]
    [ -dataauditops <+|- operations> ]
    [ -dbindexlagsecalarmthresh
<threshold> ]
    [ -dbrepllagsecalarmthresh
<threshold> ]
    [ -disableddataauditops
<operations> ]
    [ -ecenable true|false ]
    [ -ecscheme <ec_scheme> ]
    [ -ectopology <path> ]

    [ -enforceminreplicationforio
true|false ]

    [ -forceauditenable true|false ]
    [ -group <list of
group:allowMask> ]
    [ -maxinodesalarmthreshold
<threshold> ]
    [ -maxnssizembalarmthreshold
<threshold> ]
    [ -metricsenabled true|false ]
    [ -minreplication <minimum
replication> ]
    [ -mirrorschedule <mirror
schedule ID> ]
    [ -mirrorthrottle true|false ]
    [ -namecontainerdatathreshold
<size>  ] (available from version
6.0.1)
    [ -nsminreplication <minimum
replication factor> ]
    [ -nsreplication <replication
factor> ]

    [ -offloadschedule <schedule ID> ]
    [ -quota <quota> ]
    [ -readAce <Access Control
Expression> ]
    [ -readonly <readonly> ]
    [ -recallexpirytime <expiry
time> ]
    [ -replication <replication> ]
    [ -rereplicationtimeoutsec
<timeout in seconds> ]
    [ -schedule <schedule ID> ]


    [ -source <source volume> ]
    [ -tierencryption true|
false ]
    [ -tieringrule <rule name> ]
    [ -tierkey <tier encryption key> ]
    [ -tiername <tier name> ]
    [ -type rw|mirror ]
    [ -user <list of user:allowMask> ]
```

```
    [ -wiresecurityenabled true|
false ]
    [ -writeAce <Access Control
Expression> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/modify?<parameters>`

Parameters

Parameter: `advisoryquota`

Possible Values: 0 or any other integer value.

Description: The advisory quota for the volume as `integer` plus `unit. Example: quota=500G; Units: B`, `K`, `M`, `G`, `T`, `P`

Parameter: `ae`

Possible Values: Name of the entity that owns the volume.

Description: The accounting entity that owns the volume.

Parameter: `aetype`

Possible Values:

- `0=user`
- `1=group`

Description: Type of accounting entity.

Parameter: `allowgrant`

Possible Values:

- `truetrue`
- `false`

Description: Specifies whether the volume as a parent, grants permission for a child volume to inherit its properties.

Parameter: `allowreadforexecute`

Possible Values:

- `true`
- `false`

Description: Allows execution of SUID binaries with only their executable bit set, on a FUSE filesystem. This parameter works in conjunction with the `fuse.mount.setuid` FUSE option. For more information, see Configuring the MapR FUSE-Based POSIX Client on page 1240.

Parameter: `auditenabled`

Possible Values:

- `true`
- `false`

Description: Specifies whether to turn on auditing for the volume. If you enable auditing at the cluster level

	with the audit data on page 1553 command, setting this value to `true` causes auditing to start for any directories, files, tables, or streams that are already enabled for auditing. If none are yet enabled, enabling auditing on any of them causes auditing of them to start. Set `auditenabled` to `true` to enable auditing on directories, files, tables, and streams in the volume. You must have the `fc` permission on the cluster to use this parameter. See `acl` for details about this permission.
Parameter: `autooffloadthresholdgb`	Possible Values: Any positive integer. Description: The size of the volume in GB (threshold). When this threshold is reached or exceeded, volume data is automatically offloaded by the Automatic Tiering Scheduler. To use the global size threshold (of 1024 GB), set the value to `0`.
Parameter: `cluster`	Possible Values: Any valid cluster. Description: The cluster on which to run the command.
Parameter: ***`coalesce`***	Possible Values: Set this parameter to a large number of minutes to prevent audit logs from growing quickly. Description: The interval of time (in minutes) during which READ, WRITE, or GETATTR operations on one file from one IP address or UID are logged only once for a particular operation, if auditing is enabled. For example, suppose that a client application reads a single file three times in 6 minutes, so that there is one read at 0 minutes, another at 3 minutes, and a final read at 6 minutes. If the coalesce interval is at least 6 minutes, then only the first read operation is logged. However, if the interval is 4 minutes, then only the first and third read operations are logged. If the interval is 2 minutes, all three read operations are logged. Now however, if the client was also writing to the file, irrespective of the coalesce interval for the read operation in the example stated previously, the write operation is logged, as it is a different operation from reading.
Parameter: `compactionoverheadthreshold`	Possible Values: 0-100% Description: Specifies the percentage of offloaded data that must have been deleted on the MapR cluster to qualify the data for compaction (or deletion from the tier).
Parameter: `compactionschedule`	Possible Values: Any valid schedule ID. Set this parameter to `0` to disable the compactor. Description: Specifies the schedule to use for running the compactor. By default, the compactor runs on an automatic internal schedule.
Parameter: `containerallocationfactor`	Recommended value: 2* SP count in the volume topology. Description: Specifies the number of containers to create when the first write from a remote client is sent to the volume. The pre-created containers are distributed equally across topologies, servers, MapR File System instances, and storage pools. CLDB also

takes into consideration the load (IO/Space) when selecting target storage pools for containers. The value must be a positive integer.

Parameter: `criticalrereplicationtimeoutsec`

Possible Values: Any integer between 300 and 3600 (seconds)

Description: Timeout (in seconds) before re-replicating only the critically under-replicated containers . If you set both `rereplicationtimeoutsec` and `criticalrereplicationtimeoutsec`, and if the value of:

- `rereplicationtimeoutsec` is less than `criticalrereplicationtimeoutsec`, `rereplicationtimeoutsec` overrides the `criticalrereplicationtimeoutsec` setting for both under-replicated and critically under-replicated containers.
- `rereplicationtimeoutsec` is greater than `criticalrereplicationtimeoutsec`, `criticalrereplicationtimeoutsec` overrides the `rereplicationtimeoutsec` setting only for critically under-replicated containers; `rereplicationtimeoutsec` setting is still applicable for under-replicated containers.

Parameter: `dataauditops`

Possible Values: Any audit operations that you want to enable.

Description: The comma separated list of filesystem operations to include (specified with a preceding plus sign (+)) or exclude (specified with a preceding minus sign (-)) from auditing.

To exclude the first operation in the list (of operations) from auditing, precede it by two minus (`--`) signs. To exclude subsequent operations, precede them by only a single minus (`-`) sign, irrespective of whether the first operation was included (using a plus (+) sign) or excluded (using two minus (`--`) signs). If neither sign is specified, the given operation is included for auditing.

The operations that can be included (+) or excluded (`-`) from auditing are listed here. You can, alternatively, group all the operations using the keyword `all`, which:

- If included (+), cannot be specified with a list of other included operations.
- If excluded (`-`), cannot be specified with a list of other excluded operations.

You can specify a mixed list of included and excluded operations. There is no change to operations that are not specified with the command.

> **Tip:** For more information, see Selective Auditing of Filesystem and Table Operations.

Parameter: `disableddataauditops`

Possible Values: Any audit operations that you want to disable.

Description: The comma-separated list of disabled filesystem audit operations to set. This parameter is an alternate way of setting audit operations as compared

last-updated: Apr 11, 2022

to the `dataauditops` option. Plus (+) or minus signs (-) are not allowed for this option. Any audit operation that is specified with this option replaces any existing disabled audit operations configured for this security policy, while any audit operations that are not specified, are enabled.

Merging of the specified audit operations with existing audit operations is not performed, as with the `dataauditops` option.

Parameter: `dbindexlagsecalarmthresh`

Possible Values: Any integer value.

Description: Specifies the threshold (in seconds) to raise an alarm for index update lag.

Parameter: `dbrepllagsecalarmthresh`

Possible Values: Any integer value.

Description: Specifies the threshold (in seconds) to raise an alarm for DB replication lag.

Parameter: `ecenable`

Possible Values:

- `true`
- `false`

Description: Enable (`true`) warm tiering for the volume only if it is already not enabled. When specified, you cannot specify `tiername` to use an existing warm-tier; when the command runs, a new tier and rule are automatically created for the volume.

`ecenable` works only if you have set `tieringenable` to `true` at the time of volume creation.

When modifying volumes, `ecenable` does not automatically set `tieringenable` to `true` as in the case of volume creation.

Setting this parameter to `false` is the same as not specifying this parameter, and does nothing.

Parameter: `ecscheme`

Possible Values: Any valid EC scheme.

Description: The number of data chunks and the number of parity chunks separated by a plus (+) sign. The default scheme is `4+2`. For information on the supported schemes, see Erasure Coding Scheme for Data Protection and Recovery on page 926.

Note: This parameter is applicable only for EC volumes, and only when you set the `ecenable` parameter to `true`.

Parameter: `ectopology`

Possible Values: Any topology that exists in your environment.

Description: Sets the topology of the erasure coded volume if it is not set.

Note: This parameter is applicable only for EC volumes.

Once set, you cannot change the topology of an erasure coded volume using this command. To change the topology of an erasure coded volume, use `volume move` on page 2021

last-updated: Apr 11, 2022

Parameter: `enforceminreplicationforio`	Possible Values: • `true` • `false` Description: Specifies whether (`true`) or not (`false`) to enforce minimum number of replicas for the (read-write) volume during IO. This flag ensures that further updates (writes) to volume are successful only when the minimum number of copies of the container are available. Setting this parameter to `true` ensures that if writes succeed, then it has been applied to at least the minimum number of copies; if writes fail, it may have been applied to zero or more copies. Enabling this parameter, may stall `volume dump` and `volume snapshot create` operations, if the minimum number of copies of the container are not available. If you do not set this parameter on a volume, or if you modified this parameter from `false` to `true`, then you need to restart all the nodes where the containers associated with the volume exist, for the changes to take effect. This flag is ignored on mirror volumes.
Parameter: `forceauditenable`	Possible Values: • `true` • `false` Description: Specifies whether (`true`) or not (`false`) to force audit of operations on all files, tables, and streams in the volume if auditing is enabled at the cluster and volume levels, irrespective of the audit setting on the individual directory, file, table, and stream.
Parameter: `group`	Possible Values: Any user with `Create Volume` privileges. Description: Space-separated list of `group:permission` pairs.
Parameter: `maxinodesalarmthreshold`	Possible Values: Any positive integer. Description: The number of inodes, which when exceeded raises the INODES_EXCEEDED alarm.
Parameter: `maxnssizembalarmthreshold`	Possible Values: Any positive integer. Description: The namespace container size, which when exceeded raises the INODES_EXCEEDED alarm.
Parameter: `metricsenabled`	Possible Values: • `true` • `false` Description: Specifies whether (`true`) or not (`false`) to enable metrics collection for a volume.
Parameter: `minreplication`	Possible Values: Can be any value that you desire based on the replication you need.

last-updated: Apr 11, 2022

Description: The minimum replication level. When the replication factor falls below this minimum, re-replication occurs as aggressively as possible to restore the replication level. If any containers in the CLDB volume fall below the minimum replication factor, writes are disabled until aggressive re-replication restores the minimum level of replication.

> **Tip:** For more information, see Understanding Replication on page 454.

Parameter: `mirrorschedule`

Possible Values: `0` or a valid schedule ID.

Description: The schedule ID corresponding to the schedule to be used for mirroring. If you specify a mirror schedule ID, the mirror volume automatically syncs with its source volume on the specified schedule. Pre-assigned IDs include `1` for critical data, `2` for important data, and `3` for normal data. Custom schedules are assigned ID numbers in sequence. To determine the ID number, use the `schedule list` command. To disable the schedule, set this parameter to `0`.

Parameter: `mirrorthrottle`

Possible Values:

- `true`
- `false`

Description: Specifies whether mirror throttling is enabled (`true`) or disabled (`false`). Throttling is set on the source volume and applies to all its mirrors.

Parameter: `namecontainerdatathreshold`

Possible Values: Any integer value.

If you set this parameter to `0`, there is no limit on the size of user data that can be stored in the name container.

Description: Limits the size of user data that can be placed in the name container. The value is interpreted as being in MB. If the user data size limit:

- Has not yet been reached, the first 64 KB of data is stored in name container, and the rest of the data is stored in data containers.
- Has already been reached, only meta data is stored in the name container, and the data is stored in data containers. For example, if you set the current name container size to 200GB and the limit to 100GB, then all new user data is stored in data containers.

Parameter: `nsminreplication`

Possible Values: Any integer value.

Description: When the replication factor falls below this value, re-replication occurs as aggressively as possible to restore the replication level. If any containers in the CLDB volume fall below the minimum replication factor, writes are disabled until aggressive re-replication restores the minimum level of replication.

When enabled, the CLDB manages the namespace container replication separate from the data container replication. You use this capability when you have low volume replication but want to have higher namespace replication.

Set the value to be the same or larger than the value of the equivalent data replication parameter, `minreplication`.

See also: Understanding Replication on page 454.

Parameter: `nsreplication`

Possible Values: Any integer value.

Description: The desired namespace container replication level.

When the number of copies falls below the desired replication factor, but remains equal to or above the minimum replication factor, re-replication occurs after the timeout specified in the `cldb.fs.mark.rereplicate.sec` parameter. This timeout is the time given for a node that is down to come back online. After this timeout period, the CLDB takes the action required to restore the replication factor.

When enabled, the CLDB manages the namespace container replication separate from the data container replication. Use this capability when you have low volume replication but want to have higher namespace replication.

By default, the value of this parameter is the same or larger than the value of the equivalent data replication parameter, `replication`. However, to set the value of this parameter lower than the `replication` value, first set `engg.manual.override` to `true` in `cldb.conf`.

See also: Understanding Replication on page 454.

Parameter: `name`

Possible Values: Not Applicable.

Description: The name of the volume to modify.

Parameter: `offloadschedule`

Possible Values: Any valid schedule ID. To disable schedule-based offload, set this value to `0`.

Description: The ID of the schedule to associate with the volume for offloading volume data to the tier.

Note: This parameter is required only for Cold/EC tiered volumes.

Parameter: `quota`

Possible Values: Any integer value along with a unit.

Description: The quota for the volume as `integer` plus `unit. Example: quota=500G; Units: B`, `K`, `M`, `G`, `T`, `P`

Do not use two-letter abbreviations for quota units, such as `GB` and `MB`.

When you set a quota for a tiering-enabled volume, the quota is the total space allocated for the volume irrespective of the location (cluster or tier) where the volume data is stored. For example, if you allocate 1GB of hard quota for a tiering-enabled volume, writes fail after you write 1GB of data whether or not the volume data is local (on the cluster) or offloaded (to the tier).

Note that quotas for source and mirror volumes must match.

Parameter: `readAce`

Possible Values: Any valid permissions.

Description: Specifies Access Control Expressions(ACEs) that grant permissions at the volume level to read files and tables in the volume. The default value is `p`, which grants access to all users.

See *ACE*s.

Parameter: `readonly`

Possible Values:

- `0`
- `1`

Description: Specifies whether the volume is read-only.

- `0` - read/write
- `1` - read-only

Parameter: `recallexpirytime`

Possible Values: Any integer between `1` and `7500`.

Description: The amount of time (in days) to keep the recalled data before purging or offloading it.

Parameter: `replication`

Possible Values: Any integer starting at `0`.

Description: The desired replication level. When the number of copies falls below the desired replication factor, but remains equal to or above the minimum replication factor, re-replication occurs after the timeout specified in the `cldb.fs.mark.rereplicate.sec` parameter. Note that this timeout is the time given for a node that is offline to come back online. After this timeout period, the CLDB takes action to restore the replication factor.

> **Tip:** For more information, see Understanding Replication on page 454.

Parameter: `rereplicationtimeoutsec`

Possible Values: Any positive integer.

Description: Timeout (in seconds) before attempting re-replication of replica containers. This volume property defines the timeout period until CLDB starts re-replicating the containers on the node of the volume after CLDB stops receiving a heartbeat from the node.

When a node is down, CLDB gives the node an hour to come back online before it takes any action for the containers on this node. You can set this parameter on volumes to reduce the default 1 hour to a shorter time period. This option is provided mainly for local volumes, so that when the MapR File System is down, CLDB can give up quickly and decide that the container has no master. This forces the TT to give up on local containers, and take the appropriate recovery action of deleting the mapred volume and creating another one.

Parameter: `schedule`

Possible Values: `0` or a valid schedule ID.

Description: The ID of a schedule. Use the `schedule list` command to find the ID of the named schedule that you want to apply to the volume.

To disable the schedule, set this parameter to `0`.

Parameter: `skipwiresecurityfortierinternalops`

Possible Values:

- `true`
- `false`

Description: Skips wire security for internal operations.

Parameter: `source`

Possible Values: Any volume.

Description: The source volume from which a mirror volume receives updates, specified in the format `<volume>@<cluster>`.

Parameter: `tierencryption`

Possible Values:

- `true`
- `false`

Description: Specifies whether to enable (`true`) or disable (`false`) encryption of data on the object store. This parameter is applicable only for cold-tier volumes. If you enable this parameter, user data is encrypted before being written to the object, and the HTTPS protocol is used for communication with the object store to ensure that data is encrypted both on the wire and on the tier.

You can set this parameter only if you specify a tier name (see the `tiername` parameter) as well. You cannot modify this parameter after you set it.

If you set the value to `true`, you can also specify a custom key using the `tierkey` parameter. Once set to `true`, the MAST Gateway uses HTTPS to upload data to the cold-tier. If the cold tier does not support HTTPS, all tier related operations fail. If the cold-tier does not support HTTPS, you must explicitly set the value for this to `false` at the time of associating a tier with the volume because the default value for this parameter is `true`.

> **Tip:** For warm tier, use `-dare` option on the front-end volume to enable or disable encryption of data-at-rest.

Parameter: `tieringrule`

Possible Values: Name of any valid rule

Description: The name of the rule (referred to as storage policy in the Control System) to use for offloading data to the tier. If you do not specify a rule, the default rule, which is all files (p), is associated with the volume. See Creating a Rule in Creating a Storage Tier Policy on page 972 for more information.

Parameter: `tierkey`

Possible Values:Any 32-character HEX string, or let CLDB auto-generate this string

Description: The 32-character HEX string to use for encryption only for cold tier volumes. If you do not specify a string, CLDB generates a 32 character HEX string to use for encrypting the data to offload to the tier.

Restriction: You cannot modify the tierkey that is already associated with the volume.

Parameter: `tiername`

Possible Values: Not Applicable

last-updated: Apr 11, 2022

Description: The name of the tier to use for offloading data. You can set this name only once and cannot modify it.

For warm tiering, you cannot specify this parameter if `ecenable` is set to `true`.

Parameter: `type`

Possible Values:

- `mirror`
- `rw`
- `0`
- `1`

Description: The type of volume to create.

The following values are accepted:

- `mirror` - standard mirror (read-only) volume (promotable to standard read-write volume)
- `rw` - standard (read-write) volume (convertible to standard mirror volume)
- `0` - standard (read-write) volume (for backward compatibility)
- `1` - non-convertible mirror (read-only) volume (for backward compatibility)

Parameter: `user`

Possible Values: Any valid permissions

Description: Space-separated list of `user:permission` pairs.

Use comma to separate permissions. For example: `user:permission,permission,...`

Parameter: `wiresecurityenabled`

Default Value: true

Possible Values:

- `true`
- `false`

Description: Enables (`true`) or disables (`false`) on-wire encryption for all files, tables, and streams in the volume for secure clusters. This parameter is not supported on insecure clusters.

If `true`, this setting overrides all file, table, and stream level encryption settings (set using the `hadoop mfs` command) and enables on-wire encryption for all files, tables, and streams. If you disable (`false`) this parameter at the volume level, but enable it at the file, table, or stream level, the file, table, or stream level encryption setting overrides this setting on those files, tables, and streams where it is enabled; for all other files, tables, and streams where encryption is not enabled at the file, table, or stream level, the on-wire encryption is disabled.

Parameter: `writeAce`

Possible Values: Any valid permissions

Description: Specifies Access Control Expressions (ACEs) that grant permission at the volume level to write to files and tables in the volume. The default value is p, which grants access to all users.

See *ACE*s.

Examples

Change the source volume of the mirror "test-mirror":

CLI

```
/opt/mapr/bin/maprcli volume
modify -name test-mirror -source
volume-2@my-cluster
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/modify?
name=test-mirror&source=volume-2@my-cl
uster&' --user mapr:mapr
```

Create a volume with namespace container replicas

CLI

```
/opt/mapr/bin/maprcli
volume modify -name
testVol -nsminreplication
2 -nsreplication 4 -json
{
    "timestamp":1526528489360,
    "timeofday":"2018-05-16
08:41:29.360 GMT-0700 PM",
    "status":"OK",
    "total":0,
    "data":[

    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/modify?
name=testVol&nsminreplication=2&nsrepl
ication=4' --user mapr:mapr
{"timestamp":1526528556748,"timeofday"
:"2018-05-16 08:42:36.748 GMT-0700
PM","status":"OK","total":0,"data":[]}
```

Modify a volume to allow inheritance by a child volume

Sub-volumes (children) can inherit properties from their parent volume. The `maprcli volume create` and `volume modify` commands provide parameters for setting the inheritance feature. For a child volume to inherit from a parent volume, the parent volume must grant permission, and the child volume must be created specifying the volume name of the parent. In the following example, the parent volume, parentVol, grants inheritance to child volumes.

CLI

```
/opt/mapr/bin/maprcli volume
modify -name parentVol -allowgrant
true
```

REST

```
curl -k -X
POST 'https://abc.sj.us:8443/
rest/volume/modify?name=parentVol?
allowgrant=true' --user mapr:mapr
```

Set and modify ACEs on a volume

In the following example, the command sets and modifies access (defined using ACEs) to the volume data. When the command runs, new values:

- Overwrite existing values for access types that were previously set.
- Are set for access types that were not set.

Note: There is no change to the readAce access type, which is not specified with the command, irrespective of whether it is set or not.

CLI

```
/opt/mapr/bin/maprcli volume
modify -name testVol -writeAce
'g:group1&(!u:user1|!r:role1)'
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/modify?
name=testVol&writeAce=g%3Agroup1%26%28
%21u%3Auser1%7C%21r%3Arole1%29' --user
 mapr:mapr
```

Modify the list of operations that are audited

In the following example, the `create` operation is included for auditing and the `lookup` operation is excluded from auditing. There are no changes to operations that are not specified.

CLI

```
/opt/mapr/bin/maprcli volume
modify -name parentVol -dataauditops
+create,-lookup
```

RETST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/modify?
name=p1&dataauditops=%2Bcreate%2C-look
up' --user mapr:mapr
```

Modify an existing volume to enable on-wire encryption:

CLI

```
/opt/mapr/bin/maprcli
volume modify -name
local2 -wiresecurityenabled true -json
{
        "timestamp":1505205889697,
        "timeofday":"2017-09-12
```

```
01:44:49.697 GMT-0700",
        "status":"OK",
        "total":0,
        "data":[

        ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/modify?
name=p1&wiresecurityenabled=true' --us
er mapr:mapr
{"timestamp":1526569299139,"timeofday"
:"2018-05-17 08:01:39.139 GMT-0700
AM","status":"OK","total":0,"data":[]}
```

Associate an offload rule with a tiering-enabled volume:

CLI

```
/opt/mapr/bin/maprcli volume
modify -name sampleVol -tieringrule
ksTestRule -json
{
    "timestamp":1526569498559,
    "timeofday":"2018-05-17
08:04:58.559 GMT-0700 AM",
    "status":"OK",
    "total":0,
    "data":[

    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/modify?
name=sampleVol&tieringrule=ksTestRule'
 --user mapr:mapr
{"timestamp":1526569554743,"timeofday"
:"2018-05-17 08:05:54.743 GMT-0700
AM","status":"OK","total":0,"data":[]}
```

Modify a volume to set a schedule for data offload and set number of days to three days to keep recalled data:

CLI

```
/opt/mapr/bin/maprcli
volume modify -name
sampleVol -offloadschedule
3 -recallexpirytime 3 -json
{
    "timestamp":1526569615285,
    "timeofday":"2018-05-17
08:06:55.285 GMT-0700 AM",
    "status":"OK",
    "total":0,
    "data":[
```

```
    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/modify?
name=sampleVol&offloadschedule=3&recal
lexpirytime=3' --user mapr:mapr
{"timestamp":1526569653267,"timeofday"
:"2018-05-17 08:07:33.267 GMT-0700
AM","status":"OK","total":0,"data":[]}
```

Disable scheduled snapshot creation

To disable a schedule, set the `schedule` parameter to `0`.

For example:

CLI

```
/opt/mapr/bin/maprcli volume
modify -name mapr.apps  -schedule 0
/opt/mapr/bin/maprcli volume
info -name mapr.apps  -json | grep
schedule

"scheduleid":"0",

"schedulename":"",

"mirrorscheduleid":"0"
```

Remove file filter from a volume

To remove a file filter use the special filefilter "" .

For example:

CLI

```
/opt/mapr/bin/maprcli volume
modify -name noexec -filefilter ""
```

Related reference

disk add on page 1602
Adds one or more disks to the specified node. Permissions required: `fc` or `a`.

volume create on page 1931
Creates a volume.

volume mount
Mounts one or more specified volumes. Permissions required: fc or `m` on the volume.

Syntax

CLI

```
maprcli volume mount
    [ -cluster <cluster> ]
    -name <volume list>
    -path <path list>
    [ -createparent 0|1 ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/mount?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`name`	The name of the volume to mount.
`path`	The path at which to mount the volume. The path must be relative to `/` and cannot be in the form of a global namespace path (for example, `/mapr/<cluster-name>/`).
`createparent`	Specifies whether or not to create a parent volume: • `0` = Do not create a parent volume. • `1` = Create a parent volume.

Examples

Mount the volume "test-volume" at the path "/test":

CLI

```
maprcli volume mount -name
test-volume -path /test
{
    "timestamp":1537804971391,
    "timeofday":"2018-09-24
09:02:51.391 GMT-0700 AM",
    "status":"OK",
    "total":0,
    "data":[

    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/mount?
name=test-volume&path=/test' --user
mapr:mapr
{"timestamp":1537804971391,"timeofday"
:"2018-09-24 09:02:51.391 GMT-0700
AM","status":"OK","total":0,"data":[]}
```

volume move

Moves the specified volume or mirror to a different topology. Permissions required: `m` or `fc` on the volume.

Syntax

CLI

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/move?<parameters>`

Parameters

Parameter: `cluster`

Possible Values: Any valid cluster.

Description: The cluster on which to create the volume.

Parameter: `ectopology`

Possible Values: Any topology that exists in your environment.

Description: The new rack path for the erasure-coded volume if you are moving an erasure-coded volume.

Note: This parameter is applicable only for EC volumes.

Parameter: `label`

Possible Values: Any label.

Description: The label to use for the storage pool. See for more information on labels.

The label should contain only the following characters:

```
A-Z a-z 0-9 _ - .
```

Tip: Use the special label named `anywhere` to let a volume reside on any storage pool. Not setting a label, causes a volume to reside only on a storage pool without a label.

Parameter: `name`

Possible Values: Any valid name

Description: The name of the volume to move. For moving:

- An erasure coded volume, specify the name of the front-end volume.
- The metadata volume associated with a tier, specify the name of the metadata volume.

The name should contain only the following characters:

```
A-Z a-z 0-9 _ - .
```

For tiering-enabled volumes, the volume name cannot exceed ninety-eight characters.

Parameter: `nslabel`

Possible Values: Any value.

Description: The label to use for the namespace container. See for more information on labels.

The label should contain only the following characters:

```
A-Z a-z 0-9 _ - .
```

Parameter: `topology`	Possible Values: Any Description: The new rack path for the: • Regular or tiered standard volume, if you are moving a regular or tiered standard volume. • Regular or tiered mirror volume, if you are moving a regular or tiered mirror volume. • Metadata volume, if you are moving a metadata volume associated with a tier. This parameter is not required, if you are moving an erasure-coded volume.

Advisory Note on Storage Labels

When the volume of a label is changed, replicas cannot be migrated within the file server, from one SP with the old label to another SP with the desired label. If there no other SPs, all old copies will not be fully migrated to the new desired label.

Examples

CLI

```
maprcli volume move -name
testVolume -topology /newPath
```

REST

```
curl -k -X 'https://abc.sj.us:8443/
rest/volume/move?
name=testVolume&topology=%2FnewPath'
 --user mapr:mapr
```

Related concepts

node on page 1694
Manages nodes in the cluster

Related reference

disk add on page 1602
Adds one or more disks to the specified node. Permissions required: `fc` or `a`.

volume create on page 1931
Creates a volume.

node list on page 1705
Lists nodes in the cluster.

configure.sh on page 2053
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

volume offload
Offloads data in the volume to the tier.

Permissions Required

The user running the command must have one of the following:

- Full control (fc) on the cluster or volume
- Volume edit permissions

Syntax

CLI

```
maprcli volume offload
   [ -cluster cluster_name ]
   [ -ignorerule <true|false> ]
   -name <volume_name>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/offload?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`ignorerule`	Specify whether (`true`) or not (`false`) to ignore existing rules associated with the volume for offloading data. If value is: • `true`, all data in the volume is offloaded and rules associated with the volume for offload are ignored. • `false`, data is offloaded based on the rules set up for offloading data. Default value is `false`.
`name`	The name of the volume.

Example

Offload a volume:

CLI

```
# /opt/mapr/bin/maprcli volume
offload -name sampleVol -json
{
    "timestamp":1501104289006,
    "timeofday":"2017-07-26
02:24:49.006 GMT-0700",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully started
offload."
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/offload?
```

```
name=sampleVol' --user mapr:mapr
{"timestamp":1519947659597,"timeofday"
:"2018-03-01 03:40:59.597 GMT-0800
PM","status":"OK","total":0,"data":
[],"messages":["Successfully started
offload."]}
```

volume recall
Recalls the offloaded data for the specified volume.

Permissions Required

The user running the command must have one of the following:

- Full control (fc) on the cluster or volume
- Volume edit permissions

Syntax

CLI

```
maprcli volume recall
        [ -cluster cluster_name ]
         -name <volume_name>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/recall?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`name`	The name of the volume.

Example

Recall a volume:

CLI

```
# /opt/mapr/bin/maprcli volume
recall -name sampleVol -json
{
    "timestamp":1520007453541,
    "timeofday":"2018-03-02
08:17:33.541 GMT-0800 AM",
    "status":"OK",
    "total":0,
    "data":[

    ],
    "messages":[
        "Successfully started recall."
```

```
    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/recall?
name=sampleVol' --user mapr:mapr
{"timestamp":1520007538784,"timeofday"
:"2018-03-02 08:18:58.784 GMT-0800
AM","status":"OK","total":0,"data":
[],"messages":["Successfully started
recall."]}
```

volume remove
Removes the specified volume or mirror. Permissions required: d or fc on the volume.

Syntax

CLI

```
maprcli volume remove
    [ -cluster <cluster> ]
    [ -force true|false ]
    [ -filter <filter> ]
    -name <volume name>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/remove?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`force`	Forces the removal of the volume, even if there are dependencies.
`name`	The volume name.
`filter`	All volumes with names that match the filter are removed.

Examples

CLI

```
maprcli volume remove -name testVolume
```

REST

```
https://abc.sj.us:8443/rest/volume/
remove?name=testVolume
```

volume rename
Renames the specified volume or mirror. Permissions required: `fc` or `d` on the volume.

Note: If you rename a volume, you must unmount and re-mount the volume to allow applications and/or users to continue accessing the volume.

Syntax

CLI

```
maprcli volume rename
    [ -cluster <cluster> ]
    -name <volume name>
    -newname <new volume name>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/rename?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`name`	The volume name.
`newname`	The new volume name. For tiering-enabled volumes, volume name cannot exceed ninety-eight characters.

Examples

Rename a standard volume:

CLI

```
# maprcli volume
rename -name testVolume -newname
newVolumeName -json
{
    "timestamp":1537994815889,
    "timeofday":"2018-09-26
01:46:55.889 GMT-0700 PM",
    "status":"OK",
    "total":0,
    "data":[

    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/rename?
name=testVolume&newname=newVolumeName'
 --user mapr:mapr
{"timestamp":1537994918599,"timeofday"
:"2018-09-26 01:48:38.599 GMT-0700
PM","status":"OK","total":0,"data":[]}
```

volume showmounts
Returns a list of mount points for the specified volume.

last-updated: Apr 11, 2022

Note: The three dots in the output indicate hierarchical mounts within a volume. Use `-json` to format the output.

Syntax

CLI

```
maprcli volume showmounts
    [ -cluster <cluster name> ]
    -name <volume name>
    -json
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/volume/showmounts?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster hosting the volume.
`json`	(Required) Returns the output in JSON format.
`name`	The name of the volume to return a list of mount points for.

Examples

Return the mount points for volume mapr.user.volume for the cluster my.cluster.com:

CLI

```
maprcli volume showmounts -cluster
my.cluster.com -name
mapr.user.volume -json
```

REST

```
curl -k -X GET 'https://
abc.sj.us:8443/rest/volume/showmounts?
cluster=my.cluster.com&name=mapr.user.
volume' --user mapr:mapr
```

volume snapshot create

Creates a snapshot of the specified volume, using the specified snapshot name.

- License required: Enterprise Edition
- Permissions required: `fc` or `m` on the volume

Syntax

CLI

```
maprcli volume snapshot create
    [ -cluster <cluster> ]
    [ -retain <positive_integer>mi|h|
```


last-updated: Apr 11, 2022

```
d|w|m|y ]
    -snapshotname <snapshot>
    -volume <volume>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/snapshot/create?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`retain`	Specifies how long to retain the snapshot data. Value can be specified in: • Minutes by appending `mi` to the integer. For example, `30mi` can be specified as the value for this parameter to retain snapshot data for 30 minutes. • Hours by appending `h` to the integer. For example, `1h` can be specified as the value for this parameter to retain snapshot data for 1 hour. • Days by appending `d` to the integer. For example, `2d` can be specified as the value for this parameter to retain snapshot data for 2 days. • Weeks by appending `w` to the integer. For example, `4w` can be specified as the value for this parameter to retain snapshot data for 4 weeks. • Months by appending `m` to the integer. For example, `10m` can be specified as the value for this parameter to retain snapshot data for 10 months. • Years by appending `y` to the integer. For example, `7y` can be specified as the value for this parameter to retain snapshot data for 7 years. If this is not specified, snapshot data will never expire.
`snapshotname`	The name of the snapshot to create.
`volume`	The volume for which to create a snapshot.

Examples

Create a snapshot called "test-snapshot" for volume "test-volume":

CLI

```
maprcli volume
snapshot create -snapshotname
test-snapshot -volume test-volume
{
    "timestamp":1537805380237,
    "timeofday":"2018-09-24
```

```
09:09:40.237 GMT-0700 AM",
    "status":"OK",
    "total":0,
    "data":[

    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/snapshot/
create?
volume=test-volume&snapshotname=test-s
napshot' --user mapr:mapr
{"timestamp":1537805548885,"timeofday"
:"2018-09-24 09:12:28.885 GMT-0700
AM","status":"OK","total":0,"data":[]}
```

volume snapshot list
Displays info about a set of snapshots.

You can specify the snapshots by volumes or paths, or by specifying a filter to select volumes with certain characteristics.

Syntax

CLI

```
/opt/mapr/bin/maprcli volume snapshot
list
    [ -cluster <cluster name> ]
    [ -columns <fields> ]
    ( -filter <filter> ]
    [ -path <volume path list> ]
    [ -volume <volume list> ]
    [ -limit <rows> ]
    [ -output (terse|verbose) ]
    [ -start <offset> ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/volume/snapshot/list[?<parameters>]`

Parameters

Either `volume` or `path` can be used if you wish the specify the snapshots. In addition, `filter` can be used with `volume` and `path`, or independently.

Parameter	Description
`cluster`	The cluster on which to run the command.
`columns`	A comma-separated list of fields to return in the query. See the Fields on page 2031 table below. Default: none
`filter`	A filter specifying snapshots to list. See Filters on page 1526 for more information.

Parameter	Description
`limit`	The number of rows to return, beginning at start. Default: `2147483647`
`output`	Specifies whether the output should be `terse` or `verbose`. Default: `verbose`
`path`	A comma-separated list of paths for which to list snapshots.
`start`	The offset from the starting row. Default: `0`
`volume`	A comma-separated list of volumes for which to list snapshots.

Fields

The following table lists the fields used in the `columns` parameter, and returned as output.

Field Name	Short Name	Description
`snapshotid`	`id`	Unique snapshot ID.
`sharedSize`	shSz	Size of data (in MB) that the snapshot shares with previous snapshots.
`volumename`	`vn`	Name of the read-write volume associated with the snapshot.
`ownername`	`on`	Owner (user or group) associated with the volume.
`cumulativeReclaimSizeMB`	`cs`	Disk space (in MB) used/owned by the snapshot
`snapshotname`	`n`	Snapshot name.
`ownedsize`	owSz	Size of data (in MB) owned by a snapshot, as opposed to sharedSize (owned by previous snapshots).
`ownertype`	`ot`	Owner type for the owner of the volume: • `0=user` • `1=group`
`volumeid`	`vid`	ID of the volume associated with the snapshot.
`creationtime`	`ct`	Snapshot creation time. Date time string (verbose output) or milliseconds since 1970 (terse output).
`volumepath`	`vp`	Path to the volume associated with the snapshot.
`expirytime`	`et`	The time until which the snapshot should be maintained. Expired snapshots are purged (deleted) periodically. Date time string (verbose output), or milliseconds since 1970 (terse output); 0 = never expires.

last-updated: Apr 11, 2022

Field Name	Short Name	Description
volumeSnapshotAces	N/A	*ACE* permissions for read and write on the volume snapshot. Use `-json` to view the *ACE* permissions.

Output

This sample output is based on using the following code to create a snapshot called `uservolume` for the volume named `users`.

```
/opt/mapr/bin/maprcli volume snapshot create -snapshotname
uservolsnap -volume users
```

Sample Output

```
# maprcli volume snapshot list
snapshotid  ownedsize  sharedSize  volumename
ownername  ownertype  cumulativeReclaimSizeMB  volumeid
snapshotname    creationtime                  volumepath
volumeSnapshotAces
256000051   0          0           users       mapr       1
0                      212450174  uservolsnap     Wed Sep 26 10:45:27 PDT
2018  /user       ...
```

Examples

List all snapshots:

CLI

```
# maprcli volume snapshot list
snapshotid  ownedsize
sharedSize  volumename  ownername
ownertype  cumulativeReclaimSizeMB
volumeid   snapshotname
creationtime
volumepath  volumeSnapshotAces
256000049   0          0
egVol       mapr       1
0                        29379677
egVol-snapshot  Wed Sep
26 10:42:52 PDT 2018  /
egVol      ...
256000051   0          0
users       mapr       1
0                        212450174
uservolsnap     Wed Sep
26 10:45:27 PDT 2018  /
user       ...
256000050   0          0
egVol       mapr       1
0                        29379677
egVolSnapshot   Wed Sep 26 10:43:12
PDT 2018  /egVol      ...
```

REST

```
curl -k -X GET 'https://
abc.sj.us:8443/rest/volume/snapshot/
list' --user mapr:mapr
{"timestamp":1537984492448,"timeofday"
:"2018-09-26 10:54:52.448 GMT-0700
```

```
AM","status":"OK","total":3,"data":
[{"ownername":"mapr","ownertype":"1","
volumeid":"29379677","volumename":"egV
ol","volumepath":"/
egVol","snapshotid":"256000049","snaps
hotname":"egVol-snapshot","creationtim
e":"Wed Sep 26 10:42:52 PDT
2018","cumulativeReclaimSizeMB":"0","o
wnedsize":"0","sharedSize":"0","volume
SnapshotAces":
{"readAce":"p","writeAce":"p"}},
{"ownername":"mapr","ownertype":"1","v
olumeid":"212450174","volumename":"use
rs","volumepath":"/
user","snapshotid":"256000051","snapsh
otname":"uservolsnap","creationtime":"
Wed Sep 26 10:45:27 PDT
2018","cumulativeReclaimSizeMB":"0","o
wnedsize":"0","sharedSize":"0","volume
SnapshotAces":
{"readAce":"p","writeAce":"p"}},
{"ownername":"mapr","ownertype":"1","v
olumeid":"29379677","volumename":"egVo
l","volumepath":"/
egVol","snapshotid":"256000050","snaps
hotname":"egVolSnapshot","creationtime
":"Wed Sep 26 10:43:12 PDT
2018","cumulativeReclaimSizeMB":"0","o
wnedsize":"0","sharedSize":"0","volume
SnapshotAces":
{"readAce":"p","writeAce":"p"}}]}
```


List all snapshots and format the output:


```
# maprcli volume snapshot list -json
{
    "timestamp":1537984231452,
    "timeofday":"2018-09-26 10:50:31.452 GMT-0700 AM",
    "status":"OK",
    "total":3,
    "data":[
        {
            "ownername":"mapr",
            "ownertype":"1",
            "volumeid":"29379677",
            "volumename":"egVol",
            "volumepath":"/egVol",
            "snapshotid":"256000049",
            "snapshotname":"egVol-snapshot",
            "creationtime":"Wed Sep 26 10:42:52 PDT 2018",
            "cumulativeReclaimSizeMB":"0",
            "ownedsize":"0",
            "sharedSize":"0",
            "volumeSnapshotAces":{
                "readAce":"p",
                "writeAce":"p"
            }
        },
        {
            "ownername":"mapr",
            "ownertype":"1",
            "volumeid":"212450174",
```

```
            "volumename":"users",
            "volumepath":"/user",
            "snapshotid":"256000051",
            "snapshotname":"uservolsnap",
            "creationtime":"Wed Sep 26 10:45:27 PDT 2018",
            "cumulativeReclaimSizeMB":"0",
            "ownedsize":"0",
            "sharedSize":"0",
            "volumeSnapshotAces":{
                "readAce":"p",
                "writeAce":"p"
            }
        },
        {
            "ownername":"mapr",
            "ownertype":"1",
            "volumeid":"29379677",
            "volumename":"egVol",
            "volumepath":"/egVol",
            "snapshotid":"256000050",
            "snapshotname":"egVolSnapshot",
            "creationtime":"Wed Sep 26 10:43:12 PDT 2018",
            "cumulativeReclaimSizeMB":"0",
            "ownedsize":"0",
            "sharedSize":"0",
            "volumeSnapshotAces":{
                "readAce":"p",
                "writeAce":"p"
            }
        }
    ]
}
```

volume snapshot preserve
Preserves one or more snapshots from expiration.

Specify the snapshots by volumes, paths, filter, or IDs.

- License required: Enterprise Edition
- Permissions required: `fc` or `m` on the volume

Syntax

CLI

```
maprcli volume snapshot preserve
    [ -cluster <cluster> ]
    ( -filter <filter> | -path
<volume path list> | -snapshots
<snapshot list> | -volume <volume
list> )
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/snapshot/preserve[?<parameters>]`

Parameters

Specify exactly one of the following parameters: volume, path, filter, or snapshots.

Parameter	Description
`cluster`	The cluster on which to run the command.
`filter`	A filter specifying snapshots to preserve. See Filters on page 1526 for more information.
`path`	A comma-separated list of paths for which to preserve snapshots.
`snapshots`	A comma-separated list of snapshot IDs to preserve.
`volume`	A comma-separated list of volumes for which to preserve snapshots.

Examples

Preserve two snapshots by ID:

First, use `volume snapshot list` to get the IDs of the snapshots you wish to preserve. Example:

```
# maprcli volume snapshot list
snapshotid  ownedsize  sharedSize  volumename  ownername
ownertype  cumulativeReclaimSizeMB  volumeid   snapshotname
creationtime                  volumepath  volumeSnapshotAces
256000049   0          0           egVol       mapr       1
0                       29379677   egVol-snapshot  Wed Sep 26 10:42:52 PDT
2018  /egVol      ...
256000051   0          0           users       mapr       1
0                       212450174  uservolsnap     Wed Sep 26 10:45:27 PDT
2018  /user       ...
256000050   0          0           egVol       mapr       1
0                       29379677   egVolSnapshot   Wed Sep 26 10:43:12 PDT
2018  /egVol      ...
```

Use the IDs in the `volume snapshot preserve` command. For example, to preserve the first two snapshots in the above list, run the commands as follows:

CLI

```
maprcli volume
snapshot preserve -snapshots
256000049,256000051 -json
{
    "timestamp":1537986060505,
    "timeofday":"2018-09-26
11:21:00.505 GMT-0700 AM",
    "status":"OK",
    "total":0,
    "data":[

    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/snapshot/
preserve?
snapshots=256000049,256000051' --user
mapr:mapr
{"timestamp":1537986132998,"timeofday"
```

```
:"2018-09-26 11:22:12.998 GMT-0700
AM","status":"OK","total":0,"data":[]}
```

volume snapshot remove
Removes one or more snapshots.

- License required: Enterprise Edition
- Permissions required: `fc` or `m` on the volume

Syntax

CLI

```
maprcli volume snapshot remove
    [ -cluster <cluster> ]
    ( -snapshotname <snapshot name> ]
    [ -snapshots <snapshots> ]
    [ -volume <volume name> ]
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/snapshot/remove[?<parameters>]`

Parameters

Specify both snapshot name and volume, or just snapshot ID.

Parameter	Description
`cluster`	The cluster on which to run the command.
`snapshotname`	The name of the snapshot to remove. You must also specify the volume name using the `volume` parameter.
`snapshots`	A comma-separated list of IDs of snapshots to remove.
`volume`	The name of the volume from which to remove the snapshot. This is required if you are removing snapshot by specifying the snapshot name (using `snapshotname` parameter).

Examples

Remove the snapshot named "test-snapshot" associated with volume named "test-volume":

CLI

```
maprcli volume
snapshot remove -snapshotname
test-snapshot -volume test-volume
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/api/volume/snapshot/
remove?
snapshotname=test-snapshot&volume=tes
t-volume' --user mapr:mapr
```

Remove two snapshots by ID:

First, use `volume snapshot list` to get the IDs of the snapshots you wish to remove. Example:

```
# maprcli volume snapshot list
snapshotid  ownedsize  sharedSize  volumename  ownername
ownertype  cumulativeReclaimSizeMB  volumeid   snapshotname
creationtime                  volumepath  volumeSnapshotAces
256000049   0          0           egVol       mapr       1
0                       29379677   egVol-snapshot  Wed Sep 26 10:42:52 PDT
2018  /egVol      ...
256000051   0          0           users       mapr       1
0                       212450174  uservolsnap     Wed Sep 26 10:45:27 PDT
2018  /user       ...
256000050   0          0           egVol       mapr       1
0                       29379677   egVolSnapshot   Wed Sep 26 10:43:12 PDT
2018  /egVol      ...
```

Use the IDs in the `volume snapshot remove` command. For example, to remove the first two snapshots in the above list, run the commands as follows:

CLI

```
maprcli volume snapshot
remove -snapshots 256000049,256000051
{
    "timestamp":1537986405764,
    "timeofday":"2018-09-26
11:26:45.764 GMT-0700 AM",
    "status":"OK",
    "total":0,
    "data":[

    ]
}
```

REST

```
# curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/snapshot/
remove?
snapshots=256000049,256000051' --user
mapr:mapr
{"timestamp":1537987406574,"timeofday"
:"2018-09-26 11:43:26.574 GMT-0700
AM","status":"OK","total":0,"data":[]}
```

volume tierjobabort
Terminates an ongoing offload or recall operation for a volume (specified by name).

Permissions Required

The user running the command must have one of the following:

- Full control (fc) on the cluster or volume
- Volume edit permissions

Syntax

CLI

```
maprcli volume tierjobabort
    [ -cluster cluster_name ]
     -name name
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/tierjobabort?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`name`	The name of the volume.

Example

Stop offloading data to the tier:

CLI

```
/opt/mapr/bin/maprcli volume
tierjobabort -name sampleVol -json
{
        "timestamp":1503504450211,
        "timeofday":"2017-08-23
04:07:30.211 GMT+0000",
        "status":"OK",
        "total":0,
        "data":[

        ],
        "messages":[
                "Successfully started
to abort."
        ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/
tierjobabort?name=sampleVol' --user
mapr:mapr
{"timestamp":1503504450211,"timeofday"
:"2017-08-23 04:07:30.211
GMT+0000","status":"OK","total":0,"dat
a":[],"messages":["Successfully
started to abort."]}
```

volume tierjobstatus

Retrieves the status of the currently running operation (such as offload, recall, or terminate) for a volume.

Permissions Required

The user running the command must have one of the following:

- Full control (fc) on the cluster or volume
- Volume edit permissions

Syntax

CLI

```
maprcli volume tierjobstatus
     [ -cluster <cluster_name> ]
     -name <volume_name>
     [ -verbose true|false ]
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/volume/tierjobstatus?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`name`	The name of the volume.
`verbose`	Specifies whether the command output should be verbose. The value for this must be `true` to retrieve the status of a compaction operation. The default value is `false`.

Output

The command returns the following:

`state`	The status of the offload, recall, or terminate operation. See Statuses on page 2040 below for more information.
`offloadedDataSize`	The amount of data offloaded. This is returned only when returning the status of an offload operation.
`progress`	The percentage of containers that have been processed so far.
`recalledDataSize`	The amount of data recalled. This is returned only when returning the status of a recall operation.
`reclaimedDataSize`	The amount of data purged. This is returned only when returning the status of a compaction job.
`startTime`	The date and timestamp for when the offload operation started.
`endTime`	The date and timestamp for when the offload operation completed.

last-updated: Apr 11, 2022

`gateway`	The IP address of the MAST Gateway used for the tiering operation.

Statuses

The value for the `state` field (statuses) can be one of the following:

State	Description

Scheduled	Indicates the job request has reached CLDB, but has not yet been forwarded to any MAST Gateway service. For example:
	CLI
	`{ "timestamp":1532093619983, "timeofday":"2018-07-20 06:33:39.983 GMT-0700 AM", "status":"OK", "total":1, "data":[ { "compaction":{ "state":"Scheduled", "scheduleTime":"2018-07-20 06:33:38.953 GMT-0700", "gateway":"10.10.108.116:8660" } } ] }`
	REST
	`{"timestamp":1532093619983,"timeofday":"2018-07-20 06:33:39.983 GMT-0700 AM","status":"OK","total":1,"data":[{"compaction":{"state":"Scheduled","scheduleTime":"2018-07-20 06:33:38.953 GMT-0700","gateway":"10.10.108.116:8660"}}]}`

<table>
<tr><th>State</th><th>Description</th></tr>
<tr><td>Running</td><td>Indicates the offload or recall job has been forwarded to MAST Gateway service. The MAST Gateway service can either still be waiting for resources to run the job or is actually performing the requested job. For example:

CLI

<pre>{
"timestamp":1532095481297,
"timeofday":"2018-07-20 07:04:41.297 GMT-0700 AM",
"status":"OK",
"total":1,
"data":[
{
"offload":{
"state":"Running",
"progress":"61%",
"startTime":"2018-07-20 07:00:02.277 GMT-0700",
"gateway":"10.10.108.115:8660"
},
"compaction":{
"state":"Success",
"progress":"100%",
"startTime":"2018-07-20 06:34:06.628 GMT-0700",
"endTime":"2018-07-20 06:40:25.334 GMT-0700",
"reclaimedDataSize":"0 MB",
"gateway":"10.10.108.115:8660"
}
}
]
}</pre>
REST

<pre>{"timestamp":1532095481297,"timeofday":"2018-07-20 07:04:41.297 GMT-0700 AM","status":"OK","total":1,"data":[{"offload":</pre></td></tr>
</table>

State	Description

FailureFatal	Indicates the job has failed with non-retriable error. You must resolve the issue and retry the operation. For example:
	CLI
	{
	"timestamp":1531778057385,
	"timeofday":"2018-07-16 09:54:17.385 GMT+0000 PM",
	"status":"OK",
	"total":1,
	"data":[
	{
	"offload":{
	"state":"FailureFatal",
	"progress":"50%",
	"startTime":"2018-07-16 21:54:01.779 GMT+0000",
	"endTime":"2018-07-16 21:54:05.339 GMT+0000",
	"offloadedDataSize":"0 MB",
	"gateway":"10.10.88.198:8660"
	}
	}
	]
	}
	REST
	{"timestamp":1531778057385,"timeofday":"2018-07-16 09:54:17.385 GMT+0000 PM","status":"OK","total":1,"data":[{"offload":{"state":"FailureFatal","progress":"50%","startTime":"2018-07-16 21:54:01.779 GMT+0000","endTime":"2018-07-16 21:54:05.339 GMT+0000","offloadedDataSize":"0 MB","gateway":"10.10.88.198:8660"}}]
	}

State	Description

FailureRetriable	Indicates the job has failed with an error for which CLDB will retry the job based on the configuration parameters, `cldb.gateway.retry.count` and `cldb.gateway.retry.waittime`. But if the job is restarted manually or terminated, CLDB will not retry. For example:
	CLI
	{
	"timestamp":1532624516372,
	"timeofday":"2018-07-26 10:01:56.372 GMT-0700 AM",
	"status":"OK",
	"total":1,
	"data":[
	{
	"offload":{
	"state":"FailureRetry, RetryCount: 5",
	"progress":"50%",
	"startTime":"2018-07-25 17:43:27.924 GMT-0700",
	"endTime":"2018-07-25 17:43:59.108 GMT-0700",
	"offloadedDataSize":"0 MB",
	"gateway":"10.10.25.29:8660"
	}
	}
	]
	}
	REST
	{"timestamp":1532624656640,"timeofday":"2018-07-26 10:04:16.640 GMT-0700 AM","status":"OK","total":1,"data":[{"offload":{"state":"FailureRetry, RetryCount: 5","progress":"50%","startTime":"2018-07-25 17:43:27.924 GMT-0700","endTime":"2018-07-25 17:43:59.108 GMT-0700","offloadedDataSize":"0

<table>
<tr><th>State</th><th>Description</th></tr>
<tr><td>Success</td><td>Indicates the job has been successfully completed. For example:
CLI
<pre>{
"timestamp":1531311128469,
"timeofday":"2018-07-11 12:12:08.469 GMT+0000 PM",
"status":"OK",
"total":1,
"data":[
{
"offload":{
"state":"Success",
"progress":"100%",
"startTime":"2018-07-11 12:10:26.290 GMT+0000",
"endTime":"2018-07-11 12:10:35.521 GMT+0000",
"offloadedDataSize":"353.16 MB",
"gateway":"10.10.20.12:8660"
},
"compaction":{
"state":"Success",
"progress":"100%",
"startTime":"2018-07-11 12:12:01.335 GMT+0000",
"endTime":"2018-07-11 12:12:02.264 GMT+0000",
"reclaimedDataSize":"353.097 MB",
"gateway":"10.10.20.12:8660"
}
}
]
}</pre>REST
<pre>{"timestamp":1531311128469,"timeofday":"2018-07-11</pre></td></tr>
</table>

<table>
<tr><th>State</th><th>Description</th></tr>
<tr><td>Aborted</td><td>Indicates the job has been terminated. For example:
CLI
<pre>{
"timestamp":1503504464179,
"timeofday":"2017-08-23 04:07:44.179 GMT+0000",
  "status":"OK",
  "total":1,
  "data":[{
      "offload":{
"state":"Aborted",
"startTime":"2017-08-23 04:06:06.867 GMT+0000",
"endTime":"2017-08-23 04:06:38.910 GMT+0000",
"gateway":"10.10.88.199:8660"
      }
  }]
}</pre>REST
<pre>{"timestamp":1503504464179,"timeofday":"2017-08-23 04:07:44.179 GMT+0000","status":"OK","total":1,"data":[{"offload":{"state":"Aborted","startTime":"2017-08-23 04:06:06.867 GMT+0000","endTime":"2017-08-23 04:06:38.910 GMT+0000","gateway":"10.10.88.199:8660"}}]}</pre></td></tr>
</table>

<table>
<tr><th>State</th><th>Description</th></tr>
<tr><td>AbortInProgress</td><td>Indicates that the terminate operation is in progress. For example:
CLI
<pre>{
"timestamp":1533005375001,
"timeofday":"2018-07-30 07:49:35.001 GMT-0700 PM",
"status":"OK",
"total":1,
"data":[
{
"offload":{
"state":"AbortInProgress",
"progress":"98%",
"startTime":"2018-07-30 19:02:37.108 GMT-0700",
"gateway":"10.10.101.121:8660"
}
}
]
}</pre>
REST
<pre>{"timestamp":1533005375001,"timeofday":"2018-07-30 07:49:35.001 GMT-0700 PM","status":"OK","total":1,"data":[{"offload":{"state":"AbortInProgress","progress":"98%","startTime":"2018-07-30 19:02:37.108 GMT-0700","gateway":"10.10.101.121:8660"}}]}</pre></td></tr>
</table>

<table>
<tr><th>State</th><th>Description</th></tr>
<tr><td>AbortedInternal</td><td>Indicates the offload operation was terminated by another internal process, such as when promoting a mirror volume to a read-write volume when offload is in progress. For example:
CLI
<pre>{
"timestamp":1515488569411,
"timeofday":"2018-01-09 01:02:49.411 GMT-0800",
"status":"OK",
"total":1,
"data":[{
"recall":{
"state":"AbortedInternal",
"progress":"36%",
"startTime":"2018-01-09 01:01:57.824 GMT-0800",
"endTime":"2018-01-09 01:02:43.329 GMT-0800",
"gateway":"10.10.108.150:8660"
}
}]
}</pre>REST
<pre>{"timestamp":1515488569411,"timeofday":"2018-01-09 01:02:49.411 GMT-0800","status":"OK","total":1,"data":[{"recall":{"state":"AbortedInternal","progress":"36%","startTime":"2018-01-09 01:01:57.824 GMT-0800","endTime":"2018-01-09 01:02:43.329 GMT-0800","gateway":"10.10.108.150:8660"}}]}</pre></td></tr>
</table>

Example

CLI

```
# maprcli volume tierjobstatus -name
testVol -json -verbose true
{
   "timestamp":1533005419522,
   "timeofday":"2018-07-30
07:50:19.522 GMT-0700 PM",
   "status":"OK",
   "total":1,
   "data":[
    {
          "offload":{
              "state":"Success",
              "progress":"100%",
              "startTime":"2018-07-30
19:00:06.185 GMT-0700",
              "endTime":"2018-07-30
19:19:58.303 GMT-0700",

"offloadedDataSize":"2487.911 MB",

"gateway":"10.10.108.117:8660"
          },
          "compaction":{

"state":"AbortInProgress",
              "progress":"45%",
              "startTime":"2018-07-30
19:23:33.504 GMT-0700",

"gateway":"10.10.101.121:8660"
          }
    }
   ]
}
```

REST

```
# curl -k -X GET 'https://
abc.sj.us:8443/rest/tierjobstatus?
name=testVol&verbose=true' --user
mapr:mapr
{"timestamp":1533005419522,"timeofday"
:"2018-07-30 07:50:19.522 GMT-0700
PM","status":"OK","total":1,"data":
[{"offload":
{"state":"Success","progress":"100%","
startTime":"2018-07-30 19:00:06.185
GMT-0700","endTime":"2018-07-30
19:19:58.303
GMT-0700","offloadedDataSize":"2487.91
1
MB","gateway":"10.10.108.117:8660"},"c
ompaction":
{"state":"AbortInProgress","progress":
"45%","startTime":"2018-07-30
19:23:33.504
GMT-0700","gateway":"10.10.101.121:866
0"}}]}
```

volume tierstats
Retrieves statistics on the offload and recall operation.

Permissions Required

The user running the command must have one of the following:

- Full control (fc) on the cluster or volume
- Volume edit permissions

Syntax

CLI

```
maprcli volume tierstats
            [ -cluster cluster_name ]
            -name <volume_name>
```

REST

Request Type	GET
Request URL	`http[s]://<host>:<port>/rest/volume/tierstats?<parameters>`

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`name`	The name of the volume.

Output

The command returns the following:

`offloadThroughput`	The amount of data (in MB) offloaded per second.
`totalTierDataSize`	The total size of tiered data (in MB).
`totalTierReclaimableSize`	The size of deleted data (in MB) that is under the compaction operation threshold.
`recallThroughput`	The amount of data (in MB) recalled per second.

Example

Retrieve statistics for a volume specified by name:

CLI

```
# /opt/mapr/bin/maprcli volume
tierstats -name sampleVol -json
{
    "timestamp":1520275614872,
    "timeofday":"2018-03-05
06:46:54.872 GMT+0000",
    "status":"OK",
    "total":1,
```

```
    "data":[
        {

"totalTierDataSize":"404.323 MB",

"offloadThroughput":"17.063 MB/s",

"recallThroughput":"14.071 MB/s"
        }
    ]
}
```

REST

```
curl -k -X GET 'https://
abc.sj.us:8443/rest/volume/tierstats?
name=sampleVol' --user marp:mapr
{"timestamp":1520275614872,"timeofday"
:"2018-03-05 06:46:54.872
GMT+0000","status":"OK","total":1,"dat
a":[{"totalTierDataSize":"404.323
MB","offloadThroughput":"17.063 MB/
s","recallThroughput":"14.071 MB/s"}]}
```

Retrieve statistics for a volume after a compaction operation:

CLI

```
# maprcli volume tierstats -name
test1 -json
{
        "timestamp":1527048672887,
        "timeofday":"2018-05-23
04:11:12.887 GMT+0000 AM",
        "status":"OK",
        "total":1,
        "data":[
                {

"totalTierDataSize":"3001.926 MB",

"totalTierReclaimableSize":"100 MB",

"offloadThroughput":"31.064 MB/s"
                }
        ]
}
```

REST

```
curl -k -X GET 'https://
abc.sj.us:8443/rest/volume/tierstatus?
name=test1' --user mapr:mapr
{"timestamp":1527048672887,"timeofday"
:"2018-05-23 04:11:12.887 GMT+0000
AM","status":"OK","total":1,"data":
[{"totalTierDataSize":"3001.926
MB","totalTierReclaimableSize":"100
MB","offloadThroughput":"31.064 MB/
s"}]}
```

volume unmount

Unmounts one or more mounted volumes. Permissions required: `fc` or `m` on the volume.

Syntax

CLI

```
maprcli volume unmount
    [ -cluster <cluster> ]
    [ -force 0|1 ]
    -name <volume name>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/unmount?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`force`	Specifies whether (1) or not (0) to force the volume to unmount.
`name`	The name of the volume to unmount.

Examples

Unmount the volume "test-volume":

CLI

```
maprcli volume unmount -name
test-volume
{
    "timestamp":1537804903335,
    "timeofday":"2018-09-24
09:01:43.335 GMT-0700 AM",
    "status":"OK",
    "total":0,
    "data":[

    ]
}
```

REST

```
curl -k -X POST 'https://
abc.sj.us:8443/rest/volume/unmount?
name=test-volume' --user mapr:mapr
{"timestamp":1537805053854,"timeofday"
:"2018-09-24 09:01:43.335 GMT-0700
AM","status":"OK","total":0,"data":[]}
```

volume upgradeformat
Upgrades and old-type volume to a new-type volume, which can in turn be used as a promotable mirror volume. Permissions required: `m` or `fc` on the volume.

Syntax

CLI

```
maprcli volume upgradeformat
    [ -cluster <cluster> ]
    -name <volume name>
```

REST

Request Type	POST
Request URL	`http[s]://<host>:<port>/rest/volume/upgradeformat?<parameters>`

Parameters

Parameter	Description
`cluster`	The cluster on which to run the command.
`name`	The volume name.

Examples

CLI

```
maprcli volume upgradeformat -name
vol999 -json
```

REST

```
https://abc.sj.us:8443/rest/volume/
upgradeformat?name=vol999
```

Utilities

Contains information about various scripts and utilities, that help setup, maintain, and monitor clusters.

The following scripts and utilities help you configure clusters, setup cross cluster security, setup storage pools, monitor CLDB activity, perform consistency checks and repair errors on volumes and snapshots, and maintain clusters with ease.

Script or Utility	Description
Cluster Configuration	
configure.sh on page 2053	Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks, including setting up MapR client nodes, and configuring services for a node.
configure-crosscluster.sh on page 2065	Sets up cross-cluster security between two clusters.
disksetup on page 2092	Formats specified disks for use by MapR storage, and adds those disks to the disktab file.
mrconfig on page 2138	Lets you create, remove, and manage storage pools, disk groups, and disks; and provides information about containers.

last-updated: Apr 11, 2022

Script or Utility	Description
pullcentralconfig on page 2180	Pulls master configuration files from `/var/mapr/configuration` on the cluster to the local disk, on each node.
fcdebug on page 2098	Dynamically sets the loglevel to debug a library.
Auditing and Monitoring	
cldbguts on page 2080	Monitors CLDB activity. This utility prints information about the CLDB service that is running on the node from which you run the utility.
ectool on page 2094	Dumps or checks the validity of the stripelets in the backend volume that is associated with the volume configured for warm tiering.
expandaudit on page 2096	Expands IDs captured in the audit logs to their corresponding names.
fsck on page 2100	Detects and fixes inconsistencies in the filesystem.
gfsck on page 2102	Performs consistency checks and appropriate repairs on a volume, or a volume snapshot.
mapr-support-collect.sh on page 2121	Collects information about a cluster's recent activity, to help MapR Support diagnose problems.
mapr-support-dump.sh on page 2127	Collects node and cluster-level information for the node on which you invoke the script.
mrdirectorystats on page 2177	Prints the space usage for each directory, for a container.
mrfscmd on page 2179	Returns the path to the file specified by ID (fid).
stubfuse on page 2181	Determines the read and write performance of a FUSE mount point.
Authentication	
maprlogin on page 2130	Authenticates logins to secure MapR clusters.

configure.sh
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

Note: The `configure.sh` script must always be run as `root`.

You run `configure.sh` to set up a MapR cluster node, or to set up a MapR client node for communication with one or more clusters. You can also run `configure.sh` to update the configuration of a node. For example, you can use `configure.sh` to change the services running on a node, or specify the user that runs MapR services.

Attention: On a Windows client, the `configure.sh` script is named `configure.bat`. The script requires the `-c` parameter and does not accept the `-Z` parameter, but otherwise works similarly as on a Linux client.

Steps Performed by configure.sh
`configure.sh` performs the following steps, each time you run it:

- **Updates /opt/mapr/conf/mapr-clusters.conf with the cluster name.** It creates or modifies a line in `/opt/mapr/conf/mapr-clusters.conf` containing a cluster name followed by a list of CLDB nodes. New entries are added to `mapr-clusters.conf` when the cluster name passed to the `-N` parameter is different from the existing cluster name in that file.

last-updated: Apr 11, 2022

- **Checks that the node has at least 4GB of RAM, and that the /tmp and /opt partitions each have at least 1 GB of free space.** If these conditions are not met, the script asks for confirmation before continuing.
- **Disables standard NFS daemons.** If the node has the `mapr-nfs` role, the script disables the standard Linux NFS daemon, since both NFS processes cannot run on the same node.
- **Updates additional *.conf and *.xml files related to the cluster and the services running on the node.** For example, `yarn-site.xml`, `warden.conf`, and `cldb.conf` may be updated based on input to `configure.sh`.
- **On cluster nodes, it creates a group named *shadow*, adds the MapR user to this group, and then enables members of the *shadow* group to view the */etc/shadow* file.** Read access to the `/etc/shadow` file enables MapR users to authenticate with the MapR cluster.
- **Starts newly installed services.** Automatically starts new services, if Warden is running at the time you run `configure.sh`.
- **All changes to configuration options or system files are logged to /opt/mapr/logs/configure.log.** You can use the `-L` parameter to specify a different log file name.

When you include disk-setup options (`-D` or `-F`) on nodes with the `mapr-fileserver` role, the script performs the following additional steps:

- **Runs disksetup to create the disktab file.** `configure.sh` takes the values that you specify in the `-disk-opts` option, and passes the value to disksetup. For example, if you include `-disk-opts FW5` when you run `configure.sh`, `configure.sh` runs `disksteup -F -W5`. If disksetup fails, `configure.sh` exits with an error.
- **Starts Zookeeper and Warden.** When the `configure.sh` script starts services, the message `starting <servicename>` is echoed to the standard output to enable the user to see which services are starting. When Warden starts, the Warden and ZooKeeper services are added to the `inittab` file as the first available `inittab` IDs, enabling these services to restart automatically on failure.

  You can specify the `-no-autostart` option to prevent the script from starting Zookeeper or Warden when you run `configure.sh` with the `-F` or `-D` options.

Syntax

```
/opt/mapr/server/configure.sh
        -C <cldb_list>
        -Z <zookeeper_list>
        -EZ <ext_zookeeper_list>
        [<parameters>]
```

```
/opt/mapr/server/configure.sh
        -C <cldb_list>
        [ -M <cldb_mh_list ...> ]
        -Z <zookeeper_list>
        [<parameters>]
```

```
/opt/mapr/server/configure.sh
        -c
```

```
        [ -R ]
        [<parameters>]
```

```
/opt/mapr/server/configure.sh
        -R
        [ -c ]
        [<parameters>]
```

Options

-C

Use the `-C` option for CLDB servers that only have a single IP address. This option takes a comma-separated list of the CLDB nodes that this machine uses to connect to the MapR cluster. The list is in the following format:

```
hostname[:port_no]
[,hostname[:port_no]...]
```

-c

Specifies client setup. The `-C` option is required, while the `-Z` option is optional. See set up a MapR client node for communication with one or more clusters.

-EZ

The `-EZ` option is optional when configuring the cluster, and is not applicable when configuring a client. This option takes a comma-separated list of the external IP addresses of the ZooKeeper nodes in the cluster. The list is in the following format:

```
hostname[:port_no]
[,hostname[:port_no] ...]
```

-M

Use the `-M` option only for multihomed CLDB servers that have more than one IP address. This option takes a comma-separated list of the multihomed CLDB nodes that this machine uses to connect to the MapR cluster. The list is in the following format:

```
hostname[:port_no][,
[hostname[:port_no]...]]
```

-R

After initial node configuration, specifies that `configure.sh` should use the previously configured ZooKeeper and CLDB nodes. The `-C` and `-Z` parameters are not required when you specify `-R`. When `-R` is specified, the CLDB credentials are read from `mapr-clusters.conf`, while the ZooKeeper credentials are read from `warden.conf`. Use the `-R` option when you make changes to the services configured on a node without changing the CLDB and ZooKeeper nodes. Specify the `--noRecalcMem` parameter to skip recalculating memory settings when refreshing roles.

last-updated: Apr 11, 2022

Note: This parameter impacts the JMX parameters in `/opt/mapr/conf/env_override.sh` in the following ways:

- When you set `MAPR_JMXLOCALBINDING` to `true`, running `/opt/mapr/server/configure.sh -R` sets `MAPR_JMXAUTH` to `false`, since JMX is only accessible from the local machine and does not require authentication.
- When you set `MAPR_JMXLOCALBINDING` to `false` but set `MAPR_JMXLOCALHOST` to `true`, running `/opt/mapr/server/configure.sh -R` sets `MAPR_JMXAUTH` to `true` and `MAPR_JMXSSL` to `false`, since JMX is only accessible from the local network and does not require secure authentication.
- When you set `MAPR_JMXLOCALBINDING` to `false` but set `MAPR_JMXREMOTEHOST` to `true`, running `/opt/mapr/server/configure.sh -R` sets `MAPR_JMXAUTH` to `true` and `MAPR_JMXSSL` to `true`, since JMX is now accessible remotely and requires secure authentication.

-Z

The `-Z` option is required unless you specify the `-c` (lowercase), or the `-R` option. The `-Z` option takes a comma-separated list of the ZooKeeper nodes in the cluster. The list is in the following format:

```
hostname[:port_no]
[,hostname[:port_no]...]
```

Parameters

-certdomain

Specifies a DNS domain for generated SSL wildcard certificates. This domain overrides the default DNS domain.

--create-user | -a

Creates a local user to run MapR services, using the user specified either with the `-u` parameter, or from the environment variable $MAPR_USER.

-D

Specifies a comma-delimited list of disks to use with the MapR filesystem. With the `-D` option, you cannot specify partitions. By default, the `configure.sh` script automatically starts cluster services, after the configuration finishes successfully. If you do not want cluster services to be restarted, include the `-no-autostart` option along with the `-D` option.

-d

The host and port of the MySQL database to use for storing MapR Metrics data.

-dare

Enables on-disk encryption at the cluster-level. When run on the first CLDB node with the `-genkeys` option, the utility generates the data-at-rest encryption master key file at `/opt/mapr/conf/dare.master.key`.

-defaultdb	Sets the default database (HBase or MapR Database) to which the HBase clients connect. If you do not explicitly configure this option, it defaults to `hbase` (HBase) when you have `mapr-hbase-regionserver` or `mapr-hbase-master` installed on the node. Otherwise, it defaults to `maprdb` (MapR Database). You can also change the database setting using `hbase-site.xml` or the HBase client connection. For more information, see Configure the Default Database for HBase Clients on page 3389.
-disk-opts	Denotes disksetup formatting options. Do not include spaces or commas between the disksetup options. For example, you can specify `-disk-opts FW5` to format the disks (F), and configure five disks per storage pool (W5).
-dp	Specifies the password for logging into the MySQL database used for storing MapR Metrics data.
-ds	Specifies the name of the database schema to use for the MySQL database used for storing MapR Metrics data. The default schema name is *metrics*.
-du	Specifies the username for logging into the MySQL database used for storing MapR Metrics data.

-EC	Specifies a host or hosts that contain the Hive Metastore. Use this parameter and the `-hiveMetastoreHost` argument to configure an ecosystem component, such as Drill, to communicate with the Hive Metastore. Use the following format to specify a list of hosts:
	`hostname[:port_no]`
	`[,hostname[:port_no]...]`
	See the `-EC` parameter example later on this page. If you do not specify a host port number, Drill uses the default Hive Metastore port number (9083) for every host.

-EP	Specifies an option that is passed directly to an ecosystem `configure.sh` script. These commands follow the form `-EP<ecosystem component name> <option>`. In general, `-EP` options are not documented, and should be used only if the documentation specifically instructs you to use them.
	In MapR 6.0 and later, some ecosystem components have their own `configure.sh` scripts. The server `configure.sh` script or a user, can pass options directly to the ecosystem component by using the `-EP` syntax. For example, in the following command:
	`/opt/mapr/server/`
	`configure.sh -R -EPkibana`
	`'-kibanaPort 5610'`
	`-EPkibana '-kibanaPort 5610'` changes the default port for Kibana to 5610.
	As ecosystem components are updated more frequently than MapR Core (which contains the server `configure.sh` script), implementing some

last-updated: Apr 11, 2022

`configure.sh` functions through an ecosystem `configure.sh` script can accelerate the introduction of new features.

`-ES`

Specifies a comma-separated list of host names or IP addresses that identify the Elasticsearch nodes. The Elasticsearch nodes can either be part of the current MapR cluster, or part of a different MapR cluster. Do not use this option when you configure a node for the first time. Use this option along with the `-R` parameter.

The list is in the following format:

```
hostname/IPaddress[:port_no]
[,hostname/IPaddress[:port_no]...]
```

Note: The default Elasticsearch port is `9200`. If you want to use a different port, specify the port number when you list the Elasticsearch nodes.

`-ESDB`

Specifies a non-default location for writing index data on Elasticsearch nodes. To configure an index location, you only need to include this parameter on Elasticsearch nodes.

Elasticsearch requires a lot of disk space. Therefore, a separate filesystem for the index is recommended. It is not recommended to store index data under the `/` or the `/var` file system.

For more information, see Log Aggregation and Storage on page 1391.

`-F`

Specifies a path to a text file that lists the disks and partitions to use with the MapR filesystem. By default, the `configure.sh` script automatically starts cluster services after the configuration finishes successfully. If you do not want cluster services to be restarted, include the `-no-autostart` option along with the `-F` option.

`-f`

Specifies that the node should be configured without performing the system prerequisite check.

`-forceSecurityDefaults`

Instructs `configure.sh` to undo any custom security settings for a cluster, and reconfigure security to the default MapR values for `-unsecure` or `-secure`. You must specify either the `-secure` or the `-unsecure` option. Using the `-forceSecurityDefaults` option removes the `/opt/mapr/conf/.customSecure` file. Use the following syntax:

```
/opt/mapr/server/
configure.sh -forceSecurityDefaults
[ -unsecure | -secure ] -C
<CLDB_node> -Z <ZK_node>
```

For more information, see Customizing Security in MapR on page 1474.

	Important: It is possible that the `-forceSecurityDefaults` operation might not undo all custom security settings since `configure.sh` cannot know all of the custom settings that were implemented. Therefore, you might have to edit some configuration files and settings to restore the cluster to full functionality.
-G	The group ID to use when creating $MAPR_USER with the `-create-user` or `-a` option; corresponds to the `-g` or `-gid` option of the `useradd` command in Linux.
-g	The group name under which MapR services run.
-genkeys	Generates needed keys and certificates for the initial CLDB node in a secure cluster. If specified with the `-dare` option, the `-genkeys` option generates a master key at `/opt/mapr/conf/dare.master.key` on the first CLDB node. Without the master key, you cannot start the cluster, nor can you access the data.
-H	Specifies the HTTPS port number for connecting to the CLDB node. The default port is `7443`.
-HS	Specifies the IP or hostname of the node in the cluster that performs the HistoryServer role. This parameter is required only when a node in the cluster performs the HistoryServer role. In MapR 5.1 and later, this parameter is expanded to support the Mesos DNS-style name with format for Job History. The format is <myriad-fwk-name>.mesos. For example, if the `-MF` parameter is myriadA, the name is: `jobhistory.myriadA.mesos`.
--isvm	Specifies the virtual machine setup. Required when `configure.sh` is run on a cluster node, that is on a virtual machine. This option configures the script to use less memory.
-J	Specifies the JMX port for the CLDB. Default: `7220`
-K | -kerberosEnable	Indicates that Kerberos security has been enabled. Kerberos security is disabled by default.
-L	Specifies a log file. If not specified, `configure.sh` logs errors to `/opt/mapr/logs/configure.log`.
--logHTTPFS	Specifies the hostname to enable centralized logging using `fluentd`.
-MCL	Specifies the top-level directory where all the staging data as well as shuffle data is written for a specific Myriad framework. Used when multiple clusters are implementing Myriad.
-MP	Specifies the name of the Myriad framework that is displayed in the Mesos UI.
-MHA	Enables Myriad high availability.
-M7	Deprecated as of MapR 4.0.1.
-maprpam	When specified, the `configure.sh` script installs the MapR version of Pluggable Authentication Modules (PAM). This option is ignored if `-S` is not set.

last-updated: Apr 11, 2022

-N	Specifies the cluster name. If you do not specify a name, `configure.sh` applies a default name (my.cluster.com) to the cluster. Whenever you run `configure.sh`, be aware of the existing cluster name or names in `mapr-clusters.conf` and specify the `-N` parameter accordingly. If you specify a name that does not exist, a new line is created in `mapr-clusters.conf` and is treated as a configuration for a separate cluster. Subsequent runs of `configure.sh` without the `-N` parameter operate on this default cluster. If you specify a name when you first run `configure.sh`, you can modify the CLDB and ZooKeeper settings corresponding to the named cluster by specifying the same name and running `configure.sh` again. Whenever you need to re-run `configure.sh` on a given cluster (to add or rename nodes, for example), be sure to specify the same cluster name that you used when you ran `configure.sh` for the first time.
-no-autostart	Specifies that the script should not start Zookeeper or Warden when you run `configure.sh`.
-no-auto-permission-update	Pass this option to prevent MapR from silently altering permissions in `/etc/shadow`.
-nocerts	When specified, the `configure.sh` script does not generate SSL certificates even when the `-genkeys` option is specified.
-noDB	Specifies that MapR Database is not in use.
-noRecalcMem	Skips recalculating memory settings when refreshing roles. Can be used only with the `-R` option.
-OT	Specifies a comma-separated list of host names or IP addresses that identify the OpenTSDB nodes. The OpenTSDB nodes can be part of the current MapR cluster or part of a different MapR cluster. Do not use this option when you configure a node for the first time. Use this option along with the `-R` parameter. The Warden service must be running when you use `configure.sh -R -OT`. Use the following format to list the hostnames: `hostname/IP address[:port_no] [,hostname/IP address[:port_no]...]` **Note:** The default OpenTSDB port is `4242`. If you want to use a different port, specify the port number when you list the OpenTSDB nodes.
-on-prompt-cont	Specify: • y to automatically respond Yes to all prompts. • n to automatically respond No to all prompts.
-P	Specifies the Kerberos instance that is used to form a CLDB Kerberos principal in the form of `mapr/<instance-name>@<realm-name>`. Enclose this value in quotes ("). This value is ignored if Kerberos security is not enabled.

last-updated: Apr 11, 2022

-QS	Use the `-QS` option to configure the OJAI Distributed Query Service. See Configure the OJAI Distributed Query Service on page 182.
-RM	In MapR 5.1, this parameter is expanded to support the Mesos DNS-style hostname for Myriad configuration. The Mesos-style hostname is `<application name>.marathon.mesos`. When starting ResourceManager from Marathon, the `.<application name> rm`, for example, is `rm.marathon.mesos`. In MapR 4.0.2, this parameter is not required unless you want to configure manual or automatic failover; zero configuration failover is enabled by default. In MapR 4.0.1, this parameter specifies the nodes in the cluster with the ResourceManager role. List the nodes in the following format: `hostname[,hostname]...]` For more information, see ResourceManager High Availability on page 1508.
-S | -secure	Specifies that this cluster is a secure cluster, and configures security on the platform and on all ecosystem components that support security. Default: `insecure`.
-syschk	Configures the system checks to be enabled or disabled. Value: Y/N.
-TL	Specifies the single node on which the timeline server is installed for the Hive-on-Tez user interface. When you install Tez manually, you must also install the timeline server and run `configure.sh -TL <timeline_server_node>` on all nodes to indicate where the timeline server resides.
-U	The user ID to use when creating $MAPR_USER with either the `--create-user` or `-a` option; corresponds to the `-u` or `--uid` option of the `useradd` command in Linux.
-u	The user name under which MapR services run.
-unsecure	Specifies that this cluster is not secure. Default: `unsecure`.
-v	In addition to logging information, also prints to `stdout`.

Examples

1. **Add a node (not CLDB or ZooKeeper) to a cluster that is running the CLDB and ZooKeeper on three nodes:**

   On the new node, run the following command:

   ```
   /opt/mapr/server/configure.sh -C nodeA,nodeB,nodeC -Z nodeA,nodeB,nodeC
   ```

2. **Configure a client to work with cluster my.cluster.com, which has one CLDB at nodeA:**

   On a Linux client, run the following command:

   ```
   /opt/mapr/server/configure.sh -N my.cluster.com -c -C nodeA
   ```

   On a Windows 7 client, run the following command:

   ```
   C:\opt\mapr\server\configure.bat -N my.cluster.com -c -C nodeA
   ```

3. **Add a second cluster to the configuration:**

   On a node in the second cluster your.cluster.com, run the following command:

   ```
   /opt/mapr/server/configure.sh -C nodeZ -N your.cluster.com -Z
   <zkNodeA,zkNodeB,zkNodeC>
   ```

4. **Add CLDB servers with multiple IP addresses to a cluster:**

   In this example, the cluster my.cluster.com has CLDB servers at nodeA, nodeB, nodeC, and nodeD. The CLDB servers nodeB and nodeD have two NICs each at *eth0* and *eth1*.

   On a node in the cluster `my.cluster.com`, run the following command:

   ```
   /opt/mapr/server/configure.sh -N my.cluster.com -C
   nodeAeth0,nodeCeth0 -M \
            nodeBeth0,nodeBeth1 -M nodeDeth0,nodeDeth1 -Z zknodeA
   ```

5. **Start a cluster in secure mode using configure.sh**

   In this example, the cluster `my.cluster.com` has two CLDB servers at nodeA and nodeB. The ZooKeeper node for this cluster is at nodeC. To start the cluster in secure mode, run the following command on nodeA:

   ```
   /opt/mapr/server/configure.sh -N my.cluster.com -C nodeA,nodeB -Z
   nodeC -secure \
            -genkeys -F <disklist file>
   ```

   This command creates the `ssl_truststore`, `ssl_keystore`, `maprserverticket`, and `cldb.key` files. Copy these files from nodeA's `/opt/mapr/conf` directory to nodeB's `/opt/mapr/conf` directory.

   On nodeB, change the permissions on the `ssl_keystore`, `maprserverticket`, and `cldb.key` files to 600 (the `mapr` user) by using the following command:

   ```
   chmod 600 ssl_keystore maprserverticket cldb.key
   ```

   On the `ssl_truststore` file, change the permissions to 644 (world readable):

   ```
   chmod 644 ssl_truststore
   ```

   On nodeB, run the following command:

   ```
   /opt/mapr/server/configure.sh -N mycluster.com -C nodeA,nodeB -Z
   nodeC -secure -F \
            <disklist file>
   ```

6. **Configure the Drill storage plugin to communicate with a Hive Metastore:**

   This example uses the `-EC` parameter to configure the Drill storage plugin to communicate with a Hive Metastore located on `nodeA`:

   ```
   /opt/mapr/server/configure.sh -EC '-hiveMetastoreHost nodeA'
   ```

7. **Configure HSM:**

   A sample session transcript using the `/opt/mapr/server/configure.sh` script with DARE enabled is as follows. The portions in **bold** relate to the common HSM features, while the portions in *italics* relate to the DARE-specific features:

```
# /opt/mapr/server/configure.sh  -secure -genkeys -N
test96.cluster.com -C perfnode96.lab:7222 -Z perfnode96.lab:5181 -F
disks.txt -dare -hsm -hsmip 10.10.30.129 -hsmlabel "SafeNet
KeySecure" -hsmsopin 12345678 -hsmclientcert /root/safenet-keysecure/
client.pem -hsmcacert /root/safenet-keysecure/CA.pem -hsmclientkey /root/
safenet-keysecure/key.pem
create /opt/mapr/conf/conf.old
CLDB node list: perfnode96.lab:7222
Zookeeper node list: perfnode96.lab:5181
External Zookeeper node list:
Node setup configuration:  cldb fileserver hadoop-util zookeeper
Log can be found at:  /opt/mapr/logs/configure.log
Initializing HSM with label SafeNet KeySecure
Generated random user PIN B$V5g%$2#%8Kc6SL
Obtained cluster name test96.cluster.com from mapr-clusters.conf
Enabling MapR HSM on cluster test96.cluster.com
Successfully generated Core KEK, UUID
CF9FE63E85EF233B583972FB6265DB33067E8DBBB300297FF8F562DFCF7EA904
Successfully generated Common KEK, UUID
32A903E6D0DF67FDBCD953A33FC2547F50D35C18666E2A0A0B5CF749FBF84D6A
Successfully set encrypted CLDB key in KMIP configuration
Successfully set encrypted DARE key in KMIP configuration

################################################################################
#####
# NOTE: The DARE master key for data at rest encryption is protected by
the  #
# HSM. All keys in the HSM, including the DARE master key, should be
safely  #
# backed up. Without the DARE master key, cluster cannot be started and
data #
# cannot be
accessed.                                                              #
################################################################################
#####

Creating 100 year self signed certificate with subjectDN='CN=*.lab'
Configuring hadoop-util
/dev/sdb added.
/dev/sdc added.
/dev/sdd added.
Zookeeper found on this node, and it is not running. Starting Zookeeper
Warden is not running. Starting mapr-warden. Warden will then start all
other configured services on this node
... Starting cldb
... Starting fileserver
... Starting hadoop-util
To further manage the system, use "maprcli", or connect browser to
https://{webserver host name}:8443/
To stop and start this node, use "systemctl start/stop mapr-warden "
No need to set label returning from SetDiskLabel
```

Troubleshooting configure.sh

When you run `configure.sh` with the `-OT` option for the first time, you might encounter an error message such as **directory /opt/mapr/conf/proxy is not owned by root**. You must ignore this transient error

message. If you repeatedly see this error during client operations, then re-run `configure.sh` with the `-R` option.

Related concepts

node on page 1694
Manages nodes in the cluster

Related reference

disk add on page 1602
Adds one or more disks to the specified node. Permissions required: `fc` or `a`.

volume create on page 1931
Creates a volume.

node list on page 1705
Lists nodes in the cluster.

configure-crosscluster.sh

Use the `configure-crosscluster.sh` utility to set up cross-cluster security between two clusters.

You can use the `configure-crosscluster.sh on page 2065` utility to set up cross-cluster security between two clusters. When you run this utility with the `create` subcommand, it establishes security between the local cluster and a remote cluster. After the setup, communication between the two clusters is bi-directional. You can run this utility on any node in the source cluster to grant secure access to users and servers (for replication, or mirroring) on the destination cluster.

The utility prompts for the passwords for both the local and remote clusters. All hosts on a cluster must have the same password. Alternatively, you can use `ssh` public key authentication between the current node and the other nodes in the local and remote clusters.

Prerequisites

Before running this utility, you must:

- Enable wire-level security for both the local and remote clusters.

  Set `secure=true` in your cluster entry for `/opt/mapr/conf/mapr-clusters.conf`, for both the local and remote clusters.

- Install the `pssh` (Parallel SSH) package from EPEL.

- Install the `expect` package.

If you plan to use a user other than the `mapr` administrative user for mirroring or gateway/streams replication, that user must already exist on both the local and remote clusters.

Syntax

```
/opt/mapr/server/configure-crosscluster.sh create <cross-cluster-type>
    [ -localcrossclusteruser <user> ]
    [ -localhosts <path_to_file> ]
    [ -localport <port_number> ]
    [ -localuser <user> ]
    [ -recover <id> ]
    [ -remotecrossclusteruser <user> ]
    [ -remotehosts <port_number> ]
    -remoteip <ip_address>
    [ -remoteport <port_number> ]
    [ -remoteuser <user> ]
```

Type Parameters

The <cross-cluster-type> parameter specifies the type of entity for which cross-cluster access must be established. The value can be one of the following:

user

Used for direct data access for the given user. When you run the utility with the `user` parameter, it performs the following tasks on both the clusters:

1. Updates the `/opt/mapr/conf/mapr-clusters.conf` file to include the first entry from the `/opt/mapr/conf/mapr-clusters.conf` file on the other cluster.
2. Imports the certificate of the other cluster in the `/opt/mapr/conf/ssl_truststore` file, and copies the updated `/opt/mapr/conf/ssl_truststore` file to all the other nodes on the cluster.

server

Used for MapR server access such as mirroring and replication. When you run the utility with the `server` parameter, it performs the following tasks on both the clusters:

1. Generates a cross-cluster ticket on this cluster for the other cluster, and copies the ticket to the CLDB node on the other cluster.
2. Merges the ticket with the `/opt/mapr/conf/maprserverticket` file on the node on the other cluster, and copies the updated `/opt/mapr/conf/maprserverticket` file to all the other CLDB nodes on the other cluster.

all

Used for both user and server access. When you run the utility with the `all` parameter, it performs the following actions on both the clusters:

1. Updates the `/opt/mapr/conf/mapr-clusters.conf` file to include the first entry from the `/opt/mapr/conf/mapr-clusters.conf` file on the other cluster.
2. Imports the certificate of the other cluster in the `/opt/mapr/conf/ssl_truststore` file, and copies the updated `/opt/mapr/conf/ssl_truststore` file to all the other nodes on the cluster.
3. Generates a cross-cluster ticket for the other cluster, copies the ticket to the CLDB node on the other cluster, merges the ticket with the `/opt/mapr/conf/maprserverticket` file on the node in the other cluster, and copies the updated `/opt/mapr/conf/maprserverticket` file to all other CLDB nodes on the other cluster.

Options

The configure-crosscluster.sh on page 2065 utility supports the following options:

localcrossclusteruser

Default value: `local user`

This option applies only to the server parameter. Specifies the name of the local cross-cluster user if different from the local user, for mirroring and replication of tables, and streams.

localhosts

Default value: `No Default Value`

Contains the full or relative path to the file containing the list of IP addresses or host names of the hosts to update in the local cluster. Specify one host per line in the file, excluding the current host. If you specify this option, the utility updates the configuration, both on the host on which you are running the utility, and on the other nodes specified in the file. If you do not specify this option, the utility copies both the:

- Updated server security configuration in the `/opt/mapr/conf/maprserverticket` file to only the CLDB nodes in the local cluster.
- Updated user security configuration in the `/opt/mapr/conf/mapr-clusters.conf` file, and the `/opt/mapr/conf/ssl_truststore` file, to all the nodes in the local cluster.

localport

Default value: `22`

Indicates the port to use to connect (using `ssh` or `scp`) to local cluster hosts.

localuser

Default value: `mapr`

Specifies the name of the user for the local cluster.

recover

Default value: `No Default Value`

Defines the option to recover from the failure to copy files to nodes in the local or remote cluster, due to failed cluster nodes. Use the special ID keyword `latest` to run with the contents of the most recent run. See Troubleshooting and Recovery on page 2071 for more information.

remotecrossclusteruser

Default value: `remote user`

This option applies only to the server parameter. Specifies the name of the remote cross-cluster user, if different from the remote user.

remotehosts

Default value: `No Default Value`

Contains the full or relative path to the file containing the list of IP addresses or host names of the hosts to update in the remote cluster. Specify one host per line in the file. If you do not specify this option, the utility copies both the:

- Updated server security configuration in the `/opt/mapr/conf/maprserverticket` file to only the CLDB nodes in the remote cluster.
- Updated user security configuration in the `/opt/mapr/conf/mapr-clusters.conf` file, and the `/opt/mapr/conf/ssl_truststore` file, to all the nodes in the remote cluster.

For example, if you have a file `myhosts.txt` in the current directory with the following contents:

```
10.10.20.100
10.10.20.101
10.10.20.102
```

then, specify `-remotehosts myhosts.txt` for this parameter.

Attention: All hosts specified in this file must be directly reachable from the local hosts from which you run the `configure-crosscluster.sh` on page 2065 utility.

remoteip

Default value: `No Default Value`

This option is mandatory. Specifies the host name or IP address of a host in the remote cluster.

remoteport

Default value: `22`

Indicates the port to use to connect (using `ssh` or `scp`) to remote cluster hosts.

remoteuser

Default value: `local user`

Designates the name of the user for the remote cluster.

Verification

To verify that cross-cluster security is correctly set up, perform one of the following actions:

- If you ran the utility using the cross-cluster type `user` or `all`, and the utility completed successfully, you should be able to run remote commands from the local node after obtaining a user ticket using the `maprlogin` on page 2130 utility. See Configuring Secure Clusters for Running Commands Remotely on page 1484 for more information.
- If you ran the utility using the cross-cluster type `server` or `all`, and the utility completed successfully, you should be able to perform various service operations from the local to the remote cluster and vice versa, including mounting volumes over NFS, mirroring volumes, and replicating tables and streams. See Configuring Secure Clusters for Cross-Cluster Mirroring and Replication on page 1486 for more information.

Sample Session

To configure cross-cluster security, run the utility on a CLDB host with wire-level security enabled, :

```
# /opt/mapr/server/configure-crosscluster.sh create all -remoteip
10.10.30.96
Remote IP is 10.10.30.96
WARNING: Strict host key checking will be disabled for this script.
Local user unset, defaulting to mapr
Remote user unset, defaulting to local user mapr
Enter password for mapr user (mapr) for local cluster:
Enter password for mapr user (mapr) for remote cluster:
Local cross-cluster user unset, defaulting to local user mapr
Remote cross-cluster user unset, defaulting to remote user mapr
Verifying connectivity to 10.10.30.96 and presence of mapr-clusters.conf
MapR credentials of user 'mapr' for cluster 'myCluster.cluster.com' are
written to '/tmp/maprticket_0'
Local host is running the CLDB
```

```
chyelin101.cluster.com secure=true qa-cnode101.lab:7222
Configuring cross-cluster communication for users
Certificate stored in file </tmp/mapr-xcs/29668/local_mapcert>
Certificate stored in file </tmp/mapr-xcs/29668/remote_mapcert>
Successfully exported certificate for remote cluster to /tmp/mapr-xcs/29668/
remote_mapcert
Certificate was added to keystore
Certificate was added to keystore
Configuring cross-cluster communication for server-side operations
Generating cross-cluster ticket for user mapr on remote node
Generating cross-cluster ticket for mirroring for user mapr
MapR credentials of user 'mapr' for cluster 'myCluster.cluster.com' are
written to '/tmp/mapr-xcs/29668/local_crosscluster_ticket'
SUCCESS
This script has logged in to both the local and remote clusters. Please log
out of
the clusters if needed.
```

Cleanup

After running the utility, you must perform two cleanup actions:

1. Log out of the local and remote clusters if needed, using the `maprlogin logout` command.
2. Delete the `/tmp/mapr-xcs` directory, if it is present, after verifying that the cross-cluster setup is correct.

   If you run this utility without the `-recover` option, the utility creates temporary files in the `/tmp/mapr-xcs` directory under the current process ID. These directories contain sensitive information such as server tickets that are protected by Unix permissions. The utility preserves these tickets, so that you can perform troubleshooting and recovery actions, as needed. You must delete this directory after verifying that the cross-cluster setup is correct:

   ```
   $ /bin/rm -rf /tmp/mapr-xcs
   ```

Post-Configuration Tasks

After you run this utility, cross-cluster security should be successfully set up between the local and the remote cluster. If you specified either the `user` or `all` cross-cluster type when running the utility, to perform any operations on the remote cluster from the local node, login to the remote cluster to obtain a user ticket using the `maprlogin` `on page 2130` command.

```
$ maprlogin password -cluster <remote-cluster-name>
```

Note: You must obtain a new ticket when your current ticket expires.

Examples of Using the Cross-Cluster Utility

For examples on how to run the `configure-crosscluster.sh` `on page 2065` utility, see Configure-crosscluster.sh Examples on page 2069.

Related information

Configure-crosscluster.sh Examples on page 2069
Demonstrates how to use the `configure-crosscluster.sh` utility.

Troubleshooting and Recovery on page 2071

Configure-crosscluster.sh Examples

Demonstrates how to use the `configure-crosscluster.sh` utility.

last-updated: Apr 11, 2022

Example 1

Suppose both the local and remote cluster administrator usernames are `mapr`, and both the local and remote cross-cluster users for mirroring and gateway/streams replication are also `mapr`, specify only the `-remoteip` argument for a fresh run:

```
$ /opt/mapr/server/configure-crosscluster.sh create server -remoteip
10.10.1.1
```

Example 2

Assume that the local MapR administrator username defaults to `mapr`, and the remote MapR administrator username defaults to the local MapR administrator username. Specify different user names for the local and remote MapR administrator using the `-localuser` and `-remoteuser` arguments. For example, if the local MapR administrator username is `admin` and the remote MapR administrator username is `mapr`:

```
$ /opt/mapr/server/configure-crosscluster.sh create server -remoteip
10.10.1.1 -localuser admin -remoteuser mapr
```

Example 3

Assume that the local cross-cluster user defaults to the local MapR administrative user, and the remote cross-cluster user defaults to the remote MapR administrative user. To use a different cross-cluster user for mirroring or gateway/streams replication, specify the `-localcrossclusteruser` and/or `-remotecrossclusteruser` parameters. For example, if the local cross-cluster username is `crosscluster`, run the utility as follows:

```
$ /opt/mapr/server/configure-crosscluster.sh create server -remoteip
10.10.1.1 -localcrossclusteruser crosscluster
```

Example 4

By default, the utility performs `ssh` and `scp` operations between the node where the utility is running and the other nodes in the local and remote clusters, using the default SSH port 22. To use a non-default SSH port, either for the local or remote clusters, specify the port number using the `-localport` or the `-remoteport` option. For example, if the SSH port for the local cluster is 10022, run the utility as follows. The remote SSH port is the default value of 22 if the `-remoteport` argument is not specified:

```
$ /opt/mapr/server/configure-crosscluster.sh create server -remoteip
10.10.1.1 -localport 10022
```

Example 5

By default, the utility runs the `maprcli node list` command on both the local and remote nodes to determine the list of hosts in the local and remote clusters, and updates the configuration for all the nodes in the local and remote clusters. To update the configuration only for a subset of nodes in either the local or remote cluster, such as when you want to update only the CLDB nodes, specify the path to the file containing the list of hosts, one per line, using the `-localhosts` and `-remotehosts` options respectively. For example, to update the configuration for only the local nodes specified in the file `/tmp/local` and the remote nodes specified in the file `/tmp/remote`, run:

```
$ /opt/mapr/server/configure-crosscluster.sh create server -remoteip
10.10.1.1 -localhosts /tmp/local -remotehosts /tmp/remote
```

Example 6

To configure cross-cluster functionality for a user without setting up server cross-cluster functionality, specify `user` as the parameter. This parameter allows users to run commands such as `maprcli node list` using the `-cluster` parameter, and the remote cluster name:

```
$ /opt/mapr/server/configure-crosscluster.sh create user -remoteip 10.10.1.1
```

Example 7

To configure both user and server cross-cluster functionality, specify `all` as the parameter, instead of user or server:

```
$ /opt/mapr/server/configure-crosscluster.sh create all -remoteip 10.10.1.1
```

Example 8

To copy the configuration files from the most recently failed run, use the `-recover` option. To update the configuration for a specified list of local or remote hosts, use the `-recover` option together with the `-localhosts` and `-remotehosts` options.

```
$ /opt/mapr/server/configure-crosscluster.sh create server -remoteip
10.10.1.103 -localuser admin -remoteuser mapr -recover latest -localhosts
local -remotehosts remote
```

See Sample Failure, Troubleshooting, and Recovery Session on page 2075 for more information.

Troubleshooting and Recovery

Typically, the utility succeeds and if the utility fails partially or completely, try the troubleshooting and recovery steps described here.

Completely Failed Runs

A completely failed run indicates that the utility did not set up cross-cluster communication between any of the local or remote nodes. Typical reasons for complete failure include:

- The prerequisites for running thes utility were not met.

  Refer to Prerequisites on page 2065 for running this utility.

- The utility was not run as `mapr` or administrative user.

  The user running the utility must be able to run commands like `maprlogin password`, `maprlogin generateticket`, and `maprcli node list`.

- The `-localuser` option was not specified when there was a non-default username (like `admin`) for the `mapr` user on the local node.

- The `-remoteuser` option was not specified when there was a non-default username (like `admin`) for the `mapr` user on the remote node.

- The username specified in the `-localcrosscluster` option does not exist in the local cluster.

  This utility requires that the username specified in the `-localcrossclusteruser` option to exist before running the utility.

- The username specified in the `-remotecrosscluster` option does not exist in the remote cluster.

  This utility requires that the username specified in the `-remotecrossclusteruser` option to exist before running the utility.

- The wrong password was specified for the local `mapr` user for the local cluster.

  This utility uses commands like `ssh` and `scp` to access other nodes in the local cluster. So, if the public key authentication between the local node and the other nodes in the local cluster is not setup, you must have set a password for the mapr administrative user specified in the `-localuser` argument.

- The wrong password was specified for the remote `mapr` user for the remote cluster.

  This utility uses commands like `ssh` and `scp` to access other nodes in the local cluster. So, if the public key authentication between the local node and the node specified in the `-remoteip` option is not setup, there must be a password for the `mapr` administrative user specified in the `-remoteuser` option.

For completely failed runs, examine the log file in `/opt/mapr/logs/crosscluster.log` and the output of the latest run in `/tmp/mapr-xcs` directory. The `crosscluster.log` looks something like the following:

```
Script started at Fri Sep 29 15:56:33
PDT 2017
Entering recovery mode. Using
cross-cluster directory /tmp/mapr-xcs/
13194
Verifying that pssh is present ... ok
Verifying that expect is present ...
ok
Verifying that trust store is
present ... ok
Verifying that cluster file is
present and cluster name is
set ... clustername is set to
node95.cluster.com
ok
Verifying that security is
configured ...
Verifying that keytool exists ... ok
Verifying that local user exists ...
ok
Verifying that local cross-cluster
```

```
user exists ... ok
Verifying that remote cross-cluster
user exists ... ok
Verifying connectivity to 10.10.1.103
and presence of mapr-clusters.conf
Running command: ssh -o
StrictHostKeyChecking=no -o
UserKnownHostsFile=/dev/null
 -o GlobalKnownHostsFile=/dev/null -o
LogLevel=quiet -p 22 mapr@10.10.1.103
ls /
opt/mapr/conf/mapr-clusters.conf
Logging in to local cluster
...
```

Partially Failed Runs

The utility may also report partial success. The utility does not fail due to inability to copy the updated files to the nodes in the local or remote clusters, so the most likely cause of partial failure is improperly configured or failed cluster nodes.

For partially failed runs, examine the contents of the latest run in `/tmp/mapr-xcs` directory in addition to the contents of the `/opt/mapr/logs/crosscluster.log` file.

In the following example, the latest run of the utility is in `/tmp/mapr-xcs/13194`, since this directory has the most recent modification date:

```
[admin@node95 ~]# ls -lt /tmp/mapr-xcs
total 8
drwx------ 14 mapr mapr 4096 Sep 29
15:49 13194
drwx------ 30 mapr mapr 4096 Sep 29
14:44 23802
```

The following is the sample content in `/tmp/mapr-xcs/13194`:

```
[admin@node95 mapr-xcs]# ls -l 13194
total 52
-rw-r--r-- 1 mapr mapr  59 Sep 29
15:49 local_clusterentry
-rw-r--r-- 1 mapr mapr  90 Sep 29
15:49 localclusterhosts_full.txt
-rw-r--r-- 1 mapr mapr  12 Sep 29
15:56 localclusterhosts.txt
-rw------- 1 mapr mapr 315 Sep 29
15:49 local_crosscluster_ticket
-rw-r--r-- 1 mapr mapr 299 Sep 29
15:49 local_maprserverticket_entries
drwxr-xr-x 2 mapr mapr  58 Sep 29
15:49 lpscp_clusterhosts_edir
drwxr-xr-x 2 mapr mapr  44 Sep 29
15:49 lpscp_clusterhosts_odir
drwxr-xr-x 2 mapr mapr  58 Sep 29
15:49 lpscp_server_edir
drwxr-xr-x 2 mapr mapr  44 Sep 29
15:49 lpscp_server_odir
drwxr-xr-x 2 mapr mapr  58 Sep 29
15:49 lpssh_server_edir
```

```
drwxr-xr-x 2 mapr mapr  44 Sep 29
15:49 lpssh_server_odir
-rw-r--r-- 1 mapr mapr 115 Sep 29
15:49 remote_clusterconf
-rw-r--r-- 1 mapr mapr  56 Sep 29
15:49 remote_clusterentry
-rw-r--r-- 1 mapr mapr 138 Sep 29
15:49 remoteclusterhosts_full.txt
-rw-r--r-- 1 mapr mapr  12 Sep 29
15:56 remoteclusterhosts.txt
-rw------- 1 mapr mapr 320 Sep 29
15:49 remote_crosscluster_ticket
-rw------- 1 mapr mapr 914 Sep 29
15:49 remote_maprserverticket
-rw-r--r-- 1 mapr mapr 300 Sep 29
15:49 remote_maprserverticket_entries
drwxr-xr-x 2 mapr mapr  58 Sep 29
15:49 rpscp_clusterhosts_edir
drwxr-xr-x 2 mapr mapr  44 Sep 29
15:49 rpscp_clusterhosts_odir
drwxr-xr-x 2 mapr mapr  58 Sep 29
15:49 rpscp_server_edir
drwxr-xr-x 2 mapr mapr  44 Sep 29
15:49 rpscp_server_odir
drwxr-xr-x 2 mapr mapr  58 Sep 29
15:49 rpssh_server_edir
drwxr-xr-x 2 mapr mapr  44 Sep 29
15:49 rpssh_server_odir
-rw-r--r-- 1 mapr mapr   2 Sep 29
15:56 STATUS
```

Troubleshooting

1. Look at `/tmp/mapr-xcs/<latest>/STATUS`. A non-zero value indicates an overall error.
2. If you encounter a non-zero overall status, look at the `STATUS` files in each of the subdirectories.
3. For the subdirectories reporting a non-zero status, look at the contents of the files in that subdirectory.
4. If there is an error in updating the local cluster hosts, and you did not use the `-localhosts` option when running the script, also look at `/tmp/mapr-xcs/<latest>/localclusterhosts.txt` file.

   The `/tmp/mapr-xcs/<latest>/localclusterhosts.txt` file contains the list of IP addresses of the local cluster hosts, which is the first IP address of each node if there are multiple IP addresses, obtained from the following command:

   ```
   maprcli node list -cluster <local-cluster-name> -columns hostname
   ```

   Verify the contents of the file to ensure that the list of local cluster hosts is correct. The original output of the above command is in `/tmp/mapr-xcs/<latest>/localclusterhosts_full.txt`, and you should also check the output to ensure that the list is correct. Otherwise, fix the errors and re-run the script with the `-localhosts` option.

5. If there is an error in updating the remote cluster hosts, and you did not use the `-remotehosts` option when running the script, you should also look at `/tmp/mapr-xcs/<latest>/remoteclusterhosts.txt` file.

   The `/tmp/mapr-xcs/<latest>/remoteclusterhosts.txt` file contains the list of IP addresses of remote cluster hosts, which is the first IP address of each node if there are multiple IP addresses, obtained from the following command:

   ```
   ssh <remoteuser>@<remote-ip> maprcli node list -cluster
   <remote-cluster-name> -columns hostname
   ```

   Verify the contents of the file to ensure that the list of remote cluster hosts is correct. The original output of the above command is in `/tmp/mapr-xcs/<latest>/remoteclusterhosts_full.txt`, and you should also check the output to ensure that the list is correct. Otherwise, fix the errors and re-run the script with the `-remotehosts` option.

6. If you have an error copying to some or all of the local or remote cluster hosts, try doing an `ssh` to the local or remote cluster host (respectively) to ensure that it is accessible using the supplied username and password. If this fails, the copy operation in the script will also fail, since it relies on either public key authentication or username/password authentication to access the nodes. Specify the correct username and/or password and then re-run the script.

Sample Failure, Troubleshooting, and Recovery Session

Suppose the utility is run where one of the nodes in the local cluster (10.10.30.96) has a password that is different from other local cluster nodes, causing the `ssh` and `scp` commands to this node to fail.

1. Run the utility on the local node (10.10.30.95).

   The highlighted text below are the warning messages. The utility continues to run, despite the warnings, to update the cross-cluster configuration on as many nodes as possible:

```
[admin@node95 cross-cluster]$ /opt/mapr/server/configure-crosscluster.sh
create server -remoteip 10.10.1.103 -localuser admin -remoteuser mapr
Remote IP is 10.10.1.103
WARNING: Strict host key checking will be disabled for this script.
Enter password for mapr user (admin) for local cluster:
Enter password for mapr user (mapr) for remote cluster:
Local cross-cluster user unset, defaulting to local user admin
Remote cross-cluster user unset, defaulting to remote user mapr
Verifying connectivity to 10.10.1.103 and presence of mapr-clusters.conf
MapR credentials of user 'admin' for cluster 'node95.cluster.com' are
written to '/tmp/maprticket_0'
node95.cluster.com secure=true node95.perf.lab:7222
WARNING: Copying local /opt/mapr/conf/mapr-clusters.conf to all hosts in
local cluster complete, but the operation failed for at least one node
in the cluster.
For details, look at the output directory /tmp/mapr-xcs/
14043/lpscp_clusterhosts_odir or error directory /tmp/mapr-xcs/14043/
lpscp_clusterhosts_edir.
Configuring cross-cluster communication for server-side operations
Generating cross-cluster ticket for user mapr on remote node
WARNING: Changing permissions of local maprserverticket complete, but
the operation failed for at least one node in the cluster.
For details, look at the output directory /tmp/mapr-xcs/
14043/lpssh_server_odir or error directory /tmp/mapr-xcs/14043/
lpssh_server_edir
WARNING: Cannot change permissions for local MapR server ticket for at
least 1 node
WARNING: Copy local maprserverticket to all hosts in local cluster
complete, but the operation failed for at least one node in the cluster.
For details, look at the output directory /tmp/mapr-xcs/
14043/lpscp_server_odir or error directory /tmp/mapr-xcs/14043/
lpscp_server_edir.
WARNING: Cannot copy local MapR server ticket for at least 1 node
Generating cross-cluster ticket for mirroring for user admin
MapR credentials of user 'admin' for cluster 'node95.cluster.com' are
written to '/tmp/mapr-xcs/14043/local_crosscluster_ticket'
An error has been encountered in configuring cross-cluster communication.
For more information, refer to the log file at /opt/mapr/logs/
crosscluster.log.
If the error is caused by non-functioning local and remote cluster
nodes, more information on the precise errors can be found in /tmp/
mapr-xcs/14043. The list of local cluster hosts is in /tmp/mapr-xcs/
14043/localclusterhosts.txt, and the list of remote cluster hosts is
in /tmp/mapr-xcs/14043/remoteclusterhosts.txt.
In such cases, you can normally fix the error by editing the list
of local and/or remote cluster hosts file and then re-run the script
using the -r option, specifying the local or remote hosts file in
the -localhosts or -remotehosts option respectively.
This script has logged in to both the local and remote clusters. Please
log out of the clusters if needed.
```

2. Look at the specified directory, `/tmp/mapr-xcs/14043`, because the utility resulted in an error.

   The overall status is 1 (indicating an error) as shown in bold below:

```
$ cat /tmp/mapr-xcs/14043/STATUS
1
```

3. Look at the STATUS files in each of the subdirectories to determine the content reporting error.

   Content has non-zero status as shown in bold below:

```
[admin@node95 14043]$ find /tmp/mapr-xcs/14043 -print | grep STATUS |
xargs more
::::::::::::::
./rpscp_clusterhosts_edir/STATUS
::::::::::::::
0
::::::::::::::
./lpscp_clusterhosts_edir/STATUS
::::::::::::::
1               ←FAIL
::::::::::::::
./lpssh_server_edir/STATUS
::::::::::::::
1               ←FAIL
::::::::::::::
./lpscp_server_edir/STATUS
::::::::::::::
1               ←FAIL
::::::::::::::
./rpssh_server_edir/STATUS
::::::::::::::
0
::::::::::::::
./rpscp_server_edir/STATUS
::::::::::::::
0
::::::::::::::
./STATUS
::::::::::::::
1               ←Overall status is FAIL
```

4. Look at each of the files in the subdirectories reporting a non-zero status, for example, `lpscp_clusterhosts_edir`.

   The error “lost connection” for 10.10.30.96 indicates that the local node 10.10.30.95 could not run the `scp` command to that node:

```
[admin@node95 14043]$ more lpscp_clusterhosts_edir/*
::::::::::::::
lpscp_clusterhosts_edir/10.10.30.95
::::::::::::::
::::::::::::::
lpscp_clusterhosts_edir/10.10.30.96
::::::::::::::
lost connection
::::::::::::::
lpscp_clusterhosts_edir/STATUS
::::::::::::::
1
```

5. Try to `ssh` to that node (10.10.30.96), using the same local password used for running the utility.

   The `ssh` (and therefore also `scp`) command fails:

```
[admin@node95 14043]$ ssh admin@10.10.30.96 ls
Password:
Password:
Password:
admin@10.10.30.96's password:
Permission denied, please try again.
admin@10.10.30.96's password:
Received disconnect from 10.10.30.96: 2: Too many authentication
failures for admin
```

6. Run the utility again.

   The utility detects that the previous run did not complete successfully and prompts you to run the utility with the recovery option. It also detects the directories that contain the detailed error information as shown below:

```
[admin@node95 cross-cluster]$ /opt/mapr/server/configure-crosscluster.sh
create server -remoteip 10.10.1.103 -localuser admin -remoteuser mapr
Remote IP is 10.10.1.103
WARNING: Strict host key checking will be disabled for this script.
The previous run of this script with ID 14043 did not complete
successfully. Examine the error directories in /tmp/mapr-xcs/14043
for details of the error:
/tmp/mapr-xcs/14043/lpscp_clusterhosts_edir
/tmp/mapr-xcs/14043/lpscp_server_edir
/tmp/mapr-xcs/14043/lpssh_server_edir
If the failure is down to partially failed nodes, you should exit now,
and re-run this script in recovery mode using the -recover option to
copy the configured tickets and files to the remaining nodes, instead of
continuing and generating new tickets.
Exit now? Enter y to exit, or n to continue: y

Exiting. Re-run this script with the -recover option.
```

7. Fix the error (in this example, by setting/changing the password for 10.10.30.96), and run the utility again with the `-recover` option to update the configuration for the previously failed operation for the local cluster node.

   To copy the configuration again to all the nodes in the local and remote clusters, run the utility without the `-localhosts` and `-remotehosts` option. To rerun the utility to update the configuration for the failed nodes only, specify the IP addresses of the failed nodes only.

   Note: Specify at least one node in the `-localhosts` and `-remotehosts` option. You can use hostnames instead of IP addresses, as long as you ensure that DNS is working properly between the local node you are running the utility on, and the nodes you specify in the `-localhosts` and `-remotehosts` options.

   The output of the recovery session is shown below. Note that the utility returned a SUCCESS result:

```
[admin@node95 cross-cluster]$ cat local
10.10.30.96
[admin@node95 cross-cluster]$ cat remote
10.10.1.101
[admin@node95 cross-cluster]$ /opt/mapr/server/configure-crosscluster.sh
create server -remoteip 10.10.1.103 -localuser admin -remoteuser
mapr -recover latest -localhosts local -remotehosts remote
Remote IP is 10.10.1.103
WARNING: Strict host key checking will be disabled for this script.
Looking for most recent log file
Entering recovery mode. Using cross-cluster directory /tmp/mapr-xcs/14043
Enter password for mapr user (admin) for local cluster:
Enter password for mapr user (mapr) for remote cluster:
Local cross-cluster user unset, defaulting to local user admin
Remote cross-cluster user unset, defaulting to remote user mapr
Verifying connectivity to 10.10.1.103 and presence of mapr-clusters.conf
MapR credentials of user 'admin' for cluster 'chyelin95.cluster.com' are
written to '/tmp/maprticket_0'
Recovery option, using configured remote mapr-clusters.conf in /tmp/
mapr-xcs/14043/remote_clusterconf
Recovery option, using configured local mapr-clusters.conf in /opt/mapr/
conf/mapr-clusters.conf
Configuring cross-cluster communication for server-side operations
Recovery option, using configured local maprserverticket in /opt/mapr/
conf/maprserverticket
Recovery option, using configured remote maprserverticket in /tmp/
mapr-xcs/14043/remote_maprserverticket
SUCCESS
This script has logged in to both the local and remote clusters. Please
log out of the clusters if needed.
```

Multiple Runs of the Utility

When running the utility with the `all` or `user` argument, you may see the following error if you run the utility multiple times:

```
keytool error: java.lang.Exception: Certificate not imported, alias
<remote.cluster.com> already exists
ERROR: Unable to import remote cluster certificate from /tmp/mapr-xcs/17056/
remote_mapcert into local SSL trust store
Please delete the certificate with the same alias remote.cluster.com from
the truststore first
```

Certificates with the same alias should be imported to the trust store only once. If you are able to run commands like `maprcli volume mount` on the remote cluster from the local cluster, you can ignore this error. If you really want to re-import the remote cluster certificate into the trust store, contact MapR support.

Also, note that when you run the utility multiple times, there are at least 2 entries with the same alias in `/opt/mapr/conf/maprserverticket` file. The utility generates a new cross-cluster ticket (useful for volume mirroring and table and streams replication) every time it is run, and does not delete any tickets in `/opt/mapr/conf/maprserverticket` file. Service tickets have a long lifetime, so this can be ignored if you are able to successfully perform volume mirroring, and table and streams replication operations. However, if you want to clean up the tickets, you can do the following:

1. Delete all the tickets with the remote cluster alias in the `/opt/mapr/conf/maprserverticket` file on the local node.
2. Delete all the tickets with the local cluster alias in the `/opt/mapr/conf/maprserverticket` file on the remote node referenced in the `-remoteip` parameter.
3. Re-run the utility with the `server` argument to set up the service tickets again.

cldbguts
Monitors the activity of the Container Location Database (CLDB). This utility prints information about the CLDB service that is running on the node from which you run the utility.

Monitoring the progress of the *container location database (CLDB)* may be useful when troubleshooting cluster issues.

The `cldbguts` utility prints information about active container reports, full container reports, registration requests, MapR filesystem heartbeats, NFS server heartbeats, and containers. You can run `cldbguts` from any *container location database (CLDB)* node; however, running this command from the *container location database (CLDB)* master node provides the most relevant information.

Note: After you run `cldbguts`, it continues to print the output until you kill the process. To prevent `cldbguts` from printing indefinitely, specify the `-n` parameter that denotes the number of times `cldbguts` should print the output.

Syntax:

```
/opt/mapr/bin/cldbguts [[acr | rpc | heartbeat | containers | alarms |
table | all] [-n iterations-count]
```

Output Fields:

Note: When you run `cldbguts` without any parameters, only the `fcr`, `clrpc`, `regn`, `mfs hb`, `nfs hb`, `assigns`, `roles`, `progress`, and the `con-chain` fields are displayed.

acr	Represents active container requests (ACR). This column includes the following information: • `nr`: Number of ACRs completed in the previous second. The first entry displays the total number of ACRs completed since the start of the CLDB service on the node.

- `pt`: Processing time (in milliseconds) for the ACRs completed in the previous second. The first entry displays the total time (in milliseconds) spent processing the ACRs since the start of the CLDB service on the node.
- `to`: Number of ACRs that took longer than expected in the previous second. The first entry displays the total number of ACRs that took longer than expected since the start of the CLDB service on the node.
- `d`: Number of duplicate ACRs received in the previous second. The first entry displays the total number of duplicate ACRs since the start of the CLDB service on the node.
- `dp`: Number of duplicate ACRs that required additional work in the previous second. The first entry displays the total number of duplicate ACRs that required additional work since the start of the CLDB service on the node.

fcr

Represents full container report (FCR).

This column includes the following information:

- `nr`: Number of FCRs completed in the previous second. The first entry displays the total number of FCRs completed since the start of the CLDB service on the node.
- `pt`: Processing time (in milliseconds) for the FCRs completed in the previous second. The first entry displays the total time (in milliseconds) spent processing the FCRs since the start of the CLDB service on the node.
- `to`: Number of FCRs that took longer than expected in the previous second. The first entry displays the total number of FCRs that took longer than expected since the start of the CLDB service on the node.

regn

Represents registration requests.

This column includes the following information:

- `nr`: Number of registration requests completed in the previous second. The first entry displays the total number of registration requests completed since the start of the CLDB service on the node.
- `pt`: Processing time (in milliseconds) for the registration requests completed in the previous second. The first entry displays the total time (in milliseconds) spent processing the registration requests since the start of the CLDB service on the node.

last-updated: Apr 11, 2022

- `to`: Number of registration requests that took longer than expected in the previous second. The first entry displays the total number of registration requests that took longer than expected since the start of the CLDB service on the node.
- `d`: Number of duplicate registration requests received in the previous second. The first entry displays the total number of duplicate registration requests since the start of the CLDB service on the node.
- `dp`: Number of duplicate registration requests that required additional work in the previous second. The first entry displays the total number of duplicate registration requests that required additional work since the start of the CLDB service on the node.

mfs hb

Information about MapR filesystem heartbeats.

This column includes the following information:

- `nr`: Number of MapR filesystem heartbeats completed in the previous second. The first entry displays total number of MapR filesystem heartbeats completed since the start of the CLDB service on the node.
- `pt`: Processing time (in microseconds) for the MapR filesystem heartbeats completed in the previous second. The first entry displays total time (in microseconds) spent processing MapR filesystem heartbeats since the start of the CLDB service on the node.
- `to`: Number of MapR filesystem heartbeats that took longer than expected in the previous second. The first entry displays total number of MapR filesystem heartbeats that took longer than expected since the start of the CLDB service on the node.
- `bmc`: Number of times the Become Master Command (`bmc`) has been sent to this MFS.
- `otc`: Number of times the other commands (apart from `bmc`) has been sent to this MFS.

nfs hb

Information about NFS server heartbeats.

This column includes the following information:

- `nr`: Number of NFS server heartbeats completed in the previous second. The first entry displays total number of NFS server heartbeats completed since the start of the CLDB service on the node.
- `pt`: Processing time (in microseconds) for the NFS server heartbeats completed in the previous second. The first entry displays the total time (in microseconds) spent processing MapR filesystem heartbeats since the start of the CLDB service on the node.

last-updated: Apr 11, 2022

assigns

This column includes the following information:

- `nr`: Number of container assign requests in the previous second. The first entry displays the total number of container assign requests since the start of the CLDB service on the node.
- `nc`: Number of containers created as part of the above container assign requests in the previous second. The first entry displays the total number of containers created since the start of the CLDB service on the node.
- `nrt`: Number of container assign requests for tablets in the previous second. The first entry displays the total number of container assign requests for tablets since the start of the CLDB service on the node.
- `nct`: Number of containers created as part of the above container assign requests for tablets in the previous second. The first entry displays the total number of container created in tablets since the start of the CLDB service on the node.
- `pt`: Time taken by container assignment RPC in milliseconds
- `tpt`: Time taken by container assignment tablet RPC in milliseconds
- `cas`: Number of storage pools scanned for container assignment requests

roles

Represents the roles of the various replica containers.

This column includes the following information:

- `bm`: Number of replica containers that are in the process of becoming master
- `ms`: Number of replica containers that the CLDB thinks have valid masters
- `wr`: Number of replica containers that are waiting for CLDB to assign a role to them
- `rs`: Number of replica containers that are re-syncing
- `vr`: Number of non-master replica containers that have finished resynchronization
- `uu`: Number of replica containers that are unused. For example, the number of replica containers that are on nodes or storage pools which have been offline or unavailable for more than an hour.

Attention: It may take some time for the CLDB to be aware of role changes.

progress

This column includes the following information:

last-updated: Apr 11, 2022

	• `m%`: Percentage of containers that have valid masters • `uc`: Number of unique containers • `v%`: Percentage of replica containers that are valid (that is, have completed resynchronization) • `tr`: Total number of replica containers
con-chain	This column includes the following information: • `ms`: Number of unique containers that have a master • `1r`: Number of unique containers that have 2 valid copies of the data • `2r`: Number of unique containers that have 3 valid copies of the data
location	This column includes the following information: • `lu`: Number of container location lookups • `up`: Number of container location updates • `dl`: Number of container location deletes • `sc`: Number of container location scans
size	This column includes the following information: • `lu`: Number of container size lookups • `up`: Number of container size updates • `dl`: Number of container size deletes • `sc`: Number of container size scans
sptable	This column includes the following information: • `lu`: Number of lookups on the SP-Container-Vol table • `up`: Number of updates on the SP-Container-Vol table • `dl`: Number of deletes on the SP-Container-Vol table • `sc`: Number of scans on the SP-Container-Vol table
nodes	This column includes the following information: • `nn`: Number of nodes in the cluster • `of`: Number of offline nodes

- `nsp`: Number of storage pools
- `of`: Number of offline storage pools

Example Output:

```
/opt/mapr/bin/cldbguts all  -n 3
            2019-09-15 22:08:39,981
            mfs hb                nfs
hb                 assigns                          roles
progress            con-chain       location            size
sptable            fcr              clrpc            regn
nodes
            nr        pt     to  bmc  otc    nr  pt     nr  nc  nrt  nct
pt  tpt  cas    bm  ms  wr  rs  vr  uu      m%    uc     v%    tr    ms
1r  2r    lu   up  dl  sc   lu   up  dl  sc   lu  up  dl  sc     nr   pt
to     nr    pt   to    nr   pt  to   nn  of  sp  of
            428807  112504140   0   57   0      0   0    120   5   0    0
294   0    2      0  61   0   0   0   0    98.39%  62  100.00%  61    61
0   0     0  113   0  32    0  416   0  16    0   1   0   0    476  5073
0    5650  3971   0     3  178   0    1   0   1   0
            1       245       0   57   0      0   0     0    0   0    0
0    0    0      0  61   0   0   0   0    98.39%  62  100.00%  61    61
0   0     0   0    0   0    0   0    0   0    0   0   0   0     0     0
0      0     0    0     0   0    0    1   0   1   0
            1       288       0   57   0      0   0     0    0   0    0
0    0    0      0  61   0   0   0   0    98.39%  62  100.00%  61    61
0   0     0   0    0   0    0   0    0   0    0   0   0   0     0     0
0      0     0    0     0   0    0    1   0   1   0
```

Related reference

Retrieving Tiering Statistics Using guts on page 943
Explains how to use the `guts` utility to retrieve tiering statistics.

guts on page 2110
`guts` is a tool to measure/analyse performance. In the default mode, it prints one line every second, and counts the number of operations or bytes-processed in one second intervals. `guts` is an internal utility, and is subject to change without notice.

cldbguts acr

The `acr` option displays active container requests (ACR).

Syntax

```
/opt/mapr/bin/cldbguts acr
```

Output Fields

Field	Description
`nr`	Number of ACRs completed in the previous 1 second. The first entry displays the total number of ACRs completed since the start of the CLDB service on the node.
`pt`	Processing time (in milliseconds) for the ACRs completed in the previous 1 second. The first entry displays the total time (in milliseconds) spent processing the ACRs since the start of the CLDB service on the node.

last-updated: Apr 11, 2022

Field	Description
to	Number of ACRs that took longer than expected in the previous 1 second. The first entry displays the total number of ACRs that took longer than expected since start of the CLDB service on the node.
d	Number of duplicate ACRs received in the previous 1 second.The first entry displays the total number of duplicate ACRs since start of the CLDB service on the node.
dp	Number of duplicate ACRs that required additional work in the previous 1 second. The first entry displays the total number of duplicate ACRs that required additional work since start of the CLDB service on the node.

Example Output

```
# /opt/mapr/bin/cldbguts acr
2017-09-01 15:15:25,034
             acr
    nr      pt   to  d  dp
  269644  63338   0  0   0
     1      0     0  0   0
     0      0     0  0   0
```

cldbguts containers

The `containers` option displays information on the containers.

Syntax

```
/opt/mapr/bin/cldbguts containers
```

Output Fields

assigns

This column includes the following information:

Field	Description
nr	Number of ContainerAssign requests in the previous 1 second. The first entry displays the total number of ContainerAssign requests since the start of the CLDB service on the node.
nc	Number of containers created as part of the above ContainerAssign requests in the previous 1 second. The first entry displays the total number of containers created since the start of the CLDB service on the node.

Field	Description
nrt	Number of ContainerAssign requests for tablets in the previous 1 second. The first entry displays total number of ContainerAssign requests for tablets since the start of the CLDB service on the node.
nct	Number of containers created as part of the above ContainerAssign requests for tablets in previous 1 second. The first entry displays the total number of container created in tablets since the start of the CLDB service on the node.

roles

Represents the roles of the various replica containers. This column includes the following information:

Field	Description
bm	Number of replica containers that are in the process of becoming master
ms	Number of replica containers that the CLDB thinks have valid masters
wr	Number of replica containers that are waiting for CLDB to assign a role to them
rs	Number of replica containers that are re-syncing
vr	Number of non-master replica containers that have finished resynchronization
uu	Number of replica containers that are unused. For example, the number of replica containers that are on nodes or storage pools which have been offline or unavailable for more than an hour.

Attention: It may take some time for the CLDB to be aware of any role changes.

progress

This column includes the following information:

last-updated: Apr 11, 2022

Field	Description
m%	Percentage of containers that have valid masters
uc	Number of unique containers
v%	Percentage of master + replica containers that are valid container copies
tr	Total number of replica containers

con-chain

This column includes the following information:

Field	Description
ms	Number of unique containers that have a master
1r	Number of unique containers that have 2 valid copies of the data
2r	Number of unique containers that have 3 valid copies of the data

Example Output

```
# /opt/mapr/bin/cldbguts containers

2019-10-03 03:05:15,846
            assigns                          roles
progress          con-chain
  nr  nc  nrt  nct  pt  tpt  cas    bm  ms  wr  rs  vr  uu     m%   uc
v%   tr    ms  1r  2r
   0   0   0    0    0   0    0      0   0   0   0   0   0    -DZ-
0  -DZ-   0     0   0   0
   0   0   0    0    0   0    0      0   0   0   0   0   0    -DZ-
0  -DZ-   0     0   0   0
   0   0   0    0    0   0    0      0   0   0   0   0   0    -DZ-
0  -DZ-   0     0   0   0
```

cldbguts heartbeat

The `heartbeat` option displays information on the heartbeat sent by the MapR File System and NFS.

Syntax

```
/opt/mapr/bin/cldbguts heartbeat
```

Output Fields

mfs hb

Information about MapR filesystem heartbeats. This column includes the following information:

Field	Description
nr	Number of MapR filesystem heartbeats completed in the previous 1 second. The first entry displays the total number of MapR filesystem heartbeats completed since the start of the CLDB service on the node.
pt	Processing time (in microseconds) for the MapR filesystem heartbeats completed in the previous 1 second. The first entry displays total time (in microseconds) spent processing MapR filesystem heartbeats since the start of the CLDB service on the node.
to	Number of MapR filesystem heartbeats that took longer than expected in the previous 1 second. The first entry displays total number of MapR filesystem heartbeats that took longer than expected since the start of the CLDB service on the node.
bmc	Number of Become Master commands (such as resync, reconnect, etc.) sent by CLDB since the start of CLDB.

nfs hb

Information about NFS server heartbeats. This column includes the following information:

Field	Description
nr	Number of NFS server heartbeats completed in the previous 1 second. The first entry displays total number of NFS server heartbeats completed since the start of the CLDB service on the node.

last-updated: Apr 11, 2022

Field	Description
pt	Processing time (in microseconds) for the NFS server heartbeats completed in the previous 1 second.The first entry displays the total time (in microseconds) spent processing MapR filesystem heartbeats since the start of the CLDB service on the node.

Example Output

```
# /opt/mapr/bin/cldbguts heartbeat

2019-10-03 03:14:56,811
         mfs hb           nfs hb
  nr  pt  to  bmc  otc    nr  pt
   0   0   0   0    0     0   0
   0   0   0   0    0     0   0
   0   0   0   0    0     0   0
```

cldbguts rpc

The `rpc` option returns a count of the RPCs that CLDB is processing from clients.

Syntax

```
/opt/mapr/bin/cldbguts rpc
```

Output Fields

clrpc

Represents a count of the client RPCs per second. This count includes:

1. ClusterInfoProc
2. ContainerLookupProc
3. ContainerRootLookupProc

This column includes the following information:

Field	Description
nr	Number of Client RPC's completed in the previous 1 second. The first entry displays total number of client RPCs completed since the start of the CLDB service on the node.

Field	Description
pt	Processing time (in milliseconds) for the Client RPCs completed in the previous 1 second. The first entry displays total time (in milliseconds) spent processing the client RPCs since the start of the CLDB service on the node.
to	Number of Client RPCs that took longer than expected in the previous 1 second. The first entry displays the total number of client RPCs that took longer than expected since the start of the CLDB service on the node.

fcr

Represents full container report (FCR). This column includes the following information:

Field	Description
nr	Number of FCRs completed in the previous 1 second. The first entry displays total number of FCRs completed since the start of the CLDB service on the node.
pt	Processing time (in milliseconds) for the FCRs completed in the previous 1 second. The first entry displays total time (in milliseconds) spent processing the FCRs since the start of the CLDB service on the node.
to	Number of FCRs that took longer than expected in the previous 1 second. The first entry displays the total number of FCRs that took longer than expected since the start of the CLDB service on the node.

regn

Represents registration requests. This column includes the following information:

last-updated: Apr 11, 2022

Field	Description
nr	Number of registration requests completed in the previous 1 second. The first entry displays total number of registration requests completed since the start of the CLDB service on the node.
pt	Processing time (in milliseconds) for the registration requests completed in the previous 1 second. The first entry displays the total time (in milliseconds) spent processing the registration requests since the start of the CLDB service on the node.
to	Number of registration requests that took longer than expected in the previous 1 second. The first entry displays total number of registration requests that took longer than expected since the start of the CLDB service on the node.

Example Output

```
# /opt/mapr/bin/cldbguts rpc

2019-10-03 03:17:48,863
      fcr             clrpc            regn
  nr  pt  to      nr     pt   to    nr  pt  to
   0   0   0   16116  74239   0     0   0   0
   0   0   0       0      0   0     0   0   0
   0   0   0       0      0   0     0   0   0
```

disksetup
Describes the `disksetup` command that formats disks for use by MapR storage.

Description

Note: The `disksetup` command must be run as `root`.

The `disksetup` command formats specified disks for use by MapR storage, and adds those disks to the disktab file.

You do not need to set up Redundant Array of Independent Disks (RAID) on disks used by the MapR File System. MapR uses `disksetup` to set up storage pools. In most cases, you should let MapR calculate storage pools using the default *stripe width* of two or three disks. If you anticipate a high volume of random-access I/O, you can use the `-W` option to specify larger storage pools of up to 8 disks each.

See Setting Up Disks for MapR for more information about when and how to use `disksetup`.

Important: On RHEL 8.1, run the following command to symlink `/usr/bin/python` to `/usr/bin/python3`. The disksetup command fails if `/usr/bin/python` is not found.

```
sudo alternatives --set python /usr/bin/python3
```

Syntax

```
/opt/mapr/server/disksetup
    [-F]
    [-G]
    [-X]
    [-M]
    [-W <stripe_width>]
    <disk list file>
```

Options

Option	Description
`-F`	Forces formatting of all specified disks. Disks that are already formatted for MapR are not reformatted by `disksetup` unless you specify this option. The `-F` option fails when a filesystem has an entry in the `disktab` file, is mounted, or is in use. Call `maprcli disk remove` to remove a disk entry from the `disktab` file.
`-G`	Generates the `disktab` file contents from input disk list, but does not format disks. Use this option if the `disktab` file is completely lost, and you need to regenerate it based on an input list of disks assigned to MapR-FS. This option reads the GUID from the provided disks, and generates the `disktab` output to stdout. You can redirect the output to a file.
`-X`	Fixes `disktab` contents from `/proc/partitions`, but does not format disks. Use this option if there is a change in the names of the disk devices referenced by `disktab`, but the disks themselves are still usable. For example, if `/dev/sdb` has been renamed to `/dev/sdf` but the device itself has the same GUID, only the `disktab` contents need to be updated to point to `/dev/sdf`.
`-M`	Uses the maximum available number of disks per storage pool.
`-W`	Specifies the number of disks per storage pool.

Examples

Setting up disks specified in the file /tmp/disks.txt:

```
/opt/mapr/server/disksetup -F /tmp/disks.txt
```

Reformatting all disks

To reformat all disks, remove the `disktab` file and issue the `disksetup -F` command to format the disk:

```
/opt/mapr/server/disksetup -F
```


last-updated: Apr 11, 2022

To reformat a particular disk from the `disktab`, use the `maprcli disk remove` on page 1608 and `maprcli disk add` on page 1602 commands. For more information, see Setting Up Disks for MapR on page 835.

Specifying disks

To specify the disks to be formatted for use by the MapR cluster, create a text file `/tmp/disks.txt` listing the disks and partitions for use by MapR on the node. Each line lists either a single disk, or all applicable partitions on a single disk. When listing multiple partitions on a line, separate each partition by spaces. For example:

```
/dev/sdb
/dev/sdc1 /dev/sdc2 /dev/sdc4
/dev/sdd
```

Later, when you run `disksetup` to format the disks, specify the `disks.txt` file. For example:

```
/opt/mapr/server/disksetup -F /tmp/disks.txt
```

Important:

The `disksetup` command removes all data from the specified disks. Ensure that you specify the disks correctly, and that you have backed up any data that you wish to keep.

If you are re-using a node that was used previously in another cluster, be sure to format the disks to remove any traces of data from the old cluster.

Warning: Run `disksetup` on page 2092 only after you run the `configure.sh` on page 2053 .

Test Purposes Only: Using a Flat File for Storage

When setting up a small cluster for evaluation purposes, if a particular node does not have physical disks or partitions available to dedicate to the cluster, you can use a flat file on an existing disk partition as the node's storage. Create at least a 16GB file, and include a path to the file in the disk list file for the `disksetup` on page 2092 script.

The following example creates a 20 GB flat file (`bs=1G` specifies 1 gigabyte blocks, multiplied by `count=20`) at `/root/storagefile`:

```
dd if=/dev/zero of=/root/storagefile bs=1G count=20
```

Next, add the following entry to the disk list file `/tmp/disks.txt` to be used by `disksetup`:

```
/root/storagefile
```

ectool

Dumps or checks the validity of the stripelets in the backend volume that is associated with the volume configured for warm tiering.

You can use the `/opt/mapr/server/tools/ectool` utility to dump or check the validity of the stripelets in the backend volume that is associated with the volume configured for warm tiering.

Pre-Requirements

Before you run the utility, you must ensure that the node on which you plan to run this utility meets the following requirements:

- Set the path to the library (`libjvm.so`) in the `LD_LIBRARY_PATH` environment variable. If the path is not already set, use the `export` command to set the environment variable. For example:

```
export LD_LIBRARY_PATH=/opt/jdk1.8.0_141/jre/lib/amd64/server/
```

Syntax

```
/opt/mapr/server/tools/ectool <cmd> <params>
```

Commands

Command	Description
`dumpStripelet`	Dumps the content of a stripelet to the given file.
`listStripes`	Lists all the stripes in a given container.
`validateStripe`	Validates if the given parity stripelet is valid and matches with other stripelets of the stripe.
`validateCG`	Iterates over all the stripes of the Container Group and checks the validity using the rebuild operation.
`getFid`	Returns the fid for a Virtual Cluster Descriptor (VCD) ID.

Parameters

Parameter	Description
`cid`	The ID of the container.
`fid`	The ID of the file.
`file`	The path to the file.

`volid`	The ID of the (backend) volume, referred to as `ecstorevolume` in the CLI output, associated with the tier. The ID can be retrieved using the `volume info` on page 1965 command. For example:
	`/opt/mapr/bin/maprcli volume info -name a4 -json | grep ecstorevolume`
	`"ecstorevolume":"mapr.internal.ec.a4.87379483",`
	`/opt/mapr/bin/maprcli volume info -name mapr.internal.ec.a4.87379483 -json | grep volumeid`
	`"volumeid":105118862,`

vcdid	The ID of the Virtual Cluster Descriptor (VCD).

Usage

```
/opt/mapr/server/tools/ectool dumpStripelet volid fid file
/opt/mapr/server/tools/ectool listStripes volid cid
/opt/mapr/server/tools/ectool validateStripe volid fid
/opt/mapr/server/tools/ectool validateCG volid cid
/opt/mapr/server/tools/ectool getFid cid vcdid
```


last-updated: Apr 11, 2022

Examples

Dump the stripelet content to the file `/tmp/t` for the file specified by ID 2271.160.131606 in the volume specified by ID 116581327:

```
# /opt/mapr/server/tools/ectool dump 116581327 2271.160.131606 /tmp/t
Stripelet Read done!
```

List all the stripes in the container specified by ID 2271 for the volume specified by ID 116581327:

```
# /opt/mapr/server/tools/ectool list 116581327 2271
Inum:160 Uniq:131606 Size:4194304
Inum:161 Uniq:131608 Size:4194304
Inum:162 Uniq:131610 Size:4194304
```

Validates if the given stripelet matches with other stripelets of the stripe for the file specified by ID 2271.160.131606 in the volume specified by ID 116581327

```
# /opt/mapr/server/tools/ectool validateStripe 116581327 2271.160.131606
Valid Stripe
```

For the container specified by ID 2271, validate if all stripelets match with other stripelets of the corresponding stripe:

```
# /opt/mapr/server/tools/ectool validateCG 116581327 2271
Inum:160 Valid Stripe
Inum:161 Valid Stripe
Inum:162 Valid Stripe
```

expandaudit

Describes how to use the `expandaudit` utility to expand IDs captured in the audit logs to their corresponding names.

As you perform operations on the directories, files, and tables that you are auditing, the audit logs capture records of those operations. Those records identify the affected directories, files, and tables by means of file IDs, the volumes on which the operations took place by means of volume identifiers, and the users who performed the operations by means of user IDs. These IDs are used instead of names in the audit records because fetching the actual names of these objects and users in real-time is costly in terms of performance.

You can use the `expandaudit` utility to create copies of your logs files in which the IDs are resolved into names and inserted into the audit records.

This utility acts on audit logs that exist in the current MapR cluster at the time that the utility is run.

Restrictions

This utility operates on audit logs for filesystem operations and MapR Database operations, which are logged in a local MapR volume on each node where the operations are performed. These operations are logged in `FSAudit` and `DBAudit` log files.

File identifiers are converted to names only when either of the following conditions is met:

- The file exists at the time that `expandaudit` is run.
- The file has been deleted but the deletion of the file was logged and the log files being processed by `expandaudit` include the record of the file deletion.

If a volume is deleted, `expandaudit` does not convert identifiers for files that were in the volume unless the creation of the volume and files were logged.

If the creation of a file is audited and the file is later renamed, the file ID is converted to the current name.

Permissions

Although the permissions on the tool are `755`, the tool generates output only when run by `root` or the user `mapr`.

Syntax

```
/opt/mapr/bin/expandaudit
    [-volumename volume name]
    [-volumeid volume ids. Either volume name or id must be specified]
    -o output directory
    [-d Specify for deleted volumes only]
    [-cluster cluster name]
    [-t number of threads used for parallel expansion across cluster nodes.
default 10]
```

Parameters

Parameter	Description
`cluster`	The name of the cluster on which to run the command.
`d`	Required for deleted volumes as it indicates that the volume is deleted. If you specify this parameter, you must specify a volume ID to be used during expansion. The deleted volume is tracked by the specified volume ID. You can optionally specify a volume name. This specified volume name is used for the expanded output.
`o`	The directory in the MapR filesystem in which to create the copies of the audit logs. The directory must already exist. The directory structure is: `<output directory>/<volume id>/<node>/<day>/<expanded audit log files>` The file names are the same as the names of the input files, though you might see the following extensions: • `.part`: If present, this extension is on the log file with the most recent date. The input log file that corresponds to this output file might still have been receiving new audit records at the time that the `expandaudit` utility was run. If the utility is run again with the same output directory, the utility will update the .`part` file by including the most recent records and converting the identifiers in those records. • `.pending`: This extension indicates files that contain one or more identifiers that the utility could not convert. **Note:** Sometimes, you might see a combination of these two types of files, `part.pending`, which indicates that there is a problem converting identifiers in the most recent audit file.
`t`	The number of threads to use for parallel expansion across cluster nodes. The default value is 10.

last-updated: Apr 11, 2022

Parameter	Description
`volumename`	The name of the volume being audited. You must specify either the `volumename` or the `volumeid` parameter.
`volumeid`	The ID of the volume being audited. You must specify either the `volumename` or the `volumeid` parameter.

Sample Expansion of a Record for Filesystem Operations

Original record

```
{"timestamp":
{"$date":"2015-06-06T13:02:23.746Z"},"operation":"GETATTR","uid":"1","ipAddr
ess":
"10.10.104.53","srcFid":"2049.652.263696","volumeId":68048396,"status":0}
```

Record processed by the expandaudit utility

```
{"timestamp":
{"$date":"2015-06-06T13:02:23.746Z"},"operation":"GETATTR","user":
"userA","uid":"1","ipAddress":"10.10.104.53","srcPath":"/customers/
US_Western_Region.json",
"srcFid":"2049.3296.268968","volumeName":"data_analysis","volumeId":68048396
,"status":0}
```

Attention: Here, `uid` expands to `user`, `srcFid` expands to `srcPath`, and `volumeID` expands to `volumeName`. The original fields are also preserved in the output.

Sample Expansion of a Record for MapR Database Table Operations

Original record

```
{"timestamp":
{"$date":"2015-06-06T13:08:54.474Z"},"operation":"DB_PUT","uid":"1","ipAddre
ss":
"10.10.104.51","volumeId":68048396,"columnFamily":"fam63","columnQualifier":
"col_96","tableFid":
"2049.56.262518","status":0}
```

Record processed by the expandaudit utility

```
{"timestamp":"{$date=2015-06-06T13:08:54.474Z}","operation":"DB_PUT","user":
"userA","uid":
"1","ipAddress":"10.10.104.51","volumeName":"mapr.cluster.root","volumeId":"
68048396",
"columnFamily":"fam63","columnQualifier":"col_96","tablePath":"/
mytable","tableFid":"2049.56.262518",
"status":"0"}
```

Attention: Here, `uid` expands to `user`, `volumeID` expands to `volumeName`, and `tableFid` expands to `tablePath`. The original fields are also preserved in the output.

fcdebug
Dynamically sets the log level to debug a library.

You can modify the `core-site.xml` file to set the log level of all modules using the `fs.mapr.trace` property. However, you must restart FUSE for the change to take effect. As an alternative, you can use the `fcdebug` utility to debug a specific library (at runtime) without restarting FUSE.

last-updated: Apr 11, 2022

Note: You may have to run this command once per library (to debug).

Syntax

```
/opt/mapr/server/tools/fcdebug [-i] [-p <process ID>] [-s <shm ID>][-m
<module>] [-l <level>] [-o <slowOpsTraceThreshold>]
```

Parameters

Parameter	Description
`-i`	Lists the current debug level of all modules.
`-l`	Specifies the log level. Value can be one of the following: FATAL, ERROR, WARN, INFO, DEBUG. **Note:** If you do not specify the log level, the default level is applied for the module.
`-m`	Specifies the module for which the log level is to be set. You can retrieve the list of modules with the `fcdebug -i -p <process ID>` or the `fcdebug -i -s <shmid>` command. **Note:** If you do not specify the module, the log level is set on all modules.
`-p`	Specifies the process ID of either the file client, or the FUSE-based POSIX client.
`-s`	Specifies the shared memory ID (shmid) of either the file client, or the FUSE-based POSIX client.

Attention: Specify either the `process ID (-p)` or the `shmid (-s)` option. If you specify both the options, only the `shmid (-s)` option is used.

Examples

The following command retrieves the list of modules:

```
Note: Use either the -p or the -s option.

/opt/mapr/server/tools/fcdebug -i -p 196614  (OR)
/opt/mapr/server/tools/fcdebug -i -s 335020032
```

Note: You can run this command after dynamically setting the log level to verify the setting.

The following command dynamically sets the log level to DEBUG on the given module:

```
Note: Use either the -p or the -s option.

/opt/mapr/server/tools/fcdebug -p 196614 -m FuseOps -l DEBUG (OR)
/opt/mapr/server/tools/fcdebug -s 335020032 -m FuseOps -l DEBUG
```

Note: It may take up to30 seconds for the changes to take effect.

last-updated: Apr 11, 2022

The following command sets the log level to DEBUG on all the modules:

```
Note: Use either the -p or the -s option.

/opt/mapr/server/tools/fcdebug -p 196614 -l DEBUG (OR)
/opt/mapr/server/tools/fcdebug -s 335020032 -l DEBUG
```

The following command resets the log level to the default value on all the modules:

```
Note: Use either the -p or the -s option.

/opt/mapr/server/tools/fcdebug -p 196614 (OR)
/opt/mapr/server/tools/fcdebug -s 335020032
```

fsck
Detects and fixes inconsistencies in the filesystem.

Use the filesystem check (fsck) utility to detect and fix inconsistencies in the filesystem.

Every storage pool has its own log to journal updates to the storage pool. The system performs all operations to a storage pool transactionally by journaling all operations to the log, before applying them to storage pool metadata. If MapR File System is not shutdown cleanly, some metadata blocks may not persist. However, on the next load of the storage pool, log recovery takes care of these metadata blocks by replaying the records in the log. The fsck utility also replays the log before it checks the metadata consistency in a storage pool. The `fsck` utility walks the storage pool in question to verify all MapR filesystem metadata (and data correctness if specified on the command line), and reports all potentially lost or corrupt containers, directories, tables, files, filelets, and blocks in the storage pool. The `fsck` utility:

- Checks whether all files and directories are reachable and all directory entries are valid.
- Checks whether BTrees are consistent for various inode types (such as files and directories).
- Walks the container file and visits every inode in the container to check that no block is owned by two inodes. Aso, verifies the consistency of bitmaps of inodes and blocks.
- Checks consistency of snapshots.
- Visits every allocated block in the storage pool and recovers any blocks that are part of corrupted inodes.
- Checks consistency of MapR Database metadata.
- Checks consistency of tabletmap, tablets, buckets, and spill files.

The `fsck` utility can be used on an offline storage pool after a node failure, after a disk failure, or after a MapR filesystem process crash, or simply to verify the consistency of data for suspected software bugs.

Typical process flow:

- Take the affected storage pools offline with the mrconfig sp offline on page 2175 command.
- Execute the `fsck` command on the storage pools (or disks) as specified in the following discussion.
- Bring the storage pools back online with the mrconfig sp online on page 2176 command.
- Execute the gfsck on page 2102 command on the cluster, volumes, or snapshots that were affected.

You can run the `fsck` command in two modes:

- Verification mode - `fsck` only reports errors; it does not attempt to fix or modify any data on disk. You can run `fsck` in verification mode on an offline storage pool at any time, and it will report errors if there is inconsistency. If it does not report any errors, you can bring up the storage pool online without any risk of data loss. To run the `fsck` utility in verification mode, use any parameter *except* the `-r` parameter.
- Repair mode - `fsck` attempts to repair a bad storage pool. When you run the `fsck` utility in repair mode on a storage pool, some volumes might need a global fsck (gfsck on page 2102) after bringing the storage pool online. There is potential for loss of data in this case. To run the `fsck` utility in repair mode, use the `-r` parameter.

Using the `/opt/mapr/server/fsck` utility with the `-r` option produces different results depending on the scenario. The `fsck` utility does not interpret the scenario nor does it have a safe mode.

- If a disk is offline because of an imbalanced b-tree, using `fsck -r` may result in data loss from bad containers, and data loss if additional replicas are unavailable.
- If a disk is offline because of an I/O error, using `fsck -r` produces indeterminate results. A disk that is returning I/O errors is questionable in terms of data content and reliability. For example, an operation that completed on the disk but was never returned, may have partial data remaining on the disk. Using `fsck -r` retains any partial data.
- If a disk is offline because of slow I/O, using `fsck -r` does not produce data loss.

The most conservative usage of `fsck` is to first run `fsck` without the `-r` option (verification mode) and check the output. If the output returns errors, then run `fsck` with the `-r` option.

Syntax

```
/opt/mapr/server/fsck [{<device-paths>}] or [-n <sp name>]
     -l <log filename> ; default /opt/mapr/logs/fsck.log.<ts>.<pid>
     -p <mfs port> ; default 5660
     -N to disable status bar
     -P to purge deleted containers in repair
     -h for help
     -j to skip log replay
     -m <memory in MB> to set cache size for blocks
     -d to check data blocks crc
     -b to check db consistency
     -C to specify the container-id when using -d
     -I to specify inode-number when using -d and -C
     -r to repair ; USE WITH CAUTION AS IT CAN LEAD TO LOSS OF DATA
```

Parameters

Parameter	Description
-b	Checks database consistency.
-C	Optional with the `-d` option. Specifies the read-write container ID on which CRC (cyclic redundancy check) must be performed. If this option is not specified with the `-d` option, the CRC is performed on all the containers on the storage pool.
-d	Performs a CRC on data blocks. By default, `fsck` will not validate the CRC of user data pages. Enabling this check causes the check to take a while to complete.

last-updated: Apr 11, 2022

Parameter	Description

<device-paths>	Paths to the disks that make up the storage pool.
	  **Note:** Before running `fsck`, use the mrconfig disk remove on page 2151 command to remove all the disks from MapR File System. For example:
	`/opt/mapr/server/mrconfig disk remove /dev/sdb`
	`/opt/mapr/server/fsck /dev/sdb`

-h	Help
-I	Optional with the `-C` option. Specifies the inode number on which CRC (cyclic redundancy check) must be performed on the container specified by the `-C` option. If this option is not specified with the `-C` option, the CRC is performed on all the inodes.
-j	Skips log replay. Should be set only when log recovery fails. Log recovery can fail if the damaged blocks of a disk belong to the log, or if log recovery finds some CRC errors in the metadata blocks. *Using this parameter will typically lead to larger data loss. *
-l	The log filename. Default: `/opt/mapr/logs/fsck.log.<ts>.<pid>`
-m	Sets the cache size for blocks (MB).
-n	Storage pool name. This option works only if all the disks are in `disktab`. Otherwise, you must individually specify all the disks that make up the storage pool, using the `<device-paths>` parameter.
-N	Disables the status bar.
-p	The MapR File System port. Default: `5660`
-P	Purges deleted containers in repair.
-r	Runs in repair mode. **USE WITH CAUTION AS THIS CAN LEAD TO LOSS OF DATA.**

gfsck

Describes how you can use the `gfsck` command, under the supervision of Map R Support or Engineering, to perform consistency checks and appropriate repairs on a volume, or a volume snapshot.

You can use the `gfsck` command when the local `fsck` either repairs or loses some containers at the highest epoch.

For an overview of using the GFSCK command, see Using Global File System Checking on page 980.

 **Important:** You need to install a patch to use some options such as `-Dfull` and `--crc`

Permissions Required

Although you need to be the *root* user to run this command, checking tiering-enabled volumes requires you to be the *mapr* user.

last-updated: Apr 11, 2022

Syntax

```
/opt/mapr/bin/gfsck
    [-h] [--help]
    [-c] [--clear]
    [-d] [--debug]
    [-b] [--dbcheck]
    [-r] [--repair]
    [-y] [--assume-yes]
    [-Gquick] [--check-tiermetadata-only]
    [-Gfull]  [--check-tiermetadata-full]
    [-Dquick] [--check-tierdata-presence]
    [-Dfull]  [--check-tierdata-crc]
    [-J] [--skip-tier-log-replay]
    [-D] [--crc]

    [cluster=cluster-name (default=default)]
    [rwvolume=volume-name (default=null)]
    [snapshot=snapshot-name (default=null)]
    [snapshotid=snapshot-id (default=0)]
    [fid=fid (default=null)]
    [cid=cid (default=0)]

    [rIdx=<repl index>] (replication index, only enabled with [-D]  [--crc]
    [fidThreads=<check crc thread count for fid>] (default:16, max:128)
    [cidThread=<check crc thread count for cid>] (default:16, max:128)
    [scanthreads=inode scanner threads count (default:10, max:1000)]
```

Parameters

-h|--help	*Description*: Prints usage text *User who must use this option*: Either `root` or `mapr`.
-c|--clear	*Description*: Clears previous warnings before performing the global filesystem check. *User who must use this option*: Either `root` or `mapr`.
-d|--debug	*Description*: Provides information for debugging. *User who must use this option*: Either `root` or `mapr`.
-b|--dbcheck	*Description*: Checks that every key in a tablet is within that tablet's startKey and endKey range. This option is I/O intensive, so use this option only if you suspect database inconsistency. *User who must use this option*: `root` When used with S3 volumes , this option validates that *versionIds* of objects in a given partition are less than *maxVersionId* stored in Partition Map Entry. *User who must use this option*: `mapr`.
-r|--repair	*Description*: Indicates and repairs the inconsistencies detected by `-GQuick`, `-GFull`, `-DQuick`, and `-DFull`. Repair is not supported for snapshots and mirrors. *User who must use this option*: `root`
-y|--assume-yes	*Description*: If specified, assumes that containers without valid copies (as reported by CLDB) are deleted automatically. If not specified, `gfsck` pauses for user input: **yes** to delete, **no** to exit `gfsck`, or **ctrl-C** to quit.

last-updated: Apr 11, 2022

	User who must use this option: Either `root` or `mapr`.
-D|--crc	*Description*: Provides validation of the CRC of the data present in the volume. The data can either be local or offloaded. You can use this option at the volume, container, snapshot, and the filelet levels. `gfsck` reports corruption found at each level. *User who must use this option*: `root`
cluster	*Description*: Specifies the name of the cluster (default: default cluster) *User who must use this option*: Either `root` or `mapr`.
rwvolume	*Description*: Specifies the name of the volume (default: default cluster) *User who must use this option*: Either `root` or `mapr`.
fid	*Description*: Checks data CRC for the master copy of the specified fid. To check any other copy, use the `rIdx` option. You must use **fid** only with the `--crc` option. *User who must use this option*: `mapr`
cid	*Description*: Checks data CRC for the master copy of the specified container ID. To check any other copy, use the `rIdx` option. The default value of `0` denotes that all containers are checked. You must use **cid** only with the `--crc` option. *User who must use this option*: `mapr`
rIdx	*Description*: Specifies the index (either `fid` or `cid`) of the copy of the data to check for errors. Use only with `-D` or `--crc` and either `fid` or `cid`. For example, `-D fid:2510.32.131204 rIdx=0` only checks the data for copy 1 of the specified fid. *User who must use this option*: `mapr`
fidThreads	*Description*: Specifies the number of threads for scanning fids (default:16, max:128). You must use **fidThreads** only with the `--crc` option. *User who must use this option*: `mapr`
cidThreads	*Description*: Specifies the number of threads for scanning container IDs (default:16, max:128). You must use **cidThreads** only with the `--crc` option. *User who must use this option*: `mapr`
scanthreads	*Description*: Specifies the number of threads for scanning inodes (default:10, max:1000) *User who must use this option*: Either `root` or `mapr`.
snapshot	*Description*: Specifies the name of the snapshot (default: null) *User who must use this option*: Either `root` or `mapr`.
snapshotid	*Description*: Specifies the snapshot ID (default: 0) *User who must use this option*: Either `root` or `mapr`.

Tier Options

last-updated: Apr 11, 2022

-Gquick|--check-tiermetadata-only	*Description*: Checks if the entries in the meta data tables maintained internally for objects and tiers (the mapping between the Virtual Cluster Descriptor (VCD) map and object map) , are consistent, and reports an error if not. *User who must use this option*: `mapr`
-Gfull|--check-tiermetadata-full	*Description*: Checks if the entries in the meta data tables maintained internally for objects and containers (the mapping between the VCD map and object map, along with the mapping between the VCD map and the MFS meta data), are consistent and reports an error if not. *User who must use this option*: `mapr`
-Dquick|--check-tierdata-presence	*Description*: Specified with either `-Gquick` or `-Gfull`. Checks and reports if the object in the meta data tables exists in the tier or not. *User who must use this option*: `mapr`
-Dfull|--check-tierdata-crc	*Description*: Specified with either `-Gquick` or `-Gfull`. Validates the data CRC for the object in the meta data tables. *User who must use this option*: `mapr`
-J|--skip-tier-log-replay	*Description*: Skips replaying transactions from internal dot files if a tier operation ends abruptly. MapR recommends that you use this option when running the GFSCK utility on tiered volumes. *User who must use this option*: Either `root` or `mapr`.

last-updated: Apr 11, 2022

Examples

1. **Debug Mode**

   In debug mode, run the `gfsck` command on the read/write volume named `mapr.cluster.root`:

```
/opt/mapr/bin/gfsck rwvolume=mapr.cluster.root -d
```

   Sample output is as follows:

```
Starting GlobalFsck:
  clear-mode            = false
  debug-mode            = true
  dbcheck-mode          = false
  repair-mode           = false
  assume-yes-mode       = false
  cluster               = my.cluster.com
  rw-volume-name        = mapr.cluster.root
  snapshot-name         = null
  snapshot-id           = 0
  user-id               = 0
  group-id              = 0

  get volume properties ...
    rwVolumeName = mapr.cluster.root (volumeId = 205374230,
rootContainerId = 2049, isMirror = false)

  put volume mapr.cluster.root in global-fsck mode ...

  get snapshot list for volume mapr.cluster.root ...

  starting phase one (get containers) for volume
mapr.cluster.root(205374230) ...
    container 2049 (latestEpoch=3, fixedByFsck=false)
    got volume containers map
  done phase one

  starting phase two (get inodes) for volume
mapr.cluster.root(205374230) ...
    get container inode list for cid 2049
      +inodelist: fid=2049.32.131224 pfid=-1.16.2 typ=4 styp=0 nch=0
dMe:false dRec: false
      +inodelist: fid=2049.33.131226 pfid=-1.16.2 typ=2 styp=0 nch=0
dMe:false dRec: false
      +inodelist: fid=2049.34.131228 pfid=-1.33.131226 typ=4 styp=0
nch=0 dMe:false dRec: false
      +inodelist: fid=2049.35.131230 pfid=-1.16.2 typ=4 styp=0 nch=0
dMe:false dRec: false
      +inodelist: fid=2049.36.131232 pfid=-1.16.2 typ=4 styp=0 nch=0
dMe:false dRec: false
      +inodelist: fid=2049.38.262312 pfid=-1.16.2 typ=2 styp=0 nch=0
dMe:false dRec: false
      +inodelist: fid=2049.39.262314 pfid=-1.38.262312 typ=1 styp=0
nch=0 dMe:false dRec: false
    got container inode lists (totalThreads=1)
  done phase two

  starting phase three (get fidmaps & tabletmaps) for volume
mapr.cluster.root(205374230) ...
    got fidmap lists (totalFidmapThreads=0)
    got tabletmap lists (totalTabletmapThreads=0)
  done phase three
```

```
=== Start of GlobalFsck Report ===

file-fidmap-filelet union --
2049.39.262314:P     --> primary (nchunks=0)      --> AllOk
no errors

table-tabletmap-tablet union --
empty

orphan directories --
none

orphan kvstores --
none

orphan files --
none

orphan fidmaps --
none

orphan tables --
none

orphan tabletmaps --
none

orphan dbkvstores --
none

orphan dbfiles --
none

orphan dbinodes --
none

containers that need repair --
none

incomplete snapshots that need to be deleted --
none

user statistics --
containers          = 1
directories         = 2
kvstores            = 0
files               = 1
fidmaps             = 0
filelets            = 0
tables              = 0
tabletmaps          = 0
schemas             = 0
tablets             = 0
segmaps             = 0
spillmaps           = 0
overflowfiles       = 0
bucketfiles         = 0
spillfiles          = 0

=== End of GlobalFsck Report ===

remove volume mapr.cluster.root from global-fsck mode (ret = 0) ...
```

```
GlobalFsck completed successfully (7142 ms); Result: verify succeeded
```

To verify if the object is present on the tier, run the `gfsck` command on the tiering-enabled read/write volume named `for_test5`:

```
/opt/mapr/bin/gfsck rwvolume=for_test5 -Gfull -Dquick
```

Sample output is as follows:

```
Starting GlobalFsck:
  clear-mode            = false
  debug-mode            = false
  dbcheck-mode          = false
  repair-mode           = false
  assume-yes-mode       = false
  cluster               = Cloudpool19
  rw-volume-name        = for_test5
  snapshot-name         = null
  snapshot-id           = 0
  user-id               = 2000
  group-id              = 2000

  get volume properties ...

  put volume for_test5 in global-fsck mode ...

  get snapshot list for volume for_test5 ...

  starting phase one (get containers) for volume
for_test5(16558233) ...
    got volume containers map

done phase one

  starting phase two (get inodes) for volume for_test5(16558233) ...
    got container inode lists
  done phase two

  starting phase three (get fidmaps & tabletmaps) for volume
for_test5(16558233) ...
    got fidmap lists
    got tabletmap lists
    completed secondary index field path info gathering
    completed secondary index consistency check
    Starting DeferMapCheck..
    completed DeferMapCheck
  done phase three

  === Start of GlobalFsck Report ===

  file-fidmap-filelet union --
    no errors

  table-tabletmap-tablet union --
    empty

  containers that need repair --
    none

  user statistics --
    containers          = 6
```

```
    directories         = 6
    files               = 1
    filelets            = 2
    tables              = 0
    tablets             = 0

  === End of GlobalFsck Report ===
Putting volume into TierGlobalFsck mode . . . . .

=== Start of TierGlobalFsck Report ===
TierVolumeGfsck completed, corruption not found
  total number of containers scanned           6
  total number of vcds verified                6722
  total number of objects verified             18
  total number of vcds skipped                 0
  total number of objects skipped              0
  total number of vcds that need repair        0
  total number of objects that need repair     0
=== End of TierGlobalFsck Report ===

removing volume from TierGlobalFsck mode
remove volume for_test5 from global-fsck mode (ret = 0)

GlobalFsck completed successfully (37039 ms); Result: verify succeeded
```

2. Verifying CRC of FIlelet

```
# /opt/mapr/bin/gfsck  -D fid=2085.32.131412  --debug
verifying data crc
  mode          =       fid
  fid           =       2085.32.131412
  debug-mode    =       true
  repair-mode   =       false
  cluster       =       default
  replication index     =       -1
  user-id       =       0
  group-id      =       0

crc validate result for fid : 2085.32.131412
  total local cluster/vcds verified : 51
  total local cluster/vcds corrupted : 0
  total local cluster/vcds skipped: 0
  total purged cluster/vcds verified : 0
  total purged cluster/vcds corrupted : 0
  total purged cluster/vcds skipped: 0
```

3. **Verifying CRC at a Container Level**

   For CRC checks at the container level, the output is not displayed on the terminal. Instead it is written to the `/opt/mapr/log/gfsck.log` file. Sample output is as follows:

```
/opt/mapr/bin/gfsck  -D rwvolume=rocky
verifying data crc
  mode          =       volume
  rwVolumeName          =       rocky
  fid thread count      =       16
  cid thread count      =       16
  debug-mode    =       false
  repair-mode   =       false
  cluster       =       default
  replication index     =       -1
  user-id       =       0
  group-id      =       0
  total containers : 6
  total container skipped : 0
  data crc verification completed with no errors
```

Related tasks

Using Global File System Checking on page 980
Describes how to use the `gfsck` command to check and repair filesystem errors.

guts

`guts` is a tool to measure/analyse performance. In the default mode, it prints one line every second, and counts the number of operations or bytes-processed in one second intervals. `guts` is an internal utility, and is subject to change without notice.

`guts` provides information on entities such as:

CPU	Indicates whether the CPU is idle or busy
RPCs	Number of RPCs, RPCs-in, RPCs-out, bytes-in, and bytes-out
MFS	Number of local-writes, local-reads, and other operations (such as lookup, create and remove)
Log	Number of log writes, log flushes, and log force-flushes
IO	Number of disk-operations/second (read/write), and the disk-io/second in MB (read/write)

Syntax

```
/opt/mapr/bin/guts
    guts
    -help
  instance:<id>  time:unix time:all time:none (add timestamp to output)
  key:5660 (is server port)
  shmid:<shared memory id> (client's shared memory id, check ipcs)
  threadcpu:core threadcpu:all
  cpu:none  cpu:all
  net:sum  net:msum  net:ksum  net:all  net:none
  disk:none disk:ops disk:mb disk:all
  diskMajor:major# of disk
  ssd:none ssd:all
  cache:none cache:small cache:med cache:all
  cleaner:small  cleaner:all  cleaner:none
```

```
  fs:rw   fs:all  fs:none
  kv:all  kv:none
  btree:all  btree:none
  allocator:all allocator:none
  rpc:none  rpc:op  rpc:all rpc:debug
  db:none db:op db:get db:put db:scan db:all
  dbrepl:none dbrepl:op dbrepl:all
  streams:none streams:op streams:all
  dsec:infinity (run time in sec.)
  period:n (output every n sec.)
  cache:small  cache:med  cache:all  cache:none
  log:all  log:none
  btree:all btree:none
  resync:all resync:none
  io:all io:small
  hb:all io:none
  gateway:all gateway:op gateway:lc gateway:none
  mastgateway:all mastgateway:tier mastgateway:db mastgateway:mfsops
mastgateway:none
  fstier:all fstier:none
  nfs:all nfs:none
  moss:all moss:basic moss:none
  client:none client:db client:fs client:all (requires shmid parameter)
  clientpid:<process id of a running client process>
  nfs4client:all
  fuse:all shmid:<shared memory id> (posix client's shared memory id, check
fuse logs)
  header:all  header:none (doesn't seem to work)
  flush:none  flush:line (if line, then output is flushed on every output
line)
defaults: time:none net:none disk:none rpc:op db:op db:put dbrepl:none
streams:none fs:rw cache:small kv:none
  cleaner:short log:none btree:none resync:none period:1
```

Interpreting Output

The prefix *c* identifies client metrics. The suffix *P* refers to the number of pending RPCs. The suffix *C* denotes the number of completed RPCs.

The pending metrics are a snapshot of pending RPCs when the output is printed. The completed metrics are the increase that happened in the last print interval.

Parameters and Output

CPU	cpu:`all` — Percentage of idle time of each CPU on the system in the last second.
IO	The metrics are `ior` and `iow`, which are displayed by default. • `ior` — The first number reports the number of I/O reads for a machine in the last second. The second number reports the amount of I/O reads in MB in the last second. • `iow` — The first number reports the number of I/O writes for a machine in the last second. The second number reports the amount of I/O writes in MB in the last second.

Disk

- `disk:ops` — Number of I/O requests (read+write) for each disk in the last second.
- `disk:mb` — Amount of I/O in MB (read+write) for each disk in the last second.
- `disk:all` — The preceding two numbers for each disk in the last second. The first number is from `disk:ops`, the second number is from `disk:mb`.

Filesystem

`fs:rw` — Reports MFS file system activities. Reported metrics are:

- `read` — The first number reports the number of remote reads in the last second. The second number reports the amount of data read in MB in the last second.
- `write` — The first number reports the number of remote writes in the last second. The second number reports the amount of data written in MB in the last second.
- `lread` / `lwrite` — are similar to the `read` and `write` metrics, but are applicable for local reads/ writes.

In addition, `guts` displays the following *filesystem* metrics:

- `crP` — Total pending *read* RPCs in the last second.
- `crC` — Total completed *read* RPCs in the last second.
- `cwP` — Total pending *write* RPCs in the last second.
- `cwC` — Total completed *write* RPCs in the last second.
- `ccP` — Total pending *create* RPCs in the last second.
- `ccC` — Total completed *create* RPCs in the last second.
- `cuP` — Total pending *unlink* RPCs in the last second.
- `cuC` — Total completed *unlink* RPCs in the last second.

RPC

Reports the following metrics:

- `rpc:none` — Does not display any RPC related metrics.
- `rpc:op` — *rpc* metric
- `rpc:all` — *rpc*, *im*, and *om* metrics.

- `rpc` — Number of RPC calls received in the last second.
- `im` — Amount of RPC calls received in MB in the last second.
- `om` — Amount of RPC calls sent in MB in the last second.

Cache

`cache:small` — Metrics on *inode* and *dentry* cache, which are displayed by default. The metrics reported are:

- `icache` (inode cache) — The first number reports the number of inode cache lookups in the last second. The second number reports the number of inode cache lookup misses in the last second.
- `dcache` (dentry cache) — The first and second numbers report dcache lookups and lookup misses in the last second, respectively.

Network

- `net:sum` — Total network traffic in bytes received and transmitted from all network interfaces for a machine.
- `net:msum` — Total network traffic in megabytes.
- `net:ksum` — Total network traffic in kilobytes.
- `net:all` — Not yet implemented.
- `net:none` — Does not display any network related metrics.

Metrics returned are:

- `nI` — Total amount of network traffic *received* in bytes in the last second. This is a summation of network traffic from all network interfaces for a machine.
- `nO` — Total amount of network traffic *sent* in bytes in the last second. This is a summation of network traffic from all network interfaces in a machine.

Database

`db:get` — Metrics related to `gets`. The output columns are as follows:

- `rOP` — Number of RPCs completed for type OP in the last second.
- `rOPR` — Number of rows processed from all RPCs of type OP in the last second.
- `tOPR` — Number of rows processed from all RPCs in the last second.
- `cOP` — Number of in-progress RPCs for the OP (not differential).

Cleaner Metrics

`guts` displays the following *cleaner* metrics:

last-updated: Apr 11, 2022

- `di` — Number of inodes dirtied by update operations in the last second.
- `ic` — Number of inodes cleaned by the drainer in the last second.
- `dd` — Number of data blocks dirtied by update operations in the last second.
- `dc` — Number of data blocks cleaned by the drainer in the last second.

Operational Metrics

`guts` displays the following *operational* metrics:

- `rput` — Number of *put* RPCs completed in the last second.
- `rputR` — Sum of *put* rows completed in the last second, from all *put* rpcs.
- `tputR` — Sum of *put* rows completed in the last second, from **all** rpcs (*put*, *increment*, *checkAndPut*, *Append* ..)
- `cput` — Number of *put* RPCs in progress currently. This is not a differential, but displays the number of outstanding *put* RPCs at that particular instant.
- `rget` — Number of *get* RPCs completed in the last second.
- `rgetR` — Sum of *get* rows completed in the last second, from all *get* RPCs.
- `tgetR` — Sum of *get* rows completed in the last second, from **all** rpcs (*get*, *increment*, *checkAndPut*, *Append* ..)
- `cget` — Number of *get* RPCs in progress currently. This is not a differential, but displays the number of outstanding *get* RPCs at that particular instant.
- `rsc` — Number of scan RPCs completed in the last second.
- `rscR` — Sum of scan rows returned in the last second, from **all** scan RPCs.
- `csc` — Number of scan RPCs currently in progress. This is not a differential, but shows the number of outstanding scan RPCs at that particular instant.
- `rinc` — Number of increment RPCs completed in the last second.
- `cinc` — Number of increment RPCs currently in progress. This is not a differential, but shows the number of outstanding increment RPCs at that particular instant.
- `rchk` — Number of *checkAndPut*/*checkAndDelete* RPCs completed in the last second.

- `rapp` — Number of append RPCs completed in the last second.
- `rtlk` — Number of tablet lookup RPCs completed in the last second.
- `ctlk` — Number of tablet lookup RPCs currently in progress. This is not a differential, but shows the number of outstanding lookup RPCs at that particular instant.
- `rbulkb` — Number of bulk-import-bucket RPCs completed in the last second.
- `rbulks` — Number of bulk-import-segment RPCs completed in the last second.

Put Metrics

`guts` displays the following *put* metrics:

- `rput` — Number of *put* RPCs completed in the last second.
- `rputR` — Sum of *put* rows completed in the last second, from all *put* rpcs.
- `tputR` — Sum of *put* rows completed in the last second, from **all** rpcs (*put*, *increment*, *checkAndPut*, *Append* ..)
- `cput` — Number of *put* RPCs in progress currently. This value is not a differential, but displays the number of outstanding *put* RPCs at that particular instant.
- `rsf` — Reserved free memory in MemIndex in MB. If this value falls very low, *put* RPCs can get throttled. This value is not a differential.
- `bucketWR`:
  - Column1 : Number of bucket writes (calls to MFS) in the last second.
  - Column2 : Amount of bucket writes in MB in the last second.
- `fl` — Number of bucket flushes fired in the last second.
- `ffl` — Number of force-flushes of buckets in the last second. If the bucket was flushed before it reached its optimal size, then the flush is counted as a force-flush.
- `sfl` — Number of segments touched by the bucket-flushes in the last second.
- `mcom` — Number of segments mini-packed in the last second.
- `fcom` — Number of segments packed fully in the last second.

last-updated: Apr 11, 2022

- `ccom` — Number of segment packs running currently. This value is not a differential.
- `scr` — Number of segment creates in the last second.
- `spcr` — Number of spill creates in the last second.

Get Metrics

`guts` displays the following *get* metrics:

- `rget` — Number of *get* RPCs completed in the last second.
- `rgetR` — Sum of *get* rows completed in the last second, from all *get* RPCs.
- `tgetR` — Sum of *get* rows completed in the last second, from **all** rpcs (*get*, *increment*, *checkAndPut*, *Append* ..)
- `cget` — Number of *get* RPCs currently in progress. This is not a differential, but displays the number of outstanding *get* RPCs at that particular instant.
- `vcM` — Size of the value-cache in MB. This value is not differential.
- `cL` — Number of value-cache lookups in the last second.
- `vcH` — Number of value-cache hits in the last second.
- `bget` — Number of bucket *gets* in the last second. Will be 0 if there are no active buckets.
- `sg` — Number of segment *gets* in the last second. Will normally be equal to `tgetR minus the number of value-cache hits`.
- `spg` — Number of spill *gets* in the last second. This value is calculated as `sigma(segments * spill-per-segment) - bloomFilterSkips`
- `bskp` — Number of spill *gets* that were avoided/ saved by the bloom filter in the last second.

Scan Metrics

`guts` displays the following *scan* metrics:

- `rsc` — Number of scan RPCs completed in the last second.
- `rscR` — Sum of scan rows returned in the last second, from **all** scan RPCs.
- `csc` — Number of scan RPCs currently in progress. This is not a differential, but shows the number of outstanding scan RPCs at that particular instant.
- `bsc` — Number of buckets scanned in the last second.

- `ssc` — Number of segments scanned in the last second.
- `spsc` — Number of spills scanned in the last second.
- `spscR` — Number of rows scanned from spills in the last second.
- `ldbr` — Number of *ldb* blocks read in the last second.
- `blkr` — Number of data blocks read in the last second (over *spills*, *buckets* ..)
- `raSg` — Number of segments for which read-ahead was done in the last second.
- `raSp` — Number of spills for which read-ahead was done in the last second.
- `nAdv` — Number of *fadvise* calls made to MFS for scan read-ahead in the last second.
- `raBl` — Sum of blocks in the *fadvise* calls made to MFS for scan read-ahead in the last second.

Cumulative Metrics

`guts` displays the following *cumulative* metrics:

- cmP — Total pending RPCs from the client in the last second.
- cmC — Total completed RPCs from the client in the last second.

DB Metrics

`guts` displays the following *database* metrics:

- `cgP` — Total pending *get* RPCs.
- `cgC` — Total completed *get* RPCs.
- `cpP` — Total pending *put* RPCs.
- `cpC` — Total completed *put* RPCs.
- `csP` — Total pending *scan* RPCs.
- `csC` — Total completed *scan* RPCs.
- `ciP` — Total pending *increment* RPCs.
- `ciC` — Total completed *increment* RPCs.
- `caP` — Total pending *append* RPCs.
- `caC` — Total completed *append* RPCs.
- `cgR` — Total client *get* rows.
- `cpR` — Total client *put* rows.
- `csR` — Total client *scan* rows.

last-updated: Apr 11, 2022

- `ciR` — Total client *increment* rows.
- `caR` — Total client *append* rows.

Example Usage

The following example demonstrates viewing client metrics. Perform the following steps:

1. Find the process ID of the client program.
2. Find all the shared memory segments (*shmem*) for this program:

```
ipcs -mp | grep <pid>
  998080521  root       30030      21850
  998113290  root       30030      30030
  ^^^^^^^^^
  shmem ID
```

   Here, there are two shared memory segments — one between the client and MFS, and the other between the client and `guts`.

3. Identify the correct *shmem* segment for `guts`:

```
ipcs | grep 998113290
  0x00000000 998113290  root       666        2288       1          dest
ipcs | grep 998080521
  0x00000000 998080521  root       660        20971520   1          dest
                                              ^^^^^^^^
                                              size
```

   The *shmem* with size 20M is between client and MFS. Here, we select *shmem* with ID 998113290.

4. Run `guts`:

```
/opt/mapr/bin/guts client:all shmid:998113290
  Printing only client statistics
  cmP    cmC    cgP    cgC    cpP    cpC    csP    csC    ciP    ciC
caP   caC    crP    crC    cwP    cwC    ccP    ccC    cuP    cuC
    0      0      0      0      0      0      0      0      0
0      0      0      0      0      0      0      0      0      0      0
    0      0      0      0      0      0      0      0      0
0      0      0      0      0      0      0      0      0      0      0
```

   Pass the *shmem* ID and one of the client options. Client options are one of:

   - `none` — Used when printing MFS/dbserver statistics
   - `db` — Prints client statistics for DB operations
   - `fs` — Prints client statistics for filesystem operations
   - `all` — Prints all client statistics

CLDB Guts

The `cldbguts` utility prints information about active container reports, full container reports, registration requests, MapR-FS heartbeats, NFS server heartbeats, and containers. For more information, see cldbguts on page 2080.

last-updated: Apr 11, 2022

NFS Guts

`guts` displays the following *NFS* metrics:

- `req` — Number of requests received from all the NFS clients to this NFS server in the last second.
- `dpC` — Number of dropped calls from NFS client due to running out of ONC handles (probably cluster is responding slow OR NFS client is bombarding the NFS server).
- `inReadReq` — Number of incoming read requests from NFS clients.
- `outReadResp` — Number of outgoing read request responses to NFS Clients.
- `inReadDataReq` — Size/Length of incoming read requests (buffer size) from NFS Clients.
- `outReadDataResp` — Size/Length of outgoing read request response (buffer size) to NFS Clients.
- `inWriteReq` — Number of incoming write requests from NFS clients.
- `outWriteResp` — Number of outgoing read request responses to NFS clients.
- `inWriteDataReq` — Size/Length of incoming write request (buffer size) from NFS clients.
- `outWriteDataResp` — Size/Length of outgoing write request response (buffer size) to NFS Clients.

Running Guts

Start `guts` on the node for which you need to collect metrics.

```
/opt/mapr/bin/guts
  00 01 02 03 04 05 06 07   rpc   lpc     write   lwrite   bwrite
read    lread      icache       dcache     di  ic    dd  dc       ior
iow  rput  rputR  cput  tputR  rget  rgetR  cget  tgetR   rsc    rscR  csc
  86 90 84 84 87 93 81 84     5     6     0   0    1   0    0   0     0
0    3   0     8     0    163     1    337  22    13  16     1   0   73
4     0      0     0      1     0      0     0      0     0       0    0
  62 77 70 82 93 61 50 84    12    20     0   0    3   0    0   0     0
0   10   0    27     0     41     0      6   0     3   0     0   0    0
0     0      0     0      3     0      0     0      0     0       0    0
  63 78 59 56 84 64 32 86     4     5     0   0    5   0    0   0     0
0    0   0     5     0     27     0      8   0    22   0     0   0    0
0     3   1506     0   1506     0      0     0      0     0       0    0
  83 76 77 82 68 69 82 67     1     0     0   0    0   0    0   0     0
0    0   0     0     0      0     0      0   0     0   0     0   0    0
0     0      0     0      0     0      0     0      0     0       0    0
  94 49 91 56 75 48 57 92     1     0     0   0    0   0    0   0     0
0    0   0     0     0      0     0      0   0     0   0     0   0    0
0     0      0     0      0     0      0     0      0     0       0    0
  97 96 99 89 93 94 82 95     2     0     0   0    1   0    0   0     0
0    0   0     1     0      8     0      2   0     1   0     0   0    0
0     0      0     0      0     0      0     0      0     0       0    0
  99 99 96 97 99 98 99 82    19     6     0   0    1   0    0   0     0
0    3   0   186     0     18     0      0   0     0   0     0   0    0
0     0      0     0      1     0      0     0      0     0       0    0
```

To stop collecting metrics, press ^*C*.

Related reference

cldbguts on page 2080
Monitors the activity of the Container Location Database (CLDB). This utility prints information about the CLDB service that is running on the node from which you run the utility.

last-updated: Apr 11, 2022

Retrieving Tiering Statistics Using guts on page 943
Explains how to use the `guts` utility to retrieve tiering statistics.

manageSSLKeys.sh
Use the `manageSSLKeys.sh` utility to create and manage SSL certificates.

Syntax

```
# /opt/mapr/server/manageSSLKeys.sh
manageSSLKeys.sh is a tool to create and manage the SSL certificates.
it is run once on the first node from configure.sh
Usage: manageSSLKeys and one of
     create [-d DNSDOMAIN] [-N clustername]
        creates the SSL key and trust stores needed for HTTPS traffic
        -d specifies DNS domain used in wildcard certificate. Default is
detected from Local OS
        -N clustername
        -ug MapR user/group, e.g., mapr:mapr
     merge <in trust store> <out trust store>
        merges the certificates from the in trust store into the existing
out trust store
     convert [-N <clustername> ] [-k] [-n] [-p <passwd>] [-srcType JKS|
pkcs12] [-dstType JKS|pkcs12] <in key/trust store> <out key/trust store>
        converts an existing key/trust store into a new PEM type key/trust
store
        if srcType and dstType are not specified, it is assumed that you
are
        converting from JKS to PEM(via pkcs12)
        -N <clustername>
        -k denotes you are converting a keystore
        -n Skips creating the PEM file when converting the key/trust store
from JKS to PKCS12 format
        -p <passwd> store password - needed if you are converting custom
stores
        -srcType JKS|pkcs12 denotes the source format of the store
        -dstType JKS|pkcs12 denotes the destination format of the store
```

Operations

`manageSSLKeys.sh` performs the following operations:

create

Description: Creates the SSL key and trust stores needed for HTTPS traffic.

Format: `create [-d DNSDOMAIN] [-N clustername] -ug <maprUserGroup>`

Parameters:

- `d`: DNS domain used for the wildcard certificate. The default domain is detected from the Local OS.
- `N`: Name of the cluster.
- ug: *User:Group* to use for the key. For example: *mapr:mapr*.

merge

Description: Merges the SSL certificates from the *in trust* store into the existing *out trust* store.

Format: `merge <in trust store> <out trust store>`

Parameters:

last-updated: Apr 11, 2022

- `in trust store`: Source trust store from which to obtain the SSL certificates.
- `out trust store`: Destination trust store to merge the SSL certificates.

convert

Description: Converts an existing key/trust store into a new PEM type key/trust store. If you do not specify the type of the source and the destination key/trust store, it is assumed that you are converting from JKS to PEM (via `pkcs12`).

Format:
```
convert [-N <clustername> ] [-k] [-n] [-p <passwd>] [-srcType JKS|pkcs12] [-dstType JKS|pkcs12] <in key/trust store> <out key/trust store>
```

Parameters:

- `N`: Cluster name.
- `k`: Indicates that a keystore is being converted.
- `n`: Skips creating the PEM file when converting the key/trust store from JKS to PKCS12 format
- `p <passwd>`: store password - needed if you are converting custom stores.
- `srcType`: Format of the source key/trust store - either `JKS` or `pkcs12`.
- `dstType`: Format of the destination key/trust store - either `JKS` or `pkcs12`.
- `in key/trust store`: The existing key/trust store to convert.
- `out key/trust store`: The name to use for the converted key/trust store.

Examples

The following links demonstrate using the `manageSSLKeys.sh` utility.

- Merge trust store: Configuring Secure Clusters for Running Commands Remotely on page 1484
- Generate trust store and key store files: Step 1: Restart and Check Cluster Services on page 323 and Configuring Encryption for ODBC Connection on page 3535
- Convert type of keystore file: Upgrading the MapR Data Access Gateway on page 361

mapr-support-collect.sh
Collects information about a cluster's recent activity, to help MapR Support diagnose problems.

The "mini-dump" option limits the size of the support output. When the `-m` or `--mini-dump` option is specified along with a size, `mapr-support-collect.sh` collects only a head and tail, each limited to the specified size, from any log file that is larger than twice the specified size. The total size of the output is therefore limited to approximately 2 * size * number of logs. The size can be specified in bytes, or using the following suffixes:

- b - bytes

last-updated: Apr 11, 2022

- k - kilobytes (1024 bytes)
- m - megabytes (1024 kilobytes)

Syntax

```
 /opt/mapr/support/tools/mapr-support-collect.sh
       -h, --hosts HOST_FILE
          hosts file, each line has entries [user@]host[:port]
       -H, --host HOST_ENTRY
          additional host entry of the form [user@]host[:port], multiple
can be specified
       -Q, --no-cldb
          do not query CLDB for list of nodes
       -n, --name NAME
          name of output file
       -d, --output-dir DIR_PATH
          absolute path of output directory
       -l, --no-logs
          do not include log files
       --no-hadoop-logs
          do not include hadoop log files
       --hbase-logs
          include hbase log files
       --sqoop-logs
          include sqoop log files
       --eco-logs
          include all ecosystem log files
       --oozie-logs
          include oozie log files
       --spark-logs
          include spark log files
       --pig-logs
          include pig log files
       --impala-logs
          include impala log files
       --hue-logs
          include hue log files
       --hive-logs
          include hive log files
       --flume-logs
          include flume log files
       --drill-logs
          include drill log files
       --no-kibana-logs
          do not include kibana log files
       --no-grafana-logs
          do not include grafana log files
       --no-elasticsearch-logs
          do not include elasticsearch log files
       --no-opentsdb-logs
          do not include opentsdb log files
       --no-collectd-logs
          do not include collectd log files
       --no-fluentd-logs
          do not include fluentd log files
       --no-vol-info
          do not collect volume information
       -L, --libraries
          include libraries
       -s, --no-statistics
          do not include statistics
       -c, --no-conf
```

```
          do not include configurations
       -i, --no-sysinfo
          do not include system information
       -x, --exclude-cluster
          do not collect cluster diagnostics
       -u, --user USER
          username for ssh connections
       -K, --strict-hostkey
          check for strict host key in ssh connection
       -p, --par PAR
          maximum number of nodes from which support dumps will be gathered
concurrently (default: 10)
       -t, --dump-timeout DUMPTIMEOUT
          timeout for execution of mapr-support-dump command on a node
(default: 3600 seconds, 0 = no limit)
       -T, --scp-timeout SCPTIMEOUT
          timeout for copy of support dump output from a remote node to
local filesystem (default: no limit)
       -y, --yes
          do not require acknowledgement of the number of nodes that will
be affected
       -O, --online
          Leverage MapR APIs to gather support dumps. When not specified,
SSH and SCP will be used.
       -S, --scp-port SCPPORT
          the local port to which remote nodes will establish an SCP session
       --collect-cores
          Collect cores of running mfs processes from all nodes (off by
default)
       --move-cores
          Move mfs and nfs cores from coresDir from all nodes (off by
default)
       --use-hostname
          Use hostname to ssh instead of IP addresses (off by default)
       --cldb CLDBNODE
          Use this option when the CLDB Service is down to point to a CLDB
node
       --port PORT
          port number used by FileServer (default: 5660)
       --nfsport NFS_MGMT_PORT
          port number used by NFSServer (default: 9998)
       -m, --mini-dump SIZE
          Collects only first and last number of bytes of each log file if
file is greater then 2*SIZE.
          SIZE may have a multiplier suffix: b 512, k 1024, m 1024*1024
       -A, --logs-age DAYS
          Use this option to collect logs newer than specified DAYS
          (default: 7, nolimit: 0)
       -f, --filter FILTER_STRING
          Use this option to filter nodes for which support dump should be
collected
       -?, --help
          display usage
```

Parameters

Parameter	Description
-h or --hosts	A file containing a list of hosts. Each line contains one host entry, in the format `[user@]host[:port]`.

last-updated: Apr 11, 2022

Parameter	Description
-H or --host	One or more hosts in the format `[user@]host[:port]`.
-Q or --no-cldb	If specified, the command does not query the CLDB for list of nodes.
-n or --name	Specifies the name of the output file. If not specified, the default is a date-named file in the format `YYYY-MM-DD-hh-mm-ss.tar`.
-d or --output-dir	The absolute path to the output directory. The default path is `/opt/mapr/support/collect/`.
-l or --no-logs	If specified, the command output does not include any log files.
--no-hadoop-logs	If specified, the command output does not include Hadoop log files.
--hbase-logs	If specified, the command output includes HBase log files.
--sqoop-logs	If specified, the command output includes Sqoop log files.
--eco-logs	If specified, the command output includes the log files for all MapR ecosystem components.
--oozie-logs	If specified, the command output includes Oozie log files.
--spark-logs	If specified, the command output includes Spark log files.
--pig-logs	If specified, the command output includes Apache Pig log files.
--impala-logs	If specified, the command output includes Apache Impala log files.
--hue-logs	If specified, the command output includes Apache Hue log files.
--hive-logs	If specified, the command output includes Apache Hive log files.
--flume-logs	If specified, the command output includes Apache Flume log files.
--drill-logs	If specified, the command output includes Apache Drill log files.
--no-kibana-logs	If specified, the command output does not include Kibana log files.
--no-grafana-logs	If specified, the command output does not include Grafana log files.
--no-elasticsearch-logs	If specified, the command output does not include Elasticsearch log files.
--no-opentsdb-logs	If specified, the command output does not include OpenTSDB log files.
--no-collectd-logs	If specified, the command output does not include Collectd log files.
--no-fluentd-logs	If specified, the command output does not include Fluentd log files.

last-updated: Apr 11, 2022

Parameter	Description
--no-vol-info	If specified, the command output does not include any volume information.
-L or --no-libraries	If specified, the command output does not include libraries.
-c or --no-conf	If specified, the command output does not include configurations.
-i or --no-sysinfo	If specified, the command output does not include system information.
-x or --exclude-cluster	If specified, the command does not collect cluster diagnostics. Even if cluster diagnostics are excluded, the script still collects local logs and local system diagnostic information.
-u or --user	The username for ssh connections.
-K or --strict-hostkey	If specified, checks for strict host key in the SSH connection. When specified, ssh never automatically adds host keys to the `~/.ssh/known_hosts` file, and refuses to connect to a host whose host key has changed. This provides maximum protection against trojan horse attacks, but can be troublesome when the `/etc/ssh/ssh_known_hosts` file is poorly maintained or connections to new hosts are frequently made. This option forces the user to manually add all new hosts.
-p or --par	The maximum number of nodes from which support dumps are gathered concurrently (default: 10).
-t or --dump-timeout	The timeout in seconds for execution of the `mapr-support-dump` command on a node (default: 3600 seconds or 0 = no limit).
-T or --scp-timeout	The timeout in seconds for copy of support dump output from a remote node to the local filesystem (default: no limit).
-y or --yes	If specified, the command does not require acknowledgement of the number of nodes that are affected.
-O or --online	Specifies a space-separated list of nodes from which to gather support output, and uses the MapR APIs instead of ssh for transmitting the support data.
-S or --scp-port	The local port to which remote nodes establish a SCP session. The default port is 22.
--collect-cores	If specified, the command collects cores of running MFS processes from all nodes (default: off).
--move-cores	If specified, the command moves MFS and NFS cores from /opt/cores to /opt/mapr/logs/cores/ on all nodes (default: off).

last-updated: Apr 11, 2022

Parameter	Description
--use-hostname	If specified, uses hostnames instead of IP address for SSH (default: off).
--cldb <cldbnode>	Use this option when the CLDB Service is down to point to a CLDB node.
--port	The port number used by FileServer (default: 5660).
--nfsport	The port number used by NFSServer (default: 9998).
-m or --mini-dump <size>	For any log file greater than 2 * <size>, collects only a head and tail each of the specified size. The <size> may have a suffix specifying units: • b - blocks (512 bytes) • k - kilobytes (1024 bytes) • m - megabytes (1024 kilobytes)
-A or --logs-ag	Use this option to collect logs newer than specified days (default: 7, nolimit: 0)
-f or --filter <filter string>	Use this option to specify a filter string. Support information is only collected for nodes with names that match the filter string.
-? or --help	Displays usage help text

Examples

Collect support information and dump it to the file /opt/mapr/support/collect/mysupport-output.tar:

```
/opt/mapr/support/tools/mapr-support-collect.sh -n mysupport-output
2019-09-16 21:37:28.884 INFO Creating nodes file
2019-09-16 21:37:28.907 INFO Querying CLDB for nodes in the cluster
2019-09-16 21:37:31.883 INFO Created nodes file
Diagnostics will be collected from 1 nodes. Press enter to continue:
2019-09-16 21:37:35.650 INFO Collecting cluster information
2019-09-16 21:38:35.282 INFO Collecting dump on <ip>
Password:
2019-09-16 21:41:12.917 INFO Copying dump from <ip>
Password:
2019-09-16 21:44:25.137 INFO Making a tarball of all the dumps

--------------------------------------------------------------------------------

------------------ Finished collecting diagnostics
information --------------------
--------------------------------------------------------------------------------

Successfully collected support information on cluster from CLDB.

Total no. of nodes from which dump collection was attempted: 1
Nodes from which support information gathering succeeded:    1
Number of nodes from which dump collection failed:           0
Number of nodes from which dump file could not be copied:    0

The tar ball of the dumps is available at: /opt/mapr/support/collect/
mysupport-output.tar
```


last-updated: Apr 11, 2022

mapr-support-dump.sh
Collects node and cluster-level information for the node on which you invoke the script.

The information collected is used to help MapR Support diagnose problems. Use mapr-support-collect.sh on page 2121 to collect diagnostic information from all nodes in the cluster.

The "mini-dump" option limits the size of the support output. When you specify the `-m` or `--mini-dump` option along with a size, `mapr-support-dump.sh` collects only a head and tail, each limited to the specified size, from any log file that is larger than twice the specified size. The total size of the output is therefore limited to approximately 2 * size * number of logs. You can specify the size using the following suffixes:

- b - bytes
- k - kilobytes (1024 bytes)
- m - megabytes (1024 kilobytes)

Syntax

```
/opt/mapr/support/tools/mapr-support-dump.sh
      [ -n| --name <name> ]
      [ -l| --no-logs ]
      [ --no-hadoop-logs ]
      [ --hbase-logs ]
      [ --sqoop-logs ]
      [ --eco-logs ]
      [ --oozie-logs ]
      [ --spark-logs ]
      [ --pig-logs ]
      [ --impala-logs ]
      [ --hue-logs ]
      [ --hive-logs ]
      [ --flume-logs ]
      [ --drill-logs ]
      [ -L| --libraries ]
      [ -d| --output-dir <path> ]
      [ -s| --no-statistics ]
      [ -c| --no-conf ]
      [ -i| --no-sysinfo ]
      [ -o| --exclude-cluster ]
      [ -O| --online ]
      [ -z| --only-cluster ]
      [ --collect-cores ]
      [ --move-cores ]
      [ --port <port> ]
      [ --nfsport <port> ]
      [ -m| --mini-dump <size> ]
      [ -A| --logs-age <days> ]
      [ -?| --help ]
```

Parameters

Parameter	Description
-n or --name	Specifies the name of the output file. If not specified, the default is a date-named file in the format YYYY-MM-DD-hh-mm-ss.tar
-l or --no-logs	Does not include log files.

last-updated: Apr 11, 2022

Parameter	Description
--no-hadoop-logs	Does not include Hadoop log files
--hbase-logs	Includes Hbase log files.
--sqoop-logs	Includes Sqoop log files
--eco-logs	Includes all ecosystem log files.
--oozie-logs	Includes Oozie log files.
--spark-logs	Includes Spark log files.
--pig-logs	Includes Pig log files.
--impala-logs	Includes Impala log files.
--hue-logs	Includes Hue log files.
--hive-logs	Includes Hive log files.
--flume-logs	Includes Flume log files.
--drill-logs	Includes Drill log files.
-L or --libraries	Includes libraries.
-d or --output-dir	The absolute path to the output directory. If not specified, the default is `/opt/mapr/support/collect/`
-s or --no-statistics	Does not include statistics.
-c or --no-conf	Does not include configurations.
-i or --no-sysinfo	Does not include system information.
-o or --exclude-cluster	Does not collect cluster diagnostics.
-O or --online	Saves the support dump output file to the /var/mapr/cluster/support directory in maprfs. When not specified, the output file is stored at /opt/mapr/support/dump on the local machine filesystem. Specifies a space-separated list of nodes from which to gather support output, and uses the warden instead of ssh for transmitting the support data.
-z or --only-cluster	Collects diagnostic information at the cluster level only.
--collect-cores	Collects cores of running mfs processes from all nodes (off by default)
--move-cores	Moves mfs and nfs cores from /opt/cores from all nodes (off by default)
--port	The port number used by the FileServer. Default: 5660
--nfs-port	The port used by the NFS server. Default: 9998

Parameter	Description
-m, --mini-dump <size>	For any log file greater than 2 * <size>, collects only a head and tail each of the specified size. The <size> may have a suffix specifying units: • b - blocks (512 bytes) • k - kilobytes (1024 bytes) • m - megabytes (1024 kilobytes)
-A or --logs-age <days>	Collects logs newer than the specified number of days. The default value for this parameter is 7. Specify a value of 0 to have the `mapr-support-dump.sh` script collect logs of any age.
-? or --help	Displays usage help text

Output

The example produces a tar file at `/opt/mapr/support/dump/mysupport-output.tar`. To extract the tar file, use `tar -xf mysupport-output.tar`. The directory structure is as follows:

```
$ ls
mysupport-output    mysupport-output.tar

$ ls mysupport-output
MapRBuildVersion       hostid            mfsstate_commands
cluster                hostname          roles
cluster_summary.txt    linux-release     support_dump.log
conf                   logs              system_info
conf_file_metadata     mapr-clusters.conf
```

- **mfsstate_commands** - Contains all information collected about the state of the filesystem are stored the the mfsstate_commands directory.
- **cluster** - Contains information associated with maprcli_commands, resourcemanager, and volume_dumps.
- **system_info** - Contains all information related to the system state and configuration.
- **support_dump.log** - Contains console output of the individual commands launched by the support dump script.
- Commonly required files - cluster_summary.txt, conf_file_metadata, hostid, hostname, linux-release, MapRBuildVersion, mapr-clusters.conf, and support_dump.log are place in the dump root for easier access.

Example

Collect support information and dump it to the file /opt/mapr/support/collect/mysupport-output.tar:

```
/opt/mapr/support/tools/mapr-support-dump.sh -n mysupport-output
```

Example Output

```
/opt/mapr/support/tools/mapr-support-dump.sh -n mysupport-output
2015-06-25 11:51:38.397 INFO Starting Support dump collection. For
```


last-updated: Apr 11, 2022

```
diagnostics, refer to supportèdump.log inside the dump
2015-06-25 11:51:38.402 INFO Collecting system information
2015-06-25 11:51:49.596 INFO Collecting mapr logs
2015-06-25 11:52:00.734 INFO Log collection from maprfs is succesful.
2015-06-25 11:52:00.804 INFO Collecting cluster configuration
2015-06-25 11:53:07.626 INFO Skipping /opt/mapr/logs/prerequisitecheck.log
since it does not exist
2015-06-25 11:53:07.639 INFO Skipping /opt/mapr/hbase/hbase-0.94.17/conf
since it does not exist
2015-06-25 11:53:10.436 INFO Collecting cluster summary information
2015-06-25 11:53:19.183 INFO Collecting detailed cluster information
2015-06-25 11:54:03.219 INFO Dump collection is succesful. Tar file: /opt/
mapr/support/dump/mysupport-output.tar
```

maprlogin
Authenticates logins to secure MapR clusters.

The `/opt/mapr/bin/maprlogin` command line tool enables users to log into secure MapR clusters. Users authenticate themselves to the cluster with a `maprticket` that can be generated in the following ways:

- Run `maprlogin password` to authenticate with username and password.
- Run `maprlogin generateticket` to request a service, tenant, or cross-cluster ticket for use by an external application or user account (based on the current user's ticket).
- Run `maprlogin kerberos` after generating a Kerberos ticket with the `kinit` command.

Note: Tickets contain keys, and are used to authenticate users and MapR servers. Every user who wants to access a cluster must have a MapR user ticket (`maprticket_<uid>`) and every node in the cluster must have a MapR server ticket (`maprserverticket`).

For more details about different ways to generate tickets, see Tickets.

Syntax

```
/opt/mapr/bin/maprlogin <argument> <option>
```

Arguments

Argument	Description
`authtest`	Simulates runtime behavior during authentication. The following is the syntax for running the `maprlogin` command with this argument: `/opt/mapr/bin/maprlogin authtest [ -cluster mapr cluster name ]` For more information, see Options on page 2132.
`end|logout`	Logs out of the cluster. The following is the syntax for running the `maprlogin` command with this argument: `/opt/mapr/bin/maprlogin end|logout [ -cluster mapr cluster name ]` For more information, see Options on page 2132.

Argument	Description

`generateticket`	Generates a ticket for another user or application. The user who runs the `maprlogin` command with this option must already have a user ticket and must have `fc` (full control) ACL authorization on the cluster. See acl set.
	The following is the syntax for running the `maprlogin` command with this argument:
	`/opt/mapr/bin/maprlogin generateticket`
	`-type service|crosscluster|servicewithimpersonation|tenant`
	`-user <UNIX user name>`
	`[ -cluster <cluster name> ]`
	`-out <ticket location>`
	`[ -duration <[Days:]Hours:Minutes OR -duration Seconds> ]`
	`[ -renewal <[Days:]Hours:Minutes OR -duration Seconds> ]`
	`[ -impersonateduids <uids to impersonate> ]`
	`[ -impersonatedgids <gids to impersonate> ]`
	For more information, see Options on page 2132.

`kerberos`	Indicates the presence of a Kerberos ticket. The following is the syntax for running the `maprlogin` command with this argument:
	`/opt/mapr/bin/maprlogin kerberos`
	`[ -cluster <cluster name> ]`
	`[ -duration <ticket duration> ]`
	For more information, see Options on page 2132.

`password`	The user's UNIX password. The following is the syntax for running the `maprlogin` command with this argument:
	`/opt/mapr/bin/maprlogin password`
	`[ -cluster <cluster name> ]`
	`[ -user <user name> ]`
	`[ -duration <ticket duration> ]`
	`[ -out <ticket location> ]`
	For more information, see Options on page 2132.

`print`	Prints ticket of any type and contains information including the cluster name, the user ID, the date when the ticket was created, the ticket expiration date, and whether user can impersonate other users, and whether the ticket is for a tenant.
	In the service tickets, the value for `CanImpersonate` is `true` if impersonation is enabled for user and `false` if impersonation is disabled for the user. In the regular cluster ticket for the user, the value of `CanImpersonate` is always `false`. In the tenant ticket, the value for `CanImpersonate` is always `true`.
	The following is the syntax for running the `maprlogin` command with this argument:
	`/opt/mapr/bin/maprlogin print`
	`[ -ticketfile <location of ticket file> ]`
	For more information, see Options on page 2132.

last-updated: Apr 11, 2022

Argument	Description

`renew`	Renews the ticket, given a duration that does not cause the ticket to exceed its maximum lifetime. The original `-renewal` value for the ticket determines its maximum lifetime. The following is the syntax for running the `maprlogin` command with this argument:
	`/opt/mapr/bin/maprlogin renew`
	`[ -cluster <cluster name> ]`
	`[ -duration <ticket renew duration> ]`
	`[ -ticketfile <input ticket file> ]`
	`[ -out <ticket location> ]`
	For more information, see Options on page 2132.

Options

Option	Description	Default
`-cluster`	Name of the cluster to log into.	First cluster name in the `/opt/mapr/conf/mapr-clusters.conf` file.

`-duration`	Length of time before the ticket expires, specified in one of the following formats:
	`-duration [Days:]Hours:Minutes`
	`- duration Seconds`
	Password-generated tickets are bounded by the CLDB duration and renewal properties that are set for the cluster:
	• `cldb.security.user.ticket.duration.seconds` (default=`1209600`) is used if duration is not specified while generating the ticket.
	• `cldb.security.user.ticket.max.duration.seconds` (default=`2592000`) is the maximum duration allowed for a ticket.
	For password-generated tickets, if `-duration` is not set with the `maprlogin` command, the CLDB duration property is used by default.
	See config.
	Note: The `service`, `servicewithimpersonation`, tenant, and `crosscluster` tickets may have a very long lifetime; their duration is not bounded by these properties. For service and crosscluster tickets, the default value is LIFETIME.
		• 1209600 seconds (14 days) for user tickets
		• LIFETIME for service and cross-cluster tickets

last-updated: Apr 11, 2022

Option	Description	Default
`-impersonatedgids`	The comma-separated list of GIDs to impersonate. This can only be specified when generating a servicewithimpersonation ticket. If this is specified, the ticket owner can only impersonate the specified groups or users belonging to the specified groups. If `impersonatedgids` and `impersonareduids` are not specified, the ticket holder can impersonate all users on the cluster except the root user or the mapr user.	No default
`-impersonateduids`	The comma-separated list of UIDs to impersonate. This can only be specified when generating a servicewithimpersonation ticket. If this is specified, the ticket owner can only impersonate the specified users. If `impersonatedgids` and `impersonareduids` are not specified, the ticket holder can impersonate all users on the cluster except the root user or the mapr user.	No default
`-out`	A safe directory location where the ticket will be stored. Can be used with `generateticket`, `password`, and `renew` commands. You must specify a location when generating service and tenant tickets. (This requirement ensures that other tickets are not overwritten.)	`/tmp/maprticket_<uid>` (default applies to non-service tickets only)

`-renewal`	Total lifetime of the ticket, specified in one of the following formats:
	`-renewal [Days:]Hours:Minutes`
	`-renewal Seconds`
	If `-renewal` is not set with the `maprlogin` command, the CLDB renewal property is set by default (`cldb.security.user.ticket.renew.duration.seconds`). You can also set the `cldb.security.user.ticket.renew.max.duration.seconds` property, which is the maximum duration (7776000, by default) allowed for a ticket renewal.
	Note: Service, tenant, and crosscluster tickets are not bounded by these properties.
	For example, assume that the `maprlogin` command passes the following options for a service ticket:
	`-duration 30:0:0 -renewal 90:0:0`
	The ticket will expire after 30 days unless it is renewed. If a `maprlogin renew` command is submitted for the ticket before the initial 30 days pass, the ticket's lifetime may be extended up to a total maximum lifetime of 90 days. Tickets do not renew automatically; administrators must renew them with the `maprlogin renew` command, specifying a valid renewal period, and they must do this before the duration period ends. The renewal period must be less than or equal to the remaining amount of time allowed on the ticket.
	Using the same example, if you renew a ticket on the 29th day of its life, you can renew it for up to 61 days. You can renew a ticket incrementally, for some number of days at a time, as long as you do not exceed the original renewal value.
		2592000 seconds (30 days)

Option	Description	Default
`-ticketfile`	Optional with `print` and `renew` commands. Specifies the path to ticket file, if different from default. If this is not specified, the command looks for the ticketfile (`maprticket_<uid>`) in the default location, which is `/tmp` on Linux and `%TEMP%` on Windows systems or in the location specified by the environment variable, `$MAPR_TICKETFILE_LOCATION`.	• Linux: `/tmp` • Windows: `%TEMP%`
`-type`	Required ticket type for the `generateticket` command; value must be `service`, `servicewithimpersonation`, `tenant`, or `crosscluster`: • `service` is used to generate service tickets for regular cluster operations. • `servicewithimpersonation` is used to generate tickets for regular cluster operations, including allowing user to impersonate other users. • `tenant` is used to generate tickets for tenant users/hosts. • `crosscluster` is used to generate tickets for inter-cluster operations, such as remote mirroring. The `crosscluster` option only works with the `mapr` user.	No default; `-type` must be set in the `maprlogin generateticket` command.
`-user`	Required with the `generateticket` command. The UNIX user name of the user on the MapR cluster. For `crosscluster` tickets, the user must be `mapr`.	No default

maprlogin Command Examples
Describes common scenarios associated with `maprlogin` usage.

Generating and Displaying User Ticket

Generate a user ticket:

```
$ maprlogin password
[Password for user 'juser' at cluster 'my.cluster.com': ]
MapR credentials of user 'juser' for cluster 'my.cluster.com'
are written to '/tmp/maprticket_1000'
```

Display the ticket for the current user. Sample output is shown below.

```
$ maprlogin print
Opening keyfile /tmp/maprticket_1000
my.cluster.com: user = juser,
created = 'Mon Sep 17 08:30:26 PDT 2018', expires = 'Mon Oct 01 08:30:26
PDT 2018',
RenewalTill = 'Wed Oct 17 08:30:26 PDT 2018', uid = 20001, gids = 54261,
CanImpersonate = false
```

Generating and Displaying mapr User Ticket

Generate a ticket for the `mapr` user:

```
# su mapr
$ maprlogin password
[Password for user 'mapr' at cluster 'test.cluster.com': ]
MapR credentials of user 'mapr' for cluster 'test.cluster.com'
are written to '/tmp/maprticket_5000'
```

Display the ticket for the current user. Sample output is as follows.

```
$ maprlogin print
Opening keyfile /tmp/maprticket_5000
test.cluster.com: user = mapr, created = 'Mon Sep 17 09:18:19 PDT 2018',
expires = 'Mon Oct 01 09:18:19 PDT 2018', RenewalTill = 'Wed Oct 17
09:18:19 PDT 2018',
uid = 5000, gids = 5000, 0, 5001, CanImpersonate = true
```

Generating and Displaying Service Ticket

Generate a service ticket, *longlived_ticket*, in `/tmp` for *maprUser1*:

```
$ maprlogin generateticket -type service -out /tmp/longlived_ticket
-duration 30:0:0 -renewal 90:0:0 -user maprUser1
MapR credentials of user 'maprUser1' for cluster 'JSKCluster129_secure'
are written to '/tmp/longlived_ticket'
```

Display the service ticket in a specified location:

```
$ maprlogin print -ticketfile /tmp/ticketwithduration
Opening keyfile /tmp/ticketwithduration
JSKCluster129_secure: user = maprUser1,
created = 'Tue Jun 14 11:12:01 PDT 2017', expires = 'Thu Jul 14 11:12:01
PDT 2017',
RenewalTill = 'Mon Sep 12 11:12:01 PDT 2017',
uid = 0, gids = 0, CanImpersonate = false
```

Generating and Printing Service with Impersonation Ticket

Generate a service with impersonation ticket (in `/var/tmp`) for *maprUser1*:

```
$ maprlogin generateticket -type servicewithimpersonation -user maprUser1
-out /var/tmp/impersonationTicketMapRuser1
```

After generating the ticket, ensure that `maprUser1` has read permissions on the ticket. If you move the ticketfile to a different location, set the `$MAPR_TICKETFILE_LOCATION` environment variable.

Display the service with impersonation ticket in the specified location:

```
$ maprlogin print -ticketfile /var/tmp/impersonationTicketMaprUser1
Opening keyfile /var/tmp/impersonationTicketMaprUser1
JSKCluster129_secure: user = maprUser1,
created = 'Mon Apr 18 13:46:38 PDT 2017', expires = 'Mon May 02 13:46:38
PDT 2017',
RenewalTill = 'Wed May 18 13:46:38 PDT 2017',
uid = 501, gids = 502, CanImpersonate = true
```

To allow a user to impersonate only specific users and/or groups, use the `impersonateduids` and/or `impersonatedgids` options with the `maprlogin` command. For example:

```
$ maprlogin generateticket -type servicewithimpersonation -user
mapruser1 -out /var/tmp/impersonation_ticket -duration
30:0:0 -impersonateduids 1002,1003 -impersonatedgids 1005,1006 -renewal
90:0:0
```

The command generates a service with impersonation ticket. The ticket holder can impersonate users whose UIDs are 1002 and 1003, and users in the groups with GIDs 1005 and 1006. The ticket expires after 30 days and is stored in `/var/tmp/impersonation_ticket`. The ticket may be renewed at any time within 30 days and can be extended up to a maximum of 90 days. The ticket must be renewed explicitly before its expiration date; it does not renew automatically when it expires.

Generating a Tenant Ticket that is Valid for Specific IPs

Generate a tenant ticket (in `/tmp`) for user *test* that is valid for specific IPs:

```
$ maprlogin generateticket -type tenant -out /tmp/ticketip -ips
10.9.0.1,10.9.0.2 -user test
MapR credentials of user 'test' for cluster 'my.cluster.com' are written to
'/tmp/ticketip'
```

Note: The `-ips` argument is only valid for the `tenant` ticket type.

Display the generated tenant ticket:

```
$ maprlogin print -ticketfile /tmp/ticketip
Opening keyfile /tmp/ticketip
my.cluster.com: user = test, created = 'Tue Aug 25 00:34:14 PDT 2020',
expires = 'Tue Aug 25 00:34:14
PDT 12020', RenewalTill = 'Tue Aug 25 00:34:14 PDT 12020', uid = 5001, gids
= 7001,
CanImpersonate = true, isExternal = true, ips = 10.9.0.1,10.9.0.2,,
IsTenant = true
```

Generating and Displaying Cross-Cluster Ticket

Generate a cross-cluster ticket (in `/tmp`) for *maprUser1*:

```
$ maprlogin generateticket -type crosscluster -out /tmp/
crossclusterTicket -user maprUser1
MapR credentials of user 'maprUser1' for cluster 'JSKCluster128_secure'
are written to '/tmp/crossclusterTicket'
```

Display the contents of a cross-cluster ticket in the specified location:

```
$ maprlogin print -ticketfile /tmp/crossclusterTicket
Opening keyfile /tmp/crossclusterTicket
ClusterSecure: user = root,
created = 'Fri May 27 14:29:40 PDT 2017', expires = 'Fri May 27 14:29:40
PDT 12017',
RenewalTill = 'Fri May 27 14:29:40 PDT 12017',
uid = 0, gids = 0, CanImpersonate = false
```

Running an Authentication Test

`authtest`: This troubleshooting option simulates the behavior of the runtime during authentication, going through the authentication flow.

Options: [`-cluster`] Specifies the name of the cluster.

Ending a Session Before the Ticket Expires

`end` or `logout`: Destroys tickets and logs out.

Options: [`-cluster`] Specifies the name of the cluster. By default, deletes all tickets for all clusters.

Renewing a Ticket Before It Expires

`renew`: Renews an existing ticket for a specified time period.

Options:

- `[ -cluster ]` - Specifies the name of the cluster.
- `[ -duration ]` - Specifies the ticket duration.

The duration you specify must be valid for the ticket in question, given the original `-renewal` value for the ticket and the life of the ticket when the renew command is run:

- You cannot renew a ticket that has already expired.
- You can renew the same ticket multiple times.
- The renewal period (or periods) cannot exceed the available time left for the ticket.

For example, assume that a ticket is created with a duration of 10 days and a renewal of 30 days:

```
maprlogin password -duration 10:0:0 -renewal 30:0:0
```

- On the 11th day, the ticket expires and cannot be renewed at all.
- On the 9th day, you can renew the ticket for any number of days up to a maximum of 21.
- On the 23rd day, you can renew the ticket for any number of days up to a maximum of 7.

Example: Renew a ticket and display the renewed ticket in the specified location:

```
$ maprlogin renew -out /tmp/RenewedsecureClusterTicket
-ticketfile /tmp/secureClusterTicket -duration 1:0:0

$ maprlogin print -ticketfile /tmp/RenewedsecureClusterTicket
Opening keyfile /tmp/RenewedsecureClusterTicket
JSKCluster129_secure: user = root,
created = 'Tue Jun 07 11:53:29 PDT 2017',
expires = 'Wed Jun 08 11:56:56 PDT 2017',
RenewalTill = 'Thu Jul 07 11:53:29 PDT 2017',
uid = 0, gids = 0, CanImpersonate = false
```

Troubleshooting maprlogin Failures

While the root causes of most failure cases with maprlogin can be quickly diagnosed, the following cases can prove challenging:

- When security is enabled for a cluster, the cluster's CLDB listens for connections on port 7443. If security for the cluster is disabled, the `maprlogin` utility is unable to reach the CLDB.

- The utility's connection uses HTTPS, which requires the file `conf/ssl_truststore` to exist on the client. If the file is not present, a secure connection cannot be negotiated.

Detailed error logs for `maprlogin` connection attempts are kept at `logs/maprlogin-<USERID>-nnnn.log`.

mrconfig
The `mrconfig` commands let you create, remove, and manage storage pools, disk groups, and disks; and provide information about containers.

Warning: The `mrconfig` commands provide direct control and low-level access to the MapR filesystem. If you are not careful, or do not know what you are doing, you can irrevocably destroy valuable data.

mrconfig cntr
Discusses the `mrconfig cntr` commands that allow you to manage containers and container replicas.

mrconfig cntr disablethrottle
Permits disabling throttling for resync of a container.

The `mrconfig cntr disablethrottle` command allows you to disable throttling for resync of a container (specified by ID). Run this command on the node that is the source for the resync.

Note: By default, throttling is disabled if resync is not complete after 30 minutes, or when there is only one replica container.

Syntax

```
/opt/mapr/server/mrconfig cntr disablethrottle <cid>
```

Parameters

Parameter	Description
`cid`	The ID of the container.

Example

Command

```
/opt/mapr/server/mrconfig cntr disablethrottle 2049
```

Output

```
----------------------
|From Instance 5660::|
----------------------
Changing throttling on container 2049 throttle flag disable
```

mrconfig cntr resetthrottle
Permits resetting the throttle setting for resync of a container.

The `mrconfig cntr resetthrottle` command allows you to reset the throttle setting for resync of a container (specified by ID).

Note: Run this command only after the resync operation (on the specified container) is complete.

Syntax

```
/opt/mapr/server/mrconfig cntr resetthrottle <cid>
```

Parameters

Parameter	Description
`cid`	The ID of the container.

Example

Command

```
# /opt/mapr/server/mrconfig cntr resetthrottle 2049
```

Output

```
----------------------
|From Instance 5660::|
----------------------
Changing throttling on container 2049 throttle flag reset
```

mrconfig cntr resyncprogress
Retrieves the status of a resync operation for containers or volumes.

The `mrconfig cntr resyncprogress` command allows you to get the status of a resync operation for containers or volumes (specified by IDs).

Syntax

```
/opt/mapr/server/mrconfig cntr resyncprogress --cids|--volids <id,...>
```

Parameters

Parameter	Description
`--cids`	The comma-separated list of container IDs.
`--volids`	The comma-separated list of volume IDs.

Example

Command

```
# /opt/mapr/server/mrconfig cntr resyncprogress --cids 2104
```

Output

```
----------------------
|From Instance 5660::|
----------------------
List of Source Container Ids: 2104
List of Volume Ids:
Resync Progress Info
--------------------
Cid: 2104, Snapshot Cid: 4069905785, Vol Id: 233969254, Location: Source,
Peer Addr: 10.20.30.40:5660
```

```
ResyncType: Container Resync, Status: Resync In Progress, Total Inodes:
4095, Resync Complete: 4095
```

mrconfig dbinfo
Each instance of the file server on a node is responsible for processing and tracking activities that result from running database commands. The `mrconfig dbinfo` command displays information about the activities, including information related to containers, tablets, storage pools, tags, and threads processing operations on tables.

See `mrconfig` for instructions about running `mrconfig` commands.

mrconfig dbinfo arena
The `mrconfig dbinfo arena` command displays all the database related arenas (contiguous piece of memory), including the arena count and byte allocated.

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dbinfo arena
```

Example

Display arena information.

```
/opt/mapr/server/mrconfig dbinfo arena
----------------------
Time: 2020-08-17 11:36:40,7368 Instance 5660
----------------------
tag TagMisc cnt 0 byteCnt 0
tag TagPut cnt 0 byteCnt 0
tag TagLogWriter cnt 0 byteCnt 0
tag TagMemIndex cnt 0 byteCnt 0
tag TagBucketRec cnt 0 byteCnt 0
tag TagSpill cnt 0 byteCnt 0
tag TagPrefetchScanner cnt 0 byteCnt 0
tag TagColSet cnt 0 byteCnt 0
tag TagValueCache cnt 0 byteCnt 0
tag TagSpillGet cnt 0 byteCnt 0
tag TagSpillScan cnt 0 byteCnt 0
tag TagMarlin cnt 0 byteCnt 0
tag TagBucketRowFetcher cnt 0 byteCnt 0
tag TagJsonComparator cnt 0 byteCnt 0
tag TagArAggregator cnt 0 byteCnt 0
tag TagArSender cnt 0 byteCnt 0
tag TagInitPid1 cnt 0 byteCnt 0
tag TagInitPid2 cnt 0 byteCnt 0
tag TagTopicPurge cnt 0 byteCnt 0
tag TagRowIndexInfo cnt 0 byteCnt 0
tag TagFPTree cnt 0 byteCnt 0
tag TagArrayElementFilter cnt 0 byteCnt 0
tag TagLcUserTopic cnt 0 byteCnt 0
tag TagPartition cnt 0 byteCnt 0
tag TagRowBuilder cnt 0 byteCnt 0
tag TagInitPidTGWA cnt 0 byteCnt 0
tag TagMarlinRecoveryTGWA cnt 0 byteCnt 0
tag TagTopicAsyncTGWA cnt 0 byteCnt 0
tag TagMTGGetOneTGWA cnt 0 byteCnt 0
tag TagUpdateAndGetOneRowTGWA cnt 0 byteCnt 0
tag TagGetOneTGWA cnt 0 byteCnt 0
tag TagApplyFilterTGWA cnt 0 byteCnt 0
```


last-updated: Apr 11, 2022

```
tag TagAtomicUpdateTGWA cnt 0 byteCnt 0
tag TagTransformDeleteTopTGWA cnt 0 byteCnt 0
tag TagStack cnt 0 byteCnt 0
tag TagValueCache2 cnt 0 byteCnt 0
tag TagSiDecoder cnt 0 byteCnt 0
tag TagTopicMetaFetchTGWA cnt 0 byteCnt 0
tag TagBucketRowFetcher2 cnt 0 byteCnt 0
tag TagMergeScanner cnt 0 byteCnt 0
tag TagPartitionGetWA cnt 0 byteCnt 0
tag TagGetContext cnt 0 byteCnt 0
tag TagSpillScanner cnt 0 byteCnt 0
tag TagMergeRowDesc cnt 0 byteCnt 0
tag TagIpStateCleanup cnt 0 byteCnt 0
tag TagUpdateAndGet cnt 0 byteCnt 0
tag TagJsonUpdateAndGet cnt 0 byteCnt 0
total byteCnt 0

----------------------
Time: 2020-08-17 11:36:40,7384 Instance 5661
----------------------
tag TagMisc cnt 0 byteCnt 0
tag TagPut cnt 0 byteCnt 0
tag TagLogWriter cnt 0 byteCnt 0
tag TagMemIndex cnt 0 byteCnt 0
tag TagBucketRec cnt 0 byteCnt 0
tag TagSpill cnt 0 byteCnt 0
tag TagPrefetchScanner cnt 0 byteCnt 0
tag TagColSet cnt 0 byteCnt 0
tag TagValueCache cnt 0 byteCnt 0
tag TagSpillGet cnt 0 byteCnt 0
tag TagSpillScan cnt 0 byteCnt 0
tag TagMarlin cnt 0 byteCnt 0
tag TagBucketRowFetcher cnt 0 byteCnt 0
tag TagJsonComparator cnt 0 byteCnt 0
tag TagArAggregator cnt 0 byteCnt 0
tag TagArSender cnt 0 byteCnt 0
tag TagInitPid1 cnt 0 byteCnt 0
tag TagInitPid2 cnt 0 byteCnt 0
tag TagTopicPurge cnt 0 byteCnt 0
tag TagRowIndexInfo cnt 0 byteCnt 0
tag TagFPTree cnt 0 byteCnt 0
tag TagArrayElementFilter cnt 0 byteCnt 0
tag TagLcUserTopic cnt 0 byteCnt 0
tag TagPartition cnt 0 byteCnt 0
tag TagRowBuilder cnt 0 byteCnt 0
tag TagInitPidTGWA cnt 0 byteCnt 0
tag TagMarlinRecoveryTGWA cnt 0 byteCnt 0
tag TagTopicAsyncTGWA cnt 0 byteCnt 0
tag TagMTGGetOneTGWA cnt 0 byteCnt 0
tag TagUpdateAndGetOneRowTGWA cnt 0 byteCnt 0
tag TagGetOneTGWA cnt 0 byteCnt 0
tag TagApplyFilterTGWA cnt 0 byteCnt 0
tag TagAtomicUpdateTGWA cnt 0 byteCnt 0
tag TagTransformDeleteTopTGWA cnt 0 byteCnt 0
tag TagStack cnt 0 byteCnt 0
tag TagValueCache2 cnt 0 byteCnt 0
tag TagSiDecoder cnt 0 byteCnt 0
tag TagTopicMetaFetchTGWA cnt 0 byteCnt 0
tag TagBucketRowFetcher2 cnt 0 byteCnt 0
tag TagMergeScanner cnt 0 byteCnt 0
tag TagPartitionGetWA cnt 0 byteCnt 0
tag TagGetContext cnt 0 byteCnt 0
tag TagSpillScanner cnt 0 byteCnt 0
tag TagMergeRowDesc cnt 0 byteCnt 0
```

```
tag TagIpStateCleanup cnt 0 byteCnt 0
tag TagUpdateAndGet cnt 0 byteCnt 0
tag TagJsonUpdateAndGet cnt 0 byteCnt 0
total byteCnt 0
```

mrconfig dbinfo autosetup
The `mrconfig dbinfo autosetup` command displays information about replica autosetup for database tables.

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dbinfo autosetup
```

Example

Display replica autosetup information for database tables.

```
/opt/mapr/server/mrconfig dbinfo autosetup
----------------------
Time: 2020-08-18 14:49:10,4851 Instance 5660
----------------------
table 2049.557.263820 replicaIdx 1 replicaState 4 event 0 schedState 0
createScheduled 0
copyScheduled 1 doneRegionCount 3 retryRegionCount 1 recoveredProgressPct 0
copyProgressPct 100
backoff 0 reschedAt 0 inQuickDelayList 0 inLateDelayList 0 error 0
extendedError
```

mrconfig dbinfo cidmapcache
The `mrconfig dbinfo cidmapcache` command displays the number of entries in the container ID cache and the number of successful and unsuccessful lookups performed by each instance of the file server on the cache.

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dbinfo cidmapcache
```

Example

Display container ID information.

```
/opt/mapr/server/mrconfig dbinfo cidmapcache
----------------------
Time: 2020-08-17 11:58:27,2761 Instance 5660
----------------------
entries 88138
numLookups 0
numMisses 0
----------------------
Time: 2020-08-17 11:58:27,2769 Instance 5661
----------------------
entries 88138
numLookups 0
numMisses 0
```

mrconfig dbinfo copyregiontrackers
The `mrconfig dbinfo copyregiontrackers` command displays information about copy region progress for tables upon a replica autosetup.

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dbinfo copyregiontrackers
```

Example

Display the copy region progress for tables upon replica autosetup.

```
/opt/mapr/server/mrconfig dbinfo copyregiontrackers
----------------------
Time: 2020-08-18 14:50:32,0558 Instance 5660
----------------------
table 2049.557.263820 replicaIdx 2 startKey (nil) endKey
\x0fuser3029129807259132922 doneTillKey (nil) completionPct 0 lastUpdatedAt
1597787427 backoff 0 reschedAt 0 inQuickDelayList 0 inLateDelayList 0
table 2049.557.263820 replicaIdx 2 startKey \x0fuser3029129807259132922
endKey \x0fuser5085894088285492546 doneTillKey (nil) completionPct 0
lastUpdatedAt 1597787427 backoff 0 reschedAt 0 inQuickDelayList 0
inLateDelayList 0
table 2049.557.263820 replicaIdx 2 startKey \x0fuser5085894088285492546
endKey \x0fuser7196611286587175704 doneTillKey (nil) completionPct 0
lastUpdatedAt 1597787427 backoff 0 reschedAt 0 inQuickDelayList 0
inLateDelayList 0
table 2049.557.263820 replicaIdx 2 startKey \x0fuser7196611286587175704
endKey (nil) doneTillKey (nil) completionPct 0 lastUpdatedAt 1597787427
backoff 0 reschedAt 0 inQuickDelayList 0 inLateDelayList 0
```

mrconfig dbinfo copyregionworkers
The `mrconfig dbinfo copyregionworkers` command displays information about the worker threads for parallel copy regions.

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dbinfo copyregionworkers
```

Example

Display the

```
/opt/mapr/server/mrconfig dbinfo copyregionworkers
----------------------
Time: 2020-08-18 14:51:43,9526 Instance 5660
----------------------
table 2049.557.263820 replicaIdx 3 tablet 2134.58.131424 startKey
\x0fuser5085894088285492546 endKey \x0fuser7196611286587175704 doneTillKey
\x0fuser5571582571141067657 completionPct 50 scheduled 1 error 0
table 2049.557.263820 replicaIdx 3 tablet 2167.32.131422 startKey (nil)
endKey \x0fuser3029129807259132922 doneTillKey \x0fuser1458982478364621543
completionPct 50 scheduled 1 error 0
```

mrconfig dbinfo mem
The `mrconfig dbinfo mem` command displays memory information related to storage pools and buckets.

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dbinfo mem
```

Example

Display memory information.

```
/opt/mapr/server/mrconfig dbinfo mem

----------------------
Time: 2020-08-17 10:39:14,2774 Instance 5660
----------------------
maxSz 1583322498
poolSz 0
pendingDrainSz 0
drainThresh 1266657998
waiters false
pendingBucketFlushes 0
numActiveBuckets 0
totalActiveBucketsSz 0

----------------------
Time: 2020-08-17 10:39:14,2781 Instance 5661
----------------------
maxSz 1583322498
poolSz 0
pendingDrainSz 0
drainThresh 1266657998
waiters false
pendingBucketFlushes 0
numActiveBuckets 0
totalActiveBucketsSz 0
```

mrconfig dbinfo replbuckets
The `mrconfig dbinfo replbuckets` command displays the replication progress information for the data in each bucket for each replica of a table.

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dbinfo replbuckets
```

Example

Display replication progress.

```
/opt/mapr/server/mrconfig dbinfo replbuckets
----------------------
Time: 2020-08-18 15:01:00,5833 Instance 5660
----------------------
bucket 2167.347.132054 table 2049.557.263820 sendAfter 2162.245.131682
reschedAt 1597788066 inReschedQueue 0 inDelayList 1 flushed 0 localbackoff
```

```
0 localLastAttempAt 0
bucket 2167.347.132054 replica5 workerAlloced 1 done 0 depDone 1 doneTill 0
backoff 0 lastAttemptAt 0
bucket 2167.347.132054 replica4 workerAlloced 1 done 0 depDone 1 doneTill 0
backoff 0 lastAttemptAt 0
bucket 2167.347.132054 replica3 workerAlloced 0 done 0 depDone 1 doneTill 0
backoff 7 lastAttemptAt 1597788059
bucket 2167.347.132054 replica2 workerAlloced 0 done 0 depDone 1 doneTill 0
backoff 7 lastAttemptAt 1597788059
```

mrconfig dbinfo repltable
The `mrconfig dbinfo repltable` command displays information about all the replicas setup on a table.

See mrconfig for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dbinfo [-v] repltable <tableFid>
```

Parameters

Parameter	Description
-V	Sets the verbose option.
tableFid	The file identifier for the table.

Example

Display the information about all the replicas setup on a table.

```
/opt/mapr/server/mrconfig dbinfo repltable 2049.557.263820
----------------------
Time: 2020-08-18 15:05:43,2650 Instance 5660
----------------------
table 2049.557.263820 replicaIdx 3 replicaType table replica minPendingTS
1597788052 maxPendingTS 1597788058 bucketsPending 8 bytesPending 515935659
putsPending 462970
table 2049.557.263820 replicaIdx 2 replicaType table replica minPendingTS
1597788052 maxPendingTS 1597788058 bucketsPending 8 bytesPending 515935659
putsPending 462970
table 2049.557.263820 replicaIdx 1 replicaType table replica minPendingTS
1597788052 maxPendingTS 1597788058 bucketsPending 8 bytesPending 515935659
putsPending 462970
table 2049.557.263820 replicaIdx 6 replicaType index minPendingTS 0
maxPendingTS 0 bucketsPending 0 bytesPending 0 putsPending 0
```

mrconfig dbinfo tablets
The `mrconfig dbinfo tablets` command displays information about tablets (table regions).

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dbinfo tablets
```


last-updated: Apr 11, 2022

Example

Display information about tablets.

```
/opt/mapr/server/mrconfig dbinfo tablets

----------------------
Time: 2020-08-17 10:26:05,5505 Instance 5660
----------------------
tablet 2071.32.131314 nref 0 npartitions 1 logicalMB 0 physicalMB 0
rows 0 splitState None attrAutoSplit 1 tabletSplitThreshSizeMB 6144
partitionSplitThreshSizeMB 2048 isReadOnly 0 error 0 updateError 0
tablet 2081.32.131210 nref 0 npartitions 1 logicalMB 0 physicalMB 0
rows 0 splitState None attrAutoSplit 1 tabletSplitThreshSizeMB 6144
partitionSplitThreshSizeMB 2048 isReadOnly 0 error 0 updateError 0

----------------------
Time: 2020-08-17 10:26:05,5514 Instance 5661
----------------------
tablet 2083.32.131392 nref 0 npartitions 1 logicalMB 0 physicalMB 0
rows 0 splitState None attrAutoSplit 1 tabletSplitThreshSizeMB 6144
partitionSplitThreshSizeMB 2048 isReadOnly 0 error 0 updateError 0
tablet 2082.32.131416 nref 0 npartitions 1 logicalMB 0 physicalMB 0
rows 0 splitState None attrAutoSplit 1 tabletSplitThreshSizeMB 6144
partitionSplitThreshSizeMB 2048 isReadOnly 0 error 0 updateError 0
```

mrconfig dbinfo tabletsplits
The `mrconfig dbinfo tabletsplits` command displays information about the tablets (table regions) being split.

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dbinfo tabletsplits
```

Example

Display information about the tablets currently being split.

```
/opt/mapr/server/mrconfig dbinfo tabletsplits
----------------------
Time: 2020-08-18 15:09:34,9493 Instance 5660
----------------------
from 2167.648.132844 to 2133.623.132830 elapsedSecs 8 splitState
SplitSrcInProgress splitStart (nil) splitEnd \x0fuser5085983665551623158
stabilizeState PAUSE_PARTITION_SPLITS
```

mrconfig dbinfo threads
The `mrconfig dbinfo threads` command displays information about the throttling queue for each thread processing BatchGet operations, such as the number of free and maximum slots. The command also displays the work areas (WA) for the RPCs being processed by the file server.

You can configure the number of operations that run in parallel in mfs.conf through the `mfs.db.max.concurrent.internal.ops` option. See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dbinfo threads
```

Example

Display the throttling queue information for BatchGet operations and work areas (WA) for the RPCs currently being processed by the file server.

```
/opt/mapr/server/mrconfig dbinfo threads
----------------------
Time: 2020-08-13 12:08:33,9402 Instance 5662
----------------------
ThrottleQ : maxSlots 1024 freeSlots 1016 hasWaiters 0 totalWaits 0
InternalOpThrottleQ1 : maxSlots 24576 freeSlots 24576 hasWaiters 0
totalWaits 0
InternalOpThrottleQ2 : maxSlots 24576 freeSlots 24576 hasWaiters 0
totalWaits 0
InternalOpThrottleQ3 : maxSlots 24576 freeSlots 24576 hasWaiters 0
totalWaits 0
thread:ScanWA wa:0x25a0910000 file:fs/server/db/rpc/scan.cc line:967
cbarg:0x271cbf0000
thread:ScanWA wa:0x25b81b1e00 file:fs/server/db/rpc/scan.cc line:967
cbarg:0x2653bd0000
thread:ScanWA wa:0x26d27a2800 file:fs/server/db/rpc/scan.cc line:967
cbarg:0x26ef476000
thread:ScanWA wa:0x25a0911e00 file:fs/server/db/rpc/scan.cc line:967
cbarg:0x269d60e000
thread:SingleScanWA wa:0x271cbf0000 file:fs/server/db/rpc/scan.cc line:303
cbarg:0x0
thread:SingleScanWA wa:0x2653bd0000 file:fs/server/db/rpc/scan.cc line:303
cbarg:0x0
thread:SingleScanWA wa:0x26ef476000 file:fs/server/db/rpc/scan.cc line:303
cbarg:0x0
thread:SingleScanWA wa:0x269d60e000 file:fs/server/db/rpc/scan.cc line:303
cbarg:0x0
```

mrconfig dg

This section discusses the `mrconfig dg` commands that allow you to configure disk groups.

mrconfig dg create

Facilitiates creation of disk groups.

The `mrconfig dg create` commands let you create disk groups (after you initialize disks with the `mrconfig disk init` command and add them to the node with the `mrconfig disk load` command).

You can create a disk group with one of two formats:

- Use the `mrconfig dg create raid0` command to create a striped disk group with a RAID 0 format.
- Use the `mrconfig dg create concat` command to create a concatenated disk group (one disk after another).

After you create disk groups you will be ready to create storage pools on the disk groups.

See `mrconfig` for instructions about running `mrconfig` commands.

mrconfig dg create concat

last-updated: Apr 11, 2022

The `mrconfig dg create concat` command creates a concatenated disk group. When a disk group is created MapR assigns one of the disks as the device path of the disk group. After you create a disk group you will be ready to create a storage pool on the disk group.

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dg create concat <path>
```

Parameters

Parameter	Description
path	The device path of each of the disks to add to the disk group; example /dev/sdc /dev/sdd /dev/sde

Examples

Create a concatenated disk group on a local node

```
/opt/mapr/server/mrconfig dg create concat /dev/sdc /dev/sdd /dev/sde
```

mrconfig dg create raid0

Creates a disk group striped for RAID 0.

The `mrconfig dg create raid0` command creates a disk group striped for RAID 0. When you create a disk group, MapR assigns one of the disks as the device path of the disk group. After you create a disk group, you are ready to create a storage pool on the disk group.

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dg create raid0 [-d <stripeDepth>] <path>
```

Parameters

Parameter	Description
-h	host IP address; default `127.0.0.1`
-p	The MapR File System port; default `5660`
-d	The stripe depth in 8K blocs; default `128` (1 MB)
path	The device path of each of the disks to add to the disk group; example /dev/sdc /dev/sdd /dev/sde

Examples

Create a disk group striped for RAID 0 with a stripe depth of 24 on a local node

```
/opt/mapr/server/mrconfig dg create raid0 -d 24 /dev/sdc /dev/sdd /dev/sde
```

mrconfig dg help

The `mrconfig dg help` command displays online help for disk group commands.

last-updated: Apr 11, 2022

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dg help
```

Examples
Display online help for `mrconfig dg` commands on a local node

```
/opt/mapr/server/mrconfig dg help
```

mrconfig dg list
The `mrconfig dg list` command lists the disk groups on all the MapR File System disks on a node.

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig dg list
```

Examples
List the disk groups on all the MapR File System disks on localhost

```
/opt/mapr/server/mrconfig disk list
```

mrconfig disk
This section discusses the `mrconfig disk` commands.
mrconfig disk help

The `mrconfig disk help` command displays the help text for `mrconfig disk` commands.

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig disk help
```

Example
Display the help text for `mrconfig disk` commands on a local node

```
/opt/mapr/server/mrconfig disk help
```

mrconfig disk init

The `mrconfig disk init` command initializes a disk and formats it for the MapR filesystem.

Note:

Warning: Initializing a Disk Causes Data Loss

Initializing a disk destroys the data on the disk, so be sure that all data on a disk is backed up and replicated before initializing the disk.

After executing the `mrconfig disk init` command, add the disk to the node with the `mrconfig disk load` command.

See `mrconfig` for instructions about running `mrconfig` commands. **Tip:**

To initialize, format, and load one or more disks in one step using:

- The MapR Control System, see Adding Disks to MapR File System on page 837.
- The CLI, see `disk add` on page 1602 command.

Syntax

```
/opt/mapr/server/mrconfig disk init <path>
    [-F]
    <path>
```

Parameters

Parameter	Description
-F	Forces formatting of the disk for MapR File System, regardless of prior formatting or existing data.
path	The device path of the disk; example /dev/sdc

Examples
Initialize a disk for MapR File System on a local node

```
/opt/mapr/server/mrconfig disk init /dev/sdc
```

Initialize and format a disk for MapR File System on a local node

```
/opt/mapr/server/mrconfig disk init -F /dev/sdc
```

Initialize and format a disk for MapR File System on a remote node with an IP address of xx.xx.xx.xx

```
/opt/mapr/server/mrconfig -h xx.xx.xx.xx disk init -F /dev/sdc
```

mrconfig disk list

The `mrconfig disk list` command lists all of the disks on a node that have a MapR filesystem.

It also shows information about the disk groups and storage pools on the node including whether or not the storage pools are online.

See `mrconfig` for instructions about running `mrconfig` commands.

Tip:

To list system disks and other available disks on a node in addition to MapR File System disks using:

- The MapR Control System, see Viewing the List of Disks on page 834.
- The CLI, see `disk list` on page 1604 command.

Syntax

```
/opt/mapr/server/mrconfig disk list [<path>]
```

Parameters

Parameter	Description
path	The path of the disk; if not included shows information about all disks on the node, if included only shows information about the specified disk; example: /dev/sdc

Examples
List information about all MapR File System disks on a local node

```
/opt/mapr/server/mrconfig disk list
```

List information about MapR File System disk `/dev/sdc` on a local node

```
/opt/mapr/server/mrconfig disk list /dev/sdc
```

mrconfig disk load

After initializing a disk with the `mrconfig disk init` command, load the disk into memory with the `mrconfig disk load` command.

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig disk load <path>

    <path>
```

Parameters

Parameter	Description
path	The device path of the disk; example /dev/sdc

Examples
Load a disk on a local node

```
/opt/mapr/server/mrconfig disk load /dev/sdc
```

mrconfig disk remove

The `mrconfig disk remove` command removes a disk from MapR File System. A disk cannot be removed unless its storage pool is offline.

Note: Warning: Removing a Disk Causes Data Loss

Removing a disk destroys the data on the disk, so be sure that all data on a disk is backed up and replicated before removing a disk.

The `mrconfig disk remove` command is typically used when replacing a failed disk on a node.

Syntax

```
/opt/mapr/server/mrconfig disk remove [<path>]
```


last-updated: Apr 11, 2022

Parameters

Parameter	Description
path	The device path of the disk; example /dev/sdc

Examples
Remove a disk from a local node

```
/opt/mapr/server/mrconfig disk remove /dev/sdc
```

Removing Disks Using mrconfig

Suppose one of three disks in a storage pool has failed, and the storage pool has gone offline.

To remove a disk with `mrconfig disk remove`:

1. Ensure that the data on the surviving disks is backed up/replicated.
2. Remove the failed disk from the node's disktab with the `mrconfig disk remove` command.
3. Physically remove the failed disk.
4. Physically attach the replacement disk.
5. Run the `mrconfig disk init` command on the replacement disk and on the other two disks that were in the disk group.
6. Run the `mrconfig disk load` command on each of the three disks.
7. Use the `mrconfig dg create` command to create a new disk group with the three disks.
8. Use the `mrconfig sp make` command to create a storage pool on the new disk group.

mrconfig info
The `mrconfig info` commands provide information about memory, threads, volumes, containers and other information about the MapR filesystem.

See `mrconfig` for instructions about running `mrconfig` commands.

mrconfig info containerchain

The `mrconfig info containerchain` command displays the containerchain for a given container. Example:

```
$ /opt/mapr/server/mrconfig info containerchain 2050
Container 2050 prev 256000049 next 0.
Container 256000049 prev 0 next 2050.
```

See `mrconfig` for instructions on running `mrconfig` commands.

Syntax

```
mrconfig [-h <host>] [-p <port>] info containerchain <cid>

    <cid>
```

Parameters

Parameter	Description
-h	host IP address; default `127.0.0.1`
-p	The MapR File System port; default `5660`
cid	The container identifier

Tip:

Use the `mrconfig info dumpcontainers` command to find the container identifiers on a node.

Examples
Find the containerchain for a container with a cid of 2049 on a local node

```
/opt/mapr/server/mrconfig info containerchain 2049
```

Find the containerchain for a container with a cid of 2049 on a remote node with an IP address of xx.xx.xx.xx

```
/opt/mapr/server/mrconfig -h xx.xx.xx.xx info containerchain 2049
```

mrconfig info containerlist

The `mrconfig info containerlist` command lists read/write container IDs for a specified volume. Example:

```
$ /opt/mapr/server/mrconfig info containerlist volume1
Volume containers
2050
```

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig [-h <host>] [-p <port>] info containerlist
<volName>
```

Parameters

Parameter	Description
-h	host IP address; default `127.0.0.1`
-p	The MapR File System port; default `5660`
volName	The name of the volume

Tips:

You can see the names of volumes using:

- The **Volumes** view in the in the MapR Control System.
- The `maprcli volume list` command.

Examples
Display information about the containers in a volume named `marketing` on a local node

```
/opt/mapr/server/mrconfig info containerlist marketing
```

Display information about the containers on a volume named `marketing` on a remote node with an IP address of xx.xx.xx.xx

```
/opt/mapr/server/mrconfig -h xx.xx.xx.xx info containerlist marketing
```

mrconfig info containers
The `mrconfig info containers` command displays information about containers.

Example:

```
$ /opt/mapr/server/mrconfig info containers rw
RW containers: 1 2049 2050
$ /opt/mapr/server/mrconfig info containers resync
$ /opt/mapr/server/mrconfig info containers snapshot
Snapshot containers: 256000049
```

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig [-h <host>] [-p <port>] info containers
<container-type> [path]

    <container-type>
    [path]
```

Parameters

Parameter	Description
-h	host IP address; default `127.0.0.1`
-p	The MapR File System port; default `5660`
container-type	When specified, lists only containers of the specified type. Possible values: • `rw` • `resync` • `snapshot`
path	The path to a service pool (obtained with `mrconfig sp list`). When specified, lists only containers on the specified service pool.

Examples
Display a list of read/write containers on a local node

```
/opt/mapr/server/mrconfig info containers rw
```

Display a list of read/write containers on a remote node with an IP address of xx.xx.xx.xx

```
/opt/mapr/server/mrconfig -h xx.xx.xx.xx info containers rw
```

mrconfig info dumpcontainers

The `mrconfig info dumpcontainers` command displays information about containers including container identifiers, volume identifiers, storage pools, total and free inodes per container.

Example:

```
$ /opt/mapr/server/mrconfig info dumpcontainers
cid:2352 volid:165226505 sp:SP2:/dev/sde
spid:9d28cd7770961b3a005c9210db0a88e9 prev:0 next:0 issnap:0 isclone:0
deleteinprog:0 fixedbyfsck:0 stale:0 querycldb:0 resyncinprog:0 shared:0
owned:518 logical:1080 snapusage:0 snapusageupdated:1 ismirror:0
isrwmirrorcapable:1 role:0 awaitingrole:0 totalInodes:256 freeInodes:201
dare:0 istiered:0 numtotalblocks:0 numpurgedblocks:0 numoffloadedblocks:0
maxUniq:131298 isResyncSnapshot:0 snapId:0 port:5660
cid:2353 volid:166629060 sp:SP1:/dev/sdh
spid:2c9e72229ba0a22a005c9210d801dd4c prev:0 next:0 issnap:0 isclone:0
deleteinprog:0 fixedbyfsck:0 stale:0 querycldb:0 resyncinprog:0 shared:0
owned:577 logical:1243 snapusage:0 snapusageupdated:1 ismirror:0
isrwmirrorcapable:1 role:1 awaitingrole:0 totalInodes:256 freeInodes:197
dare:0 istiered:0 numtotalblocks:0 numpurgedblocks:0 numoffloadedblocks:0
maxUniq:131232 isResyncSnapshot:0 snapId:0 port:5660
cid:2358 volid:42237139 sp:SP2:/dev/sde
spid:9d28cd7770961b3a005c9210db0a88e9 prev:0 next:0 issnap:0 isclone:0
deleteinprog:0 fixedbyfsck:0 stale:0 querycldb:0 resyncinprog:0 shared:0
owned:545 logical:1139 snapusage:0 snapusageupdated:1 ismirror:0
isrwmirrorcapable:1 role:1 awaitingrole:0 totalInodes:256 freeInodes:199
dare:0 istiered:0 numtotalblocks:0 numpurgedblocks:0 numoffloadedblocks:0
maxUniq:131362 isResyncSnapshot:0 snapId:0 port:5660
cid:2361 volid:42237139 sp:SP2:/dev/sde
spid:9d28cd7770961b3a005c9210db0a88e9 prev:0 next:0 issnap:0 isclone:0
deleteinprog:0 fixedbyfsck:0 stale:0 querycldb:0 resyncinprog:0 shared:0
owned:502 logical:1067 snapusage:0 snapusageupdated:1 ismirror:0
isrwmirrorcapable:1 role:1 awaitingrole:0 totalInodes:256 freeInodes:201
dare:0 istiered:0 numtotalblocks:0 numpurgedblocks:0 numoffloadedblocks:0
maxUniq:131242 isResyncSnapshot:0 snapId:0 port:5660
cid:2368 volid:79742583 sp:SP2:/dev/sde
spid:9d28cd7770961b3a005c9210db0a88e9 prev:0 next:0 issnap:0 isclone:0
deleteinprog:0 fixedbyfsck:0 stale:0 querycldb:0 resyncinprog:0
shared:0 owned:451 logical:996 snapusage:0 snapusageupdated:1 ismirror:0
isrwmirrorcapable:1 role:1 awaitingrole:0 totalInodes:256 freeInodes:202
dare:0 istiered:0 numtotalblocks:0 numpurgedblocks:0 numoffloadedblocks:0
maxUniq:131306 isResyncSnapshot:0 snapId:0 port:5660
cid:2370 volid:79742583 sp:SP1:/dev/sdh
spid:2c9e72229ba0a22a005c9210d801dd4c prev:0 next:0 issnap:0 isclone:0
deleteinprog:0 fixedbyfsck:0 stale:0 querycldb:0 resyncinprog:0
shared:0 owned:452 logical:981 snapusage:0 snapusageupdated:1 ismirror:0
isrwmirrorcapable:1 role:1 awaitingrole:0 totalInodes:256 freeInodes:202
dare:0 istiered:0 numtotalblocks:0 numpurgedblocks:0 numoffloadedblocks:0
maxUniq:131356 isResyncSnapshot:0 snapId:0 port:5660
```

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig [-h <host>] [-p <port>] info dumpcontainers
```

Input Parameters

Parameter	Description
-h	host IP address; default `127.0.0.1`
-p	The MapR File System port; default `5660`

Output Fields

Field	Description
cid	Container ID
volid	Volume ID of the volume to which this container belongs.
sp	Storage pool to which this container belongs. For example: SP2:/dev/sde Here, SP2 is the name of the storage pool. /dev/sde is the disk on which the container resides.
spid	Storage pool ID
prev	Pointer to the previous snapshot cid or rw cid (for a clone container), in a snapshot chain.
next	Pointer to the next snapshot cid or rw cid (for a clone container), in a snapshot chain.
issnap	Indicates whether container is a snapshot or not, in a snapshot chain. • 0 - rw container (Not a snapshot container) • 1 - Snapshot
isclone	Indicates whether this container is a clone container created as part of the resync operation. • 0 - Not a clone • 1 - Is a clone
deleteinprog	Indicates whether this container is marked for deletion. • 0 - Not marked for deletion • 1 - Marked for deletion
fixedbyfsck	Indicates whether fsck was run on this container to fix any data or metadata errors. • 0 - fsck was not run on the container • 1 - fsck was run on the container

Field	Description
stale	Indicates whether the current cid is a stale cid or not. Container is marked as stale when it is yet to be processed for cleanup. • 0 - Not stale • 1 - Stale
querycldb	Indicates whether the container is awaiting its role from CLDB at the time of bringing up a cluster. • 0 - Not waiting as CLDB has already defined the role for the container • 1 - Waiting for CLDB to specify the role for the container
resyncinprog	Indicates if the container is resyncing from another container in the container chain. • 0 - Resyncing in progress • 1 - No resyncing
shared	Indicates whether the container is hosting any shared data. The value is the number of shared data blocks (each of size 8K).
owned	Indicates the number of data blocks (each of size 8K) that the container owns.
logical	Indicates the number of logical data blocks (each of size 8K) that are present in the container.
snapusage	Indicates the number of container blocks (each of size 8K) that are used for storing snapshot data.
snapusageupdated	Internal field.
ismirror	Indicates if this container is of a mirror volume or not. 0 - Is not of a mirror volume 1- Is of a mirror volume
isrwmirrorcapable	Indicates if the container belongs to a mirror volume, and if the volume can be converted to a rw volume, in case the primary volume goes down. • 0 - Volume cannot be converted • 1 - Volume can be converted
role	Role of the container, in the container chain. • 0 - Master • 1 - Not the master
totalInodes	Indicates the total number of inodes that can be created on the container.

last-updated: Apr 11, 2022

Field	Description
freeInodes	Indicated the number of inodes that are still available to be used.
dare	Indicates whether the container belongs to a volume that is dare-enabled or not. • 0 - Volume is not dare-enabled • 1 - Volume is dare-enabled
istiered	Indicates whether the container belongs to a tiered (either cold-tiered or EC enabled) volume. • 0 - Container belongs to a tiered volume • 1 - Container does not belong to a tiered volume
numtotalblocks	Applicable only for tiered volumes, this parameter indicates the total number of data blocks present in the container.
numpurgedblocks	Applicable only for tiered volumes, this parameter indicates the number of data blocks that are offloaded and not locally present. This value decreases in case of recalls.
numoffloadedblocks	Applicable only for tiered volumes, this parameter indicates the number of data blocks that are off-loaded to the tier.
maxUniq	Internal field.
isResyncSnapshot	Indicates whether the container belongs to the resync snapshot. • 0 - Container does not belong to the resync snapshot • 1 - Container belongs to the resync snapshot
snapId	Indicates the snapshot ID to which this container belongs, if the ID is a snap cid.
port	Indicates the MFS port number used by the container.

Examples

Display information about containers on a local node

```
/opt/mapr/server/mrconfig info dumpcontainers
```

Display information about containers on a remote node with an IP address of xx.xx.xx.xx

```
/opt/mapr/server/mrconfig -h xx.xx.xx.xx info dumpcontainers
```

mrconfig info fsstate

The `mrconfig info fsstate` command displays information about the status of the MapR filesystem, for example whether or not storage pools are loaded. See `mrconfig` for instructions about running `mrconfig` commands.

last-updated: Apr 11, 2022

Syntax

```
/opt/mapr/server/mrconfig [-h <host>] [-p <port>] info fsstate
```

Parameters

Parameter	Description
-h	host IP address; default `127.0.0.1`
-p	The MapR File System port; default `5660`

Examples
Display information about the state of the MapR filesystem on a local node

```
/opt/mapr/server/mrconfig info fsstate
```

Display information about the state of the MapR filesystem on a remote node with an IP address of xx.xx.xx.xx

```
/opt/mapr/server/mrconfig -h xx.xx.xx.xx info fsstate
```

mrconfig info fsthreads

The `mrconfig info fsthreads` command displays information about threads running on MapR File System disks on a node. See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig [-h <host>] [-p <port>] info fsthreads
```

Parameters

Parameter	Description
-h	host IP address; default `127.0.0.1`
-p	The MapR File System port; default `5660`

Examples
Display information about MapR filesystem threads on a local node

```
/opt/mapr/server/mrconfig info fsthreads
```

Display information about MapR filesystem threads on a remote node with an IP address of xx.xx.xx.xx

```
/opt/mapr/server/mrconfig -h xx.xx.xx.xx info fsthreads
```

mrconfig info mastgateway

The `mrconfig info mastgateway` command must be run on a CLDB node. The command displays the status of the MAST Gateways, the total number of volumes assigned to them, and the number of active, inflight, and pending volumes.

See `mrconfig` `on page 2138` for instructions about running `mrconfig` commands.

last-updated: Apr 11, 2022

Syntax

```
/opt/mapr/server/mrconfig info mastgateway [<gwid>]
```

Parameters

Parameter	Description
gwid	The ID of the MAST Gateway for which to display information.

Examples

Display information about MAST Gateways on the cluster:

```
# /opt/mapr/servermrconfig -h 10.20.30.400 info mastgateway
Num MastGateways: 2

Gateway : atsqa8c46.qa.lab ( 6322920922584906487 )
Active : Yes
Active Vn : 101, Inflight Vn : 101
Num Active Vols : 4
Num Inflight Vols, Adds : 0, Removes : 0
Num Pending Vols, Adds : 0, Removes : 0

Active Vols :
153675213
97789611
23539482
45553484

Gateway : atsqa8c48.qa.lab ( 8723754106996643487 )
Active : Yes
Active Vn : 100, Inflight Vn : 100
Num Active Vols : 0
Num Inflight Vols, Adds : 0, Removes : 0
Num Pending Vols, Adds : 0, Removes : 0
No vols assigned to gateway.
```

mrconfig info nfsthreads

The `mrconfig info nfsthreads` command displays information about in-progress NFS operations.

Syntax

```
/opt/mapr/server/mrconfig [-h <host>] [-p <port>] info nfsthreads
```

Parameters

Parameter	Description
-h	host IP address; default `127.0.0.1`
-p	The NFS port; default `2049`

Output

The output shows the NFS operations currently running (in progress) including the IP address of the client and the type of operation. For example:

```
# /opt/mapr/server/mrconfig info nfsthreads
NFS Threads in progress = 15

Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
Client IP:127.0.0.1:0, Op Type: NFSPROC3_WRITE
```

Examples

Display information about the NFS processes currently running:

```
/opt/mapr/server/mrconfig info nfsthreads
```

Display information about NFS operations in-progress on a remote node with an IP address of `xx.xx.xx.xx`:

```
/opt/mapr/server/mrconfig -h xx.xx.xx.xx info nfsthreads
```

mrconfig info orphanagecount

The `mrconfig info orphanagecount` command displays orphan entries for a given container (specified by ID).

Syntax

```
/opt/mapr/server/mrconfig info orphanagecount <cid>
```

Parameters

Parameter	Description
cid	The ID of the container.

Examples

Display the number of orphan entries for container 2067:

```
~# /opt/mapr/server/mrconfig info orphanagecount 2067
----------------------
Time: 2017-03-31 18:00:49,1085 Instance 5660
----------------------
orphanagecount cid 2067 count 812
```

mrconfig info orphanlist

The `mrconfig info orphanlist` command displays information about a container's orphans. See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig [-h <host>] [-p <port>] info orphanlist <cid>
    <cid>
```

Parameters

Parameter	Description
-h	host IP address; default `127.0.0.1`
-p	The MapR File System port; default `5660`
cid	The container identifier

Tip:

Use the `mrconfig info dumpcontainers` command to find the container identifiers on a node.

Examples
Display information about the orphans of a container with an identifier of 2049 on a local node

```
/opt/mapr/server/mrconfig info orphanlist 2049
```

Display information about the orphans of a container with an identifier of 2049 on a remote node with an IP address of xx.xx.xx.xx

```
/opt/mapr/server/mrconfig -h xx.xx.xx.xx info orphanlist 2049
```

mrconfig info replication

The `mrconfig info replication` command displays information about container replication. See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig [-h <host>] [-p <port>] info replication
```

Parameters

Parameter	Description
-h	host IP address; default `127.0.0.1`
-p	The MapR File System port; default `5660`

Examples
Display information about container replication on a local node

```
/opt/mapr/server/mrconfig info replication
```

Display information about container replication on a remote node with an IP address of xx.xx.xx.xx

```
/opt/mapr/server/mrconfig -h xx.xx.xx.xx info replication
```

mrconfig info slabs

The `mrconfig info slabs` command displays a report about memory usage.

This report is sometimes used for troubleshooting by MapR customer support and is typically not used by customers.

See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig [-h <host>] [-p <port>] info slabs
```

Parameters

Parameter	Description
-h	host IP address; default `127.0.0.1`
-p	The MapR File System port; default `5660`

Examples
Display information about memory usage on a local node

```
/opt/mapr/server/mrconfig info slabs
```

Display information about memory usage on a remote node with an IP address of xx.xx.xx.xx

```
/opt/mapr/server/mrconfig -h xx.xx.xx.xx info slabs
```

mrconfig info threads

The `mrconfig info threads` command displays information about threads running on MapR File System. See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig [-h <host>] [-p <port>] info threads
```

Parameters

Parameter	Description
-h	host IP address; default `127.0.0.1`
-p	The MapR File System port; default `5660`

last-updated: Apr 11, 2022

Examples
Display information about MapR filesystem threads on a local node

```
/opt/mapr/server/mrconfig info threads
```

Display information about MapR filesystem threads on a remote node with an IP address of xx.xx.xx.xx

```
/opt/mapr/server/mrconfig -h xx.xx.xx.xx info threads
```

mrconfig info volume snapshot

The `mrconfig info volume snapshot` command displays information about volume snapshots.

Snapshot and this command require an upgrade to a MapR Enterprise Edition license if you don't already have it. See `mrconfig` for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig [-h <host>] [-p <port>] info volume snapshot
<volName> <snapName>

    <volName>
    <snapName>
```

Parameters

Parameter	Description
-h	host IP address; default `127.0.0.1`
-p	The MapR File System port; default `5660`
volName	The name of the volume
snapName	The name of the snapshot

Tips:

To find volume and snapshot names:

- Navigate to the **Volume** view and the **Snapshot** view respectively in the MapR Control System, or
- Execute the `maprcli volume snapshot list` command, which creates a report that displays volume names and snapshot names.

Examples
Display information about snapshot "snap-2012-01-01" of volume "myVolume" on a local node

```
/opt/mapr/server/mrconfig info volume snapshot myVolume snap-2012-01-01
```

Display information about snapshot "snap-2012-01-01" of volume "myVolume" on a remote node with an IP address of xx.xx.xx.xx

```
/opt/mapr/server/mrconfig -h xx.xx.xx.xx info volume snapshot myVolume
snap-2012-01-01
```

mrconfig info volume mastgateway

The `mrconfig info volume mastgateway` command displays the volume assignment information. This command must be run on a CLDB node.

See `mrconfig` on page 2138 for instructions about running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig info volume mastgateway [<volName>]
```

Parameters

Parameter	Description
`volName`	The name of the volume for which to retrieve the volume assignment information.

Examples

Display the volume assignment information:

```
# ./mrconfig -h 10.20.30.400 info volume mastgateway
Num volumes : 4

Volume : vol2 ( 153675213 )
State: ASSIGNED
Curr: atsqa8c46.qa.lab ( 6322920922584906487 ), Active: Yes
Prop: **No Gw** ( -1 ), Active: No
Assign Suspended : No

Volume : vol1 ( 45553484 )
State: ASSIGNED
Curr: atsqa8c46.qa.lab ( 6322920922584906487 ), Active: Yes
Prop: **No Gw** ( -1 ), Active: No
Assign Suspended : No

Volume : vol4 ( 97789611 )
State: ASSIGNED
Curr: atsqa8c46.qa.lab ( 6322920922584906487 ), Active: Yes
Prop: **No Gw** ( -1 ), Active: No
Assign Suspended : No

Volume : vol3 ( 23539482 )
State: ASSIGNED
Curr: atsqa8c46.qa.lab ( 6322920922584906487 ), Active: Yes
Prop: **No Gw** ( -1 ), Active: No
Assign Suspended : No
```

mrconfig mastgateway

This section describes the mrconfig mastgateway commands that allow you to test PUT, GET, and DELETE operations on the corresponding tier.

mastgateway infomem

Returns memory information on the MAST Gateway node.

Syntax

```
mrconfig mastgateway infomem
```


last-updated: Apr 11, 2022

Parameters

None

Output

Name	The name of the memory pool in the gateway. Each pool has a number of memory objects (not S3 objects) that are used by the MAST Gateway as needed.
Max Free	The maximum number of objects from this pool that can be free.
Active	The number of memory objects from this pool that are allocated and used by the MAST Gateway.
Avail	The number of memory objects that are free, and can be reused when needed.
Obj Size	The size of the object.
Num Objs	The number of objects on the tier.
TierType	The type of tier. Value can be: • Cold • Ec

Example

```
# /opt/mapr/server/mrconfig mastgateway infomem
MASTGw mem info on 127.0.0.1:8660
    Name           Max Free  Active     Avail     Obj Size
     WriteFragBufsS3       128      0     0     8388616
    FullReadFragBufs       64       0     0     8388616
    FragReadFragBufs       1024     0     0     1048576

    Name             Num Objs       Obj Size     TierType
ColdTiering, Recall Frag       0       8388608       Cold
ColdTiering, Recall Full       0       8388608       Cold
ColdTiering, Offload           0       8388608       Cold
ErasureCoding, Recall Frag     0       25165824       Ec
ErasureCoding, Recall Full     0       25165824       Ec
ErasureCoding, Offload         0       25165824       Ec
```

mastgateway infothreads
Returns information on the threads processing the offload or recall operation.

Syntax

```
mrconfig mastgateway infothreads
```

Parameters

None

Output

The command returns the following if it is run during an ongoing offload or recall operation.

`threadId`	The ID of the thread processing the operation.
`volId`	The ID of the volume being offloaded.
`cid`	The ID of the container associated with the volume.
`cgid`	The gateway ID of the container associated with the volume.
`op`	The operation being processed. Value can be: • VolumeOffload • VolumeRecall

Example

Retrieve information on the threads processing the offload:

```
~# /opt/mapr/server/mrconfig -p 8660 mastgateway infothreads
InfoThreads on 127.0.0.1:8660
threadId: 0
volId: 23315726
cid: 2128

op: VolumeOffload

threadId: 1
volId: 23315726
cid: 2130

op: VolumeOffload

threadId: 10
volId: 23315726
cid: 2127

op: VolumeOffload

threadId: 11
volId: 23315726
cid: 2129

op: VolumeOffload

threadId: 12
volId: 23315726
cid: 2131

op: VolumeOffload
```

mastgateway refreshvolassignment
Triggers CLDB to re-assign specified volume to the least utilized MAST Gateway to rebalance tiering operations.

This can be used when new MAST Gateways are added or when MAST Gateways are removed from the cluster.

Note: You must run this command once for each volume to reassign. Run this command for all volumes if MAST Gateway is either newly added to the cluster or permanently removed from the cluster.

last-updated: Apr 11, 2022

Syntax

```
mrconfig mastgateway refreshvolassignment <volname>
```

Parameters

Parameter	Description
`volname`	The name of the volume to reassign.

Result

If the volume is successfully re-assigned, a success message (similar to the one shown in the Examples on page 2168 below) is printed on the console where the command was triggered.

In case of an error, the volume might or might not be assigned to the newly added MAST Gateway. However, the volume would either continue to be assigned to the same MAST Gateway or would be assigned to a different gateway. You can re-run the command in case of a failure.

Examples

Refresh the assignment of the containers associated with volume named vold23:

```
# /opt/mapr/server/mrconfig mastgateway refreshvolassignment vold23
volume assignment refreshed successfully.
```

mastgateway resumevolume
Resume tiering activities and allow reads and writes on the volume data.

Syntax

```
mrconfig mastgateway resumevolume <volname> [forceresume] [forcereset]
```

Parameters

Parameter	Description
`volname`	The name of the volume.
`forceresume`	This parameter is internal-only.
`forcereset`	This parameter is internal-only.

Examples

```
# /opt/mapr/server/mrconfig mastgateway resumevolume volTSECNEW9_3
2018-08-06 02:47:14,8585 ERROR Global mrconfig.cc:2120 ResumeVolume succeed
for volume : volTSECNEW9_3
```

mastgateway suspendvolume
Revoke and suspend a volume assigned to a MAST Gateway.

When the command is run:

1. The volume assignment to the MAST Gateway is revoked.
2. All tiering activities and client reads and overwrites on the volume are suspended.

3. The volume is reassigned to another MAST Gateway.

You must manually run the `mastgateway resumevolume` on page 2168 command to resume tiering activities, including reads and overwrites, on the volume data.

Syntax

```
/opt/mapr/server/mrconfig mastgateway suspendvolume <volname>
[ignoreflusherr deletcpfiles]
```

Parameters

Parameter	Description
`volname`	The name of the volume to reassign.
`ignoreflusherr`	This parameter is internal-only.
`deletcpfiles`	This parameter is internal-only.

Examples

Revoke volume assignment to MAST Gateway and suspend tiering activities on the volume:

```
# /opt/mapr/server/mrconfig mastgateway suspendvolume volTSECNEW9_3
2018-08-06 02:38:15,9360 INFO Global mrconfig.cc:2085 SuspendVolume :
success for volume volTSECNEW9_3
```

mastgateway tierget
Test retrieving an object from the storage tier.

Syntax

```
mrconfig mastgateway tierget
        <tierName>
        <objectID>
        <isSecure>
        [ <objectSize> ]
```

Parameters

Parameter	Description
`isSecure`	Specifies whether to use HTTPs or HTTP protocol. Value can be one of the following: • `true` - for HTTPs protocol • `false` - for HTTP protocol
`objectID`	The ID of the object to get frp, the storage tier. The ID must be the same ID (in string format) specified when offloading the object (using tierput command).
`objectSize`	The size of the object to get from the storage tier. The default value is 64KB.

last-updated: Apr 11, 2022

Parameter	Description
tierName	The name of the storage tier. If necessary, run the tier list on page 1880 command to retrieve the names of the tiers.

Example

Retrieve the object named sampleamazonobj of size 20971520 on the tier named amazonTier:

```
# /opt/mapr/server/mrconfig  mastgateway tierget amazonTier sampleamazonobj
true 20971520
time take for the operation: 8.748000 seconds
tierget successful
```

mastgateway tierdelete
Test deleting an object on the storage tier.

Syntax

```
mrconfig mastgateway tierdelete
        <tierName>
        <objectID>
        <isSecure>
```

Parameters

Parameter	Description
isSecure	Specifies whether to use HTTPs or HTTP protocol. Value can be one of the following: • true - for HTTPs protocol • false - for HTTP protocol
objectID	The ID of the object to get frp, the storage tier. The ID must be the same ID (in string format) specified when offloading the object (using tierput command).
tierName	The name of the storage tier. If necessary, run the tier list on page 1880 command to retrieve the names of the tiers.

Example

Delete the object named sampleamazonobj of size 20971520 KB in the tier named amazonTier:

```
# /opt/mapr/server/mrconfig mastgateway tierdelete amazonTier
sampleamazonobj true 20971520
time take for the operation: 0.339000 seconds
tierdelete successful
```

mastgateway tierput
Test offloading an object to the storage tier.

Syntax

```
mrconfig mastgateway tierput
    <tierName>
```


last-updated: Apr 11, 2022

```
    <objectID>
    <isSecure>
    [ <objectSize> ]
```

Parameters

Parameter	Description
`isSecure`	Specifies whether to use HTTPs or HTTP protocol. Value can be one of the following: • `true` - for HTTPs protocol • `false` - for HTTP protocol
`objectID`	The ID of the object to put on the storage tier. Value must be a string.
`objectSize`	The size of the object to put on the storage tier. The default value is 64KB.
`tierName`	The name of the storage tier. If necessary, run the `tier list on page 1880` command to retrieve the names of the tiers.

Example

Test offload of an object, whose ID is sapleamazonobj and size is 20971520, to the tier named amazonTier:

```
# /opt/mapr/server/mrconfig mastgateway tierput amazonTier sampleamazonobj
true 20971520
time take for the operation: 4.291000 seconds
tierput successful
```

mrconfig sp
The `mrconfig sp` commands create and control storage pools.

Storage pools are created on disk groups, so disk groups must be created before storage pools can be created.

MapR File System reads and writes data (and metadata) to and from logical storage units called volumes. Volumes store data in containers in storage pools.

Initially storage pools don't have any containers, the containers are automatically created for a volume as needed. When a container is created it is assigned a container identifier (cid).

Storage pools aren't associated with any particular volume – storage pools may hold containers for multiple volumes. Large files may be distributed across multiple containers, and therefore across multiple storage pools. Data replication happens at the container level.

Data cannot be written directly to containers, a volume is required.

You can create volumes in one of two ways:

- Click the **Create Volume** button in the **Data** > **Volumes** page in the MapR Control System, or
- Execute the `maprcli volume create` command.

See `mrconfig` for instructions about running `mrconfig` commands.

mrconfig sp help

The `mrconfig sp help` command displays the online help for storage pool commands.

last-updated: Apr 11, 2022

See `mrconfig` for instructions on running `mrconfig` commands.

Syntax

```
mrconfig sp help
```

Examples
Display the online help for storage pools on a local node

```
/opt/mapr/server/mrconfig sp help
```

mrconfig sp list
Displays information about configured storage pools.

The `mrconfig sp list` command displays information about storage pools including the name, size, free space and path of each storage pool, whether or not each storage pool is online or offline, and the total number of storage pools. See mrconfig for instructions on running `mrconfig` commands.

Syntax

```
/opt/mapr/server/mrconfig sp list [-v] [sp path]
```

Parameters

Parameter	Description
path	The device path of the storage pool. If you do not specify the path, information about all storage pools is displayed. If you specify the path, only information about the specified storage pool is displayed; example `/dev/sdc`
-v	Print storage pool and cluster GUID information including whether or not the storage pool is enabled for data-at-rest encryption (DARE), and service pool log (journal) size.

Examples

Display information about all storage pools on a local node:

```
/opt/mapr/server/mrconfig sp list
```

Display information about a storage pool with a path `/dev/sdc` on a local node:

```
/opt/mapr/server/mrconfig sp list /dev/sdc
```

Display storage pool information including whether or not the storage pool is DARE-enabled:

```
# /opt/mapr/server/mrconfig sp list -v
ListSPs resp: status 0:2
No. of SPs (2), totalsize 3518339 MB, totalfree 691937 MB

SP 0: name SP1, Online, size 1761217 MB, free 377127 MB, path /dev/
sdb, log 200 MB, port 5660, guid 9dd586829e179476005b0ce23f0dae3c,
clusterUuid -7600986066553737256-4524271553806028052, disks /dev/sdb /dev/
sdd, dare 1
SP 1: name SP2, Online, size 1757121 MB, free 314809 MB, path /dev/
sde, log 200 MB, port 5660, guid daa5916af8909118005b0ce2430d6d54,
```

```
clusterUuid -7600986066553737256-4524271553806028052, disks /dev/sde /dev/
sdf, dare 1
```

mrconfig sp load

The `mrconfig sp load` command loads all of the disks associated with a storage pool.

See `mrconfig` for instructions on running `mrconfig` commands.

Syntax

```
mrconfig sp load <sp name>
```

Parameters

Parameter	Description
sp name	The name of the storage pool; example SP2

Tips:

- Use the `mrconfig sp list` command to see storage pool names (examples SP1, SP2) and the device paths of the storage pools (example `/dev/sdc`).
- Use the `mrconfig disk list` command to see storage pool names (examples SP1, SP2), the device paths of the storage pools (example `/dev/sdc`), and the disks associated with each storage pool (examples `/dev/sdc`, `/dev/sdd`, `/dev/sde`).

Examples
Load the disks associated with the storage pool named SP2 on the local node

```
/opt/mapr/server/mrconfig sp load SP2
```

mrconfig sp make

The `mrconfig sp make` command creates a storage pool on a concat disk group.

Warning: Creating a Storage Pool Causes Data Loss

Creating a storage pool on a disk group destroys the data on the disks in the disk group, so be sure that all data on the disks in the disk group is backed up and replicated before creating a storage pool.

See `mrconfig` for instructions on running `mrconfig` commands.

Syntax

```
mrconfig sp make <dg path>

    [ -P <yes/no> ]
    [ -l <LogSize> ]
    [ -s <deviceSize> ]
    [ -L <Lable> ]
    [ -F ]
    [ -I <cid> ]
    <dg path>
```

Parameters

Parameter	Description
-P	Primary partition or not; yes/no
-l	Log size in number of blocks; Note that this is a lowercase letter "l" (ell), not the number "1".
-s	Disk size in GB
-L	Label for this storage pool
-F	Force the overwrite of any existing storage pool
-I	Initialize the storage pool with one container with the specified container identifier, one directory, and one file. Note that this is an uppercase letter "I" (eye), not the letter "l" (ell) or the number "1".
dg path	The device path of the disk group; example /dev/sdc

Examples

Create a storage pool on a disk group with a path of `/dev/sdc` on a local node

```
/opt/mapr/server/mrconfig sp make /dev/sdc
```

Creating a Storage Pool Using mrconfig

To create a storage pool using mrconfig:

1. Assume the disks `/dev/sdb`, `/dev/sdc`, and `/dev/sdd` are available; initialize them with `mrconfig disk init`:

   ```
   /opt/mapr/server/mrconfig disk init /dev/sdb
   /opt/mapr/server/mrconfig disk init /dev/sdc
   /opt/mapr/server/mrconfig disk init /dev/sdd
   ```

2. Create a disk group with `mrconfig dg create`:

   ```
   /opt/mapr/server/mrconfig dg create raid0 -d
   128 /dev/sdb /dev/sdc /dev/sdd
   ```

3. Create a concatenated disk group with `mrconfig dg create concat` by specifying the primary drive.

   ```
   /opt/mapr/server/mrconfig dg create concat /dev/sdb
   ```

4. At this point, you can use `mrconfig dg list` to see the layout of the disk group, and which disk is the primary disk. The primary disk can be used in other commands to refer to the disk group as a whole. Example:

   ```
   /opt/mapr/server/mrconfig dg list
   ```


last-updated: Apr 11, 2022

5. From the disk group, create a storage pool with `mrconfig sp make`:

```
/opt/mapr/server/mrconfig sp make /dev/sdb
```

mrconfig sp offline

The `mrconfig sp offline` command takes a loaded storage pool offline. When a storage pool is offline it remains loaded into memory but it is not available to MapR filesystem for reads and writes.

The main use of the `mrconfig sp offline` command is to take a storage pool offline so the `fsck` (filesystem check) command can be run on one or more disks or storage pools if there are lost or corrupt containers, directories, tables, files, filelets, or blocks.

After running `fsck` the storage pool is brought back online with the `mrconfig sp online` command, and then typically the `gfsck` (global filesystem check) command would be run on the affected cluster, volumes, or snapshots.

See `mrconfig` for instructions on running `mrconfig` commands.

Syntax

```
mrconfig sp offline <sp path>
```

Parameters

Parameter	Description
sp path	The device path of the storage pool; example /dev/sdc

Examples
Offline a loaded storage pool with a path of `/dev/sdc` on localhost

```
/opt/mapr/server/mrconfig sp offline /dev/sdc
```

mrconfig sp offline all

The `mrconfig sp offline all` command takes all of a node's loaded storage pools offline. When a storage pool is offline it remains loaded into memory, but it is not available to MapR filesystem for reads and writes.

The main use of the `mrconfig sp offline all` command is to take all storage pools on a node offline so the `fsck` (filesystem check) command can be run on disks or storage pools if there are lost or corrupt containers, directories, tables, files, filelets, or blocks.

After running `fsck` the storage pools are brought back online with the `mrconfig sp online` command, and then typically the `gfsck` (global filesystem check) command would be run on the affected cluster, volumes, or snapshots.

See `mrconfig` for instructions on running `mrconfig` commands.

Syntax

```
mrconfig sp offline all
```

Examples
Offline all storage pools on a local node

```
/opt/mapr/server/mrconfig sp offline all
```

mrconfig sp online

The `mrconfig sp online` command makes an offline storage pool online.

When a storage pool is taken offline with the `mrconfig sp offline` command, the storage pool is not available for reads and writes. Typically this is done so the `fsck` (filesystem check) command can be run to check for or repair filesystem inconsistencies.

After the storage pool is put back online with the `mrconfig sp online` command, the storage pool is once again available for reads and writes, and the `gfsck` (global filesystem check) command can be run on the affected cluster, volumes or snapshots.

See `mrconfig` for instructions on running `mrconfig` commands.

Syntax

```
mrconfig sp online <sp path>
```

Parameters

Parameter	Description
sp path	The device path of the storage pool; example /dev/sdc

Examples
Online a storage pool with a path of `/dev/sdc` on a local node

```
/opt/mapr/server/mrconfig sp online /dev/sdc
```

mrconfig sp refresh

The `mrconfig sp refresh` command reloads the `disktab` file and adds any new disks to MapR File System.

See `mrconfig` for instructions on running `mrconfig` commands.

Syntax

```
mrconfig sp refresh
```

Examples
Refresh the storage pools on the local node

```
/opt/mapr/server/mrconfig sp refresh
```

mrconfig sp shutdown

The `mrconfig sp shutdown` command offlines all storage pools and stops the MapR filesystem on their disks.

See `mrconfig` for instructions on running `mrconfig` commands.

Syntax

```
mrconfig sp shutdown
```

Examples
Offline all storage pools on the local node and stop MapR filesystem on their disks

```
/opt/mapr/server/mrconfig sp shutdown
```

mrconfig sp unload

The `mrconfig sp unload` command unloads all of the disks associated with a storage pool.

See `mrconfig` for instructions on running `mrconfig` commands.

Syntax

```
mrconfig sp unload <sp name>
```

Parameters

Parameter	Description
sp name	The name of the storage pool; example SP2

Tips:

- Use the `mrconfig sp list` command to see storage pool names (examples SP1, SP2) and the device paths of the storage pools (example `/dev/sdc`).
- Use the `mrconfig disk list` command to see storage pool names (examples SP1, SP2), the device paths of the storage pools (example `/dev/sdc`), and the disks associated with each storage pool (examples `/dev/sdc`, `/dev/sdd`).

Examples
Unload the disks associated with the storage pool named SP2 on a local node

```
/opt/mapr/server/mrconfig sp unload SP2
```

mrdirectorystats
Prints the space usage for each directory, for a container.

The `mrdirectorystats` utility, when run for a container, prints the space usage information for all directories, starting from the root of the container. This utility is considerably faster than running `ls -R` command on the root of the container and is useful, for example, in identifying directories which need to be moved out to a different volume while trying to reduce the size of the current namespace container.

Syntax

```
/opt/mapr/server/mrdirectorystats
     -c <container_id>
     [ -p ]
     [ -h ]
```

Parameters

Parameter	Description
`c`	The ID of the container.
`h`	Prints help for running the command.
`p`	Prints only parent file ID (PFid) and other information about the file IDs (fids) in the container.

Output

When you specify the `-c` option, the utility prints the following information per directory to the console:

`DirFid`	The directory inode number.
`files`	The number of regular files under the directory.
`subdir`	The number of sub-directories inside the directory.
`others`	The number of other types of files (except directories and regular files), such as device, symlinks, kvstores, tables, etc., inside the directory.
`tfiles`	The total number of regular files stored in the entire directory tree.
`tsubdir`	The total number of sub-directories stored in the entire directory tree.
`tothers`	The total number of other type of files stored in the entire directory tree.
`cntrBlocks`	The space occupied in block size (8k) by the direct blocks of the total regular files (tfiles) in the directory tree, for the current container.
`fieletBlocks`	A rough estimation of the sum of all the data blocks of all the filelets of regular files spread across different data containers.

Note: The utility also shows volume links if any volume exists in the container.

The utility prints the following if `-p` is specified with `-c`:

`Inode`	The inode of the file.
`PFid`	The parent file ID.
`Type`	The type of entity in the container. Value can be one of the following: • Directory — indicates the entity is a directory. • VolLink — indicates entity is a volume link. • KvStore — indicates entity is KvStore.

last-updated: Apr 11, 2022

SubType	The sub-type of the entity in the container. Value can be one of the following: • Directory — indicates entity is directory. • VolLink — indicates entity is a volume link. • Table — indicates entity is a table. Entity can be table only if type is KvStore. • Tabletmap — indicates entity is a tabletmap. Entity can be tabletmap only if type is KvStore. • Schema — indicates entity is a schema. Entity can be schema only if type is KvStore.

Example

Retrieve the disk space usage information for a container by running the utility with the `-c` option:

```
#  ./mrdirectorystats -c 2245
DirFid          files     subdir   others  tfiles    tsubdir  tothers
cntrBlocks  fileletBlocks
2245.16.2       5         3        3       6         4        3
0           65536
2245.39.131308  1         0        0       1         0        0
0           0
2245.40.131310  0         1        0       0         1        0
0           0
2245.41.131312  0         0        0       0         0        0
0           0
2245.45.131320  0         0        0       0         0        0
0           0
symlinks 2 fidmaps 1 tables 1 schemas 1 tabletmaps 1
```

Retrieve information about the file IDs in the container by running the utility with the `-c` and `-p` option:

```
# ./mrdirectorystats  -p -c 2049
Inode :32       PFid: 2049.16.2         Type: VolLink   SubType: VolLink
Inode :33       PFid: 2049.16.2         Type: VolLink   SubType: VolLink
Inode :34       PFid: 2049.16.2         Type: Directory SubType: Directory
Inode :35       PFid: 2049.34.131372    Type: VolLink   SubType: VolLink
Inode :36       PFid: 2049.16.2         Type: VolLink   SubType: VolLink
Inode :37       PFid: 2049.16.2         Type: VolLink   SubType: VolLink
Inode :38       PFid: 2049.16.2         Type: VolLink   SubType: VolLink
Inode :39       PFid: 2049.16.2         Type: KvStore   SubType: Table
Inode :40       PFid: 2049.16.2         Type: VolLink   SubType: VolLink
Inode :41       PFid: 2049.39.262468    Type: KvStore   SubType: Tabletmap
Inode :42       PFid: 2049.39.262468    Type: KvStore   SubType: Schema
```

mrfscmd

Returns the path to the file specified by ID (fid).

Before running the utility, ensure that the `LD_LIBRARY_PATH` environment variable is set for the path to the `libjvm.so` file. If necessary, run the following command to set the `LD_LIBRARY_PATH`:

```
export LD_LIBRARY_PATH=/usr/lib/jvm/java/jre/lib/amd64/server/
```

Syntax

```
/opt/mapr/server/mrfscmd fid path -fid <file-ID>
```

Parameters

Parameter	Description
fid	The ID of the file.

Output

On success, returns path to the file specified by ID (fid).

On failure, returns error.

Examples

The following examples show file path (on success) and errors returned by the utility.

```
# ./mrfscmd fid path -fid 2115.33.131412
/var/mapr/file1
```

```
# ./mrfscmd fid path -fid  2071.33.1313445
Error: Getting Path for Fid 2071.33.1313445 Failed - Stale file handle
(116).
```

```
# ./mrfscmd fid path -fid  2071.33.
Error: Invalid Fid 2071.33.
```

pullcentralconfig
Pulls master configuration files from `/var/mapr/configuration` on the cluster to the local disk, on each node.

The script `/opt/mapr/server/pullcentralconfig` pulls master configuration files from `/var/mapr/configuration` on the cluster to the local disk, on each node.

- If the master configuration file is newer, the local copy is overwritten by the master copy
- If the local configuration file is newer, no changes are made to the local copy

The volume `mapr.configuration` (normally mounted at `/var/mapr/configuration`) contains directories with master configuration files:

- Configuration files in the `default` directory are applied to all nodes
- To specify custom configuration files for individual nodes, create directories corresponding to individual hostnames. For example, the configuration files in a directory named `/var/mapr/configuration/nodes/host1.r1.nyc` are applied only to the machine with the hostname `host1.r1.nyc`.

The following parameters in `warden.conf` control whether central configuration is enabled, and how often `pullcentralconfig` runs:

- `centralconfig.enabled` — Specifies whether to enable central configuration.
- `pollcentralconfig.interval.seconds` — The frequency to check for configuration updates, in seconds.

stubfuse
Simulates a FUSE mount point to determine its read and write performance.

Simulates a FUSE mount point and creates a large test file named `hello`. Use the `dd` command to print the maximum read and write performance of this mount point. The values give you a fair idea of the performance to expect from a MapR POSIX FUSE client. For more information, see MapR FUSE-Based POSIX Client on page 1238

Note: Use an empty directory for the test, as the contents of this directory are emptied during the test. The files that are created during the test are not present after the test.

Attention: Export the path to `libfuse.so` before running this command. Run:

```
export LD_LIBRARY_PATH="/opt/mapr/lib"
```

Syntax

```
/opt/mapr/bin/stubfuse <mountpoint> [<options>] [-h|--help]
```

Parameters

Parameter	Description
`-h|--help`	Prints syntax and all supported options.
mountpoint	The simulated FUSE mount point. This parameter is required.
options	The options that can be specified with the command. Use `-h` or `--help` to retrieve the list of supported options.

Example

Retrieve the read and write performance for the mount point, `/tmp/egmnt`:

- For a write test: `dd if=/dev/zero of=/tmp/egmnt/hello count=100k bs=128k oflag=direct`

  Note: The name of the output file has to be `hello`. Else, the command will fail.

- For a read test: `dd if=/tmp/egmnt/hello of=/dev/null count=100k bs=128k`

  Note: The name of the input file has to be `hello`. Else, the command will fail.

The output will look similar to the following results:

```
100+0 records in
100+0 records out
5120 bytes (5.1 kB) copied, 0.000233249 s, 22.0 MB/s
```

Configuration Files

This section contains reference information about various configuration files.

last-updated: Apr 11, 2022

Configuration File Permissions

Files located in `/opt/mapr/hadoop-2.x.x/etc/hadoop` are owned by the root user account. To edit these files, you must be logged in as `root` user.

Automatic Rolloff of Old Configuration Files

Whenever you run `configure.sh`, the current `warden.conf`, `mapr-clusters.conf`, `hibernate.cfg.xml`, and `db.conf` configuration files are saved with the current timestamp appended to the name. They are saved to the following directory:

```
/opt/mapr/conf/conf.old
```

These configuration files are saved for backup purposes.

cldb.conf
Contains the configuration for CLDB nodes.

The file `/opt/mapr/conf/cldb.conf` specifies the configuration parameters for the CLDB nodes and the cluster topology.

cldb.containers.cache.entries	*Default Value*: 1000000 *Description*: The maximum number of read/write containers available in the CLDB cache.
cldb.default.topologyfileserver	*Default Value*: `/data` *Description*: The default topology for newly-created volumes.
cldb.detect.dup.hostid.enabled	*Default Value*: `false` *Description*: When `true`, CLDB disables *all* nodes with duplicate hostid, including new nodes that try to register with duplicate hostid *and* the existing node. Alarm NODE_ALARM_DUPLICATE_HOSTID is raised. This case requires administrator intervention to correct the hostid confusion. If duplicate hostid occurs on nodes running CLDB, the cluster may fail to start. Therefore, the alarm is not raised, but the `cldb.log` file in `/opt/mapr/logs/` contains an error message.
cldb.enable.memory.tracker	*Default Value*: `false` *Description*: Utility that monitors CLDB for memory usage and deadlocks. If `true`, memory allocations in CLDB are tracked. If memory usage of CLDB goes above certain limits, the utility generates core and shuts down the CLDB. Memory limit is configured as: `Xmx+non heap memory costant(default:3072MB)` You can change the non-heap memory usage by setting `cldb.memory.max.nonheap.mb` to any custom value. If CLDB memory usage goes beyond 130% of this limit, the utility dumps and shuts down CLDB. The default value is `false`.
cldb.ignore.posix.only.hb.alarm	*Default Value*: 1 *Description*: By default, this parameter is set to `1` to consider all nodes except edge nodes (nodes that

	have only POSIX clients and loopback NFS installed) for the No Heartbeat alarm . Set this parameter to `0` to include both edge as well as cluster nodes for the No Heartbeat alarm. **Note:** The edge nodes that went down before changing this parameter are not visible in alarms. However, edge nodes that go down after changing this parameter to `0` will be visible in alarms. See the `-nfsnodes` option of the node list on page 1705 command to view edge nodes.
cldb.ignore.stale.zk	*Default Value*: `false` *Description*: When this setting is `true`, the CLDB ignores the ZooKeeper's information regarding the most recent copy of CLDB data. Change this setting to `true` when the ZooKeeper information is stale. Restart the CLDB with this setting. After the CLDB starts, change the setting back to `false` then restart the CLDB again. Only change this setting on CLDB nodes that are known to have the most recent copy of the CLDB data. Shut down all CLDB processes before changing this variable.
cldb.jmxremote.port	*Default Value*: 7220 *Description*: The CLDB JMX remote port
cldb.max.security.policies	*Default Value*: 10000 *Description*: Defines the maximum number of configured security policies. To prevent users from arbitrarily creating numerous security policies and draining CLDB performance, the maximum number of security policies is limited to 10000 by default.
cldb.min.fileservers	*Default Value*: 1 *Description*: Number of file servers that must register with the CLDB before the root volume is created.
cldb.numthreads	*Default Value*: 10 *Description*: The number of threads reserved for use by the CLDB.
cldb.pbs.global.master	*Default Value*: 0 *Description*: Indicates the global primary cluster for the global namespace. Only the global primary security policy cluster can create/modify security policies. All other secondary security policy clusters can only view or import security policies.
cldb.port	*Default Value*: 7222 *Description*: The port on which the CLDB listens.
cldb.security.blacklist.cleanup.duration.seconds	*Default Value*: 36000 *Description*: Ticket blacklist cleanup interval.
cldb.security.resolve.user	*Default Value*: 0 *Description*: Resolve UID:GID on client OS or on CLDB.

last-updated: Apr 11, 2022

cldb.security.user.ticket.duration.seconds	*Default Value*: 1209600 *Description*: Default ticket duration
cldb.security.user.ticket.max.duration.seconds	*Default Value*: 2592000 *Description*: Maximum ticket duration
cldb.security.user.ticket.renew.duration.seconds	*Default Value*: 2592000 *Description*: Default ticket renew duration
cldb.security.user.ticket.renew.max.duration.seconds	*Default Value*: 7776000 *Description*: Maximum ticket renew duration
cldb.snap.cntr.count.alarm.threshold	*Default Value*: 100000000 *Description*: The threshold (in minutes) for raising the `CLUSTER_ALARM_TOO_MANY_SNAPSHOT_CONTAINERS` alarm.
cldb.snap.cntr.count.disable.threshold	*Default Value*: 128000000 *Description*: The maximum number of snapshots to allow before disabling snapshot creation.
cldb.snap.cntr.count.monitor.interval.minutes	*Default Value*: 60 *Description*: The interval of time (in minutes) to elapse between checking the number of snapshots on the cluster.
cldb.v2.features.enabled	*Default Value*: 1 *Description*: Enables new features added in MapR version 2.0. Used only during the upgrade process from v1.x to 2.x to control when new features become active. Once enabled, cannot be disabled.
cldb.v3.features.enabled	*Default Value*: 1 *Description*: Enables new features added in MapR version 3.0. Used only during the upgrade process from a pre-3.0 version to control when new features become active. Once enabled, cannot be disabled.
cldb.web.port	*Default Value*: 7221 *Description*: The port that the CLDB uses for the webserver.
cldb.zookeeper.servers	*Default Value*: Not Applicable *Description*: The nodes that are running ZooKeeper, in the format `\<host:port\>`.
hadoop.version	*Default Value*: Not Applicable *Description*: The version of Hadoop supported by the cluster.
net.topology.script.file.name	*Default Value*: Not Applicable *Description*: The path to a script that associates IP addresses with physical topology paths. The script takes the IP address of a single node as input and returns the physical topology that should be associated with the specified node. This association is used only at the time a node is initially added to the cluster. To change topology for nodes already in the cluster, use the `maprcli node move` command.
net.topology.table.file.name	*Default Value*: Not Applicable *Description*: The path to a text file that associates IP addresses with physical topology paths. Each line of

	the text file is of format `<hostname/ip> <rack>`, with the IP address or hostname of one node, followed by the topology to associate with the node. This association is used only at the time a node is initially added to the cluster. To change topology for nodes already in the cluster, use the `maprcli node move` command.
num.volmirror.threads	*Default Value*: 1 *Description*: The number of (volume mirror) threads to create to process mirroring requests. The specified number of threads will be created to process requests in parallel; the remaining requests will be in the queue till they are picked up by volume mirror thread.

Example cldb.conf file

```
#
# CLDB Config file.
# Properties defined in this file are loaded during startup
# and are valid for only CLDB which loaded the config.
# These parameters are not persisted anywhere else.
#
# Wait until minimum number of fileserver register with
# CLDB before creating Root Volume
cldb.min.fileservers=1
# CLDB listening port
cldb.port=7222
# Number of worker threads
cldb.numthreads=10
# CLDB webport
cldb.web.port=7221
# CLDB https port
cldb.web.https.port=7443
# Disable duplicate hostid detection
cldb.detect.dup.hostid.enabled=false
# Deprecated: This param is no longer supported. To configure
# the container cache, use the param cldb.containers.cache.percent
# Number of RW containers in cache
#cldb.containers.cache.entries=1000000
#
# Percentage (integer) of Xmx setting to be used for container cache
#cldb.containers.cache.percent=20
#
#Frequency of the heartbeat interval

# Topology script to be used to determine
# Rack topology of node
# Script should take an IP address as input and print rack path
# on STDOUT. eg
# $>/home/mapr/topo.pl 10.10.10.10
# $>/mapr-rack1
# $>/home/mapr/topo.pl 10.10.10.20
# $>/mapr-rack2
#net.topology.script.file.name=/home/mapr/topo.pl
#
# Topology mapping file used to determine
# Rack topology of node
# File is of a 2 column format (space separated)
# 1st column is an IP address or hostname
# 2nd column is the rack path
# Line starting with '#' is a comment
# Example file contents
```


last-updated: Apr 11, 2022

```
# 10.10.10.10 /mapr-rack1
# 10.10.10.20 /mapr-rack2
# host.foo.com /mapr-rack3
#net.topology.table.file.name=/home/mapr/topo.txt
#
# ZooKeeper address
cldb.zookeeper.servers=10.10.82.22:5181
# Hadoop metrics jar version
hadoop.version=2.7.0
# CLDB JMX remote port
cldb.jmxremote.port=7220
num.volmirror.threads=1
# Set this to set the default topology for all volumes and nodes
# The default for all volumes is /data by default
# UNCOMMENT the below to change the default topology.
# For e.g., set cldb.default.topology=/mydata to create volumes
# in /mydata topology and to place all nodes in /mydata topology
# by default
#cldb.default.topology=/mydata
enable.replicas.invariant.check=false
```

Related concepts

Security Certificate Expiry Alarm on page 2233
Describes the NODE_ALARM_CERTIFICATE_NEAR_EXPIRATION alarm.

core-site.xml

Describes the `core-site.xml` file that contains the configuration that overrides the default core parameters.

The `/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop/core-site.xml` file contains configuration that override the default core parameters.

Note: `/opt/mapr/hadoop/hadoop-0.20.2/conf/core-site.xml` is a symlink to `/opt/mapr/hadoop/hadoop-2.x/etc/hadoop/core-site.xml`.

To override a default value, specify the new value within the `<configuration>` tags, using the following format:

```
<property>
  <name> </name>
  <value> </value>
  <description> </description>
</property>
```

Default core Parameters describes the possible entries to place in the `<name>` and `<value>` tags. The `<description>` tag is optional but recommended for maintainability.

Warning: You can examine the current configuration information for this node by using the `hadoop conf -dump` command from a command line.

Default core-site.xml file

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>

</configuration>
```

Core Parameters

See Default core Parameters.

daemon.conf

The file `/opt/mapr/conf/daemon.conf` specifies the user and group under which MapR services run, and whether all MapR services run as the specified user/group, or only ZooKeeper and FileServer. The configuration parameters operate as follows:

- If `mapr.daemon.user` and `mapr.daemon.group` are set, the ZooKeeper and FileServer run as the specified user/group. Otherwise, they run as `root`.
- If `mapr.daemon.runuser.warden=1`, all services started by the warden run as the specified user. Otherwise, they run as `root`.

Sample daemon.conf file

```
mapr.daemon.user=mapr
mapr.daemon.group=mapr
mapr.daemon.runuser.warden=1
```

db.conf

The file `/opt/mapr/conf/db.conf` specifies configuration parameters for the Metrics database.

Warning: Any time you make changes to the `db.conf` file, you must restart the `hoststats` service and Warden for those changes to take effect.

Field	Default	Description
db.url	localhost:3306	The URL and port for the MySQL server that stores Metrics data. This machine does not need to be a node in the cluster.
db.user	root	The MySQL user name.
db.passwd	mapr	The MySQL password. If the password contains the & character, it is replaced with the `&` string in the `hibernate.cfg.xml` file (following XML parsing standards).
db.schema	metrics	The name of the MySQL schema.
db.mode	mysql	Reserved for future use.
db.driverclass	com.mysql.jdbc.Driver	Reserved for future use.

last-updated: Apr 11, 2022

Field	Default	Description
db.joblastacessed.limit.hours	48	Task and task attempt data for a job are purged for jobs that have not been accessed in a number of hours equal to this parameter's value. **Note:** Note that there is an error in this parameter's name. Instead of `db.joblastaccessed.limit.hours`, spelled as the English word *accessed*, the parameter is written `db.joblastacessed.limit.hours`.
db.partition.finest.count.days	3	Integer number of days for which the finest data granularity is kept. Finest granularity is a ten-second resolution.
db.partition.fine.count.days	15	Integer number of days for which fine data granularity is kept. Fine granularity is a five-minute average of the finest resolution.
db.partition.coarse.count.years	100	Integer number of years for which the coarse data granularity is kept. Coarse granularity is a 24-hour average of the fine resolution.
metric.file.rotate	365	Integer number of days for which metrics files are kept in the local volume for each node.
metric.file.cleanupthreshold	512	Specifies a size in GB. When the total size of the metrics files exceeds the value of this parameter, all data over 30 days old is cleaned up.

Example db.conf file

```
db.url=localhost:3306
db.user=root
db.passwd=mapr
db.schema=metrics
db.mode=mysql
db.driverclass=com.mysql.jdbc.Driver
db.joblastacessed.limit.hours=48
db.partition.finest.count.days=3
db.partition.fine.count.days=15
db.partition.coarse.count.years=100
### How many files with raw node metrics data to keep
metric.file.rotate=365
```

.dfs_attributes

Each directory in MapR storage contains a hidden file called `.dfs_attributes` that controls compression and chunk size. To change these attributes, change the corresponding values in the file.

Example:

```
# lines beginning with # are treated as comments
Compression=lz4
ChunkSize=268435456
```

Valid values:

- Compression: `lz4`, `lzf`, `zlib`, or `false`
- Chunk size (in bytes): a multiple of 65535 (64 K) or zero (no chunks). Example: `131072`

You can also set compression and chunksize using the hadoop mfs command.

disktab
Describes the use of the disktab file.

On each node, the file `/opt/mapr/conf/disktab` lists all of the physical drives and partitions that have been added to the MapR File System. The `disktab` file is created by the `disksetup` command, and automatically updated when disks are added or removed (either using the MapR Control System, or with the `disk add` and `disk remove` commands).

Sample disktab file

```
# MapR Disks Mon Nov 28 11:46:16 2011

/dev/sdb
47E4CCDA-3536-E767-CD18-0CB7E4D34E00
/dev/sdc
7B6A3E66-6AF0-AF60-AE39-01B8E4D34E00
/dev/sdd
27A59ED3-DFD4-C692-68F8-04B8E4D34E00
/dev/sde
F0BB5FB1-F2AC-CC01-275B-08B8E4D34E00
/dev/sdf
678FCF40-926F-0D04-49AC-0BB8E4D34E00
/dev/sdg
46823852-E45B-A7ED-8417-02B9E4D34E00
/dev/sdh
60A99B96-4CEE-7C46-A749-05B9E4D34E00
/dev/sdi
66533D4D-49F9-3CC4-0DF9-08B9E4D34E00
/dev/sdj
44CA818A-9320-6BBB-3751-0CB9E4D34E00
/dev/sdk
587E658F-EC8B-A3DF-4D74-00BAE4D34E00
/dev/sdl
11384F8D-1DA2-E0F3-E6E5-03BAE4D34E00
```

exports

Access control for hosts

On each node, the file `/opt/mapr/conf/exports` lists the clusters and mount points available to mount with NFS.

Specify access control for hosts with a space-separated list of hosts, appending `(rw)` for read-write or `(ro)` for read-only access after each host. To specify a default access for all hosts not otherwise specified, add `(rw)` or `(ro)` after a space at the end of a line. The `exports` file follows the same semantics as a standard UNIX exports table. The following export options are supported:

last-updated: Apr 11, 2022

Export option	Definition
ro	Provides read-only access.
rw	Provides read-write access.
root_squash	Squashes root privileges for remote users. For example, you can use: `/mapr (rw,root_squash)` This entry prevents the `/mapr` directory from being written to by the root user on remote hosts.
no_root_squash	Turns off root squashing for remote users.
all_squash	Squashes every remote user, including root.
anonuid, anongid	Specifies user and group IDs to use with remote users from a particular host.

Restricting clusters to specific hosts

To restrict access to a specific export path to particular hosts, use the following format:

```
<Path> <space-separated list of hosts and access rights>
```

For example, the line `/mapr/cluster1 host01(rw) host02(ro)` restricts read-write access to the cluster in `/mapr/cluster1` to host `host01`, and restricts read-only access to host `host02`. No other hosts have access.

Note: After making changes to this file, you do not have to restart the NFS server. You can run a maprcli command to refresh the exports definition without a restart. See nfsmgmt refresh exports.

Enabling Central Configuration

To enable Central Configuration for `exports`, specify a value for the `AutoRefreshExportsTimeInterval` parameter in the `/opt/mapr/conf/nfsserver.conf` file. The value of the `AutoRefreshExportsTimeInterval` parameter determines the number of minutes after which the NFS server refreshes the `exports` file. The default value of 0 disables central configuration for NFS exports.

Sample exports file

```
# Sample Exports file

# for /mapr exports
# <Path> <exports_control>

#access_control -> order is specific to default
# list the hosts before specifying a default for all
#  host01(ro) host02(ro) host03(ro) (rw)
#  enforces ro for a.b.c.d & 1.2.3.4 and everybody else is rw

# special path to export clusters in mapr-clusters.conf. To disable
exporting,
# comment it out. to restrict access use the exports_control
#
/mapr (rw)

#to export only certain clusters, comment out the /mapr & uncomment.
```


last-updated: Apr 11, 2022

```
# Note: this will cause /mapr to be unexported
#/mapr/clustername (rw)

#to export /mapr only to certain hosts (using exports_control)
#/mapr a.b.c.d(rw) e.f.g.h(ro)

# export /mapr/cluster1 rw to a.b.c.d & ro to e.f.g.h (denied for others)
#/mapr/cluster1 a.b.c.d(rw) e.f.g.h(ro)

# export /mapr/cluster2 only to e.f.g.h (denied for others)
#/mapr/cluster2 e.f.g.h(rw)

# export /mapr/cluster3 rw to e.f.g.h & ro to others
#/mapr/cluster2 e.f.g.h(rw) (ro)
```

FileMigrate.properties

Properties		Default Value	Description
UI Setting	**Property File**		
Full Scan Frequency	`upload.fullScanFrequencyMin`	60	Frequency, in minutes, at which full scan of all files detect changes missed by lite scans and purge (per the configured policy) uploaded files. **Note:** The actual time of a full scan is computed on a per directory basis and spread randomly around the specified fullScanFrequency. Thus specifying 60 minutes does not mean that all directories get a full scan at exactly 60 minutes. Instead, directories randomly get a full scan on average every 60 minutes (somewhere between 30 and 90 minutes). This approach avoids load spikes.
Upload Scan Frequency	`upload.scanFrequencySec`	60	Frequency, in seconds, at which light scans of directories detect changes since the last scan. This value cannot be below 10 seconds.

last-updated: Apr 11, 2022

Properties		Default Value	Description
UI Setting	**Property File**		
Completion Scan Frequency	`upload.completionScanFrequencySec`	3	Frequency, in seconds, for checking upload status. This impacts the time delay before the queue is cleared for more uploads, the reporting time for completion, and statistics collection. **Note:** Values below the default may consume extra system resources.
Minimum Wait Before Upload	`upload.minWaitBeforeUploadSecs`	15	Length of time, in seconds, to wait for uploading a file since the file was last modified. Set a higher value to avoid uploading a file still being modified.
Maximum Active Uploads	`upload.maxActiveUploads`	10	Maximum number of files to upload concurrently. The scanning component of the file migration service is paused when 10 times this many files are waiting for upload.
Minimum Wait After Failure	`upload.minWaitAfterFailureSecs`	300	Amount of time, in seconds, to wait (pause) before retrying if upload fails or if there is an error in the filesystem.
Maximum Retries Per File	`upload.maxErrorsOnOneFileBeforeGiveUp`	10	Maximum number of times to attempt to upload a file. If the limit is reached, the file is dequeued to allow other files to be uploaded and the file modification time does not change. The dequeued file is rediscovered and queued for upload during the next full scan.
Maximum System Retries	`upload.maxErrorsBeforeAssumeSystemIssue`	10	Maximum number of errors to allow before pausing all uploads. If this limit is reached, uploads continue only after the amount of time specified as value for the `minWaitAfterFailureSecs` property has elapsed.
Minimum Idle Time Before Error	`upload.minIdleTimeBeforeErrorSec`	300	Amount of time, in seconds, to wait before logging a warning (in the `filemigrate.log` file) if all uploads are not progressing.

Properties		Default Value	Description
UI Setting	**Property File**		
Bucket Location Region	`aws.region`		Region information as defined here. For: • Older regions, specify the short region name. • New regions, specify the short region name and the region endpoint. The following regions can be specified using just the region name: us-east-1, us-west-1, us-west-2, ap-northeast-1, ap-southeast-1, ap-southeast-2, sa-east-1, eu-west-1, cn-north-1, us-gov-west-1. For all other regions, you might have to specify the endpoint as well. **Note:** When you create a bucket in a region, verify that the bucket is really in that region; in some cases, buckets for one region are actually hosted elsewhere.
Region Endpoint	`aws.regionendpoint`		The AWS region endpoint.
Proxy Host	`aws.proxyHost`		(Optional) Proxy host information.
Proxy Port	`aws.proxyPort`		
Proxy Username	`aws.proxyUsername`		
Proxy Password	`aws.proxyPassword`		
Access Key	`aws.accessKey`		Credentials to use for accessing the S3 bucket.
Secret Key	`aws.secretKey`		
Directory Path	`dir<N>.path`		The path from which the scan of the directory or root directory starts. Replace `<N>` with an integer; for each additional directory tree, increment the number and specify a unique path. **Note:** Do not create multiple policies or directory configurations pointing to the same filesystem location.
Target Bucket	`dir<N>.bucket`		The Amazon S3 bucket to upload to.

last-updated: Apr 11, 2022

Properties		Default Value	Description
UI Setting	**Property File**		
Purge Interval	`dir<N>.purgepolicy.hours`	0	Amount of time, in hours, after the file was last modified to wait before deleting the file. The default value is never delete (`0`). **Note:** Deletion is delayed until after the file upload occurs, regardless of the purge time set.
Delete Empty Directories	`dir<N>.deleteEmptyDirectoryDelayMins`	0	Amount of time to wait, in minutes, before deleting empty directories. The default value is never delete (`0`).
Server Side Encryption	`dir<N>.serverSideEncrypt`	false	Specify whether (`true`) or not (`false`) S3 server side encryption should be used. If the value is `true` (enabled), the TransferManager API that specifies AES256 encryption is used automatically for each uploaded file.

Properties		Default Value	Description
UI Setting	**Property File**		
Ignore Files Regex	`dir<N>.ignoreFilesRegex`		File and directory names to ignore, specified as regex pattern, as defined by `java.util.regex`. For example: • `.*\.TEMP$` - ignore all files/directories ending in .TEMP • `.*TEMP.*` - ignore all files/directories that contain the phrase TEMP • `^TEMP.*` - ignore all files/directories that start with TEMP Specify a single regular expression that matches multiple patterns, as shown in the following examples: • `dir2.ignoreFilesRegex=^IGNORE.*\|.*\.TMP$` • `dir2.ignoreFilesRegex=^IGNORE.*|.*\.TMP$\|.*\\.TXT$|.*\\.txt$` • `dir2.ignoreFilesRegex=^temp$`

Properties		Default Value	Description
UI Setting	**Property File**		
X-Attributes	`dir<N>.xattrs`		The space-separated list of AWS S3 attributes to copy from MapR File System to AWS S3 bucket when uploading a file. Make a note of the following: • While MapR File System attributes can be strings or binary, AWS S3 attributes must be strings. • While MapR File System attribute names are case sensitive, AWS S3 attribute names are case insensitive. **Note:** Only extended attributes from MapR filesystem that start with `user.` are considered for upload (since all user settable attributes begin with `user.`). The attribute must be specified without `user.`. For example, to upload a file with the extended attribute name `user.foo`, the value for this property must be foo because the `user.` is implicit. Extended attributes are uploaded when the file is first uploaded. If extended attributes are set after file upload, the extended attributes are not sent to AWS S3 unless the file modification timestamp changes.

Properties		Default Value	Description
UI Setting	**Property File**		
Authorized Users	`authorized.users`	mapr	The comma-separated list of users who can login and use the File Migration Service. The UI authorizes the following users to login and use the service: • The owner of the file migrate process (on the cluster, this is MAPR_USER). • Any users belonging to the comma-separated list of authorized users specified here. • Any users belonging to any group in the comma-separated list specified using `authorized.groups` property. The default value is the MAPR_USER (who is mapr, by default) specified with the `configure.sh` on page 2053 utility.
Authorized Groups	`authorized.groups`	mapr	The comma-separated list of groups authorized to login and use the service. The default value is the MAPR_USER (mapr, by default) specified with the `configure.sh` on page 2053 utility.

gateway.conf

Describes configuration parameters for the MapR gateway.

The `/opt/mapr/conf/gateway.conf` file specifies configuration parameters for the gateway that supports table and stream replication.

Warning: Changing the default settings in the `gateway.conf` file is not recommended and is not likely to improve performance. If you still need to make changes to the `gateway.conf` file, you must restart the gateway after doing so. See Configuring Gateways for Table and Stream Replication on page 1152.

Field	Default	Description
gateway.port	7660	The gateway listening port.
gateway.receive.numthreads	128	The number of worker threads to receive replication stream requests.
gateway.flush.numthreads	128	The number of flush threads to send put requests to replicas.

Field	Default	Description
gateway.put.mem.mb	128	The maximum size limit (in MB) of the putbuffer memory.
gateway.logfile.size.mb	1024	The maximum size limit (in MB) of the MapR gateway log file. When the size limit is reached, the logs get rolled over.
gateway.es.request.maxsize.kb	128	The maximum size limit (in KB) of replication requests distributed by MapR source clusters. *This property is no longer supported.*
gateway.es.cluster.maxClients	1	Max number of clients for the MapR Event Store For Apache Kafka. *This property is no longer supported.*

Example `gateway.conf` **file**

```
#
# Gateway Config file.
# Properties defined in this file are loaded during startup
# and are valid for only Gateway which loaded the config.
# These parameters are not persisted anywhere else.
#
# Gateway listening port
#gateway.port=7660
# Number of worker threads to receive replication stream requests
#gateway.receive.numthreads=128
# Number of flush threads to send put requests to replicas
#gateway.flush.numthreads=128
#
# Max limit on putbuffer memory in MB
#gateway.put.mem.mb=128
#
# Max limit on log file size
#gateway.logfile.size.mb=1024
#
#
# Gateway ES properties
#gateway.es.request.maxsize.kb=128
#gateway.es.cluster.maxClients=1
```

Related concepts

Administering MapR Gateways on page 1150
A MapR gateway mediates one-way communication between a source MapR cluster and a destination cluster. You can replicate MapR Database tables (binary and JSON) and MapR Event Store For Apache Kafka streams. MapR gateways also apply updates from JSON tables to their secondary indexes and propagate Change Data Capture (CDC) logs.

Configuring Gateways for Table and Stream Replication on page 1152
Configuring gateways involves installing the `mapr-gateway` package on nodes on a MapR destination cluster and then configuring the MapR source cluster to communicate with the destination cluster. The MapR source cluster is configured by specifying the destination cluster's CLDB node and gateway nodes.

Gateways for Replicating MapR Database Tables on page 621
In MapR Database table replication, MapR Database replicates updates to tables (binary and JSON) on source MapR clusters to replicas of those tables on destination MapR clusters. Gateways are services that

receive these updates and apply them to the replicas. These gateways also propagate updates from JSON tables to their secondary indexes.

Related tasks

Specifying the Location of Gateways on page 782
Describes how to set the location of the MapR gateways using either the Control System or the CLI.

Related reference

cluster gateway delete on page 1564
Deletes the list of MapR gateways from a source MapR cluster.

cluster gateway get on page 1565
Lists the MapR gateways that a source MapR cluster is using.

cluster gateway list on page 1567
Lists all of the gateways that a source MapR cluster is using.

cluster gateway local on page 1569
Lists the gateways that are configured on the MapR cluster where this command is run.

cluster gateway resolve on page 1571
Lists the gateways configured on a MapR cluster that are up and running at the time that the command is issued.

cluster gateway set on page 1573
Specifies the locations of the MapR gateways that a source MapR cluster can use for table replication to a destination MapR cluster, or for indexing table data in an Elasticsearch cluster.

Related information

Managing Gateways on page 1154
Describes the commands for listing gateways, checking status of gateways, managing gateways if they fail, and troubleshooting gateways.

mapr.login.conf

The MapR Converged Data Platform uses the Java Authentication and Authorization Service (JAAS) to control security features. The `/opt/mapr/conf/mapr.login.conf` file specifies configuration parameters for JAAS. Contact MapR support before changing any parameters in this file other than the ones listed in this document.

The MAPR_SERVER_KERBEROS Stanza

The CLDB uses this stanza to verify users that are authenticating with Kerberos. This stanza requires the `com.sun.security.auth.module.Krb5LoginModule` module.

Attribute	Default Value	Description
keyTab	"/opt/mapr/conf/mapr.keytab"	File path to the keytab file.
principal	"mapr/my.cluster.com"	The Kerberos principal to use.

The MAPR_WEBSERVER_KERBEROS Stanza

Web UIs on the cluster use this stanza to evaluate SPNEGO requests. This stanza requires the `com.sun.security.auth.module.Krb5LoginModule` module.

Attribute	Default Value	Description
keyTab	"/opt/mapr/conf/mapr.keytab"	File path to the keytab file.

Attribute	Default Value	Description
principal	"HTTP/*yourhost*"	The principal *must* be HTTP. This principal is used to negotiate authentication for Web services over SPNEGO. You can set the value for *yourhost* manually, but be aware that you must set the principal in the `mapr.keytab` file to match this value.

The jpamLogin Stanza

The MapR cluster uses this stanza to verify user ID and password authentication to all the servers on the cluster. You can modify this stanza to alter the PAM configuration used by the cluster. The `net.sf.jpam.jaas.JpamLoginModule` module is sufficient for this stanza. There are three provided default services. The order of the `serviceName` in the stanza (at cluster startup) determines which PAM configuration file to use. If a failure occurs with a configuration, MapR ignores the error and proceeds with the next entry.

Attribute	Provided Default Values	Description
serviceName	• `sudo` • `sshd` • `mapr-admin`	The PAM configurations to use for validating passwords, shown in their order of use. The configuration files are typically in `/etc/pam.d`.

Other Stanzas

The `Server`, `Client`, `Server_simple`, `Client_simple`, and `hadoop_maprsasl` stanzas control important aspects of your cluster's stability. Consult with MapR support before modifying these stanzas.

mapr-clusters.conf
Provides information and instructions for configuring clusters to create the global namespace.

You can define one or more clusters in the `/opt/mapr/conf/mapr-clusters.conf` file on a node or client. To define a cluster, you specify the CLDB nodes in the cluster. Defining a cluster creates the global namespace. Note that the `mapr-clusters.conf` file on each node or client must have the same cluster configuration and naming convention to create the global namespace.

The following section describes the configuration format of the `mapr-clusters.conf` file:

Format:

```
<cluster-name1> secure=false  <CLDB> <CLDB> ... <CLDB>
[ clustername2  <CLDB> <CLDB> <CLDB> ]
[ ... ]
```

The <CLDB> string format can contain multiple space-separated instances of the following:

- `host;ip:port` - Host, IP, and port (uses DNS to resolve hostnames, or provided IP if DNS is down)
- `host:port` - Hostname and IP (uses DNS to resolve host, specifies port)
- `ip:port` - IP and port (avoids using DNS to resolve hosts, specifies port)
- `host` - Hostname only (default, uses DNS to resolve host, uses default port)
- `ip` - IP only (avoids using DNS to resolve hosts, uses default port)

You can edit `mapr-clusters.conf` manually to add more clusters. For example:

```
<cluster-name3>  <CLDB> <CLDB> <CLDB>
```

Security enabled, with and without Kerberos

With security enabled, without Kerberos, the format for the mapr-clusters.conf file is:

```
<clustername1> secure=true  <CLDB> <CLDB> ... <CLDB>
            [ <clustername2>  <CLDB> <CLDB> ... <CLDB> ]
            [ ... ]
```

With Kerberos enabled, the format for the `mapr-clusters.conf` file is:

```
<clustername1> secure=true kerberosEnable=true <CLDB> <CLDB> ... <CLDB>
            [ <clustername2>  <CLDB> <CLDB> ... <CLDB> ]
            [ ... ]
```

Note: Before renaming a cluster using the `mapr-clusters.conf` file, stop the warden on all the nodes.

Adding multihomed CLDB entries to mapr-clusters.conf with configure.sh

In this example, the cluster `my.cluster.com` has CLDB servers at `nodeA`, `nodeB`, `nodeC`, and `nodeD`. The CLDB servers `nodeB` and `nodeD` have two NICs each at `eth0` and `eth1`. The entries in `mapr-clusters.conf` are separated by spaces for each server's entry. Within a server's entry, individual interfaces are separated by semicolons (`;`).

The command

```
configure.sh -N my.cluster.com -C nodeAeth0,nodeCeth0 -M
nodeBeth0,nodeBeth1 -M nodeDeth0,nodeDeth1 -Z zknodeA
```

generates the following entry in `mapr-clusters.conf`:

```
my.cluster.com nodeAeth0 nodeBeth0;nodeBeth1 nodeCeth0 nodeDeth0;nodeDeth1
```

Cluster Limits

There is no limit for the number of clusters for HDFS and NFSv3. However, the maximum number of clusters for FUSE is 16.

mapred-site.xml
Lists the parameters for MapReduce configuration.

MapReduce is a type of application that can run on the Hadoop 2.x framework. MapReduce configuration options are stored in the `/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop/mapred-site.xml` file and are editable by the `root` user. This file contains configuration information that overrides the default values for MapReduce parameters. Overrides of the default values for core configuration properties are stored in the MapR Parameters on page 2245 file.

To override a default value for a property, specify the new value within the `<configuration>` tags, using the following format:

```
<property>
 <name> </name>
 <value> </value>
 <description> </description>
</property>
```

Configurations for MapReduce Applications

The configuration comprises the following parameters:

mapreduce.framework.name	*Value*: yarn *Description*: Execution framework set to Hadoop YARN.
mapreduce.input.fileinputformat.split.maxblocknum	*Value*: 0 *Description*: Number of blocks that can be added to one split. A value of *0* means that a single split is generated per node. This functionality requires a patch. To install patches, see Applying a Patch.
mapreduce.map.memory.mb	*Value*: 1024 *Description*: Larger resource limit for maps.
mapreduce.map.java.opts	*Value*: -Xmx1024M *Description*: Larger heap-size for child jvms of maps.
mapreduce.reduce.memory.mb	*Value*: 3072 *Description*: Larger resource limit for reduces.
mapreduce.reduce.java.opts	*Value*: -Xmx2560m *Description*: Larger heap-size for child jvms of reduces.
mapreduce.task.io.sort.mb	*Value*: 512 *Description*: Higher memory limit while sorting data for efficiency.
mapreduce.task.io.sort.factor	*Value*: 100 *Description*: More streams merged at once while sorting files.
mapreduce.reduce.shuffle.parallelcopies	*Value*: 50 *Description*: Higher number of parallel copies run by reduces to fetch outputs from very large number of maps.

Configurations for MapReduce JobHistory Server

The configuration comprises the following parameters:

mapr.localspill.expiration.date	*Value*: *days* *Description*: Property to determine spill files expiration date in days. Default value is 30 days.
mapreduce.jobhistory.address	*Value*: MapReduce JobHistory Server *host:port* *Description*: Default port is 10020.
mapreduce.jobhistory.webapp.address	*Value*: MapReduce JobHistory Server Web UI *host:port* *Description*: Default port is 19888.
mapreduce.jobhistory.intermediate-done-dir	*Value*: `/mr-history/tmp` *Description*: Directory where history files are written by MapReduce applications.

mapreduce.jobhistory.intermediate-done-scan-timeout	*Value*: *milliseconds* *Description*: Timeout in milliseconds for rescanning the `done_intermediate` user directory to reduce JobHistory Server loading. Information about a job is received with a delay equal to the timeout. Adjust the setting based on the cluster load. Start with 5000 ms and increase timeout as needed. **Note:** This functionality requires a patch. To install patches, see Applying a Patch on page 437.
mapreduce.jobhistory.done-dir	*Value*: `/mr-history/done` *Description*: Directory where history files are managed by the MapReduce JobHistory Server.
mapreduce.jobhistory.webapp.https.address	*Value*: Secure MapReduce JobHistory Server Web UI *host:port* (HTTPS) *Description*: Default port is 19890.

Sample Hadoop 2.x mapred-site.xml File

The following `mapred-site.xml` file defines values for two job history parameters.

```
<configuration>
  <property>
    <name>mapreduce.jobhistory.address</name>
    <value>__HS_IP__:10020</value>
  </property>
  <property>
    <name>mapreduce.jobhistory.webapp.address</name>
    <value>__HS_IP__:19888</value>
  </property>
</configuration>
```

Configuration for Apache Shuffle

You can disable Direct Shuffle and enable Apache Shuffle for MapReduce applications through the following settings:

mapreduce.job.shuffle.provider.services	*Value*: `mapreduce_shuffle`
mapreduce.job.reduce.shuffle.consumer.plugin.class	*Value*: `org.apache.hadoop.mapreduce.task.reduce.Shuffle`
mapreduce.job.map.output.collector.class	*Value*: `org.apache.hadoop.mapred.MapTask$MapOutputBuffer`
mapred.ifile.outputstream	*Value*: `org.apache.hadoop.mapred.IFileOutputStream`
mapred.ifile.inputstream	*Value*: `org.apache.hadoop.mapred.IFileInputStream`
mapred.local.mapoutput	*Value*: `true`
mapreduce.task.local.output.class	*Value*: `org.apache.hadoop.mapred.YarnOutputFiles`

mfs.conf
Lists the parameters of the MFS configuration file.

The configuration file `/opt/mapr/conf/mfs.conf` specifies the following parameters about the MapR File System server on each node.

Warning: You must restart the File Server after making changes to this file.

Parameters

`mfs.server.ip`	*Default Value:* Not applicable Description: IP address of the File Server. For example, `192.168.10.10`.
`mfs.server.port`	*Default Value:* 5660 Description: Port used for communication with the server.
`mfs.cache.lru.sizes`	*Default Value:* • For version 4.0.1: `inode:6:log:6:meta:10:dir:40:small:15` • For version 4.0.2 and later versions: `inode:3:meta:6:small:27:dir:15:db:20:valc:3` Description: LRU cache configuration. See the section, **Notes on LRU Cache Configuration** for more information.
`mfs.on.virtual.machine`	*Default Value:* false Description: Specifies whether the MapR File System is running on a virtual machine.
`mfs.io.disk.timeout`	*Default Value:* 60 seconds Description: Timeout, in seconds, after which a disk is considered failed and taken offline. You can increase the timeout to tolerate slow disks.
`mfs.max.disks`	*Default Value:* 48 Description: Maximum number of disks supported on a single node.
`mfs.max.logfile.size.in.mb`	*Default Value:* 1000 MB Description: The maximum amount of disk space that the MFS logs can consume before the oldest log file is deleted; based on the following calculation: `maxSizePerLogFile = maxLogSize / MAX_NUM_OF_LOG_FILES` where • `maxLogSize` = total amount of space that MFS log files can consume • `MAX_NUM_OF_LOG_FILES` = total number of MFS log files
`mfs.max.resync.count`	*Default Value:* 16 Description: The number of parallel resync operations.
`mfs.subnets.whitelist`	*Default Value:* Not Applicable

	Description: A list of subnets (up to 256 characters) that are allowed to make requests to the File Server service and access data on the cluster.
mfs.disk.iothrottle.count	*Default Value:* 100 Description: The maximum number of outstanding requests on disk.  **Note:** You can disable throttling by setting a high value. This option is disabled if you set the value for `mfs.disk.is.ssd` to `1`.
mfs.disk.resynciothrottle.factor	*Default Value:* 20 Description: Controls the amount of time to wait before submitting a request to disk. Increasing this value reduces the wait time, and decreasing this value increases the wait time. For example, setting the value to 40, halves the wait time, while setting the value to 10, doubles the wait time.
mfs.network.resynciothrottle.factor	*Default Value:* 20 Description: Controls the amount of time to wait before sending a resync operation over the network. Increasing this value reduces the wait time, and decreasing this value increases the wait time. For example, setting the value to 40, halves the wait time, while setting the value to 10, doubles the wait time.
mfs.ssd.trim.enabled	*Default Value:* 0 Description: Set this parameter to `1` to enable TRIM operations for SSD devices.  **Note:** Enable TRIM only if it is recommended by the SSD vendor.
mfs.disk.is.ssd	*Default Value:* 0 Description: Specifies whether (`1`) or not (`0`) the drives are SSD. If the value is `0`, the drives are assumed to be rotations. If the value is `1`, the noop scheduler on the SSD is automatically enabled, and I/O throttling is disabled.
mfs.mem.debug.enabled	*Default Value:* 0 Description: Specifies whether file server should (1) or should not (0) track all memory allocations. The default value is `0`. If value is `1` , you can determine the root cause for high memory allocation, or determine the component consuming the most memory.
mfs.numrpcthreads	*Default Value:* 2 Description: Specifies the number of RPC threads per MFS instance. The valid range of values is from 1 to 4.
mfs.db.max.concurrent.internal.ops	*Default Value:* 73728 (72 * 1024) *Max Value:* 131072 (128 *1024) *Min Value:* 36864 (36*1024) Description: Regulates how many BatchGet operations can run in parallel when secondary indexes are present on the table. PUT operations on tables with secondary indexes convert to BatchGet operations on the tables. PUT operations that convert to a high volume of BatchGets can degrade performance. BatchGet operations are spread equally across

last-updated: Apr 11, 2022

	three threads (73728/3). Run `mrconfig dbinfo threads` to evaluate the throttling queue for each thread. **Important:** This parameter is only available with a patch for MapR Core-6.1.0. To install patches, see Applying a Patch.
`mfs.num.compress.threads`	*Default Value:* 1 Description: Reserved for internal use.
`mfs.max.aio.events`	*Default Value:* 5000 Description: Reserved for internal use.
`mfs.disable.periodic.flush`	*Default Value:* 0 Description: Reserved for internal use.
`mfs.ignore.container.delete`	*Default Value:* 0 Description: Reserved for internal use.
`mfs.ignore.readdir.pattern`	*Default Value:* 0 Description: Reserved for internal use.
`mfs.disable.IO.affinity`	*Default Value:* 0 Description: Reserved for internal use.
`mfs.deserialize.length`	*Default Value:* 8192 Description: Reserved for internal use.
`mfs.enable.nat`	*Default Value:* 0 Description: Reserved for internal use.
`mfs.bulk.writes.enabled`	*Default Value:* 0 Description: Reserved for internal use.

Example

```
mfs.server.ip=192.168.10.10
mfs.server.port=5660
mfs.cache.lru.sizes=inode:3:meta:6:small:27:dir:15:db:20:valc:3
mfs.on.virtual.machine=0
mfs.io.disk.timeout=60
mfs.max.disks=48
```

Notes on LRU Cache Configuration

The cache values are expressed as percentages, which vary based on the expected size of the data that the node is required to cache. The goal is to achieve a state in which most of the required data comes directly from the cache. You may need to tune the cache percentages based on your cluster configuration and the workload on specific nodes. Non-default allocations tend to work better for nodes that run only CLDB and nodes that do not have CLDB but do have a heavy MapR Database workload. Note the following recommendations.

- For CLDB-only nodes, increase the size of the cache for Dir LRU to 40%: change `dir:15` to `dir:40`A CLDB-only node is a file server node that hosts only the CLDB volume `mapr.cldb.internal` (no user volume data is hosted on the node). Dir LRU is used to host B-tree pages.
- For non-CLDB nodes with no MapR Database workload, optimize the cache to host as many file pages as possible. Change the value of the parameter to: `inode:3:meta:6:small:27:dir:6`

The remainder of the cache is used to cache file data pages.

Note: You need to restart MFS for the change in `mfs.conf` to take effect.

nfsserver.conf
Lists the parameters for the MapR NFS server.

The file `/opt/mapr/conf/nfsserver.conf` controls parameters related to MapR services and the warden. Most of the parameters are not intended to be edited directly by users. The following list shows the parameters of interest:

Compression	*Default Value*: `true` *Description*: Indicates whether compression is on (`true`) or off (`false`).
ChunkSize	*Default Value*: `67108864 bytes (64 MB)` *Description*: Size of each chunk.
CompThreads	*Default Value*: `2` *Description*: Number of threads for compression or decompression.
DrCacheSize	*Default Value*: `20480` *Description*: Duplicate request cache size.
DrCacheTimeout	*Default Value*: `62 seconds` *Description*: Duplicate request cache timeout in seconds.
DRCacheTimeOutOpt	*Default Value*: `0.5` *Description*: If the operations take more than `DrCacheTimeout * DRCacheTimeOutOpt`, the operations are not cached. For example, by default, if the operation takes more than 31 seconds — `(62 * .5) = 31` seconds — the operation is not cached. A value of `0` disables the cache.
HighMemLimitMB	*Default Value*: `disabled` (Parameter is commented out in the file) *Description*: The maximum amount of memory (in MB) that the NFS server process can use. If the NFS server process uses more memory than this value, then the server is automatically shutdown, and a Core file is generated for debugging. For example: `HighMemLimitMB=10000` indicates that the NFS server is shutdown if it consumes more than 10GB of memory. This parameter is effective only if you enable the `MemDebugEnable` parameter.
LogLevel	*Default Value*: `INFO` *Description*: Sets the level of log messages displayed in the output. Levels include: • DEBUG • INFO • WARN

last-updated: Apr 11, 2022

- ERROR
- CRITICAL
- OFF

MaxLogFileSize

Default Value: `1024 MB`

Description: The maximum amount of disk space that the NFS server logs can consume before the oldest log file is deleted, based on the following calculation:

```
maxSizePerLogFile = maxLogFileSize /
MAX_NUM_OF_LOG_FILES
```

where:

- `maxLogFileSize` is the total amount of space that NFS server log files may consume
- `MAX_NUM_OF_LOG_FILES` is the total number of NFS server log files

Logrotate support for both the `.log` and the `.err` files honor this setting.

Attention: `MaxLogFileSize` is not a combined size of `.log` and `.err` files. The `.log` and `.err` files can individually grow up to this size.

MemDebugEnable

Default Value: `false` (Parameter is commented out in the file)

Description: Set this parameter to `true` to enable memory tracking for the NFS server. This parameter works along with the `HighMemLimitMB` parameter.

MinLenForDeserialization

Default Value: `8192`

Description:Deserialize (if value is > 0) or do not deserialize (if value = 0) the response in the compression thread. If value is greater than 0, MapR deserializes requests with length >= value in the compression thread. If value is 0, requests of length < value are deserialized in the RPC thread itself..

RamfsMntDir

Default Value: `/ramfs/mapr`

Description: Mount point for the `ramfs` file for `mmap`.

RamfsSize

Default Value: `0.25`

Description: Size of the ramfile to use (percent of total physical memory). A value of `0` disables the use of ramfs.

WindowsAceSupport

Default Value: `false`

Description: Allow (`true`) or deny (`false`) access to a Windows client when ACEs are set. If `true`, the mode bits are set to `777`, the Windows client is granted access, and the operation is allowed based on the permissions enforced using mode bits and/or ACEs. If value is `false`, the mode bits are set to 000

and the Windows client is denied access. For more information, see Mounting NFS on a Windows Client on page 1188.

> **Tip:** Use separate NFS servers for Windows clients and non-Windows clients.

warden.conf

Lists the configuration parameters for Warden.

The file `/opt/mapr/conf/warden.conf` controls parameters related to MapR services and the Warden. Most of the parameters are not intended to be edited directly by users. The following list describes the parameters of interest:

Note: When defining heapsize values for services, keep in mind that `service.heapsize.percent` is bound by `service.heapsize.min`, if defined, and `service.heapsize.max`.

centralconfig.enabled

Sample Value: true

Description: Specifies whether to enable central configuration.

cldb.port

Sample Value: 7222

Description: The port to use for communicating with the CLDB.

enable.overcommit

Sample Value: true

Description:Set this value to `true` to allow services to start up, even if their memory demands exceed the memory provided by the node.

hoststats.port

Sample Value: 5660

Description: The port to use for communicating with the HostStats service.

hs.port

Sample Value: 1111

Description: Hoststats listening port for Metrics RPC activity.

hs.rpcon

Sample Value: true

Description: Indicates whether or not to configure Job Management.

hs.host

Sample Value: localhost

Description: Hoststats hostname for RPC activity.

isDB

Sample Value: true

Description: Specifies if MapR Database is in use. When this value is `false`, the `service.command.mfs.heapsize.percent` is set to 20. Do not manually edit this value. For more information, see Allocating Memory for Nodes on page 818..

kvstore.port

Sample Value: 5660

Description: The port for communicating with the Key/Value Store.

log.retention.exceptions

Sample Value: `mfs.log-*`

last-updated: Apr 11, 2022

	Description: Retains the following log files instead of removing them during the log file cleanup that occurs every ten days: `cldb.log`, `hoststats.log`, `configure.log` and `mfs.log-*`. You can modify the list. This parameter accepts partial file names and asterisks. Log files listed as exceptions are retained indefinitely. To disable all exceptions, comment out this parameter, that is, `#log.retention.exceptions`. When this parameter is null, that is, `log.retention.exceptions=`, no files are picked for log cleanup.
log.retention.time	*Sample Value*: 864000000 *Description*: All `.log` and `.out` files in the cluster are kept for a time period defined in milliseconds by the value of the `log.retention.time` parameter. The default value is ten days. Restart the Warden after changing this value.
mapr.home.dir	*Sample Value*: `/opt/mapr` *Description*: MapR installation directory.
mfs.port	*Sample Value*: 7222 *Description*: The port to use for communicating with the Fileserver.
pollcentralconfig.interval.seconds	*Sample Value*: 300 *Description*: The frequency (in seconds) to check for central configuration updates.
rpc.drop	*Sample Value*: false *Description*: Drop outstanding metrics when the queue to send to hoststats is too large.
service.command.cldb.heapsize.max	*Sample Value*: 4000 *Description*: The maximum heap space, specified in MB, that the CLDB can use.
service.command.cldb.heapsize.min	*Sample Value*: 256 *Description*: The minimum heap space, specified in MB, that the CLDB can use.
service.command.cldb.heapsize.percent	*Sample Value*: 8 *Description*: The percentage of heap space reserved for CLDB.
service.command.cldb.retryinterval.time.sec	*Sample Value*: 600 *Description*: Specifies an interval in seconds. The warden attempts to restart a failed CLDB service when this interval expires.

Note: The warden restarts the CLDB service only if the service has been stopped unintentionally - for example, the service crashed. Warden does not restart a CLDB service that has been stopped intentionally using the `maprcli node services` command.

service.command.mfs.heapsize.maxpercent

Sample Value: 85

Description: The maximum percentage of heap space that can be allocated to the MapR File System. Restart the Warden after modifying this setting.

service.command.mfs.heapsize.min

Sample Value: 512

Description: The minimum heap space, specified in MB, that can be allocated to the MapR File System. Restart the Warden after modifying this setting.

service.command.mfs.heapsize.percent

Sample Value: 35

Description: The percentage of heap space reserved for the MapR File System. If the value for `isDB` is `true`, you cannot set this property to a value lower than 35. If you set this parameter to a value lower than 35, the value reverts to 35 when the Warden restarts, to ensure that when the MapR Database is enabled, corresponding cache allocation occurs. If you want to lower the heapsize allocated to the MapR File System, you must change `service.command.mfs.heapsize.maxpercent` instead. Restart the Warden after modifying this setting.

service.command.nfs.heapsize.max

Sample Value: 1000

Description: The maximum heap space, specified in MB, that the NFS can use.

service.command.nfs.heapsize.min

Sample Value: 64

Description: The minimum heap space, specified in MB, that the NFS can use.

service.command.nfs.heapsize.percent

Sample Value: 3

Description: The percentage of heap space reserved for the NFS.

service.command.os.heapsize.max

Sample Value: 750

Description: The maximum heap space, specified in MB, that can be used by the operating system.

service.command.os.heapsize.min

Sample Value: 256

Description: The minimum heap space, specified in MB, for use by the operating system.

service.command.os.heapsize.percent

Sample Value: 3

Description: The percentage of heap space reserved for the operating system.

service.command.webserver.heapsize.min

Sample Value: 512

last-updated: Apr 11, 2022

	Description: The minimum heap space, specified in MB, for use by the MapR Control System.
service.command.webserver.heapsize.percent	*Sample Value*: 3 *Description*: The percentage of heap space reserved for the MapR Control System.
service.nice.value	*Sample Value*: -10 *Description*: The `nice` priority under which all services run.
services.memoryallocation.alarm.threshold	*Sample Value*: 95 *Description*: The maximum amount of system memory that services running on the node can use before triggering the NODE_ALARM_MEMORY_ALLOCATION_EXCEEDED alarm.
services.resetretries.time.sec	*Sample Value*: 3600 *Description*: Specifies a time interval in seconds. The `services.retries` parameter sets the number of times that the warden attempts to restart failing services within this interval.
services.retries	*Sample Value*: 3 *Description*: The number of times the Warden tries to restart a service that fails.
services.retryinterval.time.sec	*Sample Value*: 1800 *Description*: The number of seconds after which the warden attempts several times to start a failed service. The number of attempts after each interval is specified by the parameter `services.retries`.
warden.enable.jmxremote	*Sample Value*: `false` *Description*: Set to `true` to enable the Warden JMX server.
zookeeper.servers	*Sample Value*: 10.250.1.61:5181 10.10.1.230:5181 *Description*: Space separated list of Zookeeper servers.

For information on configuration files for additional services, see warden.<servicename>.conf.

warden.<servicename>.conf

Describes the service configuration files that Warden supports.

The `warden.conf` configuration file is associated with the standard services that are provided by MapR. Warden supports service monitoring for additional services.

Each of these supported services requires a configuration file, `warden.<servicename>.conf`, which is included with the package for that service. When you install any of these service packages, its corresponding configuration file is stored in `/opt/mapr/conf/conf.d`. The configuration files and their packages are as follows:

collectd	*Configuration File*: `warden.collectd.conf` *Description*: Installed with the `mapr-collectd` package. This package is supported only for internal MapR Monitoring uses cases.

last-updated: Apr 11, 2022

drill	*Configuration File*: `warden.drill-bits.conf` *Description*: Installed with the `mapr-drill` package.
elasticsearch	*Configuration File*: `warden.elasticsearch.conf` *Description*: Installed with the `mapr-elasticsearch` package. This package is supported only for internal MapR Monitoring uses cases.
fluentd	*Configuration File*: `warden.fluentd.conf` *Description*: Installed with the `mapr-fluentd` package. This package is supported only for internal MapR Monitoring uses cases.
gateway	*Configuration File*: `warden.gateway.conf` *Description*: Installed with the `mapr-gateway` package.
grafana	*Configuration File*: `warden.grafana.conf` *Description*: Installed with the `mapr-grafana` package. This package is supported only for internal MapR Monitoring uses cases.
hue	*Configuration File*: `warden.hue.conf` *Description*: Installed with the `mapr-hue` package.
httpfs	*Configuration File*: `warden.httpfs.conf` *Description*: Installed with the `mapr-httpfs` package.
hbase thrift server	*Configuration File*: `warden.hbasethrift.conf` *Description*: Installed with the `mapr-hbasethrift` package.
hbase rest gateway	*Configuration File*: `warden.hbase-rest.conf` *Description*: Installed with the `mapr-hbase-rest` package.
historyserver	*Configuration File*: `warden.historyserver.conf` *Description*: Installed with the `mapr-historyserver` package.
hive metastore	*Configuration File*: `warden.hivemeta.conf` *Description*: Installed with the `mapr-hivemetadata` package.
hiveserver2	*Configuration File*: `warden.hs2.conf` *Description*: Installed with the `mapr-hiveserver2` package.
kibana	*Configuration File*: `warden.kibana.conf` *Description*: Installed with the `mapr-kibana` package. This package is supported only for internal MapR Monitoring uses cases.
nodemanager	*Configuration File*: `warden.nodemanager.conf`

last-updated: Apr 11, 2022

	Description: Installed with the `mapr-nodemanager` package.
oozie	*Configuration File*: `warden.oozie.conf` *Description*: Installed with the `mapr-oozie` package.
opentsdb	*Configuration File*: `warden.opentsdb.conf` *Description*: Installed with the `mapr-opentsdb` package. This package is supported only for internal MapR Monitoring uses cases.
resourcemanager	*Configuration File*: `warden.resourcemanager.conf` *Description*: Installed with the `mapr-resourcemanager` package.
sentry	*Configuration File*: `warden.sentry.conf` *Description*: Installed with the `mapr-sentry` package. However, Sentry is not automatically monitored by the Warden. When Sentry is configured to use the database storage model, you can manually copy the `/opt/mapr/sentry/sentry-<version>/conf.d/warden.sentry.conf` file to the `/opt/mapr/conf/conf.d` directory to add Sentry to the list of services that the Warden monitors.
spark master	*Configuration File*: `warden.spark-master.conf` *Description*: Installed with the `spark-master` package.

Configuring Service Properties

You can configure the following properties in the `warden.<servicename>.conf` file:

services	*Description*: Service name and number of nodes this service should run on, along with service dependencies. Format is `serviceName:N[depServiceName]`. Values for N = `1` or `all`
service.alarm.label	*Description*: Specifies the alarm name for this service. This is the alarm name that appears in the CLI when you do not request a terse output. Once tWarden starts the service, you cannot edit this value.
service.alarm.tersename	*Description*: Specifies the abbreviated alarm name for this service. This is the alarm name that appears in the Control System. Once Warden starts the service, you cannot edit this value.
service.command.monitor	*Description*: Monitor string (if the service monitor command does not provide sufficient monitoring).
service.command.monitorcommand	*Description*: Specifies a command that checks whether the service is running.
service.command.start	*Description*: Service `start` command.
service.command.stop	*Description*: Service `stop` command.

service.command.type	*Description*: Indicates whether the script runs in background (and exits) or inline (script does not exit). Type is either `BACKGROUND` or `INLINE`.
service.depends.local	*Description*: Indicates whether the service depends on a service instance locally, or on the master. Values = 1 (local) or 0 (master).
service.displayname	*Description*: The name of the service to display.
service.env	*Description*: Specifies environment variables to be use by the service. By default, it may include `MAPR_MAPREDUCE_MODE=default`. You can include a comma-separated list of environment variables. For example, `service.env=MAPR_MAPREDUCE_MODE=default, ABC=1,XYZ=2`.
service.heapsize.max	*Description*: Maximum heapsize in MB.
service.heapsize.min	*Description*: Minimum heapsize in MB.
service.heapsize.percent	*Description*: Specifies heapsize percent.
service.logs.location	*Description*: Location of the service log files.
service.port	*Description*: Port where the service is running (for example, the hue webserver runs on port 8888).
service.process.type	*Description*: Specifies the type of process. For example, `service.process.type=JAVA` indicates that the process is a Java process.
service.uri	*Description*: To include a link to a user interface associated with this service in the Control System, enter a Uniform Resource Identifier (URI) in this property, and specify the port in the `service.ui.port` property. For example, enter `/service1` for this property and then enter `8080` in the `service.ui.port` property to provide the following UI link for this service in the MCS: `http://<hostname>:8080/service1`
service.uri.port	*Description*: If you want to include a link to the user interface associated with this service in the Control System, enter the port in this property and also specify the URI in the `service.uri` property.

Memory Management for Services

The following memory parameters are used to reserve memory for the service:

- The `service.<servicename>.heapsize.percent` parameter controls the percentage of system memory allocated to the named service.
- The `service.<servicename>.heapsize.max` parameter defines the maximum heapsize used when invoking the service.
- The `service.<servicename>.heapsize.min` parameter defines the minimum heapsize used when invoking the service.

For example, the `service.command.gateway.heapsize.percent`, `service.command.gateway.heapsize.max`, and `service.command.gateway.heapsize.min` parameters in the `warden.gateway.conf` file control the amount of memory that Warden allocates to the gateway service before allocating memory to other services.

The actual heap size used when invoking a service is a combination of the three parameters according to the formula `max(heapsize.min, min(heapsize.max, total-memory * heapsize.percent / 100))`.

warden.hs2.conf Example

The hiveserver2 configuration file, `warden.hs2.conf`, looks like this:

```
services=hs2:1
service.displayname=HiveServer2
service.command.start=/opt/mapr/hive/hive-0.13/bin/hive --start --service
hiveserver2
service.command.stop=/opt/mapr/hive/hive-0.13/bin/hive --stop --service
hiveserver2
service.command.type=BACKGROUND
service.command.monitorcommand=/opt/mapr/hive/hive-0.13/bin/
hive --status --service hiveserver2
service.port=9083
service.ui.port=9083
service.uri=about
service.logs.location=/tmp/mapr
service.process.type=JAVA
```

When `hiveserver2` is installed, the `warden.hs2.conf` file is placed in the directory `/opt/mapr/conf/conf.d`. If Warden is running, it detects the file and starts the service. If Warden is not running, the file is picked up when Warden starts. Warden monitors the service and displays the status on the Control System UI.

yarn-site.xml

Describes the YARN configuration options.

YARN configuration options are stored in the `/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop/yarn-site.xml` file and are editable by the `root` user. This file contains configuration information that overrides the default values for YARN parameters. Overrides of the default values for core configuration properties are stored in the Default YARN parameters file.

To override a default value for a property, specify the new value within the `<configuration>` tags, using the following format:

```
<property>
 <name> </name>
   <value> </value>
 <description> </description>
</property>
```

The following configuration lists describe the possible entries that you can place between the `<name>` tags and between the `<value>` tags. The `<description>` tag is optional but recommended for maintainability.

Configuration for ResourceManager

Comprises the following parameters:

yarn.resourcemanager.hostname	The hostname of the ResourceManager.

	The configure.sh command automatically sets this value to the IP address that you provide with the `-RM` option. *Default value*: `{IP Address}`
yarn.resourcemanager.scheduler.address	The hostname and port of the Scheduler Interface. *Example value*: `${yarn.resourcemanager.hostname}:8030`
yarn.resourcemanager.resource-tracker.address	The hostname and port of the Resource Manager. *Example value*: `${yarn.resourcemanager.hostname}:8025`
yarn.resourcemanager.address	The address of the Applications Manager interface that is contained in the Resource Manager. *Example value*: `${yarn.resourcemanager.address}:8041`

Configuration for NodeManager

Comprises the following parameters:

yarn.nodemanager.container-localizer.log.level	*Default Value:*INFO *Description:*You can change the log level for the container localizer by setting the configuring options in this property. Different configuring options available are INFO, DEBUG, and WARN. By default logs will be available in the Application Master logs location but based on your cluster configuration, they will be available in the application's localized log directory. This functionality is available by default starting in EEP 7.1.0. For previous EEP versions, request the patch. See Applying a Patch on page 437.
yarn.nodemanager.max-retry-file-delete	*Default Value:* 2 *Description:* Defines how many times the NodeManager can attempt to delete application-related directories from a volume when Spark is configured to use the mounted NFS directory instead of the /tmp directory on the local filesystem. Increasing the value for this property can prevent application cache data from accumulating in the volume. This functionality is available by default starting in EEP 7.1.0. For previous EEP versions, request the patch. See Applying a Patch on page 437.
yarn.nodemanager.kill-container-child-process	*Default Value:* `false` *Description:* Enables NodeManager to automatically run the `kill -9` command to end processes that hang after YARN stops containers. Set to `true` to enable this behavior. This functionality is available by default starting in EEP 7.1.0. For previous EEP versions, request the patch. See Applying a Patch on page 437.
yarn.nodemanager.container-executor.class	*Default Value*: `org.apache.hadoop.yarn.server.nodemanager.LinuxContainerExecutor` *Description*: Identifies how containers are executed.

	Set to `LinuxContainerExecutor` by default, so that jobs can run as the user that submits the job. **Note:** If a system user (a user with *userID<500*) wants to submit a job, you must add the user in the *container-executor.cfg* file. The user `mapr` is already configured as an allowed system user.
yarn.nodemanager.aux-services	*Default Value*: `mapreduce_shuffle, mapr_direct_shuffle` *Description*: Selects a shuffle service that needs to be set for MapReduce to run.
yarn.nodemanager.aux-services.mapreduce_shuffle.class	*Default Value*: `org.apache.hadoop.mapred.ShuffleHandler` *Description*: This property, in conjunction with other properties, sets *direct shuffle* as the default shuffle for MapReduce.
yarn.nodemanager.aux-services.mapr_direct_shuffle.class	*Default Value*: `com.mapr.hadoop.mapred.LocalVolumeAuxService` *Description*: This property, in conjunction with other properties, sets *direct shuffle* as the default shuffle for MapReduce.

Configuration for Timeline Server Security with MapR-SASL

Comprises the following parameter:

yarn.timeline-service.http-authentication.type	*Default Value*: `com.mapr.security.maprauth.MaprDelegationTokenAuthenticationHand ler` *Description*: The authentication used for the timeline server HTTP endpoint.

Configuration for Timeline Server Security with Kerberos

Comprises the following parameter:

yarn.timeline-service.http-authentication.type	*Default Value*: `com.mapr.security.maprauth.MaprDelegationTokenAuthenticationHand ler` *Description*: The authentication used for the timeline server HTTP endpoint.
yarn.timeline-service.http-authentication.kerberos.principal	*Default Value*: `principal(HTTP/nodex@NODEx)` *Description*: The Kerberos service principal for the timeline server HTTP endpoint.
yarn.timeline-service.http-authentication.kerberos.keytab	*Default Value*: `path to keytab(/opt/mapr/conf/mapr.keytab)` *Description*: The Kerberos keytab for the timeline server HTTP endpoint.
yarn.timeline-service.principal	*Default Value*: `mapr/nodex@NODEX` *Description*: The Kerberos principal for the timeline reader. NodeManager principal is used for the timeline collector as it runs as an auxiliary service inside NodeManager.

yarn.timeline-service.keytab

Default Value: `path to keytab(/opt/mapr/conf/mapr.keytab)`

Description: The Kerberos keytab for the timeline reader. NodeManager keytab is used for the timeline collector as it runs as an auxiliary service inside NodeManager.

Configuration for MapReduce

Comprises the following parameter:

mapreduce.job.shuffle.provider.services

Default Value: `mapr_direct_shuffle`

Description: This is the default shuffle handler for MapReduce. Contains a value from the `yarn.nodemanager.aux-services` property.

Configuration for Container Logs

Comprises the following parameters:

yarn.nodemanager.log-dirs

Default Value: `/opt/mapr/hadoop/hadoop-<version>/logs/userlogs/<applicationID>/<containerID>/<filename>.log`

Description: The location to store container logs on the node. An application's log directory is `${yarn.nodemanager.log-dirs}/application_${appid}`. Individual containers' log directories are named `container_{$contid}`. Each container directory will contain the files *stderr*, *stdin*, and *syslog* generated by that container.

Note: You can find the application ID associated with your job in the Control System.

yarn.log-aggregation-enable

Default Value: `false`

Description: Indicates whether the logs are aggregated.

yarn.nodemanager.log.retain-seconds

Default Value: `10800 (3 hours)`

Description: Specifies the duration for which user logs are maintained, when log aggregation is disabled.

yarn.log-aggregation.retain-seconds

Default Value: `-1`

Description: Specifies the number of seconds to retain logs, when log aggregation is enabled. The default value of *-1*, disables the deletion of logs.

yarn.log-aggregation.retain-check-interval-seconds

Default Value: `-1`

Description: The interval between aggregated log retention checks. If set to *0* or a negative value, then the value is computed as one-tenth of the aggregated log retention time.

Note: Setting this to a low value may cause unnecessary log retention checks.

yarn.nodemanager.remote-app-log-dir

Default Value: `/tmp/logs`

last-updated: Apr 11, 2022

	Description: The location on the filesystem where the logs are aggregated.
yarn.nodemanager.remote-app-log-dir-suffix	*Default Value*: `logs` *Description*: The suffix for the directory that stores the aggregated logs for each user.

Configuration for Oozie

Comprises the following parameter:

yarn.resourcemanager.principal	*Default Value*: `mapr` *Description*: The name of the administrative user.

Configuration for Apache Shuffle

You can disable Direct Shuffle and enable Apache Shuffle for MapReduce applications through the following setting:

yarn.nodemanager.aux-services	*Value*: `mapreduce_shuffle`

zoo.cfg
Lists the ZooKeeper configuration file.

Example zoo.cfg File

The file `/opt/mapr/zookeeper/zookeeper-`*`$version`*`/conf/zoo.cfg` specifies ZooKeeper configuration parameters.

```
# The number of milliseconds of each tick
tickTime=2000
# The number of ticks that the initial
# synchronization phase can take
initLimit=20
# The number of ticks that can pass between
# sending a request and getting an acknowledgement
syncLimit=10
# the directory where the snapshot is stored.
dataDir=/opt/mapr/zkdata
# the port at which the clients will connect
clientPort=5181
# max number of client connections
maxClientCnxns=1000
#autopurge interval - 24 hours
autopurge.purgeInterval=24
#superuser to allow zk nodes delete
superUser=mapr
#readuser to allow read zk info for authenticated clients
readUser=anyone
# cldb key location
mapr.cldbkeyfile.location=/opt/mapr/conf/cldb.key
#security provider name
authMech=MAPR-SECURITY
# security auth provider
authProvider.1=org.apache.zookeeper.server.auth.SASLAuthenticationProvider
# use maprserverticket not userticket for auth
mapr.usemaprserverticket=true
#
# ZK-to-ZK server authentication using MAPR-SASL
# Set quorum.auth.enableSasl=false for insecure cluster, =true for secure
cluster
```

```
quorum.auth.enableSasl=true
quorum.auth.learnerRequireSasl=true
quorum.auth.serverRequireSasl=true
quorum.auth.learner.loginContext=QuorumLearner
quorum.auth.server.loginContext=QuorumServer
quorum.cnxn.threads.size=20
server.0=xxx.xxx.xxx.xxx:2888:3888
server.1=xxx.xxx.xxx.xxx:2888:3888
server.2=xxx.xxx.xxx.xxx:2888:3888
```

Warning: `maxClientCnxns` limits the number of concurrent ZooKeeper connections that a single client machine may make. This value does not set a limit for the whole cluster. The default is 100. If you plan to run more than 100 jobs from a single node, increase this value.

Attention: By default, only **authenticated** users (users with a valid ticket) are allowed to execute ZooKeeper related commands. To allow **all** users to execute ZooKeeper related commands, add the entry `sessionRequireClientSASLAuth=false` to this file and restart ZooKeeper.

Related tasks

Enabling Security on page 1405
Describes how to enable security for the cluster, platform, ecosystem components, and network-based connections.

zookeeper-env.sh
Use this file to load or unload JMX parameters for ZooKeeper.

Important: MapR version 6.0.x, and MapR version 6.1.x require a patch to use this functionality.

By default, in all MapR versions, the ZooKeeper JMX parameters are not loaded.

To load ZooKeeper JMX parameters, set the JMXDISABLE parameter to `false` in the `zookeeper-env.sh` file within the `/opt/mapr/zookeeper/zookeeper-$version/conf` directory.

```
JMXDISABLE=false
```

Restart the ZooKeeper process: `service mapr-zookeeper restart`.

To unload ZooKeeper JMX parameters, set:

```
JMXDISABLE=true
```

in `zookeeper-env.sh`. Restart the ZooKeeper process: `service mapr-zookeeper restart`.

Alarms Reference

The pages in this section provide details about all of the types of alarms.

User/Group Alarms

User/group alarms indicate problems with user or group quotas. The following tables describe the MapR user/group alarms.

- Entity Advisory Quota Alarm
- Entity Quota Alarm

Entity Advisory Quota Alarm

UI Column	User Advisory Quota Alarm

Logged As	AE_ALARM_AEADVISORY_QUOTA_EXCEEDED
Meaning	A user or group has exceeded its advisory quota. See Setting Quota Defaults for Users and Groups on page 781 for more information about user/group quotas.
Resolution	No immediate action is required. To avoid exceeding the hard quota, clear space on volumes created by the user or group, or stop further data writes to those volumes.
Configuration	Configurable when setting/modifying entity properties. See Configuring the Alarm Threshold Using the CLI on page 786 for more information.

Entity Quota Alarm

UI Column	User Quota Alarm
Logged As	AE_ALARM_AEQUOTA_EXCEEDED
Meaning	A user or group has exceeded its quota. Further writes by the user or group will fail. See Setting Quota Defaults for Users and Groups on page 781 and Set or Modify Quotas for Users and/or Groups on page 952 for more information about user/group quotas.
Resolution	Free some space on the volumes created by the user or group, or increase the user or group quota.
Configuration	Configurable when setting/modifying entity properties. See Configuring the Alarm Threshold Using the CLI on page 786 for more information.

Cluster Alarms

Cluster alarms indicate problems that affect the cluster as a whole. The following sections describe the data-fabric cluster alarms.

CLDB Low Memory Alarm

UI Column	Cluster freespace above CLDB heapsize
Logged As	CLUSTER_ALARM_CLDB_HEAPSIZE
Meaning	The CLDB process needs more memory to cache containers.
Resolution	The CLDB heap size is no longer sufficient for the CLDB to cache containers. The solution is to increase the CLDB memory settings on all CLDB nodes, using the same value for the minimum and maximum heap sizes. The text the alarm code provides will include the minimum amount of memory required to be sufficient; however, to accommodate future growth, you should set these values to a somewhat higher number. For example, if the alarm indicates that the CLDB needs 4000 MB, you should set the minimum and maximum heap sizes to a larger value such as 4400 MB. The CLDB memory settings are controlled by the following parameters in the `warden.conf` file located in `$MAPR_HOME/conf/:`:

	`service.command.cldb.heapsize.max=<max heap size>`
	`service.command.cldb.heapsize.min=<min heap size>`
	Restart the Warden service on each CLDB node after you edit the `warden.conf file`.

License Near Expiration

UI Column	License Near Expiration Alarm
Logged As	CLUSTER_ALARM_LICENSE_NEAR_EXPIRATION
Meaning	The Enterprise Edition license associated with the cluster is within 30 days of expiration.
Resolution	Renew the Enterprise Edition license.
Configuration	Configurable at cluster level. See Configuring the Alarm Threshold Using the CLI on page 786 for more information.

License Expired

UI Column	License Expiration Alarm
Logged As	CLUSTER_ALARM_LICENSE_EXPIRED
Meaning	The Enterprise Edition license associated with the cluster has expired. Enterprise Edition features have been disabled.
Resolution	Renew the Enterprise Edition license.

Cluster Almost Full

UI Column	Cluster Almost Full
Logged As	CLUSTER_ALARM_CLUSTER_ALMOST_FULL
Meaning	The cluster storage is almost full. The percentage of storage used before this alarm is triggered is 90% by default, and is controlled by the configuration parameter `cldb.cluster.almost.full.percentage`.
Resolution	Reduce the amount of data stored in the cluster. If the cluster storage is less than 90% full, check the `cldb.cluster.almost.full.percentage` parameter via the `config load` command, and adjust it if necessary via the `config save` command.
Configuration	Configurable at cluster level. See Configuring the Alarm Threshold Using the CLI on page 786 for more information.

Cluster Full

UI Column	Cluster Full
Logged As	CLUSTER_ALARM_CLUSTER_FULL
Meaning	The cluster storage is full. MapReduce operations have been halted.
Resolution	Free up some space on the cluster.

Maximum Licensed Nodes Exceeded alarm

UI Column	Licensed Nodes Exceeded Alarm
Logged As	CLUSTER_ALARM_LICENSE_MAXNODES_EXCEEDED
Meaning	The cluster has exceeded the number of nodes specified in the license.
Resolution	Remove some nodes, or upgrade the license to accommodate the added nodes.

New Cluster Features Disabled

UI Column	New Cluster Features Disabled
Logged As	CLUSTER_ALARM_NEW_FEATURES_DISABLED
Meaning	Features added in version 2.0 or 3.0 are not enabled on the cluster.
Resolution	Enable the latest features for the data-fabric version that you are currently running.

Upgrade in Progress

UI Column	Software Installation & Upgrades
Logged As	CLUSTER_ALARM_UPGRADE_IN_PROGRESS
Meaning	A rolling upgrade of the cluster is in progress.
Resolution	No action is required. Performance may be affected during the upgrade, but the cluster should still function normally. After the upgrade is complete, the alarm is cleared.

VIPAssignment Failure

UI Column	VIP Assignment Alarm
Logged As	CLUSTER_ALARM_UNASSIGNED_VIRTUAL_IPS
Meaning	Core software was unable to assign a VIP to any NFS servers.
Resolution	Check the VIP configuration, and make sure at least one of the NFS servers in the VIP pool are up and running. See Setting Up VIPs for NFS. This alarm can also indicate that a VIP's hostname exceeds the maximum allowed length of 16. Check the log file `/opt/mapr/logs/nfsmon.log` for additional information.

DARE Enabled

UI Column	DARE Enabled Alarm
Logged As	CLUSTER_ALARM_DARE_COPY_MASTER_KEY
Meaning	Data-at-rest encryption (DARE) is enabled on the cluster.
Resolution	When DARE is enabled on the cluster, a data-at-rest encryption master key file is generated and stored in `/opt/mapr/conf/dare.master.key` on the

	CLDB node. Before dismissing the alarm, make a copy of the master key file because loss of the master key file can be catastrophic and irreversible and might result in loss of data.

DARE Incompatible

UI Column	DARE Incompatible Alarm
Logged As	CLUSTER_ALARM_DARE_INCOMPATIBLE
Meaning	Not all nodes on the cluster are enabled for data-at-rest encryption (DARE).
Resolution	When DARE is enabled on certain nodes in the cluster, there may still be some nodes that are not (yet) enabled for DARE. Enable DARE on all the nodes before dismissing the alarm.

Too Many Snapshots

UI Column	Too Many Snapshots
Logged As	CLUSTER_ALARM_TOO_MANY_SNAPSHOT_CONTAINERS
Meaning	There are too many snapshots on this cluster.
Resolution	Delete snapshots from the cluster before dismissing the alarm.

Node Alarms

Node alarms indicate problems in individual nodes. The following tables describe the MapR node alarms.

CLDB Service Alarm

UI Column	CLDB Alarm
Logged As	NODE_ALARM_SERVICE_CLDB_DOWN
Meaning	The CLDB service on the node has stopped running.
Resolution	Go to the node information page or the **Services** page in the Control System to check whether the CLDB service is running. The warden will try three times to restart the service automatically. After an interval (30 minutes by default) the warden will again try three times to restart the service. The interval can be configured using the parameter `services.retryinterval.time.sec` in `warden.conf` on page 2209. If the warden successfully restarts the CLDB service, the alarm is cleared. If the warden is unable to restart the CLDB service, see more troubleshooting information.

Core Present Alarm

UI Column	Core Present
Logged As	NODE_ALARM_CORE_PRESENT
Meaning	A service on the node has crashed and created a core dump file. When all core files are removed, the alarm is cleared.

last-updated: Apr 11, 2022

Resolution	See troubleshooting information.

Debug Logging Active

UI Column	Excess Logs Alarm
Logged As	NODE_ALARM_DEBUG_LOGGING
Meaning	Debug logging is enabled on the node.
Resolution	Debug logging generates enormous amounts of data, and can fill up disk space. If debug logging is not absolutely necessary, turn it off: use the `setloglevel on page 1751` command. If it is absolutely necessary, make sure that the logs in `/opt/mapr/logs` are not in danger of filling the entire disk.

Disk Failure

UI Column	Disk Failure Alarm
Logged As	NODE_ALARM_DISK_FAILURE
Meaning	A disk has failed on the node.
Resolution	Check the disk health log (`/opt/mapr/logs/faileddisk.log`) to determine which disk failed and view any SMART data provided by the disk. See Managing Disks on page 834.

Duplicate Host ID

UI Column	Duplicate Host Id
Logged As	NODE_ALARM_DUPLICATE_HOSTID
Meaning	Two or more nodes in the cluster have the same host ID.
Resolution	Multiple nodes with the same host ID are prevented from joining the cluster, in order to prevent addressing problems that can lead to data loss. To correct the problem and clear the alarm, make sure all host IDs are unique and use the `maprcli node allow-into-cluster` command to un-ban the affected host IDs.

FileServer Service Alarm

UI Column	FileServer Alarm
Logged As	NODE_ALARM_SERVICE_FILESERVER_DOWN
Meaning	The FileServer service on the node has stopped running.
Resolution	Go to the node information page or the **Services** page in the Control System to check whether the FileServer service is running. The warden will try three times to restart the service automatically. After an interval (30 minutes by default) the

	warden will again try three times to restart the service. The interval can be configured using the parameter `services.retryinterval.time.sec` in `warden.conf` on page 2209 file. If the warden successfully restarts the FileServer service, the alarm is cleared. If the warden is unable to restart the FileServer service, see more troubleshooting information.

Heartbeat Processing Slow

UI Column	Heartbeat Processing Slow Alarm
Logged As	NODE_ALARM_HB_PROCESSING_SLOW
Meaning	The time that has elapsed since the CLDB processed the previous heartbeat from the MapR File System node has exceeded 5 seconds.
Resolution	When the CLDB is processing a heartbeat from a node, it will compare the current time to the time at which the previous heartbeat from that node was processed. If the elapsed time exceeds 5 seconds then this alarm is raised. If this alarm occurs frequently, investigate what might be causing the relevant node or nodes to be busy, or whether the CLDB nodes have enough resources to handle their load.

HBMaster Service Alarm

UI Column	HBase Master Alarm
Logged As	NODE_ALARM_SERVICE_HBMASTER_DOWN
Meaning	The HBMaster service on the node has stopped running.
Resolution	To check whether the HBMaster service is running, go to the node information page or the **Services** page in the Control System. Warden will try three times to restart the service automatically. After an interval (30 minutes by default), Warden will again try three times to restart the service. The interval can be configured using the `services.retryinterval.time.sec` parameter in the `warden.conf` file. If Warden successfully restarts the HBMaster service, the alarm is cleared. If Warden is unable to restart the HBMaster service, it might be necessary to contact technical support.

HBRegion Service Alarm

UI Column	Hbase RegionServer Alarm
Logged As	NODE_ALARM_SERVICE_HBREGION_DOWN
Meaning	The HBRegion service on the node has stopped running.
Resolution	To check whether the HBRegion service is running, go to the node information page or the **Services** page in the Control System. Warden will try three times to

	restart the service automatically. After an interval (30 minutes by default), Warden will again try three times to restart the service. The interval can be configured using the `services.retryinterval.time.sec` parameter in the `warden.conf` file. If Warden successfully restarts the HBRegion service, the alarm is cleared. If Warden is unable to restart the HBRegion service, it might be necessary to contact technical support.

High MAST Gateway Memory Alarm

UI Column	High Memory Usage
Logged As	NODE_ALARM_HIGH_MASTGATEWAY_MEMORY
Meaning	Memory consumption of MAST Gateway exceeds the memory allocated for MAST Gateway.
Resolution	Tune the percentage of node memory allocated for MAST Gateway in the `/opt/mapr/conf/conf.d/warden.mastgateway.conf` file. See Configuring MAST Gateway for more information. If core is generated for this error, contact MapR support.

HistoryServer Alarm

UI Column	HistoryServer Alarm
Logged As	NODE_ALARM_SERVICE_HISTORYSERVER_DOWN
Meaning	The HistoryServer on the node has stopped running.
Resolution	Go to the node information page or the **Services** page in the Control System to check whether HistoryServer is running. Warden will try three times to restart the service automatically ever 30 minutes (by default). This 30 minute interval can be reconfigured using the parameter `services.retryinterval.time.sec` in the `warden.conf` on page 2209 file. If warden successfully restarts the HistoryServer, the alarm is cleared. If Warden is unable to restart the HistoryServer, see more troubleshooting information.

HiveMeta Alarm

UI Column	HiveMeta Alarm
Logged As	NODE_ALARM_SERVICE_HIVEMETA_DOWN
Meaning	The HiveMeta service on the node has stopped running.
Resolution	Go to the node information page or the **Services** page in the Control System to check whether Hive Metastore is running. Warden will try three times to restart the service automatically ever 30 minutes (by default). This 30 minute interval can be reconfigured using the parameter `services.retryinterval.time.sec` in the `warden.conf` on page 2209 file.

	If Warden successfully restarts the Hive Metastore service, the alarm is cleared. If Warden is unable to restart the Hive Metastore service, see more troubleshooting information.

HiveServer 2 Alarm

UI Column	HiveServer 2 Alarm
Logged As	NODE_ALARM_SERVICE_HS2_DOWN
Meaning	The HiveServer 2 service on the node has stopped running.
Resolution	Go to the node information page or the **Services** page in the Control System to check whether HiveServer 2 is running. Warden will try three times to restart the service automatically ever 30 minutes (by default). This 30 minute interval can be reconfigured using the parameter `services.retryinterval.time.sec` in the `warden.conf` on page 2209 file. If Warden successfully restarts the HiveServer 2 service, the alarm is cleared. If Warden is unable to restart the HiveServer 2 service, see more troubleshooting information.

Hoststats Alarm

UI Column	HostStats
Logged As	NODE_ALARM_SERVICE_HOSTSTATS_DOWN
Meaning	The Hoststats service on the node has stopped running.
Resolution	Go to the node information page or the **Services** page in the Control System to check whether the Hoststats service is running. The warden will try three times to restart the service automatically. After an interval (30 minutes by default) the warden will again try three times to restart the service. The interval can be configured using the parameter `services.retryinterval.time.sec` in `warden.conf` on page 2209 file. If the warden successfully restarts the service, the alarm is cleared. If the warden is unable to restart the service, review more troubleshooting information.

Incorrect Topology Alarm

UI Column	CLDB Alarm
Logged As	NODE_ALARM_INCORRECT_TOPOLOGY_ALARM
Meaning	The `mapr.cldb.internal` volume's topology (normally `/cldb`) must include all CLDB nodes. This alarm signifies that one or more CLDB nodes are outside the CLDB volume's topology.
Resolution	There are two ways to resolve this alarm:

last-updated: Apr 11, 2022

- Move any stray CLDB nodes into the topology in which `mapr.cldb.internal` resides. See Setting Up Volume Topology on page 915 for more information.
- Change the volume topology of `mapr.cldb.internal` to include the stray CLDB nodes. See Administering Volumes on page 856 for more information.

Installation Directory Full Alarm

UI Column	Installation Directory Full
Logged As	NODE_ALARM_OPT_MAPR_FULL
Meaning	The partition `/opt/mapr` on the node is running out of space (95% full).
Resolution	Free up some space in `/opt/mapr` on the node.

Instance Mismatch Alarm

UI Column	Instance Mismatch Alarm
Logged As	NODE_ALARM_NUM_INSTANCES_MISMATCH
Meaning	The number of MapR File System instances is not as configured.
Resolution	Restart warden on the node by running the following command: `service mapr-warden restart`

High FileServer Memory Alarm

UI Column	High FileServer Memory Alarm
Logged As	NODE_ALARM_HIGH_MFS_MEMORY
Meaning	Memory consumed by **fileserver** service on the node is in excess of the allotted amount.
Resolution	Log on as root to the node for which the alarm is raised, and restart the Warden: `service mapr-warden restart`
Configuration	Configurable at cluster level. See Configuring the Alarm Threshold Using the CLI on page 786 for more information.

MapR User Mismatch

UI Column	MapR User Mismatch Alarm
Logged As	NODE_ALARM_MAPRUSER_MISMATCH
Meaning	The cluster nodes are not all set up to run MapR services as the same user (for example, some nodes

	are running MapR as `root` while others are running as `mapr_user`.
Resolution	For the nodes on which the User Mismatch alarm is raised, follow the steps in Changing the User for MapR Services from the Command-Line on page 832.

Memory Allocation Alarm

UI Column	Memory Allocation Alarm
Logged As	NODE_ALARM_MEMORY_ALLOCATION_EXCEEDED
Meaning	The percentage of system memory required to run services on the node exceeds the set threshold and could potentially overload the node. If you installed a service on the node that causes the sum of memory used by the services on the node to exceed the threshold set, the system raises the alarm.
Resolution	To clear the alarm, you can add more memory to the node, stop a service from running on the node, or remove a service from the node. You can run the `service list` command to see the memory allocated to each service on the node. See service list for more information. The `services.memoryallocation.alarm.threshold` property in `warden.conf` defines the maximum amount of system memory that services running on the node can use before triggering the alarm. The default setting for this property is 95 percent: `services.memoryallocation.alarm.threshold=95` The percentage of system memory that services can use on the node should not exceed 95. Restart the Warden service on the node after you edit the `warden.conf` file.

Memory Usage Alarm

UI Column	Memory Usage Alarm
Logged As	NODE_ALARM_MEMORY_SWAPPING
Meaning	The HostStats service raises this alarm for swap space when the delta of swap in memory and the delta of swap out memory exceeds the threshold set over a specific time period.
Resolution	To clear the alarm, you can increase the physical memory or reduce the load running on the node. You can run the `service list` command to see the memory allocated to each service on the node. See `service list` for more information. The memory swapping alarm is controlled by the following properties in `/opt/mapr/conf/hoststats.conf`:

last-updated: Apr 11, 2022

	• `alarm.swapping.threshold` • `alarm.swapping.counter` The memory threshold for swap in and swap out is defined by the `alarm.swapping.threshold` property, which is set to 100MB by default. The duration over which HostStats checks the delta of the memory is defined by the `alarm.swapping.counter`, which is set to 100 seconds by default.

NFS Gateway Alarm

UI Column	NFS Service Down
Logged As	NODE_ALARM_SERVICE_NFS_DOWN
Meaning	The NFS service on the node has stopped running.
Resolution	Go to the node information page or the **Services** page in the Control System to check whether the NFS service is running. The warden will try three times to restart the service automatically. After an interval (30 minutes by default) the warden will again try three times to restart the service. The interval can be configured using the parameter `services.retryinterval.time.sec` in `warden.conf` on page 2209 file. If the warden successfully restarts the NFS service, the alarm is cleared. If the warden is unable to restart the NFS service, see more troubleshooting information.

NFSv4 Service Alarm

UI Column	NFSv4 Service Down
Logged As	NODE_ALARM_SERVICE_NFS4_DOWN
Meaning	The NFSv4 service on the node has stopped running.
Resolution	Go to the node information page or the **Services** page in the Control System to check whether the NFS service is running. The warden will try three times to restart the service automatically. After an interval (30 minutes by default) the warden will again try three times to restart the service. The interval can be configured using the parameter `services.retryinterval.time.sec` in `warden.conf` on page 2209 file. If the warden successfully restarts the NFS service, the alarm is cleared. If the warden is unable to restart the NFS service, refer to NFSv4 Troubleshooting on page 1219 to restart the service.

NodeManager Alarm

Explains how to resolve the issue with the Node Manager service stopping on the node.

UI Column	NodeManager Alarm
Logged As	NODE_ALARM_SERVICE_NODEMANAGER_DOWN

Meaning	The NodeManager service on the node has stopped running.
Resolution	Go to the node information page or the **Services** page in the Control System to check whether NodeManager is running. Warden will try three times to restart the service automatically ever 30 minutes (by default). This 30 minute interval can be reconfigured using the parameter `services.retryinterval.time.sec` in the `warden.conf` on page 2209 file. If warden successfully restarts the NodeManager, the alarm is cleared. If warden is unable to restart the NodeManager, see more troubleshooting information.

No Disk Attached Alarm

UI Column	No Disk Attached Alarm
Logged As	NODE_ALARM_NO_DISK_ATTACHED
Meaning	There are one or more MapR File System instances on a node with no SP assigned to them.
Resolution	To clear the alarm, assign at least one SP per MapR File System.

No Heartbeat Alarm

Describes the NODE_ALARM_NO_HEARTBEAT alarm.

UI Column	No Heartbeat Alarm
Logged As	NODE_ALARM_NO_HEARTBEAT
Meaning	Node is not undergoing maintenance, and no heartbeat detected for over 5 minutes.
Resolution	Check the status of the node manually.
Configuration	Configurable at cluster level. See Configuring the Alarm Threshold Using the CLI on page 786 for more information.

This alarm is raised when a node is down for more than 5 minutes. By default, this alarm is not raised if an edge node is down. To raise an alarm for edge nodes as well, set the CLDB parameter `cldb.ignore.posix.only.hb.alarm` to `0` using the command:

```
/opt/mapr/bin/maprcli config save -values
'{cldb.ignore.posix.only.hb.alarm:"0"}'
```

Security Certificate Expiry Alarm

Describes the NODE_ALARM_CERTIFICATE_NEAR_EXPIRATION alarm.

Note: On MapR version 6.1, you need to install the latest patch to enable this functionality.

UI Column	SSL Certificate Expiry
Logged As	NODE_ALARM_CERTIFICATE_NEAR_EXPIRATION
Meaning	SSL certificates are expiring within the number of days denoted by the CLDB setting

last-updated: Apr 11, 2022

	`cldb.ssl.cert.expiring.alarm.days`. See cldb.conf on page 2182 for more information.
Resolution	Renew the SSL certificates. See Importing a Certificate Authority Signed (CA Signed) SSL Certificate Into a MapR Cluster for more information.
Configuration	None.

Specification	This alarm is raised when any of the first ten security certificates in `/opt/mapr/conf/ssl_keystore` or in `/opt/mapr/conf/ssl_truststore` are set to expire within the number of days denoted by the CLDB setting `cldb.ssl.cert.expiring.alarm.days`. Once the alarm is raised, the administrator needs to find out the certificates that are expiring, and renew them.
	To find out the certificates that are expiring, use the `/opt/mapr/server/getSSLExpiryCerts.py` Python script. For example:
	`python /opt/mapr/server/getSSLExpiryCerts.py -print`
	`Below certificates expiring in the next 120 days`
	`Truststore:`
	`Alias: 100day valid until: Mon Jul 13 04:04:15 PDT 2020`
	`Alias: 65day valid until: Mon Jun 08 03:45:44 PDT 2020`
	`Alias: 70day valid until: Sat Jun 13 03:46:00 PDT 2020`
	`Alias: 80day valid until: Tue Jun 23 03:46:14 PDT 2020`
	`Alias: 90day valid until: Fri Jul 03 04:03:57 PDT 2020`
	`Keystore:`
	`Alias: 3daymay17 valid until: Thu May 21 04:20:26 PDT 2020`

Related reference

cldb.conf on page 2182
Contains the configuration for CLDB nodes.

Node Too Many Containers

UI Column	Too Many Containers Alarm
Logged As	NODE_ALARM_TOO_MANY_CONTAINERS
Meaning	Number of containers on this node reached the maximum limit.
Resolution	Delete unused volumes or Snapshots. You can reset the maximum with: `maprcli config save -values {"pernode.numcntrs.alarm.thr":"<number>"}`
Configuration	Configurable at cluster level.

	This alarm is also raised when total number of containers (including snap containers) exceed 10 times the value of `pernode.numcntrs.alarm.thr`. See Configuring the Alarm Threshold Using the CLI on page 786 for more information.

Oozie Alarm

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

UI Column	Oozie Alarm
Logged As	NODE_ALARM_SERVICE_OOZIE_DOWN
Meaning	The Oozie service on the node has stopped running.
Resolution	Go to the node information page or the **Services** page in the Control System to check whether Oozie is running. Warden will try three times to restart Oozie automatically ever 30 minutes (by default). This 30 minute interval can be reconfigured using the parameter `services.retryinterval.time.sec` in the `warden.conf` on page 2209 file. If Warden successfully restarts Oozie, the alarm is cleared. If Warden is unable to restart Oozie, see more troubleshooting information.

PAM Misconfigured Alarm

UI Column	Pam Misconfigured Alarm
Logged As	NODE_ALARM_PAM_MISCONFIGURED
Meaning	The PAM authentication on the node is configured incorrectly.
Resolution	See PAM Configuration.

ResourceManager Alarm

UI Column	ResourceManager Alarm
Logged As	NODE_ALARM_SERVICE_RESOURCEMANAGER_DOWN
Meaning	The ResourceManager service on the node has stopped running.
Resolution	Go to the node information page or the **Services** page in the Control System to check whether ResourceManager is running. Warden will try three times to restart the service automatically ever 30 minutes (by default). This 30 minute interval can be reconfigured using the parameter `services.retryinterval.time.sec` in the `warden.conf` on page 2209 file. If warden successfully restarts the ResourceManager, the alarm is cleared. If warden is unable to restart the ResourceManager, see more troubleshooting information.

last-updated: Apr 11, 2022

Root Partition Full Alarm

UI Column	Root Partition Full
Logged As	NODE_ALARM_ROOT_PARTITION_FULL
Meaning	The root partition ('/') on the node is running out of space (99% full).
Resolution	Free up some space in the root partition of the node.

Tiny Buckets Flush Alarm

UI Column	Lot of Tiny Buckets Flushed
Logged As	NODE_ALARM_TINY_BUCKET_FLUSH
Meaning	Indicates lot of small buckets (<= 8mb) are getting flushed in DB resulting in performance degradation. You may see put operation performance going down during this phase and will need to take corrective actions to fix it. This will not bring down the cluster and data is still accessible.
Resolution	Increase memory for MapR File System as number of active tablets is very high.

Time Skew Alarm

UI Column	Time Skew Alarm
Logged As	NODE_ALARM_TIME_SKEW
Meaning	The clock on the node is out of sync with the master CLDB by more than 20 seconds.
Resolution	Use NTP to synchronize the time on all the nodes in the cluster.

Version Alarm

UI Column	Version Alarm
Logged As	NODE_ALARM_VERSION_MISMATCH
Meaning	One or more services on the node are running an unexpected version or there is a mismatch in the MapR File System patch versions on the nodes.
Resolution	Stop the node, Restore the correct version of any services you have modified, and re-start the node. See Administering Nodes on page 797.

WebServer Service Alarm

UI Column	Webserver Alarm
Logged As	NODE_ALARM_SERVICE_WEBSERVER_DOWN
Meaning	The WebServer service on the node has stopped running.

Resolution	Go to the node information page or the **Services** page in the Control System to check whether the WebServer service is running. The warden will try three times to restart the service automatically. After an interval (30 minutes by default) the warden will again try three times to restart the service. The interval can be configured using the parameter `services.retryinterval.time.sec` in `warden.conf` on page 2209. If the warden successfully restarts the WebServer service, the alarm is cleared. If the warden is unable to restart the WebServer service, see more troubleshooting information.

Table-Replication Alarms

You can view table-replication alarms using the Control System and the CLI. See Viewing Active Table Replication Alarms on page 1318 for more information.

Table Replication Errors
Explains the alarm that is raised when there are table replication errors.

Logged As	VOLUME_ALARM_TABLE_REPL_ERROR
Meaning	This alarm displays the paths and names of the source tables for which the alarms were issued. Up to ten (10) source tables, that have encountered an error, are displayed.

Diagnostics

To identify the cause of the alarm, run the maprcli table replica list command. This command displays an `errors` field with additional information about the error.

```
maprcli table replica list -path <table path>
```

The `errors` field provides the following information:

- Code - table replication error code:
- Host - host that the error occurred on
- Msg - error message information

For additional information about possible causes of the error, see the following log files (located in the **/opt/mapr/logs** directory):

- mfs.log-5 - for the nameserver node of the source table
- mfs.log-5 - for the nameserver node of the destination table
- gateway.log

last-updated: Apr 11, 2022

Error Conditions

Possible Error Conditions	Description
A missing table or column family on the destination cluster	If a column family no longer exists in the replica, pause replication with the `maprcli table replica pause` command, recreate the column family with the `maprcli table cf create`command, run the CopyTable utility to copy data from the source table into the column family, and then resume replication with the `maprcli table replica resume` command.
A mismatch in column family names for the table on the destination cluster	If the column family still exists in the replica but the name of the column family was changed, run the `maprcli table cf edit` command with the parameter `-newcfname` set to the correct name. Replication will resume automatically.
Unreachable gateways on the destination cluster	This error occurs only if none of the gateways on the cluster are reachable.
Autosetup with directcopy has failed while copying data from the source to the replica	This error could be raised under the following conditions: • If this error occurs due to a connection failure, this alarm should get resolved once the connection is restored. • If this error occurs due to other error conditions such as missing tables, mis-matched column families, or an unreachable gateway, the error condition may not get resolved on its own and may need administration action. For example, if a PUT operation fails on destination cluster due to an Out-Of-Space condition, the administrator will need to add more storage before the PUT retry succeeds.

Table Replication Lag High

Logged As

VOLUME_ALARM_TABLE_REPL_LAG_HIGH

Meaning

These alarms display the paths and names of the source tables for which the replication lag is high. High lag times might be caused by these conditions:

- High load on the source or replica
- Low network bandwidth between the source and replicas
- Miscellaneous error conditions that prevent replication from proceeding
- Replication explicitly paused on the source cluster

Configuration

Configurable at the volume level. See volume create on page 1931 for more information.

Table Replication Asynchronous

Logged As

VOLUME_ALARM_TABLE_REPL_ASYNC

Meaning

These alarms display the pathnames of the source tables that are involved.

	If MapR Database is replicating synchronously and it judges the latency of the replication stream to be too high, it will switch to asynchronous replication temporarily. After a new gateway is created, an existing gateway is restarted, or after latency is sufficiently reduced, MapR Database switches the mode of replication back to synchronous. You can also check whether a source table is being replicated synchronously or asynchronously by running the command `table replica list` on page 1863 .

Elasticsearch Formatting Alarm

Logged As	NODE_ALARM_SERVICE_ELASTICSEARCH_EXCP
Meaning	The put of primary table cannot be converted/ formatted for pushing to ElasticSearch. If the primary table has ElasticSearch as one of its replicas, that may miss updates on some rows as they could not be converted to a supported format for ElasticSearch.
Resolution	Check for rows that could not be formatted for pushing to ElasticSearch in MapR File System logs and insert row with data that can be pushed to ElasticSearch.

Secondary Index Alarms

Secondary index alarms indicate issues that MapR Database might encounter while updating secondary indexes. It is important that you understand what the alarms indicate, and how to resolve the issue causing them.

Secondary Index Update Lag High

Logged As	VOLUME_ALARM_TABLE_INDEX_LAG_HIGH
Meaning	This alarm displays the paths and names of the source tables for which replication lag is high. High lag times might be caused by these conditions: • High load on the source or replica • Low network bandwidth between the source and replicas • Miscellaneous error conditions that prevent replication from proceeding • Replication explicitly paused on the source cluster
Resolution	If you have configured the threshold for this alarm too low, you can increase the theshhold. You configure this lag at the volume level by specifying the `dbindexlagsecalarmthresh` parameter. See volume modify on page 2005 for more information.

Secondary Index Update Error

Logged As	VOLUME_ALARM_TABLE_INDEX_ERROR

Meaning	This alarm occurs if the JSON table regions cannot connect to any of the internal gateways used to update fields in a secondary index. This might be caused by these conditions: • The replication gateway failed. • You have configured too few replication gateway instances.
Resolution	If the gateway failed, restart it. Otherwise, add more gateways. See Managing Gateways on page 1154 for further information.

Secondary Index Encoding Error

Logged As	VOLUME_ALARM_TABLE_INDEX_ENCODING_ERROR
Meaning	This alarm occurs when any of the following encoding errors occur during index updates: • The indexed rowkey size is too big (> 32 Kb). • The indexed field contains a CAST function, and a failure occurs evaluating the CAST function.
Resolution	You must either correct your underlying data or redefine the index to avoid missing rows. See Troubleshooting Secondary Index Encoding Errors on page 1101 for details about how to identify these errors and possible corrective actions. You also must manually clear this alarm, even if you drop the index or the table.

Volume Alarms

Volume alarms indicate problems in individual volumes. The following sections describe the MapR volume alarms.

Compaction Failed

UI Column	Volume Compaction Failed
Logged As	VOLUME_ALARM_COMPACTION_FAILURE
Meaning	Data could not be purged as the compactor did not complete the run.
Resolution	Wait for the compactor to run again or manually trigger the compactor using the `volume compact` on page 1927 command.

Data Unavailable

UI Column	Data Alarm
Logged As	VOLUME_ALARM_DATA_UNAVAILABLE
Meaning	This is a potentially very serious alarm that may indicate data loss. Some of the data on the volume

	cannot be located. This alarm indicates that enough nodes have failed to bring the replication factor of part or all of the volume to zero. For example, if the volume is stored on a single node and has a replication factor of one, the Data Unavailable alarm will be raised if that volume fails or is taken out of service unexpectedly. If a volume is replicated properly (and therefore is stored on multiple nodes) then the Data Unavailable alarm can indicate that a significant number of nodes is down.
Resolution	Investigate any nodes that have failed or are out of service. • You can see which nodes have failed by looking at the Node Health pane in the **Overview** page in the Control System. • Check the cluster(s) for any snapshots or mirrors that can be used to re-create the volume. For additional troubleshooting information, see how to handle this alarm.

Data Under-Replicated
Describe the alarm that is triggered when a volume is under replicated.

UI Column	Replication Alarm
Logged As	VOLUME_ALARM_DATA_UNDER_REPLICATED
Meaning	The volume replication factor is lower than the desired replication factor set for the volume. This can be caused by failing disks or nodes, or the cluster may be running out of storage space.
Resolution	Investigate any nodes that are failing. You can see which nodes have failed by looking at the Node Health pane in the **Overview** page on the Control System. Determine whether it is necessary to add disks or nodes to the cluster. This alarm is generally raised when the nodes that store the volumes or replicas have not sent a heartbeat for five minutes. To prevent re-replication during normal maintenance procedures, MapR waits a specified interval (by default, one hour) before considering the node dead and re-replicating its data. You can control this interval by setting the `cldb.fs.mark.rereplicate.sec` parameter using the `config save` command. For additional troubleshooting information, see how to handle this alarm.

Warm-Tier Data Node Down
Provides the resolution for the Warm-Tier Data Node Down alarm.

UI Column	Warm-Tier Data Node Down
Logged As	VOLUME_ALARM_DEGRADED_EC_STRIPES
Meaning	One of the nodes or SPs, on which either the data or parity fragments associated with the tiering enabled volume resides, is offline or down.

last-updated: Apr 11, 2022

Resolution	MapR tolerates failure of nodes equal to the number of parity fragments. However, to ensure the availability of data, add nodes to the topology or cluster.

Data Under-Encoded

UI Column	Volume Data Under-Encoded
Logged As	VOLUME_ALARM_CRITICALLY_DEGRADED_EC_STRIPES
Meaning	The number of nodes that are down is equal to the number of parity fragments.
Resolution	MapR tolerates failure of nodes equal to the number of parity fragments. However, to ensure the availability of data, add nodes to the topology or cluster.

Data Below-Parity

UI Column	Volume Data Below-Parity
Logged As	VOLUME_ALARM_EC_DATA_UNAVAILABLE
Meaning	The number of SPs or nodes that are offline or down exceed the number of parity fragments set for the tiering-enabled volume. MapR only tolerates failure of nodes equal to the number of parity fragments set for the tiering-enabled volume.
Resolution	Add more nodes to the topology or cluster.

Offload/Recall Failed

UI Column	Volume Offload/Recall Failed
Logged As	VOLUME_ALARM_OFFLOAD_RECALL_FAILURE
Meaning	The volume data could not be offloaded or could not be recalled.
Resolution	Check the log file for more information on the error. For some errors, CLDB tries to offload the data again after a brief wait. For more information, see Retrying Failed Operation on page 939.

Inodes Limit Exceeded

UI Column	Inodes Exceeded Alarm
Logged As	VOLUME_ALARM_INODES_EXCEEDED
Meaning	The volume contains too many files or the size of the namespace container size has exceeded the configured limit.
Resolution	This alarm indicates that not enough volumes are set up to handle the number of files stored in the cluster or the size of the name container exceeds the limit. Typically, each user or project should have a separate volume. To resolve the name container issue, investigate the cause for the alarm. After careful

	consideration, create one or more volumes and move data into the new volumes. See How to handle the VOLUME_ALARM_INODES_EXCEEDED alarm in MapR for more information on resolving this alarm.
Configuration	Configurable at cluster and volume level. See Configuring the Alarm Threshold Using the CLI on page 786 for more information.

Large Row

UI Label	Large Row
Logged As	VOLUME_ALARM_TABLE_LARGE_ROW_WARNING
Meaning	A row in a table within the specified volume has reached 75% of the maximum supported row size of 2 GB. The alarm provides the rowkey and the name of the table. If the rowsize exceeds 2 GB, subsequent MapR Database operations on the corresponding table region will fail with an I/O error.
Resolution	Ensure that client applications that access the table are managing row data correctly, so that no row exceeds 2 GB. The method of resolving the alarm depends on the way in which client applications were managing row data. For example, if client applications allowed too many versions of cell data, delete excess versions. If client applications neglected to remove old columns or column families, remove those manually.

Mirror Failure

UI Column	Mirror Alarm
Logged As	VOLUME_ALARM_MIRROR_FAILURE
Meaning	A mirror operation failed.
Resolution	Make sure the CLDB is running on both the source cluster and the destination cluster. Look at the CLDB log (`/opt/mapr/logs/cldb.log`) and the MapR filesystem log (`/opt/mapr/logs/mfs.log`) on both clusters for more information. If the attempted mirror operation was between two clusters, make sure that both clusters are reachable over the network. Make sure the source volume is available and reachable from the cluster that is performing the mirror operation. For more troubleshooting information, see how to handle this alarm.

No Nodes in Topology

UI Column	No Nodes in Vol Topo
Logged As	VOLUME_ALARM_NO_NODES_IN_TOPOLOGY
Meaning	The path specified in the volume's topology no longer corresponds to a physical topology that contains any nodes, either due to node failures or changes to node

	topology settings. While this alarm is raised, MapR places data for the volume on nodes outside the volume's topology to prevent write failures.
Resolution	Add nodes to the specified volume topology, either by moving existing nodes or adding nodes to the cluster. See Understanding Topology on page 457.

Snapshot Failure

UI Column	Snapshot Alarm
Logged As	VOLUME_ALARM_SNAPSHOT_FAILURE
Meaning	A snapshot operation failed.
Resolution	Make sure the CLDB is running. Look at the CLDB log (`/opt/mapr/logs/cldb.log`) and the MapR filesystem log (`/opt/mapr/logs/mfs.log`) on both clusters for more information. If the attempted snapshot was a scheduled snapshot that was running in the background, try a manual snapshot. For more troubleshooting information, see how to handle thi alarm.

Topology Almost Full

UI Column	Vol Topo Almost Full
Logged As	VOLUME_ALARM_TOPOLOGY_ALMOST_FULL
Meaning	The nodes in the specified topology are running out of storage space.
Resolution	Move volumes to another topology, enlarge the specified topology by adding more nodes, or add disks to the nodes in the specified topology.
Configuration	Configurable at cluster level. See Configuring the Alarm Threshold Using the CLI on page 786 for more information.

Topology Full Alarm

UI Column	Vol Topo Full
Logged As	VOLUME_ALARM_TOPOLOGY_FULL
Meaning	The nodes in the specified topology have out of storage space.
Resolution	Move volumes to another topology, enlarge the specified topology by adding more nodes, or add disks to the nodes in the specified topology.

Volume Advisory Quota Alarm

UI Column	Vol Advisory Quota Alarm
Logged As	VOLUME_ALARM_ADVISORY_QUOTA_EXCEEDED
Meaning	A volume has exceeded its advisory quota.

Resolution	No immediate action is required. To avoid exceeding the hard quota, clear space on the volume or stop further data writes.
Configuration	Configurable in volume properties. See Configuring the Alarm Threshold Using the CLI on page 786 for more information.

Volume Become Master Stuck

UI Column	VOLUME BECOME MASTER STUCK
Logged As	VOLUME_ALARM_BECOME_MASTER_STUCK
Meaning	This means that there are some containers (associated with the volume) that don't have Master role. The alarm description displays the containers on which Master role is not assigned.
Resolution	Run `dump containerinfo` on page 1616 command to determine the node on which role is not assigned and restart that node. If you see the alarm after the node is restarted, contact MapR support.

Volume with Non-Local Containers

UI Column	Local Volume containers non-local
Logged As	VOLUME_ALARM_DATA_CONTAINERS_NONLOCAL
Meaning	This is a local volume and its containers should all reside on the same node. Some containers were created on another node, which may cause performance issues in MapReduce applications.
Resolution	Recreate the local volume or review information on how to handle this alarm.

Volume Quota Alarm

UI Column	Vol Quota Alarm
Logged As	VOLUME_ALARM_QUOTA_EXCEEDED
Meaning	A volume has exceeded its quota. Further writes to the volume will fail.
Resolution	Free some space on the volume or increase the volume hard quota.
Configuration	Configurable in volume properties. See Configuring the Alarm Threshold Using the CLI on page 786 for more information.

MapR Environment

This section provides information associated with the MapR environment.

MapR Parameters

The following table lists user-configurable parameters and their default values. These default values reflect those in the default configuration files, plus any overrides shipped out-of-the-box in `core-site.xml`, `mapred-site.xml`, or other configuration files. You can override these values by editing or adding them

in `mapred-site.xml` or `core-site.xml`, using the `-D` option to the `hadoop jar` command when submitting a job, or by setting them explicitly in your code.

Parameter	Default
fs.mapr.bailout.on.library.mismatch	true
fs.mapr.bind.retries	false
fs.mapr.working.dir	
fs.maprfs.impl	
fs.ramfs.impl	
fs.s3.block.size	33554432
fs.s3.blockSize	33554432
fs.s3.buffer.dir	
fs.s3.impl	
fs.s3.maxRetries	4
fs.s3.sleepTimeSeconds	10
fs.s3n.block.size	33554432
fs.s3n.blockSize	33554432
fs.s3n.impl	
fs.trash.interval	0
hadoop.logfile.count	10
hadoop.logfile.size	10000000
hadoop.native.lib	TRUE
hadoop.proxyuser.root.groups	root
hadoop.proxyuser.root.hosts	
hadoop.rpc.socket.factory.class.default	
hadoop.security.authentication	simple
hadoop.security.authorization	FALSE
hadoop.security.group.mapping	
hadoop.security.uid.cache.secs	14400
hadoop.tmp.dir	
hadoop.util.hash.type	murmur
hadoop.workaround.non.threadsafe.getpwuid	FALSE
io.bytes.per.checksum	512
io.compression.codecs	
io.file.buffer.size	8192
io.mapfile.bloom.error.rate	0.005
io.mapfile.bloom.size	1048576
io.serializations	

Parameter	Default
io.skip.checksum.errors	FALSE
io.sort.factor	256
io.sort.mb	380
io.sort.record.percent	0.17
io.sort.spill.percent	0.99
ipc.client.connect.max.retries	10
ipc.client.connection.maxidletime	10000
ipc.client.idlethreshold	4000
ipc.client.kill.max	10
ipc.client.max.connection.setup.timeout	20
ipc.client.tcpnodelay	TRUE
ipc.server.listen.queue.size	128
ipc.server.tcpnodelay	TRUE
job.end.retry.interval	30000
jobclient.completion.poll.interval	5000
jobclient.output.filter	FAILED
jobclient.progress.monitor.poll.interval	1000
keep.failed.task.files	FALSE
local.cache.size	1.07E+10
map.sort.class	
mapr.centrallog.dir	logs
mapr.localoutput.dir	output
mapr.localspill.dir	spill
mapr.localvolumes.path	
mapr.map.keyprefix.ints	1
mapr.task.diagnostics.enabled	FALSE
mapreduce.heartbeat.10	300
mapreduce.heartbeat.100	1000
mapreduce.heartbeat.1000	10000
mapreduce.job.complete.cancel.delegation.tokens	TRUE
mapreduce.maprfs.use.compression	TRUE
mapreduce.reduce.input.limit	-1
mapreduce.task.classpath.user.precedence	FALSE
maprfs.openfid2.prefetch.bytes	0
tasktracker.http.threads	2
topology.node.switch.mapping.impl	

Default MapR Configurations

The default values for configuration parameters can come for various sources:

- marpred-default.xml
- core-default.xml
- yarn-default.xml
- MapR code
- Hadoop code

The topics in this section include the default values for each parameter and the source of the default.

Note: For each parameter, an entry in the <name>-site.xml file overrides the default.

Default core Parameters

Property	Description
dfs.bytes-per-checksum	Default value: 512 Default source: code
dfs.ha.fencing.ssh.connect-timeout	Default value: 30000 Default source: core-default.xml
dfs.namenode.checkpoint.dir	Default value:${hadoop.tmp.dir}/dfs/namesecondary Default source: code
dfs.namenode.checkpoint.edits.dir	Default value: ${fs.checkpoint.dir} Default source: code
dfs.namenode.checkpoint.period	Default value: 3600 Default source: code
file.blocksize	Default value: 67108864 Default source: core-default.xml
file.bytes-per-checksum	Default value: 512 Default source: core-default.xml
file.client-write-packet-size	Default value: 65536 Default source: core-default.xml
file.replication	Default value: 1 Default source: core-default.xml
file.stream-buffer-size	Default value: 4096 Default source: core-default.xml
fs.AbstractFileSystem.file.impl	Default value: org.apache.hadoop.fs.local.LocalFs Default source: core-default.xml

fs.AbstractFileSystem.ftp.impl	Default value: org.apache.hadoop.fs.ftp.FtpFs Default source: core-default.xml
fs.AbstractFileSystem.har.impl	Default value: org.apache.hadoop.fs.HarFs Default source: core-default.xml
fs.AbstractFileSystem.hdfs.impl	Default value: com.mapr.fs.MFS Default source: code
fs.AbstractFileSystem.maprfs.impl	Default value: com.mapr.fs.MFS Default source: code
fs.AbstractFileSystem.viewfs.impl	Default value: org.apache.hadoop.fs.viewfs.ViewFs Default source: core-default.xml
fs.automatic.close	Default value: TRUE Default source: core-default.xml
fs.checkpoint.size	Default value: 67108864 Default source: code
fs.client.resolve.remote.symlinks	Default value: TRUE Default source: core-default.xml
fs.defaultFS	Default value: maprfs:/// Default source: code
fs.df.interval	Default value: 60000 Default source: core-default.xml
fs.du.interval	Default value: 600000 Default source: core-default.xml
fs.file.impl	Default value: org.apache.hadoop.fs.LocalFileSystem Default source: code
fs.ftp.host	Default value: 0.0.0.0 Default source: core-default.xml
fs.ftp.host.port	Default value: 21 Default source: core-default.xml
fs.ftp.impl	Default value: org.apache.hadoop.fs.ftp.FTPFileSystem Default source: code
fs.har.impl	Default value: org.apache.hadoop.fs.HarFileSystem Default source: code
fs.har.impl.disable.cache	Default value: TRUE Default source: core-default.xml

fs.hdfs.impl	Default value: com.mapr.fs.MapRFileSystem Default source: code
fs.hftp.impl	Default value: org.apache.hadoop.hdfs.HftpFileSystem Default source: code
fs.hsftp.impl	Default value: org.apache.hadoop.hdfs.HsftpFileSystem Default source: code
fs.kfs.impl	Default value: org.apache.hadoop.fs.kfs.KosmosFileSystem Default source: code
fs.mapr.flush.unaligned	Default value: false Default source: code
fs.mapr.rathreads	Default value: 0 Default source: code
fs.mapr.working.dir	Default value: /user/$USERNAME/ Default source: code
fs.mapr.write.idleflush.timeout	Default value: 3 seconds Default source: code
fs.maprfs.impl	Default value: com.mapr.fs.MapRFileSystem Default source: code
fs.permissions.umask-mode	Default value: 22 Default source: core-default.xml
fs.ramfs.impl	Default value: org.apache.hadoop.fs.InMemoryFileSystem Default source: code
fs.s3.block.size	Default value: 33554432 Default source: code
fs.s3.blockSize	Default value: 33554432 Default source: code
fs.s3.buffer.dir	Default value: ${hadoop.tmp.dir}/s3 Default source: core-default.xml
fs.s3.impl	Default value: org.apache.hadoop.fs.s3native.NativeS3FileSystem Default source: code
fs.s3.maxRetries	Default value: 4 Default source: core-default.xml

fs.s3.sleepTimeSeconds	Default value: 10 Default source: core-default.xml
fs.s3a.attempts.maximum	Default value: 10 Default source: core-default.xml
fs.s3a.buffer.dir	Default value: ${hadoop.tmp.dir}/s3a Default source: core-default.xml
fs.s3a.connection.establish.timeout	Default value: 5000 Default source: core-default.xml
fs.s3a.connection.maximum	Default value: 15 Default source: core-default.xml
fs.s3a.connection.ssl.enabled	Default value: TRUE Default source: core-default.xml
fs.s3a.connection.timeout	Default value: 50000 Default source: core-default.xml
fs.s3a.fast.buffer.size	Default value: 1048576 Default source: core-default.xml
fs.s3a.fast.upload	Default value: FALSE Default source: core-default.xml
fs.s3a.impl	Default value: org.apache.hadoop.fs.s3a.S3AFileSystem Default source: core-default.xml
fs.s3a.max.total.tasks	Default value: 1000 Default source: core-default.xml
fs.s3a.multipart.purge	Default value: FALSE Default source: core-default.xml
fs.s3a.multipart.purge.age	Default value: 86400 Default source: core-default.xml
fs.s3a.multipart.size	Default value: 104857600 Default source: core-default.xml
fs.s3a.multipart.threshold	Default value: 2147483647 Default source: core-default.xml
fs.s3a.paging.maximum	Default value: 5000 Default source: core-default.xml
fs.s3a.threads.core	Default value: 15 Default source: core-default.xml
fs.s3a.threads.keepalivetime	Default value: 60 Default source: core-default.xml

last-updated: Apr 11, 2022

fs.s3a.threads.max	Default value: 256 Default source: core-default.xml
fs.s3n.block.size	Default value: 33554432 Default source: code
fs.s3n.blockSize	Default value: 33554432 Default source: code
fs.s3n.impl	Default value: org.apache.hadoop.fs.s3native.NativeS3FileSystem Default source: code
fs.s3n.multipart.copy.block.size	Default value: 5368709120 Default source: core-default.xml
fs.s3n.multipart.uploads.block.size	Default value: 67108864 Default source: core-default.xml
fs.s3n.multipart.uploads.enabled	Default value: FALSE Default source: core-default.xml
fs.swift.impl	Default value: org.apache.hadoop.fs.swift.snative.SwiftNativeFileSystem Default source: core-default.xml
fs.webhdfs.impl	Default value: org.apache.hadoop.hdfs.web.WebHdfsFileSystem Default source: code
ftp.blocksize	Default value: 67108864 Default source: core-default.xml
ftp.bytes-per-checksum	Default value: 512 Default source: core-default.xml
ftp.client-write-packet-size	Default value: 65536 Default source: core-default.xml
ftp.replication	Default value: 3 Default source: core-default.xml
ftp.stream-buffer-size	Default value: 4096 Default source: core-default.xml
ha.failover-controller.cli-check.rpc-timeout.ms	Default value: 20000 Default source: core-default.xml
ha.failover-controller.graceful-fence.connection.retries	Default value: 1 Default source: core-default.xml
ha.failover-controller.graceful-fence.rpc-timeout.ms	Default value: 5000 Default source: core-default.xml

last-updated: Apr 11, 2022

ha.failover-controller.new-active.rpc-timeout.ms	Default value: 60000 Default source: core-default.xml
ha.health-monitor.check-interval.ms	Default value: 1000 Default source: core-default.xml
ha.health-monitor.connect-retry-interval.ms	Default value: 1000 Default source: core-default.xml
ha.health-monitor.rpc-timeout.ms	Default value: 45000 Default source: core-default.xml
ha.health-monitor.sleep-after-disconnect.ms	Default value: 1000 Default source: core-default.xml
ha.zookeeper.acl	Default value: world:anyone:rwcda Default source: core-default.xml
ha.zookeeper.parent-znode	Default value: /hadoop-ha Default source: core-default.xml
ha.zookeeper.session-timeout.ms	Default value: 5000 Default source: core-default.xml
hadoop.common.configuration.version	Default value: 0.23.0 Default source: core-default.xml
hadoop.http.authentication.kerberos.keytab	Default value: ${user.home}/hadoop.keytab Default source: core-default.xml
hadoop.http.authentication.kerberos.principal	Default value: HTTP/_HOST@LOCALHOST Default source: core-default.xml
hadoop.http.authentication.signature.secret	Default value: com.mapr.security.maprauth.MaprSignatureSecretFactory Default source: code
hadoop.http.authentication.signature.secret.file	Default value: ${user.home}/hadoop-http-auth-signature-secret Default source: core-default.xml
hadoop.http.authentication.signer.secret.provider	Default value: random Default source: code
hadoop.http.authentication.simple.anonymous.allowed	Default value: TRUE Default source: core-default.xml
hadoop.http.authentication.token.validity	Default value: 36000 Default source: core-default.xml
hadoop.http.authentication.type	Default value: simple Default source: core-default.xml

last-updated: Apr 11, 2022

hadoop.http.filter.initializers	Default value: org.apache.hadoop.http.lib.StaticUserWebFilter Default source: core-default.xml
hadoop.http.staticuser.user	Default value: unknown Default source: core-default.xml
hadoop.jetty.logs.serve.aliases	Default value: TRUE Default source: core-default.xml
hadoop.kerberos.kinit.command	Default value: kinit Default source: core-default.xml
hadoop.logfile.count	Default value: 10 Default source: code
hadoop.logfile.size	Default value: 10000000 Default source: code
hadoop.registry.jaas.context	Default value: Client Default source: core-default.xml
hadoop.registry.rm.enabled	Default value: FALSE Default source: core-default.xml
hadoop.registry.secure	Default value: FALSE Default source: core-default.xml
hadoop.registry.system.acls	Default value: sasl:yarn@, sasl:mapred@, sasl:hdfs@ Default source: core-default.xml
hadoop.registry.zk.connection.timeout.ms	Default value: 15000 Default source: core-default.xml
hadoop.registry.zk.quorum	Default value: localhost:2181 Default source: core-default.xml
hadoop.registry.zk.retry.ceiling.ms	Default value: 60000 Default source: core-default.xml
hadoop.registry.zk.retry.interval.ms	Default value: 1000 Default source: core-default.xml
hadoop.registry.zk.retry.times	Default value: 5 Default source: core-default.xml
hadoop.registry.zk.root	Default value: /registry Default source: core-default.xml
hadoop.registry.zk.session.timeout.ms	Default value: 60000 Default source: core-default.xml

hadoop.rpc.protection	Default value: authentication Default source: core-default.xml
hadoop.rpc.socket.factory.class.default	Default value: org.apache.hadoop.net.StandardSocketFactory Default source: core-default.xml
hadoop.security.authentication	Default value: SIMPLE Default source: code
hadoop.security.authorization	Default value: FALSE Default source: core-default.xml
hadoop.security.crypto.buffer.size	Default value: 8192 Default source: core-default.xml
hadoop.security.crypto.cipher.suite	Default value: AES/CTR/NoPadding Default source: core-default.xml
hadoop.security.crypto.codec.classes.aes.ctr.nopadding	Default value: org.apache.hadoop.crypto.OpensslAesCtrCryptoCodec, org.apache.hadoop.crypto.JceAesCtrCryptoCodec Default source: core-default.xml
hadoop.security.group.mapping	Default value: org.apache.hadoop.security.JniBasedUnixGroupsMappingWithFallback Default source: core-default.xml
hadoop.security.group.mapping.ldap.directory.search.timeout	Default value: 10000 Default source: core-default.xml
hadoop.security.group.mapping.ldap.search.attr.group.name	Default value: cn Default source: core-default.xml
hadoop.security.group.mapping.ldap.search.attr.member	Default value: member Default source: core-default.xml
hadoop.security.group.mapping.ldap.search.filter.group	Default value: (objectClass=group) Default source: core-default.xml
hadoop.security.group.mapping.ldap.search.filter.user	Default value: (&(objectClass=user)(sAMAccountName={0})) Default source: core-default.xml
hadoop.security.group.mapping.ldap.ssl	Default value: FALSE Default source: core-default.xml
hadoop.security.groups.cache.secs	Default value: 300 Default source: core-default.xml
hadoop.security.groups.cache.warn.after.ms	Default value: 5000 Default source: core-default.xml

hadoop.security.groups.negative-cache.secs	Default value: 30 Default source: core-default.xml
hadoop.security.instrumentation.requires.admin	Default value: FALSE Default source: core-default.xml
hadoop.security.java.secure.random.algorithm	Default value: SHA1PRNG Default source: core-default.xml
hadoop.security.java.security.login.config.jar.path	Default value: /mapr.login.conf Default source: code
hadoop.security.kms.client.authentication.retry-count	Default value: 1 Default source: core-default.xml
hadoop.security.kms.client.encrypted.key.cache.expiry	Default value: 43200000 Default source: core-default.xml
hadoop.security.kms.client.encrypted.key.cache.low-watermark	Default value: 0.3f Default source: core-default.xml
hadoop.security.kms.client.encrypted.key.cache.num.refill.threads	Default value: 2 Default source: core-default.xml
hadoop.security.kms.client.encrypted.key.cache.size	Default value: 500 Default source: core-default.xml
hadoop.security.random.device.file.path	Default value: /dev/urandom Default source: core-default.xml
hadoop.security.uid.cache.secs	Default value: 14400 Default source: core-default.xml
hadoop.ssl.client.conf	Default value: ssl-client.xml Default source: core-default.xml
hadoop.ssl.enabled	Default value: FALSE Default source: core-default.xml
hadoop.ssl.enabled.protocols	Default value: TLSv1 Default source: core-default.xml
hadoop.ssl.exclude.cipher.suites	Default value: SSL_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA, SSL_RSA_EXPORT_WITH_DES40_CBC_SHA, SSL_RSA_EXPORT_WITH_RC4_40_MD5 Default source: code
hadoop.ssl.hostname.verifier	Default value: DEFAULT Default source: core-default.xml

hadoop.ssl.keystores.factory.class	Default value: org.apache.hadoop.security.ssl.FileBasedKeyStoresFactory Default source: core-default.xml
hadoop.ssl.require.client.cert	Default value: FALSE Default source: core-default.xml
hadoop.ssl.server.conf	Default value: ssl-server.xml Default source: core-default.xml
hadoop.tmp.dir	Default value: /tmp/hadoop-${user.name} Default source: core-default.xml
hadoop.user.group.static.mapping.overrides	Default value: dr.who=; Default source: core-default.xml
hadoop.util.hash.type	Default value: murmur Default source: core-default.xml
hadoop.work.around.non.threadsafe.getpwuid	Default value: FALSE Default source: core-default.xml
hadoop.workaround.non.threadsafe.getpwuid	Default value: FALSE Default source: code
io.bytes.per.checksum	Default value: 512 Default source: core-default.xml
io.compression.codec.bzip2.library	Default value: system-native Default source: core-default.xml
io.compression.codecs	Default value: org.apache.hadoop.io.compress.DefaultCodec, org.apache.hadoop.io.compress.GzipCodec, org.apache.hadoop.io.compress.BZip2Codec, org.apache.hadoop.io.compress.DeflateCodec, org.apache.hadoop.io.compress.SnappyCodec Default source: code
io.file.buffer.size	Default value: 8192 Default source: code
io.map.index.interval	Default value: 128 Default source: core-default.xml
io.map.index.skip	Default value: 0 Default source: core-default.xml
io.mapfile.bloom.error.rate	Default value: 0.005 Default source: core-default.xml

last-updated: Apr 11, 2022

io.mapfile.bloom.size	Default value: 1048576 Default source: core-default.xml
io.native.lib.available	Default value: TRUE Default source: core-default.xml
io.seqfile.compress.blocksize	Default value: 1000000 Default source: core-default.xml
io.seqfile.lazydecompress	Default value: TRUE Default source: core-default.xml
io.seqfile.local.dir	Default value: ${hadoop.tmp.dir}/io/local Default source: core-default.xml
io.seqfile.sorter.recordlimit	Default value: 1000000 Default source: core-default.xml
io.serializations	Default value: org.apache.hadoop.io.serializer.WritableSerialization Default source: code
io.skip.checksum.errors	Default value: FALSE Default source: core-default.xml
ipc.client.connect.max.retries	Default value: 10 Default source: core-default.xml
ipc.client.connect.max.retries.on.timeouts	Default value: 45 Default source: core-default.xml
ipc.client.connect.retry.interval	Default value: 1000 Default source: core-default.xml
ipc.client.connect.timeout	Default value: 20000 Default source: core-default.xml
ipc.client.connection.maxidletime	Default value: 10000 Default source: core-default.xml
ipc.client.fallback-to-simple-auth-allowed	Default value: FALSE Default source: core-default.xml
ipc.client.idlethreshold	Default value: 4000 Default source: core-default.xml
ipc.client.kill.max	Default value: 10 Default source: core-default.xml
ipc.client.max.connection.setup.timeout	Default value: 20 Default source: code

ipc.client.tcpnodelay	Default value: TRUE Default source: code
ipc.server.listen.queue.size	Default value: 128 Default source: core-default.xml
ipc.server.max.connections	Default value: 0 Default source: core-default.xml
ipc.server.tcpnodelay	Default value: TRUE Default source: code
mapr.home	Default value: /opt/mapr Default source: code
mapr.host	Default value: <hostname> Default source: code
mapr.localvolumes.path	Default value: /var/mapr/local Default source: code
mapr.mapred.localvolume.mount.path	Default value: ${mapr.localvolumes.path}/${mapr.host}/mapred Default source: code
mapr.mapred.localvolume.root.dir.path	Default value: ${mapr.mapred.localvolume.mount.path}/${mapr.mapred.localvolume.root.dir.name} Default source: code
net.topology.impl	Default value: org.apache.hadoop.net.NetworkTopology Default source: core-default.xml
net.topology.node.switch.mapping.impl	Default value: org.apache.hadoop.net.ScriptBasedMapping Default source: core-default.xml
net.topology.script.number.args	Default value: 100 Default source: core-default.xml
nfs.exports.allowed.hosts	Default value: * rw Default source: core-default.xml
rpc.metrics.quantile.enable	Default value: FALSE Default source: core-default.xml
s3.blocksize	Default value: 67108864 Default source: core-default.xml
s3.bytes-per-checksum	Default value: 512 Default source: core-default.xml
s3.client-write-packet-size	Default value: 65536 Default source: core-default.xml

last-updated: Apr 11, 2022

s3.replication	Default value: 3 Default source: core-default.xml
s3.stream-buffer-size	Default value: 4096 Default source: core-default.xml
s3native.blocksize	Default value: 67108864 Default source: core-default.xml
s3native.bytes-per-checksum	Default value: 512 Default source: core-default.xml
s3native.client-write-packet-size	Default value: 65536 Default source: core-default.xml
s3native.replication	Default value: 3 Default source: core-default.xml
s3native.stream-buffer-size	Default value: 4096 Default source: core-default.xml
tfile.fs.input.buffer.size	Default value: 262144 Default source: core-default.xml
tfile.fs.output.buffer.size	Default value: 262144 Default source: core-default.xml
tfile.io.chunk.size	Default value: 1048576 Default source: core-default.xml

Default YARN Parameters

Parameter	Description
mapreduce.job.hdfs-servers	Default value: ${fs.defaultFS} Default source: yarn-default.xml
yarn.acl.enable	Indicates whether ACLs are enabled. Default value: FALSE Default source: yarn-default.xml
yarn.admin.acl	ACL of who can be admin of the YARN cluster. Default value: * Default source: yarn-default.xml
yarn.am.liveness-monitor.expiry-interval-ms	The expiry interval for application master reporting. Default value: 600000 Default source: yarn-default.xml

yarn.app.mapreduce.job.update-status-max-retries	The number of job status update retries. Default value: 0 (retried only 1 time) For a value N, the update is retried 1+N times.
yarn.app.mapreduce.job.update-status-retry-interval	The interval in milliseconds for job status update retries. Default value: 2000 (2000 millisecond or 2 seconds delay is observed before each retry attempt)
yarn.client.application-client-protocol.poll-interval-ms	The interval that the yarn client library uses to poll the completion status of the asynchronous API of application client protocol. Default value: 200 Default source: yarn-default.xml
yarn.client.failover-proxy-provider	When HA is enabled, the class to be used by Clients, AMs and NMs to failover to the Active RM. It should extend org.apache.hadoop.yarn.client.RMFailoverProxyProvider. Default value: org.apache.hadoop.yarn.client.ConfiguredRMFailoverProxyProvider Default source: yarn-default.xml When you configure failover configure.sh may change the default value by adding a value for this parameter in yarn-site.xml. For more information, see ResourceManager Configuration Properties.
yarn.client.failover-retries	When HA is enabled, the number of retries per attempt to connect to a ResourceManager. In other words, it is the ipc.client.connect.max.retries to be used during failover attempts. Default value: 0 Default source: yarn-default.xml
yarn.client.failover-retries-on-socket-timeouts	When HA is enabled, the number of retries per attempt to connect to a ResourceManager on socket timeouts. In other words, it is the ipc.client.connect.max.retries.on.timeouts to be used during failover attempts. Default value: 0 Default source: yarn-default.xml

last-updated: Apr 11, 2022

yarn.client.max-nodemanagers-proxies	Maximum number of proxy connections for node manager. It should always be more than 1. NMClient and MRAppMaster will use this to cache connection with node manager. There will be at max one connection per node manager. Ex. configuring it to a value of 5 will make sure that client will at max have 5 connections cached with 5 different node managers. These connections will be timed out if idle for more than system wide idle timeout period. The token if used for authentication then it will be used only at connection creation time. If new token is received then earlier connection should be closed in order to use newer token. This and (yarn.client.nodemanager-client-async.thread-pool-max-size) are related and should be sync (no need for them to be equal). Default value: 0 Default source: yarn-default.xml
yarn.client.nodemanager-client-async.thread-pool-max-size	Max number of threads in NMClientAsync to process container management events. Default value: 500 Default source: yarn-default.xml
yarn.client.nodemanager-connect.max-wait-ms	Default value: 900000 Default source: yarn-default.xml
yarn.client.nodemanager-connect.retry-interval-ms	Default value: 10000 Default source: yarn-default.xml
yarn.dfs-logging.dir-glob	Default value: maprfs:////var/mapr/local/*/logs/yarn/userlogs Default source: Code
yarn.dfs-logging.handler-class	Default value:com.mapr.hadoop.yarn.util.MapRFSLoggingHandler Default source: Code
yarn.external.token.manager	Default value: com.mapr.hadoop.yarn.security.MapRTicketManager Default source: Code
yarn.http.policy	Configures the HTTP endpoint for YARN Daemons.The following values are supported: -HTTP_ONLY : Service is provided only on http -HTTPS_ONLY : Service is provided only on https Default value: HTTP_ONLY Default source: yarn-default.xml
yarn.ipc.rpc.class	RPC class implementation. Default value: org.apache.hadoop.yarn.ipc.HadoopYarnProtoRPC Default source: yarn-default.xml

yarn.log-aggregation.retain-check-interval-seconds	How long to wait between aggregated log retention checks. If set to 0 or a negative value, then the value is computed as one-tenth of the aggregated log retention time. Default value: -1 Default source: yarn-default.xml See YARN Log Aggregation.
yarn.log-aggregation.retain-seconds	How long to keep aggregation logs before deleting them. -1 disables. Default value: 2592000 Default source: code See YARN Log Aggregation.
yarn.log-aggregation-enable	Whether to enable log aggregation. Default value: FALSE Default source: yarn-default.xml See YARN Log Aggregation.
yarn.mapr.ticket.expiration	Default value: 604800000 Default source: code
yarn.nm.liveness-monitor.expiry-interval-ms	How long to wait until a NodeManager is considered dead. Default value: 600000 Default source: yarn-default.xml
yarn.nodemanager.container-localizer.log.level	Configuring container localizer logs Default value: INFO Default source: code
yarn.nodemanager.address	The address of the container manager in the NM. Default value: ${yarn.nodemanager.hostname}:0 Default source: yarn-default.xml
yarn.nodemanager.admin-env	Environment variables that should be forwarded from the NodeManager's environment to the container's environment. Default value: MALLOC_ARENA_MAX=$MALLOC_ARENA_MAX Default source: yarn-default.xml
yarn.nodemanager.aux-services	The valid service name should only contain a-zA-Z0-9_ and can not start with numbers. Default value: mapreduce_shuffle,mapr_direct_shuffle Default source: code
yarn.nodemanager.aux-services.mapr_direct_shuffle.class	Default value: com.mapr.hadoop.mapred.LocalVolumeAuxService Default source: code

last-updated: Apr 11, 2022

yarn.nodemanager.aux-services.mapreduce_shuffle.class	Default value: org.apache.hadoop.mapred.ShuffleHandler Default source: yarn-default.xml
yarn.nodemanager.container-executor.class	Identifies who will execute (launch) the containers. Default value: org.apache.hadoop.yarn.server.nodemanager.LinuxContainerExecutor Default source: code
yarn.nodemanager.container-manager.thread-count	Number of threads the container manager uses. Default value: 20 Default source:yarn-default.xml
yarn.nodemanager.container-monitor.interval-ms	How often to monitor containers. Default value: 3000 Default source: yarn-default.xml
yarn.nodemanager.container-monitor.procfs-tree.smaps-based-rss.enabled	Default value: FALSE Default source: yarn-default.xml
yarn.nodemanager.delete.debug-delay-sec	Number of seconds after an application finishes before the NodeManager's DeletionService will delete the application's localized file directory and log directory. To diagnose Yarn application problems, set this property's value large enough (for example, to 600 = 10 minutes) to permit examination of these directories. After changing the property's value, you must restart the NodeManager in order for it to have an effect. The roots of Yarn applications' work directories are configurable with the yarn.nodemanager.local-dirs property (see below), and the roots of the Yarn applications' log directories is configurable with the yarn.nodemanager.log-dirs property (see also below). Default value: 0 Default source: yarn-default.xml
yarn.nodemanager.delete.thread-count	Number of threads used in cleanup. Default value: 4 Default source: yarn-default.xml
yarn.nodemanager.disk-health-checker.interval-ms	Frequency of running disk health checker code. Default value: 1200000 Default source: yarn-default.xml
yarn.nodemanager.disk-health-checker.max-disk-utilization-per-disk-percentage	Default value: 90 Default source: yarn-default.xml
yarn.nodemanager.disk-health-checker.min-free-space-per-disk-mb	Default value: 0 Default source: yarn-default.xml

yarn.nodemanager.disk-health-checker.min-healthy-disks	The minimum fraction of number of disks to be healthy for the nodemanager to launch new containers. This correspond to both yarn-nodemanager.local-dirs and yarn.nodemanager.log-dirs. i.e. If there are less number of healthy local-dirs (or log-dirs) available, then new containers will not be launched on this node. Default value: 0.25 Default source: yarn-default.xml
yarn.nodemanager.docker-container-executor.exec-name	Default value: /usr/bin/docker Default source: yarn-default.xml
yarn.nodemanager.env-whitelist	Environment variables that containers may override rather than use NodeManager's default. Default value: JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,HADOOP_YARN_HOME Default source: yarn-default.xml
yarn.nodemanager.external.token.localizer	Default value: com.mapr.hadoop.yarn.nodemanager.MapRTicketLocalizer Default source: code
yarn.nodemanager.health-checker.interval-ms	Frequency of running node health script. Default value: 600000 Default source: yarn-default.xml
yarn.nodemanager.health-checker.script.timeout-ms	Script time out period. Default value: 1200000 Default source: yarn-default.xml
yarn.nodemanager.hostname	The hostname of the NM. Default value: 0.0.0.0 Default source: yarn-default.xml
yarn.nodemanager.keytab	Keytab for NM. Default value: /etc/krb5.keytab Default source: yarn-default.xml
yarn.nodemanager.linux-container-executor.cgroups.hierarchy	The cgroups hierarchy under which to place YARN proccesses (cannot contain commas). If yarn.nodemanager.linux-container-executor.cgroups.mount is false (that is, if cgroups have been pre-configured), then this cgroups hierarchy must already exist and be writable by the NodeManager user, otherwise the NodeManager may fail. Only used when the LCE resources handler is set to the CgroupsLCEResourcesHandler. Default value: /hadoop-yarn Default source: yarn-default.xml

yarn.nodemanager.linux-container-executor.cgroups.mount	Whether the LCE should attempt to mount cgroups if not found. Only used when the LCE resources handler is set to the CgroupsLCEResourcesHandler. Default value: false Default source: yarn-default.xml
yarn.nodemanager.linux-container-executor.cgroups.strict-resource-usage	Default value: FALSE Default source: yarn-default.xml
yarn.nodemanager.linux-container-executor.nonsecure-mode.limit-users	Default value: TRUE Default source: yarn-default.xml
yarn.nodemanager.linux-container-executor.nonsecure-mode.local-user	Default value: nobody Default source: yarn-default.xml
yarn.nodemanager.linux-container-executor.nonsecure-mode.user-pattern	The allowed pattern for UNIX user names enforced by Linux-container-executor when used in nonsecure mode (use case for this is using cgroups). The default value is taken from /usr/sbin/adduser. Default value: ^[_.A-Za-z0-9][-@_.A-Za-z0-9]{0,255}?[$]?$ Default source: yarn-default.xml
yarn.nodemanager.linux-container-executor.resources-handler.class	The class which should help the LCE handle resources. Default value: org.apache.hadoop.yarn.server.nodemanager.util.DefaultLCEResourcesHandler Default source: yarn-default.xml
yarn.nodemanager.local-cache.max-files-per-directory	It limits the maximum number of files which will be localized in a single local directory. If the limit is reached then sub-directories will be created and new files will be localized in them. If it is set to a value less than or equal to 36 [which are sub-directories (0-9 and then a-z)] then NodeManager will fail to start. For example; [for public cache] if this is configured with a value of 40 (4 files + 36 sub-directories) and the local-dir is /tmp/local-dir1, then it will allow 4 files to be created directly inside /tmp/local-dir1/filecache. For files that are localized further, it will create a sub-directory "0" inside /tmp/local-dir1/filecache and will localize files inside it until it becomes full. If a file is removed from a sub-directory that is marked full, then that sub-directory will be used back again to localize files. Default value: 8192 Default source: yarn-default.xml
yarn.nodemanager.local-dirs	List of directories to store localized files in. An application's localized file directory will be found in: ${yarn.nodemanager.local-dirs}/usercache/${user}/appcache/application_${appid}. Individual containers' work directories, called container_${contid}, will be subdirectories of this. Default value: ${hadoop.tmp.dir}/nm-local-dir Default source: yarn-default.xml

yarn.nodemanager.localizer.address	Address where the localizer IPC is. Default value: ${yarn.nodemanager.hostname}:8040
yarn.nodemanager.localizer.cache.cleanup.interval-ms	Interval in between cache cleanups. Default value: 60000 Default source: yarn-default.xml
yarn.nodemanager.localizer.cache.target-size-mb	Target size of localizer cache in MB, per local directory. Default value: 10240 Default source: yarn-default.xml
yarn.nodemanager.localizer.client.thread-count	Number of threads to handle localization requests. Default value: 5 Default source: yarn-default.xml
yarn.nodemanager.localizer.fetch.thread-count	Number of threads to use for localization fetching. Default value: 4 Default source: yarn-default.xml
yarn.nodemanager.log.retain-seconds	Time in seconds to retain user logs. Only applicable if log aggregation is disabled. Default value: 10800 Default source: yarn-default.xml
yarn.nodemanager.log-aggregation.compression-type	T-file compression types used to compress aggregated logs. Default value: none Default source: yarn-default.xml
yarn.nodemanager.log-aggregation.roll-monitoring-interval-seconds	Default value: -1 Default source: yarn-default.xml
yarn.nodemanager.log-dirs	Where to store container logs. An application's localized log directory will be found in ${yarn.nodemanager.log-dirs}/application_${appid}. Individual containers' log directories will be below this, in directories named container_{$contid}. Each container directory will contain the files stderr, stdin, and syslog generated by that container. Default value: ${yarn.log.dir}/userlogs Default source: yarn-default.xml
yarn.nodemanager.pmem-check-enabled	Whether physical memory limits will be enforced for containers. Default value: true Default source: yarn-default.xml
yarn.nodemanager.process-kill-wait.ms	Max time to wait for a process to come up when trying to cleanup a container. Default value: 2000 Default source: yarn-default.xml

yarn.nodemanager.recovery.enabled	Default value: TRUE Default source: yarn-default.xml
yarn.nodemanager.remote-app-log-dir	Where to aggregate logs to. Default value: /tmp/logs Default source: yarn-default.xml
yarn.nodemanager.remote-app-log-dir-suffix	The remote log dir will be created at {yarn.nodemanager.remote-app-log-dir}/${user}/{thisParam} Default value: logs Default source: yarn-default.xml
yarn.nodemanager.resource.cpu-vcores	Number of CPU cores that can be allocated for containers. Default value: ${nodemanager.resource.cpu-vcores} Default source: code
yarn.nodemanager.resource.io-spindles	Default value: ${nodemanager.resource.io-spindles} Default source: code
yarn.nodemanager.resource.memory-mb	Amount of physical memory, in MB, that can be allocated for containers. Default value: ${nodemanager.resource.memory-mb} Default source: code
yarn.nodemanager.resource.percentage-physical-cpu-limit	Default value: 100 Default source: yarn-default.xml
yarn.nodemanager.resourcemanager.minimum.version	The minimum allowed version of a resourcemanager that a nodemanager will connect to. The valid values are NONE (no version checking), EqualToNM (the resourcemanager's version is equal to or greater than the NM version), or a Version String. Default value: NONE Default source: yarn-default.xml
yarn.nodemanager.sleep-delay-before-sigkill.ms	Number of ms to wait between sending a SIGTERM and SIGKILL to a container. Default value: 250 Default source: yarn-default.xml
yarn.nodemanager.vmem-check-enabled	Whether virtual memory limits will be enforced for containers. Default value: false Default source: yarn-default.xml

yarn.nodemanager.vmem-pmem-ratio	Ratio between virtual memory to physical memory when setting memory limits for containers. Container allocations are expressed in terms of physical memory, and virtual memory usage is allowed to exceed this allocation by this ratio. Default value: 2.1 Default source: yarn-default.xml
yarn.nodemanager.webapp.address	NM Webapp address. Default value: ${yarn.nodemanager.hostname}:8042 Default source: yarn-default.xml
yarn.nodemanager.windows-container.cpu-limit.enabled	Default value: FALSE Default source: yarn-default.xml
yarn.nodemanager.windows-container.memory-limit.enabled	Default value: FALSE Default source: yarn-default.xml
yarn.resourcemanager.address	The address of the applications manager interface in the ResourceManager. Default value: ${yarn.resourcemanager.hostname}:8032 Default source: yarn-default.xml When you configure failover configure.sh may change the default value by adding a value for this parameter in yarn-site.xml. For more information, see ResourceManager Configuration Properties on page 1518.
yarn.resourcemanager.admin.address	The address of the ResourceManager admin interface. Default value: ${yarn.resourcemanager.hostname}:8033 Default source: yarn-default.xml When you configure failover configure.sh may change the default value by adding a value for this parameter in yarn-site.xml. For more information, see ResourceManager Configuration Properties on page 1518.
yarn.resourcemanager.admin.client.thread-count	Number of threads used to handle the ResourceManager admin interface. Default value: 1 Default source: yarn-default.xml
yarn.resourcemanager.am.max-attempts	The maximum number of application attempts. This is a global setting for all ApplicationMasters. Each ApplicationMaster can specify its individual maximum number of application attempts via the API, but the individual number cannot be more than the global upper bound. If it is, the ResourceManager will override it. The default number is set to 2, to allow at least one retry for the ApplicationMaster. Default value: 2 Default source: yarn-default.xml

last-updated: Apr 11, 2022

yarn.resourcemanager.application-tokens.master-key-rolling-interval-secs	Interval for the roll over for the master key used to generate application tokens. Default value: 86400 Default source: yarn-default.xml
yarn.resourcemanager.aux-services	Default value: RMVolumeManager Default source: code
yarn.resourcemanager.aux-services.HSVolumeManager.class	Default value: com.mapr.hadoop.yarn.resourcemanager.RMVolumeManager Default source: code
yarn.resourcemanager.aux-services.RMVolumeManager.class	Default value: com.mapr.hadoop.yarn.resourcemanager.RMVolumeManager Default source: code
yarn.resourcemanager.client.thread-count	The number of threads used to handle applications manager requests. Default value: 50 Default source: yarn-default.xml
yarn.resourcemanager.configuration.provider-class	Default value: org.apache.hadoop.yarn.LocalConfigurationProvider Default source: yarn-default.xml
yarn.resourcemanager.connect.max-wait.ms	Maximum time to wait to establish connection to the ResourceManager. Default value: 900000 Default source: yarn-default.xml
yarn.resourcemanager.connect.retry-interval.ms	How often to retry connecting to the ResourceManager. Default value: 30000 Default source: yarn-default.xml
yarn.resourcemanager.container.liveness-monitor.interval-ms	How often to check that containers are still alive. Default value: 600000 Default source: yarn-default.xml
yarn.resourcemanager.container-tokens.master-key-rolling-interval-secs	Interval for the roll over for the master key used to generate container tokens. It is expected to be much greater than yarn.nm.liveness-monitor.expiry-interval-ms and yarn.rm.container-allocation.expiry-interval-ms. Otherwise the behavior is undefined. Default value: 86400 Default source: yarn-default.xml
yarn.resourcemanager.delayed.delegation-token.removal-interval-ms	Interval at which the delayed token removal thread runs. Default value: 30000 Default source: yarn-default.xml

yarn.resourcemanager.dir	Default value: /var/mapr/cluster/yarn/rm Default source: code
yarn.resourcemanager.fs.state-store.num-retries	Default value: 0 Default source: yarn-default.xml
yarn.resourcemanager.fs.state-store.retry-interval-ms	Default value: 1000 Default source: yarn-default.xml
yarn.resourcemanager.fs.state-store.retry-policy-spec	hdfs client retry policy specification. hdfs client retry is always enabled. Specified in pairs of sleep-time and number-of-retries and (t0, n0), (t1, n1), ..., the first n0 retries sleep t0 milliseconds on average, the following n1 retries sleep t1 milliseconds on average, and so on. Default value: 2000, 500 Default source: yarn-default.xml
yarn.resourcemanager.fs.state-store.uri	URI pointing to the location of the FileSystem path where RM state will be stored. This must be supplied when using org.apache.hadoop.yarn.server.resourcemanager.recovery.FileSystemRMStateStore as the value for yarn.resourcemanager.store.class Default value: /var/mapr/cluster/yarn/rm/system Default source: code
yarn.resourcemanager.ha.automatic-failover.embedded	Enable embedded automatic failover. The embedded elector relies on the RM state store to handle fencing, and is primarily intended to be used in conjunction with ZKRMStateStore. Default value: TRUE Default source: yarn-default.xml
yarn.resourcemanager.ha.automatic-failover.enabled	Enable automatic failover. Default value: TRUE Default source: yarn-default.xml
yarn.resourcemanager.ha.automatic-failover.zk-base-path	The base znode path to use for storing leader information, when using ZooKeeper based leader election. Default value: /yarn-leader-election Default source: yarn-default.xml
yarn.resourcemanager.ha.custom-ha-rmaddressfinder	Default value: org.apache.hadoop.yarn.client.MapRZKBasedRMAddressFinder Default source: code

last-updated: Apr 11, 2022

yarn.resourcemanager.ha.enabled	Enable RM high-availability. When enabled, (1) The RM starts in the Standby mode by default, and transitions to the Active mode when prompted to. (2) The nodes in the RM ensemble are listed in yarn.resourcemanager.ha.rm-ids (3) The id of each RM comes from yarn.resourcemanager.ha.id (4) The actual physical addresses come from the configs of the pattern - {rpc-config}.{id} Default value: false Default source: yarn-default.xml
yarn.resourcemanager.hostname	The hostname of the ResourceManager. Default value: 0.0.0.0 Default source: yarn-default.xml
yarn.resourcemanager.keytab	The keytab for the ResourceManager. Default value: /etc/krb5.keytab Default source: yarn-default.xml
yarn.resourcemanager.leveldb-state-store.path	Default value: ${hadoop.tmp.dir}/yarn/system/rmstore Default source: yarn-default.xml
yarn.resourcemanager.max-completed-applications	The maximum number of completed applications RM keeps. Default value: 10000 Default source: yarn-default.xml
yarn.resourcemanager.nodemanager.minimum.version	The minimum allowed version of a connecting nodemanager. The valid values are NONE (no version checking), EqualToRM (the nodemanager's version is equal to or greater than the RM version), or a Version String. Default value: NONE Default source: yarn-default.xml
yarn.resourcemanager.nodemanagers.heartbeat-interval-ms	The heart-beat interval in milliseconds for every NodeManager in the cluster. Default value: 1000 Default source: yarn-default.xml
yarn.resourcemanager.principal	The Kerberos principal for the ResourceManager. Default value:mapr Default source: code
yarn.resourcemanager.proxy-user-privileges.enabled	Default value: FALSE Default source: yarn-default.xml

yarn.resourcemanager.recovery.enabled	Enable RM to recover state after starting. If true, then yarn.resourcemanager.store.class must be specified. Default value: false When you configure failover configure.sh may change the default value by adding a value for this parameter in yarn-site.xml. For more information, see ResourceManager Configuration Properties on page 1518.
yarn.resourcemanager.resource-tracker.address	Default value: ${yarn.resourcemanager.hostname}:8031 Default source: yarn-default.xml When you configure failover configure.sh may change the default value by adding a value for this parameter in yarn-site.xml. For more information, see ResourceManager Configuration Properties on page 1518.
yarn.resourcemanager.resource-tracker.client.thread-count	Number of threads to handle resource tracker calls. Default value: 50 Default source: yarn-default.xml
yarn.resourcemanager.scheduler.address	The address of the scheduler interface. Default value: ${yarn.resourcemanager.hostname}:8030 Default source: yarn-default.xml When you configure failover configure.sh may change the default value by adding a value for this parameter in yarn-site.xml. For more information, see ResourceManager Configuration Properties on page 1518.
yarn.resourcemanager.scheduler.class	The class to use as the resource scheduler. Default value: org.apache.hadoop.yarn.server.resourcemanager.scheduler.fair.FairScheduler Default source: code
yarn.resourcemanager.scheduler.client.thread-count	Number of threads to handle scheduler interface. Default value: 50 Default source: yarn-default.xml
yarn.resourcemanager.scheduler.monitor.enable	Enable a set of periodic monitors (specified in yarn.resourcemanager.scheduler.monitor.policies) that affect the scheduler. Default value: false Default source: yarn-default.xml
yarn.resourcemanager.scheduler.monitor.policies	The list of SchedulingEditPolicy classes that interact with the scheduler. A particular module may be incompatible with the scheduler, other policies, or a configuration of either. Default value: org.apache.hadoop.yarn.server.resourcemanager.monitor.capacity.ProportionalCapacityPreemptionPolicy Default source: yarn-default.xml

last-updated: Apr 11, 2022

yarn.resourcemanager.staging	Default value: /var/mapr/cluster/yarn/rm/staging Default source: code
yarn.resourcemanager.state-store.max-completed-applications	The maximum number of completed applications RM state store keeps, less than or equals to ${yarn.resourcemanager.max-completed-applications}. By default, it equals to ${yarn.resourcemanager.max-completed-applications}. This ensures that the applications kept in the state store are consistent with the applications remembered in RM memory. Any values larger than ${yarn.resourcemanager.max-completed-applications} will be reset to ${yarn.resourcemanager.max-completed-applications}. Note that this value impacts the RM recovery performance.Typically, a smaller value indicates better performance on RM recovery. Default value: ${yarn.resourcemanager.max-completed-applications} Default source: yarn-default.xml
yarn.resourcemanager.store.class	The class to use as the persistent store. If org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore is used, the store is implicitly fenced; meaning a single ResourceManager is able to use the store at any point in time. More details on this implicit fencing, along with setting up appropriate ACLs is discussed under yarn.resourcemanager.zk-state-store.root-node.acl. Default value: org.apache.hadoop.yarn.server.resourcemanager.recovery.FileSystemRMStateStore Default source: yarn-default.xml
yarn.resourcemanager.system	Default value: /var/mapr/cluster/yarn/rm/system Default source: code
yarn.resourcemanager.system-metrics-publisher.dispatcher.pool-size	Default value: 10 Default source: yarn-default.xml
yarn.resourcemanager.system-metrics-publisher.enabled	Default value: FALSE Default source: yarn-default.xml
yarn.resourcemanager.webapp.address	The http address of the ResourceManager web application. Default value: ${yarn.resourcemanager.hostname}:8088 Default source: yarn-default.xml When you configure failover configure.sh may change the default value by adding a value for this parameter in yarn-site.xml. For more information, see ResourceManager Configuration Properties on page 1518.
yarn.resourcemanager.webapp.delegation-token-auth-filter.enabled	Default value: TRUE Default source: yarn-default.xml

yarn.resourcemanager.webapp.https.address	The https adddress of the ResourceManager web application. Default value: ${yarn.resourcemanager.hostname}:8090 Default source: yarn-default.xml When you configure failover configure.sh may change the default value by adding a value for this parameter in yarn-site.xml. For more information, see ResourceManager Configuration Properties on page 1518.
yarn.resourcemanager.work-preserving-recovery.enabled	Default value: TRUE Default source: yarn-default.xml
yarn.resourcemanager.work-preserving-recovery.scheduling-wait-ms	Default value: 10000 Default source: yarn-default.xml
yarn.resourcemanager.zk-acl	ACL's to be used for ZooKeeper znodes. Default value: world:anyone:rwcda Default source: yarn-default.xml
yarn.resourcemanager.zk-num-retries	Number of times RM tries to connect to ZooKeeper. Default value: 1000 Default source: yarn-default.xml
yarn.resourcemanager.zk-retry-interval-ms	Retry interval in milliseconds when connecting to ZooKeeper. Default value: 1000 Default source: yarn-default.xml
yarn.resourcemanager.zk-state-store.parent-path	Full path of the ZooKeeper znode where RM state will be stored. This must be supplied when using org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore as the value for yarn.resourcemanager.store.class Default value: /rmstore Default source: yarn-default.xml
yarn.resourcemanager.zk-timeout-ms	ZooKeeper session timeout in milliseconds. Session expiration is managed by the ZooKeeper cluster itself, not by the client. This value is used by the cluster to determine when the client's session expires. Expirations happens when the cluster does not hear from the client within the specified session timeout period (i.e. no heartbeat). Default value: 10000 Default source: yarn-default.xml
yarn.scheduler.maximum-allocation-mb	The maximum allocation for every container request at the RM, in MBs. Memory requests higher than this won't take effect, and will get capped to this value. Default value: 8192 Default source: yarn-default.xml

last-updated: Apr 11, 2022

yarn.scheduler.maximum-allocation-vcores	The maximum allocation for every container request at the RM, in terms of virtual CPU cores. Requests higher than this won't take effect, and will get capped to this value. Default value: 4 Default source: yarn-default.xml
yarn.scheduler.minimum-allocation-mb	The minimum allocation for every container request at the RM, in MBs. Memory requests lower than this won't take effect, and the specified value will get allocated at minimum. Default value: 1024 Default source: yarn-default.xml
yarn.scheduler.minimum-allocation-vcores	The minimum allocation for every container request at the RM, in terms of virtual CPU cores. Requests lower than this won't take effect, and the specified value will get allocated the minimum. Default value: 1 Default source: yarn-default.xml
yarn.resourcemanager.zk-timeout-ms	Default value: 10000 Default source: yarn-default.xml
yarn.scheduler.minimum-allocation-mb	Default value: 1024 Default source: yarn-default.xml
yarn.scheduler.minimum-allocation-vcores	Default value: 1 Default source: yarn-default.xml
yarn.sharedcache.admin.address	Default value: 0.0.0.0:8047 Default source: yarn-default.xml
yarn.sharedcache.admin.thread-count	Default value: 1 Default source: yarn-default.xml
yarn.sharedcache.app-checker.class	Default value: org.apache.hadoop.yarn.server.sharedcachemanager.RemoteAppChecker Default source: yarn-default.xml
yarn.sharedcache.checksum.algo.impl	Default value: org.apache.hadoop.yarn.sharedcache.ChecksumSHA256Impl Default source: yarn-default.xml
yarn.sharedcache.cleaner.initial-delay-mins	Default value: 10 Default source: yarn-default.xml
yarn.sharedcache.cleaner.period-mins	Default value: 1440 Default source: yarn-default.xml
yarn.sharedcache.cleaner.resource-sleep-ms	Default value: 0 Default source: yarn-default.xml

yarn.sharedcache.client-server.address	Default value: 0.0.0.0:8045 Default source: yarn-default.xml
yarn.sharedcache.client-server.thread-count	Default value: 50 Default source: yarn-default.xml
yarn.sharedcache.enabled	Default value: FALSE Default source: yarn-default.xml
yarn.sharedcache.nested-level	Default value: 3 Default source: yarn-default.xml
yarn.sharedcache.nm.uploader.replication.factor	Default value: 10 Default source: yarn-default.xml
yarn.sharedcache.nm.uploader.thread-count	Default value: 20 Default source: yarn-default.xml
yarn.sharedcache.root-dir	Default value: /sharedcache Default source: yarn-default.xml
yarn.sharedcache.store.class	Default value: org.apache.hadoop.yarn.server.sharedcachemanager.store.InMemorySCMStore Default source: yarn-default.xml
yarn.sharedcache.store.in-memory.check-period-mins	Default value: 720 Default source: yarn-default.xml
yarn.sharedcache.store.in-memory.initial-delay-mins	Default value: 10 Default source: yarn-default.xml
yarn.sharedcache.store.in-memory.staleness-period-mins	Default value: 10080 Default source: yarn-default.xml
yarn.sharedcache.uploader.server.address	Default value: 0.0.0.0:8046 Default source: yarn-default.xml
yarn.sharedcache.uploader.server.thread-count	Default value: 50 Default source: yarn-default.xml
yarn.sharedcache.webapp.address	Default value: 0.0.0.0:8788 Default source: yarn-default.xml
yarn.timeline-service.address	Default value: ${yarn.timeline-service.hostname}:10200 Default source: yarn-default.xml
yarn.timeline-service.client.max-retries	Default value: 30 Default source: yarn-default.xml

last-updated: Apr 11, 2022

yarn.timeline-service.client.best-effort	Default value: FALSE Default source: yarn-default.xml To enable an application to run successfully after it is retried, set this to TRUE
yarn.timeline-service.client.retry-interval-ms	Default value: 1000 Default source: yarn-default.xml
yarn.timeline-service.enabled	Default value: FALSE Default source: yarn-default.xml
yarn.timeline-service.generic-application-history.aux-services	Default value: HSVolumeManager Default source: code
yarn.timeline-service.handler-thread-count	Default value: 10 Default source: yarn-default.xml
yarn.timeline-service.hostname	Default value: 0.0.0.0 Default source: yarn-default.xml
yarn.timeline-service.http-authentication.simple.anonymous.allowed	Default value: TRUE Default source: yarn-default.xml
yarn.timeline-service.http-authentication.type	Default value: simple Default source: yarn-default.xml
yarn.timeline-service.keytab	Default value: /etc/krb5.keytab Default source: yarn-default.xml
yarn.timeline-service.leveldb-state-store.path	Default value: ${hadoop.tmp.dir}/yarn/timeline Default source: yarn-default.xml
yarn.timeline-service.leveldb-timeline-store.path	Default value: ${hadoop.tmp.dir}/yarn/timeline Default source: yarn-default.xml
yarn.timeline-service.leveldb-timeline-store.read-cache-size	Default value: 104857600 Default source: yarn-default.xml
yarn.timeline-service.leveldb-timeline-store.start-time-read-cache-size	Default value: 10000 Default source: yarn-default.xml
yarn.timeline-service.leveldb-timeline-store.start-time-write-cache-size	Default value: 10000 Default source: yarn-default.xml
yarn.timeline-service.leveldb-timeline-store.ttl-interval-ms	Default value: 300000 Default source: yarn-default.xml
yarn.timeline-service.recovery.enabled	Default value: FALSE Default source: yarn-default.xml

yarn.timeline-service.state-store-class	Default value: org.apache.hadoop.yarn.server.timeline.recovery.LeveldbTimelineStateStore Default source: yarn-default.xml
yarn.timeline-service.store-class	Default value: org.apache.hadoop.yarn.server.timeline.LeveldbTimelineStore Default source: yarn-default.xml
yarn.timeline-service.ttl-enable	Default value: TRUE Default source: yarn-default.xml
yarn.timeline-service.ttl-ms	Default value: 604800000 Default source: yarn-default.xml
yarn.timeline-service.webapp.address	Default value: ${yarn.timeline-service.hostname}:8188 Default source: yarn-default.xml
yarn.timeline-service.webapp.https.address	Default value: ${yarn.timeline-service.hostname}:8190 Default source: yarn-default.xml
yarn.timeline-service.webapp.all-ifaces	Redirects all opening container logs from timeline server to 0.0.0.0. Default value: TRUE Default source: code
yarn.use-central-logging-for-mapreduce-only	Default value: FALSE Default source: code

Default mapred Parameters

Property	Description
io.sort.record.percent	Default value: 0.17 Default source: code
map.sort.class	Default value: org.apache.hadoop.util.QuickSort Default source: mapred-default.xml
mapr.localoutput.dir	Default value: output Default source: code
mapr.localspill.dir	Default value: spill Default source: code
mapr.map.keyprefix.ints	Default value: 1 Default source: code
mapr.mapred.localvolume.root.dir.name	Default value: nodeManager Default source: code
mapreduce.am.max-attempts	Default value: 2 Default source: mapred-default.xml

last-updated: Apr 11, 2022

mapreduce.app-submission.cross-platform	Default value: FALSE Default source: mapred-default.xml
mapreduce.client.completion.pollinterval	Default value: 5000 Default source: mapred-default.xml
mapreduce.client.output.filter	Default value: FAILED Default source: mapred-default.xml
mapreduce.client.progressmonitor.pollinterval	Default value: 1000 Default source: mapred-default.xml
mapreduce.client.submit.file.replication	Default value: 10 Default source: mapred-default.xml
mapreduce.cluster.acls.enabled	Default value: FALSE Default source: mapred-default.xml
mapreduce.cluster.local.dir	Default value: ${hadoop.tmp.dir}/mapred/local Default source: mapred-default.xml
mapreduce.cluster.temp.dir	Default value: ${hadoop.tmp.dir}/mapred/temp Default source: mapred-default.xml
mapreduce.fileoutputcommitter.algorithm.version	Default value: 1 Default source: mapred-default.xml
mapreduce.framework.name	Default value: yarn Default source: code
mapreduce.ifile.readahead	Default value: TRUE Default source: mapred-default.xml
mapreduce.ifile.readahead.bytes	Default value: 4194304 Default source: mapred-default.xml
mapreduce.input.fileinputformat.list-status.num-threads	Default value: 1 Default source: mapred-default.xml
mapreduce.input.fileinputformat.split.minsize	Default value: 0 Default source: mapred-default.xml
mapreduce.input.lineinputformat.linespermap	Default value: 1 Default source: mapred-default.xml
mapreduce.job.acl-modify-job	Default value: Default source: mapred-default.xml
mapreduce.job.acl-view-job	Default value: Default source: mapred-default.xml

last-updated: Apr 11, 2022

mapreduce.job.classloader	Default value: FALSE Default source: mapred-default.xml
mapreduce.job.committer.setup.cleanup.needed	Default value: TRUE Default source: mapred-default.xml
mapreduce.job.complete.cancel.delegation.tokens	Default value: TRUE Default source: mapred-default.xml
mapreduce.job.counters.max	Default value: 120 Default source: mapred-default.xml
mapreduce.job.emit-timeline-data	Default value: FALSE Default source: mapred-default.xml
mapreduce.job.end-notification.max.attempts	Default value: 5 Default source: mapred-default.xml
mapreduce.job.end-notification.max.retry.interval	Default value: 5000 Default source: mapred-default.xml
mapreduce.job.end-notification.retry.attempts	Default value: 0 Default source: mapred-default.xml
mapreduce.job.end-notification.retry.interval	Default value: 1000 Default source: mapred-default.xml
mapreduce.job.jvm.numtasks	Default value: 1 Default source: mapred-default.xml
mapreduce.job.map.output.collector.class	Default value: org.apache.hadoop.mapred.MapRFsOutputBuffer Default source: code
mapreduce.job.maps	Default value: 2 Default source: mapred-default.xml
mapreduce.job.max.split.locations	Default value: 10 Default source: mapred-default.xml
mapreduce.job.maxtaskfailures.per.tracker	Default value: 3 Default source: mapred-default.xml
mapreduce.job.queuename	Default value: default Default source: mapred-default.xml
mapreduce.job.reduce.shuffle.consumer.plugin.class	Default value: org.apache.hadoop.mapreduce.task.reduce.DirectShuffle Default source: code

last-updated: Apr 11, 2022

mapreduce.job.reduce.slowstart.completedmaps	Default value: 1.00 Default source: code
mapreduce.job.reducer.preempt.delay.sec	Default value: 0 Default source: mapred-default.xml
mapreduce.job.reduces	Default value: 1 Default source: mapred-default.xml
mapreduce.job.running.map.limit	Default value: 0 Default source: mapred-default.xml
mapreduce.job.running.reduce.limit	Default value: 0 Default source: mapred-default.xml
mapreduce.job.shuffle.provider.services	Default value: mapr_direct_shuffle Default source: code
mapreduce.job.speculative.minimum-allowed-tasks	Default value: 10 Default source: mapred-default.xml
mapreduce.job.speculative.retry-after-no-speculate	Default value: 1000 Default source: mapred-default.xml
mapreduce.job.speculative.retry-after-speculate	Default value: 15000 Default source: mapred-default.xml
mapreduce.job.speculative.slowtaskthreshold	Default value: 1 Default source: mapred-default.xml
mapreduce.job.speculative.speculative-cap-running-tasks	Default value: 0.1 Default source: mapred-default.xml
mapreduce.job.speculative.speculative-cap-total-tasks	Default value: 0.01 Default source: mapred-default.xml
mapreduce.job.split.metainfo.maxsize	Default value: 10000000 Default source: mapred-default.xml
mapreduce.job.token.tracking.ids.enabled	Default value: FALSE Default source: mapred-default.xml
mapreduce.job.ubertask.enable	Default value: FALSE Default source: mapred-default.xml
mapreduce.job.ubertask.maxmaps	Default value: 9 Default source: mapred-default.xml
mapreduce.job.ubertask.maxreduces	Default value: 1 Default source: mapred-default.xml

last-updated: Apr 11, 2022

mapreduce.job.userlog.retain.hours	Default value: 24 Default source: mapred-default.xml
mapreduce.jobhistory.address	Default value: 0.0.0.0:10020 Default source: mapred-default.xml
mapreduce.jobhistory.admin.acl	Default value: * Default source: mapred-default.xml
mapreduce.jobhistory.admin.address	Default value: 0.0.0.0:10033 Default source: mapred-default.xml
mapreduce.jobhistory.cleaner.enable	Default value: TRUE Default source: mapred-default.xml
mapreduce.jobhistory.cleaner.interval-ms	Default value: 86400000 Default source: mapred-default.xml
mapreduce.jobhistory.client.thread-count	Default value: 10 Default source: mapred-default.xml
mapreduce.jobhistory.datestring.cache.size	Default value: 200000 Default source: mapred-default.xml
mapreduce.jobhistory.done-dir	Default value: ${yarn.app.mapreduce.am.staging-dir}/history/done Default source: mapred-default.xml
mapreduce.jobhistory.http.policy	Default value: HTTP_ONLY Default source: mapred-default.xml
mapreduce.jobhistory.intermediate-done-dir	Default value: ${yarn.app.mapreduce.am.staging-dir}/history/done_intermediate Default source: mapred-default.xml
mapreduce.jobhistory.joblist.cache.size	Default value: 20000 Default source: mapred-default.xml
mapreduce.jobhistory.keytab	Default value: /etc/security/keytab/jhs.service.keytab Default source: mapred-default.xml
mapreduce.jobhistory.loadedjobs.cache.size	Default value: 5 Default source: mapred-default.xml
mapreduce.jobhistory.max-age-ms	Default value: 604800000 Default source: mapred-default.xml
mapreduce.jobhistory.minicluster.fixed.ports	Default value: FALSE Default source: mapred-default.xml

last-updated: Apr 11, 2022

mapreduce.jobhistory.move.interval-ms	Default value: 180000 Default source: mapred-default.xml
mapreduce.jobhistory.move.thread-count	Default value: 3 Default source: mapred-default.xml
mapreduce.jobhistory.principal	Default value: jhs/_HOST@REALM.TLD Default source: mapred-default.xml
mapreduce.jobhistory.recovery.enable	Default value: FALSE Default source: mapred-default.xml
mapreduce.jobhistory.recovery.store.class	Default value: org.apache.hadoop.mapreduce.v2.hs.HistoryServerFileSystemStateStoreService Default source: mapred-default.xml
mapreduce.jobhistory.recovery.store.fs.uri	Default value: ${hadoop.tmp.dir}/mapred/history/recoverystore Default source: mapred-default.xml
mapreduce.jobhistory.recovery.store.leveldb.path	Default value: ${hadoop.tmp.dir}/mapred/history/recoverystore Default source: mapred-default.xml
mapreduce.jobhistory.webapp.address	Default value: 0.0.0.0:19888 Default source: mapred-default.xml
mapreduce.local.clientfactory.class.name	Default value: org.apache.hadoop.mapred.LocalClientFactory Default source: mapred-default.xml
mapreduce.map.cpu.vcores	Default value: 1 Default source: mapred-default.xml
mapreduce.map.disk	Default value: 0.5 Default source: code
mapreduce.map.java.opts	Default value: -Xmx200m Default source: code
mapreduce.map.log.level	Default value: INFO Default source: mapred-default.xml
mapreduce.map.maxattempts	Default value: 4 Default source: mapred-default.xml
mapreduce.map.memory.mb	Default value: 1024 Default source: mapred-default.xml
mapreduce.map.output.compress	Default value: FALSE Default source: mapred-default.xml

mapreduce.map.output.compress.codec	Default value: org.apache.hadoop.io.compress.DefaultCodec Default source: mapred-default.xml
mapreduce.map.skip.maxrecords	Default value: 0 Default source: mapred-default.xml
mapreduce.map.skip.proc.count.autoincr	Default value: TRUE Default source: mapred-default.xml
mapreduce.map.sort.spill.percent	Default value: 0.99 Default source: code
mapreduce.map.speculative	Default value: TRUE Default source: mapred-default.xml
mapreduce.output.fileoutputformat.compress	Default value: FALSE Default source: mapred-default.xml
mapreduce.output.fileoutputformat.compress.codec	Default value: org.apache.hadoop.io.compress.DefaultCodec Default source: mapred-default.xml
mapreduce.output.fileoutputformat.compress.type	Default value: RECORD Default source: mapred-default.xml
mapreduce.reduce.cpu.vcores	Default value: 1 Default source: mapred-default.xml
mapreduce.reduce.disk	Default value: 1.33 Default source: code
mapreduce.reduce.input.buffer.percent	Default value: 0 Default source: mapred-default.xml
mapreduce.reduce.java.opts	Default value: -Xmx2560m Default source: code
mapreduce.reduce.log.level	Default value: INFO Default source: mapred-default.xml
mapreduce.reduce.markreset.buffer.percent	Default value: 0 Default source: mapred-default.xml
mapreduce.reduce.maxattempts	Default value: 4 Default source: mapred-default.xml
mapreduce.reduce.memory.mb	Default value: 3072 Default source: code

last-updated: Apr 11, 2022

mapreduce.reduce.merge.inmem.threshold	Default value: 1000 Default source: mapred-default.xml
mapreduce.reduce.shuffle.connect.timeout	Default value: 180000 Default source: mapred-default.xml
mapreduce.reduce.shuffle.fetch.retry.enabled	Default value: ${yarn.nodemanager.recovery.enabled} Default source: mapred-default.xml
mapreduce.reduce.shuffle.fetch.retry.interval-ms	Default value: 1000 Default source: mapred-default.xml
mapreduce.reduce.shuffle.fetch.retry.timeout-ms	Default value: 30000 Default source: mapred-default.xml
mapreduce.reduce.shuffle.input.buffer.percent	Default value: 0.7 Default source: mapred-default.xml
mapreduce.reduce.shuffle.memory.limit.percent	Default value: 0.25 Default source: mapred-default.xml
mapreduce.reduce.shuffle.merge.percent	Default value: 0.66 Default source: mapred-default.xml
mapreduce.reduce.shuffle.parallelcopies	Default value: 12 Default source:code
mapreduce.reduce.shuffle.read.timeout	Default value: 180000 Default source: mapred-default.xml
mapreduce.reduce.shuffle.retry-delay.max.ms	Default value: 60000 Default source: mapred-default.xml
mapreduce.reduce.skip.maxgroups	Default value: 0 Default source: mapred-default.xml
mapreduce.reduce.skip.proc.count.autoincr	Default value: TRUE Default source: mapred-default.xml
mapreduce.reduce.speculative	Default value: TRUE Default source: mapred-default.xml
mapreduce.shuffle.connection-keep-alive.enable	Default value: FALSE Default source: mapred-default.xml
mapreduce.shuffle.connection-keep-alive.timeout	Default value: 5 Default source: mapred-default.xml
mapreduce.shuffle.max.connections	Default value: 0 Default source: mapred-default.xml

mapreduce.shuffle.max.threads	Default value: 0 Default source: mapred-default.xml
mapreduce.shuffle.port	Default value: 13562 Default source: mapred-default.xml
mapreduce.shuffle.ssl.enabled	Default value: FALSE Default source: mapred-default.xml
mapreduce.shuffle.ssl.file.buffer.size	Default value: 65536 Default source: mapred-default.xml
mapreduce.shuffle.transfer.buffer.size	Default value: 131072 Default source: mapred-default.xml
mapreduce.task.combine.progress.records	Default value: 10000 Default source: mapred-default.xml
mapreduce.task.files.preserve.failedtasks	Default value: FALSE Default source: mapred-default.xml
mapreduce.task.io.sort.factor	Default value: 256 Default source: code
mapreduce.task.io.sort.mb	Default value: 100 Default source: mapred-default.xml
mapreduce.task.local.output.class	Default value: org.apache.hadoop.mapred.MapRFsOutputFile Default source: code
mapreduce.task.merge.progress.records	Default value: 10000 Default source: mapred-default.xml
mapreduce.task.profile	Default value: FALSE Default source: mapred-default.xml
mapreduce.task.profile.map.params	Default value: ${mapreduce.task.profile.params} Default source: mapred-default.xml
mapreduce.task.profile.maps	Default value: 0-2 Default source: mapred-default.xml
mapreduce.task.profile.params	Default value: -agentlib:hprof=cpu=samples,heap=sites,force=n,thread=y,verbose=n,file=%s Default source: mapred-default.xml
mapreduce.task.profile.reduce.params	Default value: ${mapreduce.task.profile.params} Default source: mapred-default.xml

last-updated: Apr 11, 2022

mapreduce.task.profile.reduces	Default value: 0-2 Default source: mapred-default.xml
mapreduce.task.skip.start.attempts	Default value: 2 Default source: mapred-default.xml
mapreduce.task.timeout	Default value: 600000 Default source: mapred-default.xml
mapreduce.task.userlog.limit.kb	Default value: 0 Default source: mapred-default.xml
yarn.app.mapreduce.am.command-opts	Default value: -Xmx1024m Default source: mapred-default.xml
yarn.app.mapreduce.am.container.log.backups	Default value: 0 Default source: mapred-default.xml
yarn.app.mapreduce.am.container.log.limit.kb	Default value: 0 Default source: mapred-default.xml
yarn.app.mapreduce.am.containerlauncher.threadpool-initial-size	Default value: 10 Default source: mapred-default.xml
yarn.app.mapreduce.am.hard-kill-timeout-ms	Default value: 10000 Default source: mapred-default.xml
yarn.app.mapreduce.am.job.client.port-range	Default value: blank (the range is all possible ports) Default source: mapred-default.xml When a range is specified, the YARN Mapreduce master will only open its web port within the range specified
yarn.app.mapreduce.am.job.committer.cancel-timeout	Default value: 60000 Default source: mapred-default.xml
yarn.app.mapreduce.am.job.committer.commit-window	Default value: 10000 Default source: mapred-default.xml
yarn.app.mapreduce.am.job.task.listener.thread-count	Default value: 30 Default source: mapred-default.xml
yarn.app.mapreduce.am.resource.cpu-vcores	Default value: 1 Default source: mapred-default.xml
yarn.app.mapreduce.am.resource.mb	Default value: 1536 Default source: mapred-default.xml
yarn.app.mapreduce.am.scheduler.heartbeat.interval-ms	Default value: 1000 Default source: mapred-default.xml

yarn.app.mapreduce.am.staging-dir	Default value: ${fs.defaultFS}/var/mapr/cluster/yarn/rm/staging Default source: code
yarn.app.mapreduce.client-am.ipc.max-retries	Default value: 3 Default source: mapred-default.xml
yarn.app.mapreduce.client-am.ipc.max-retries-on-timeouts	Default value: 3 Default source: mapred-default.xml
yarn.app.mapreduce.client.max-retries	Default value: 3 Default source: mapred-default.xml
yarn.app.mapreduce.shuffle.log.backups	Default value: 0 Default source: mapred-default.xml
yarn.app.mapreduce.shuffle.log.limit.kb	Default value: 0 Default source: mapred-default.xml
yarn.app.mapreduce.shuffle.log.separate	Default value: TRUE Default source: mapred-default.xml
yarn.app.mapreduce.task.container.log.backups	Default value: 0 Default source: mapred-default.xml

Environment Variables
Describes the environment variables specific to the MapR Data Platform.

For core release 6.0 and later, environment variables should be set in `/opt/mapr/conf/env_override.sh`. Editing `/opt/mapr/conf/env.sh` is no longer recommended. For more information, see About env_override.sh on page 2290.

Variable	Example Values	Description
`CLDB_EXTERNAL_RPC_PORT`	`5000`	If clients outside the cluster cannot reach CLDB on the default port, use the `CLDB_EXTERNAL_RPC_PORT` environment variable to specify the port on which CLDB can be reached.
`JAVA_HOME`	`/usr/lib/jvm/java-7-sun`	The directory where the correct version of Java is installed.
`MAPR_HOME`	`/opt/mapr` (default)	The directory in which the core software is installed.
`MAPR_EXTERNAL`	`10.10.123.25,10.10.123.30`	If your cluster nodes have multiple NICs, use the MAPR_EXTERNAL environment variable to grant external clients access to a cluster node on specific IP addresses. The value of the MAPR_EXTERNAL environment variable on a node is a comma-separated list of up to four IP addresses with no spaces.

Variable	Example Values	Description
MAPR_SUBNETS	10.10.123.0/24,10.10.124.0/24	MAPR_SUBNETS is used for MapR RPC to RPC communication. The MFS, CLDB, NFS, and LOOPBACKNFS modules use this environment variable. The NFS and LOOPBACKNFS modules use this environment variable when registering with CLDB. If you do not want MapR to use all NICs on each node, use this environment variable to restrict MapR traffic to specific NICs. Set MAPR_SUBNETS to a comma-separated list of up to four subnets in CIDR notation with no spaces. If you do not set MAPR_SUBNETS,MapR uses all NICs present on the node. When MAPR_SUBNETS is set, make sure that the node can reach all nodes in the cluster (servers and clients) using the specified subnets.
MAPR_USER	mapr (default)	Used with configure.sh on page 2053 to specify the user under which MapR runs its services. If not explicitly set, it defaults to the user mapr. After configure.sh is run, the value is stored in daemon.conf on page 2187 .
MAPR_ECOSYSTEM_LOGIN_OPTS	*hybrid*	Specifies the JAAS configuration to use with installed open source components.

About env_override.sh
Describes the purpose of the env_override.sh file.

env_override.sh is a file that you can create to store custom settings for environment variables. By default, /opt/mapr/conf/env.sh contains environment variables for a MapR cluster, but upgrading to a new MapR release causes the env.sh file to be replaced. (A backup is stored as /opt/mapr/conf/env.sh<timestemp>). When env.sh is replaced, any custom settings are removed.

For MapR 6.0 and later, keep any custom settings in /opt/mapr/conf/env_override.sh. It is no longer necessary to modify env.sh.

Upgrading a cluster does not remove or modify env_override.sh. /opt/mapr/conf/env.sh reads the env_override.sh file at the end of its execution. If the same parameter is listed in both env.sh and env_override.sh, the value specified in env_override.sh is used. If env_override.sh is not present, the values in env.sh are used.

You create env_override.sh from a blank file and insert export statements into the file. For example:

Sample env_override.sh File

```
export CLDB_EXTERNAL_RPC_PORT=5000
export JAVA_HOME=/usr/lib/jvm/java-7-sun
export MAPR_HOME=/opt/mapr
export MAPR_EXTERNAL=10.10.123.25,10.10.123.30
export MAPR_SUBNETS=10.10.123.0/24,10.10.124.0/24
export MAPR_USER=mapr
export MAPR_ECOSYSTEM_LOGIN_OPTS=*hybrid*
```

Ports Used by MapR Software
Lists the ports used by MapR services.

Avoiding Port Conflicts

To avoid trouble with port conflicts on your MapR clusters, try these tips:

- Remap the ports for the HBaseMaster and HBaseRegionServer services to ports below 32768.
- Set the ephemeral port range to stop at 50029 by changing the value in the file /proc/sys/net/ipv4/ip_local_port_range. Note that this setting changes the available number of ephemeral ports from the default of 28233 ports to 17233.

Ports Needed for POSIX Clients and File System to Communicate With Each Other

POSIX clients communicate with the CLDB and server components of the MapR filesystem. You need to open the relevant ports for TCP connectivity from POSIX clients to the MapR file-system cluster nodes. Open the CLDB, file-system server, and file-system server instances ports, as detailed in the following section.

Services and Ports Quick Reference

The following list defines the ports used by a MapR cluster, along with the default port numbers. All the ports used by MapR software are **TCP** ports.

API Server (apiserver)	*Source IP*: Cluster nodes running apiserver *Destination IP*: Cluster nodes running apiserver *Ports*: • 5701 • 5702 *Purpose*:Clustering support *Parameter and File where Port is Configured*: Not Applicable
CLDB	*Source IP*: Nodes running any MapR services, clients interacting with the file system *Destination IP*: Cluster nodes running CLDB services *Ports*: 7222 A client reads CLDB IP and port number from the /opt/mapr/conf/mapr-clusters.conf file. The client initially tries to communicate with CLDB on port 7222. Once it establishes the connection, it fetches the additional CLDB IPs and ports from the connected CLDB. By default, CLDB listens on ports 7222 and 7223. For performance reasons, additional ports may be opened, depending on the configuration parameter cldb.num.rpc.threads in the /opt/mapr/conf/cldb.conf file. For example, setting cldb.num.rpc.threads=3, opens up ports 7222, 7223 and 7224. **Note:** The cldb.num.rpc.threads parameter is hard-coded with a default value of 3. To change this value, add this parameter with the new value to the /opt/mapr/conf/cldb.conf file. **Note:** When you upgrade from MapR Core 5.2.x or Core 6 to Core 6.1 and above, the value of cldb.num.rpc.threads is not changed. The default remains as 3, which means three ports are open for each CLDB node.

last-updated: Apr 11, 2022

	The client tries connecting to the CLDBs till timeout occurs in the case of soft mount, while the client indefinitely retries in the case of hard mount. If a client cannot connect to a CLDB port, the CLDB is marked unreachable. For example, assume a CLDB with IP 10.10.10.10 and 3 ports `7222`, `7223` and `7224`. If a client fails to connect to the CLDB say on port `7223`, the CLDB 10.10.10.10 is marked unreachable, and the client will not try the two other ports for the next few minutes. It tries to connect with the next CLDB entry in the list. For load balancing at CLDB, a client will always pick a random port among the available CLDB ports. *Purpose*: MapR File System API calls *Parameter and File where Port is Configured*: • `/opt/mapr/conf/cldb.conf` • `/opt/mapr/conf/warden.conf` • `/opt/mapr/conf/mapr-clusters.conf`
CLDB JMX Monitor Port	*Source IP*: Nodes running CLDB services *Destination IP*: CLDB JMX monitor port *Ports*: 7220 *Purpose*: The port on which `Collectd` gathers CLDB metrics through JMX. *Parameter and File where Port is Configured*: Not Applicable
CLDB web port	*Source IP*: Nodes/clients connecting to the CLDB GUI *Destination IP*: Cluster nodes running CLDB services *Ports*: 7221 *Purpose*: CLDB GUI for a cluster with security disabled. For a secure cluster, the port is `7443` as defined by the `maprlogin` utility. *Parameter and File where Port is Configured*: `/opt/mapr/conf/cldb.conf`
maprlogin utility	*Source IP*: Connections using the maprlogin utility *Destination IP*: Cluster nodes running CLDB services *Ports*: 7443 *Purpose*: When security is enabled for a cluster, the CLDB listens for connections on port 7443. If security is disabled, the `maprlogin` utility is unable to reach the CLDB. *Parameter and File where Port is Configured*: Not Applicable
Data Access Gateway	*Source IP*: Clients using the MapR Database JSON REST API with HTTPS *Destination IP*:Not Applicable *Ports*: 8243 *Purpose*: The port used to connect to the Data Access Gateway using HTTPS

	Parameter and File where Port is Configured: `rest.https.port` in `/opt/mapr/data-access-gateway/conf/properties.cfg`
Data Access Gateway	*Source IP*: Node.js OJAI client *Destination IP*: Cluster nodes running the Data Access Gateway service *Ports*: 5678 *Purpose*: The port used to connect the OJAI client to the Data Access Gateway *Parameter and File where Port is Configured*: `grpc.service.port` in `/opt/mapr/data-access-gateway/conf/properties.cfg`
Data Access Gateway	*Source IP*: Python OJAI client *Destination IP*: Cluster nodes running the Data Access Gateway service *Ports*: 5678 *Purpose*: The port used to connect the OJAI client to the Data Access Gateway *Parameter and File where Port is Configured*: `grpc.service.port` in `/opt/mapr/data-access-gateway/conf/properties.cfg`
Data Access Gateway	*Source IP*: Go OJAI client *Destination IP*: Cluster nodes running the Data Access Gateway service *Ports*: 5678 *Purpose*: The port used to connect the OJAI client to the Data Access Gateway *Parameter and File where Port is Configured*: `grpc.service.port` in `/opt/mapr/data-access-gateway/conf/properties.cfg`
Data Access Gateway	*Source IP*: C# OJAI client *Destination IP*: Cluster nodes running the Data Access Gateway service *Ports*: 5678 *Purpose*: The port used to connect the OJAI client to the Data Access Gateway *Parameter and File where Port is Configured*: `grpc.service.port` in `/opt/mapr/data-access-gateway/conf/properties.cfg`
Data Access Gateway	*Source IP*: Java OJAI thin client *Destination IP*: Cluster nodes running the Data Access Gateway service *Ports*: 5678 *Purpose*: The port used to connect the OJAI client to the Data Access Gateway *Parameter and File where Port is Configured*: `grpc.service.port` in `/opt/mapr/data-access-gateway/conf/properties.cfg`
Drill JMX Port	*Source IP*: Nodes running the Drillbit service *Destination IP*: Drill JMX Port *Ports*: 6090

last-updated: Apr 11, 2022

	Purpose: The port on which `Collectd` gathers Drill metrics via JMX. *Parameter and File where Port is Configured*: Not Applicable
Drill Web UI	*Source IP*: Nodes running the Drillbit service *Destination IP*: Nodes running the Drillbit service *Ports*: 8047 *Purpose*: TCP port needed for the Drill Web UI and clients using REST API and nodes running the Drillbit service. *Parameter and File where Port is Configured*: `drill.exec.http.port` in `/opt/mapr/drill/drill-<version>/conf/drill-override.conf`
Drill (User Port)	*Source IP*: Nodes running the Drillbit service and clients using JDBC/ODBC *Destination IP*: Nodes running the Drillbit service *Ports*: 31010 *Purpose*: TCP user port address. Used between nodes in a Drill cluster. Needed for an external client, such as Tableau, to connect into the cluster nodes. Also needed for the Drill Web UI. You can also use this port to connect directly to a Drillbit. *Parameter and File where Port is Configured*: `drill.exec.rpc.user.server.port` in `/opt/mapr/drill/drill-<version>/conf/drill-override.conf`
Drill (Control Port)	*Source IP*: Nodes running the Drillbit service *Destination IP*: Nodes running the Drillbit service *Ports*: 31011 *Purpose*: TCP port that controls the port address. Used between nodes in a Drill cluster. Needed for multi-node installation of Drill. *Parameter and File where Port is Configured*: `drill.exec.rpc.bit.server.port` in `/opt/mapr/drill/drill-<version>/conf/drill-override.conf`
Drill (Data Port)	*Source IP*: Nodes running the Drillbit service *Destination IP*: Nodes running the Drillbit service *Ports*: 31012 *Purpose*: TCP data port address. Used between nodes in a Drill cluster. Needed for multi-node installation of Drill. *Parameter and File where Port is Configured*: `drill.exec.rpc.bit.server.port + 1` in `/opt/mapr/drill/drill-<version>/conf/drill-override.conf`
Drill (ZooKeeper Port)	*Source IP*: Clients using JDBC/ODBC and nodes running ZooKeeper services *Destination IP*: Nodes running the Drillbit service *Ports*: 5181 *Purpose*: ZooKeeper port used to connect to Drill through the JDBC driver.

	Parameter and File where Port is Configured: See `the ZooKeeper entry in this list.`
Elasticsearch (Components Communication Port)	*Source IP*: Non-Elasticsearch components, such a web browser, curl, and Kibana, that connect to Elasticsearch. *Destination IP*: Nodes running Elasticsearch for monitoring use cases *Ports*: 9200 *Purpose*: Non-Elasticsearch components use this port when communicating with Elasticsearch. *Parameter and File where Port is Configured*: You can configure a different port for monitoring use cases when you run the configure.sh on page 2053 script with the `-ES` parameter.
Elasticsearch (Daemons Communication Port)	*Source IP*: Nodes running Elasticsearch *Destination IP*: Nodes running Elasticsearch for monitoring use cases *Ports*: 9300 *Purpose*: Elasticsearch uses this port for communications between Elasticsearch daemons. *Parameter and File where Port is Configured*: Not Applicable
File Migration Service	*Source IP*: Not Applicable *Destination IP*: Not Applicable *Ports*: 9444 *Purpose*: File Migration service UI *Parameter and File where Port is Configured*: `/opt/mapr/conf/conf.d/warden.filemigrate.conf`
Gateway	*Source IP*: Nodes sending operations to replicate *Destination IP*: Nodes running the gateway service *Ports*: 7660 *Purpose*: The port used by gateway services to listen for incoming replication operations. *Parameter and File where Port is Configured*: Not Applicable
Grafana	*Source IP*: Web Browsers *Destination IP*: Nodes running Grafana for monitoring *Ports*: 3000 *Purpose*: Web browsers use this port when connecting to Grafana. *Parameter and File where Port is Configured*: Not Applicable
HBase Master	*Source IP*: HBase Clients *Destination IP*: Nodes running HBase Master services *Ports*: 16000 *Purpose*: HBase API and HBase shell use this port to connect to HBase Master *Parameter and File where Port is Configured*: `/opt/mapr/hbase/hbase-<version>/conf/hbase-site.xml`

HBase Master Web UI	*Source IP*: HBase Master Web UI clients *Destination IP*: Nodes running HBase Master services *Ports*: 16010 *Purpose*: Information Web UI of HBase Master *Parameter and File where Port is Configured*: `/opt/mapr/hbase/hbase-<version>/conf/hbase-site.xml`
HBase Thrift Server	*Source IP*: HBase Thrift Server clients *Destination IP*: Nodes running HBase Thrift Server *Ports*: 9090 *Purpose*: The HBase client uses this port to connect to HBase, using the Thrift protocol *Parameter and File where Port is Configured*: `/opt/mapr/hbase/hbase-<version>/conf/hbase-site.xml`
HBase Thrift Web UI	*Source IP*: HBase Thrift Web UI clients *Destination IP*: Nodes running HBase Thrift *Ports*: 9095 *Purpose*: Information Web UI of HBase Thrift *Parameter and File where Port is Configured*: `/opt/mapr/hbase/hbase-<version>/conf/hbase-site.xml`
HBase REST Server	*Source IP*: HBase REST Server clients *Destination IP*: Nodes running HBase REST Server *Ports*: 8080 *Purpose*: The HBase client uses this port to connect to HBase using the HTTP protocol *Parameter and File where Port is Configured*: `/opt/mapr/hbase/hbase-<version>/conf/hbase-site.xml`
HBase REST Web UI	*Source IP*: HBase REST Web UI clients *Destination IP*: Nodes running HBase REST *Ports*: 8086 *Purpose*: Information Web UI of HBase REST *Parameter and File where Port is Configured*: `/opt/mapr/hbase/hbase-<version>/conf/hbase-site.xml`
HBase Regionserver	*Source IP*: HBase Clients *Destination IP*: Nodes running HBase Regionserver services *Ports*: 16020 *Purpose*: HBase API and HBase shell use this port to connect to HBase RegionServer *Parameter and File where Port is Configured*: `/opt/mapr/hbase/hbase-<version>/conf/hbase-site.xml`
HBase Regionserver UI	*Source IP*: HBase Regionserver Web UI clients *Destination IP*: Nodes running HBase Regionserver *Ports*: 16030

	Purpose: Information Web UI of HBase Regionserver *Parameter and File where Port is Configured*: `/opt/mapr/hbase/hbase-<version>/conf/hbase-site.xml`
HistoryServer RPC	*Source IP*: Not Applicable *Destination IP*: Nodes running MapReduce JobHistory Server *Ports*: 10020 *Purpose*: Not Applicable *Parameter and File where Port is Configured*: Not Applicable
HistoryServer Web UI and REST APIs	*Source IP*: Clients that access Job History Server UI in a *non-secure* cluster *Destination IP*: Secure nodes running MapReduce JobHistory Server in a non-secure cluster *Ports*: 19888 *Purpose*: Non-secure HistoryServer Web UI and REST APIs *Parameter and File where Port is Configured*: See mapred-site.xml on page 2201
HistoryServer Web UI and REST APIs	*Source IP*: Clients that access Job History Server UI in a *secure* cluster *Destination IP*: Secure nodes running MapReduce JobHistory Server in a secure cluster *Ports*: 19890 *Purpose*: Secure HistoryServer Web UI and REST APIs *Parameter and File where Port is Configured*: See mapred-site.xml on page 2201
Hive Metastore	*Source IP*: Nodes/clients performing Hive queries/ operations *Destination IP*: Nodes running the Hive metastore services *Ports*: 9083 *Purpose*: Used by Hive clients to query/access the Hive metastore *Parameter and File where Port is Configured*: `/opt/mapr/hive/hive-<version>/conf/hive-site.xml`
Hiveserver2	*Source IP*: Nodes or clients performing hive queries using JDBC/ODBC *Destination IP*: Nodes running Hiveserver2 *Ports*: 10000 *Purpose*: Port through which clients perform hive queries *Parameter and File where Port is Configured*: Not Applicable
Hiveserver2 Web UI	*Source IP*: Not Applicable *Destination IP*: Nodes running Hiverserver2 Web UI *Ports*: 10002

last-updated: Apr 11, 2022

	Purpose: Provides access to Hive configuration settings, local logs, metrics, and information about active sessions and queries. *Parameter and File where Port is Configured*: Not Applicable
Hoststats	See `hoststats.port` and `hs.port` in the Warden configuration file.
Httpfs	*Source IP*: Nodes/clients accessing httpfs services *Destination IP*: Nodes running httpfs services *Ports*: 14000 *Purpose*: Used by httpfs file clients to access the httpfs server *Parameter and File where Port is Configured*: • `/opt/mapr/httpfs/httpfs-<version>` • `/etc/hadoop/httpfs-env.sh`
Hue Webserver	*Source IP*: Nodes/clients accessing Hue web services *Destination IP*: Nodes running Hue web services *Ports*: 8888 *Purpose*: Used by Hue webserver clients to access the Hue webserver *Parameter and File where Port is Configured*: `/opt/mapr/hue/hue*/desktop/conf/hue.ini`
Impala Catalog Daemon	*Source IP*: Nodes running Impala Daemon *Destination IP*: Nodes running Impala Catalog Daemon *Ports*: 25020 *Purpose*: Catalog service web interface for monitoring and troubleshooting. Available in Impala 1.2 and higher. *Parameter and File where Port is Configured*: Not Applicable
Impala Daemon	*Source IP*: Clients using JDBC/ODBC and nodes running Impala Daemon *Destination IP*: Nodes running Impala Daemon *Ports*: 21000 *Purpose*: Used to transmit commands and receive results by `impala-shell` *Parameter and File where Port is Configured*: Not Applicable
Impala Daemon	*Source IP*: Nodes running Impala Daemon *Destination IP*: Nodes running Impala Daemon *Ports*: 21050 *Purpose*: Used by applications, such as Business Intelligence tools, to transmit commands and receive results using JDBC. *Parameter and File where Port is Configured*: Not Applicable

Impala Daemon	*Source IP*: Nodes running Impala Daemon *Destination IP*: Nodes running Impala Daemon *Ports*: 25000 *Purpose*: Impala web interface for monitoring and troubleshooting. *Parameter and File where Port is Configured*: Not Applicable
Impala StateStoreDaemon	*Source IP*: Nodes running Impala Daemon *Destination IP*: Nodes running Impala StateStore Daemon *Ports*: 25010 *Purpose*: StateStore web interface for monitoring and troubleshooting *Parameter and File where Port is Configured*: Not Applicable
KSQL	*Source IP*: All cluster nodes *Destination IP*: Nodes running KSQL *Ports*: 8084 *Purpose*: KSQL *Parameter and File where Port is Configured*: `$KSQL_INSTALL_DIR/etc/ksql/ksqlserver.properties`
Kafka Connect	*Source IP*: All cluster nodes *Destination IP*: Nodes running Kafka Connect *Ports*: 8083 *Purpose*: Kafka Connect REST API calls *Parameter and File where Port is Configured*: `/opt/mapr/kafka/kafka-<version>/config/connect-distributed.properties`
Kafka REST	*Source IP*: All cluster nodes *Destination IP*: Nodes running Kafka REST *Ports*: 8082 *Purpose*: Kafka Connect REST API calls *Parameter and File where Port is Configured*: `/opt/mapr/kafka-rest/kafka-rest-<version>/config/kafka-rest.properties`
Kafka Schema Registry	*Source IP*: All cluster nodes *Destination IP*: Nodes running Kafka Schema Registry *Ports*: 8087 *Purpose*: Kafka Schema Registry API calls *Parameter and File where Port is Configured*: `/opt/mapr/schema-registry/schema-registry-<version>/config/schema-registry.properties`
Kibana	*Source IP*: Web browsers *Destination IP*: Nodes running Kibana for monitoring use cases *Ports*: 5601

	Purpose: Web browsers use this port when connecting to Grafana. *Parameter and File where Port is Configured*: Not Applicable
MAST Gateway	*Source IP*: Nodes running MAST Gateway service *Destination IP*: Nodes running MAST Gateway service *Ports*: 8660 *Purpose*: MapR clients use this port to connect to the MAST Gateway *Parameter and File where Port is Configured*: `/opt/mapr/conf/mastgateway.conf`
MapR File System server	*Source IP*: Nodes running any MapR services, clients interacting with the file system *Destination IP*: Nodes running FileServer services *Ports*: 5660, 5692, 5724, and 5756 *Purpose*: The filesystem is a random read-write distributed filesystem that allows applications to concurrently read and write directly to disk. Clients use these ports to access the file-system server. *Parameter and File where Port is Configured*: Not Applicable
MapR File System server	*Source IP*: Nodes running the gateway service *Destination IP*: Nodes running the MapR File System *Ports*: 6660 *Purpose*: The port on which gateway nodes send replicated operations to nodes in destination clusters. *Parameter and File where Port is Configured*: Not Applicable
MapR File System server instances	*Source IP*: Not Applicable *Destination IP*: Not Applicable *Ports*: See Working with Multiple Instances of the File System on page 790 *Purpose*: Multiple MapR File System instances *Parameter and File where Port is Configured*: Not Applicable
Object Store	*Source IP*: Nodes accessing the MOSS server *Destination IP*: Nodes running the MOSS server *Ports*: 9000 *Purpose*: Port for the MOSS server *Parameter and File where Port is Configured*: The *moss.port* option in `/opt/mapr/conf/moss.conf`. **CAUTION:** The default port for S3 Gateway is also 9000. If you run S3 Gateway and Object Store, change one of the ports to avoid conflicts.
S3 Gateway	*Source IP*: Nodes accessing the S3 Gateway server *Destination IP*: Nodes running the S3 Gateway server *Ports*: 9000 *Purpose*: Port for the S3 Gateway server

last-updated: Apr 11, 2022

	Parameter and File where Port is Configured: The *ports* option in `/opt/mapr/objectstore-client/objectstore-client-<version>/conf/minio.json`
NFS	*Source IP*: Nodes/clients accessing the filesystem via the NFS protocol *Destination IP*: Nodes running MapR NFS Services *Ports*: 2049 *Purpose*: NFSv3 or NFSv4 access to the MapR File System *Parameter and File where Port is Configured*: Not Applicable
NFS	*Source IP*: Nodes running NFS services *Destination IP*: Nodes running NFS services *Ports*: 9997, 9998 *Purpose*: NFS VIP Management *Parameter and File where Port is Configured*: `/opt/mapr/conf/nfsserver.conf`
NodeManager JMX Port	*Source IP*: Nodes running NodeManager *Destination IP*: NodeManager JMX Port *Ports*: 8027 *Purpose*: The port on which Collectd gathers metrics from NodeManager nodes via JMX. *Parameter and File where Port is Configured*:Not Applicable
NodeManager	*Source IP*: Nodes running NodeManager *Destination IP*: Not Applicable *Ports*: 8099 *Purpose*: The node manager manages the health of each node in the cluster. *Parameter and File where Port is Configured*:`yarn.nodemanager.address` in `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/yarn-site.xml`
NodeManager Localizer RPC	*Source IP*: Nodes running NodeManager *Destination IP*: Not Applicable *Ports*: 8040 *Purpose*: The port that node manager uses to localize resources for a node. With localization, remote resources are downloaded to the local filesystem for access. *Parameter and File where Port is Configured*: `yarn.nodemanager.localizer.address` in `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/yarn-site.xml`
NodeManager Web UI and REST APIs	*Source IP*: External Web browsers and REST clients accessing NodeManager services in a non-secure cluster *Destination IP*: Nodes running NodeManager services in a non-secure cluster *Ports*: 8042

last-updated: Apr 11, 2022

	Purpose: NodeManager HTTP port *Parameter and File where Port is Configured*: `yarn.nodemanager.webapp.address` in `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/yarn-site.xml`
NodeManager Web UI and REST APIs	*Source IP*: External Web browsers and REST clients accessing NodeManager services in a secure cluster *Destination IP*: Nodes running NodeManager services in a secure cluster *Ports*: 8044 *Purpose*: NodeManager HTTPS port *Parameter and File where Port is Configured*: `yarn.nodemanager.webapp.https.address` in `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/yarn-site.xml`
Oozie	*Source IP*: Nodes/clients accessing Oozie services in a *non-secure* cluster *Destination IP*: Nodes running Oozie services in a non-secure cluster *Ports*: 11000 *Purpose*: Used by Oozie clients to access the Oozie server in a non-secure cluster *Parameter and File where Port is Configured*: `/opt/mapr/oozie/oozie-<version>/conf/oozie-env.sh`
Oozie	*Source IP*: Nodes/clients accessing Oozie services in a *secure* cluster *Destination IP*: Nodes running Oozie services in a secure cluster *Ports*: 11443 *Purpose*: Used by Oozie clients to access the Oozie server in a secure cluster *Parameter and File where Port is Configured*: `/opt/mapr/oozie/oozie-<version>/conf/oozie-env.sh`
OpenTSDB	*Source IP*: OpenTSDB clients, such as `Collectd`. *Destination IP*: Nodes running OpenTSDB for monitoring use cases. *Ports*: 4242 *Purpose*: `Collectd` uses this port to write metrics to OpenTSDB. *Parameter and File where Port is Configured*: You can configure a different port for monitoring use cases when you run configure.sh on page 2053 script with the `-OT` parameter.
Port Mapper	*Source IP*: Nodes running MapR NFS Services *Destination IP*: Nodes/clients accessing the filesystem using the NFS protocol *Ports*: 111 *Purpose*: RPC Portmap services used to connect to the MapR File System using NFSv3

	Parameter and File where Port is Configured: Not Applicable
ResourceManager JMX Port	*Source IP*: Nodes running ResourceManager *Destination IP*: ResourceManager JMX port *Ports*: 8025 *Purpose*: The port on which `Collectd` gathers metrics from the ResourceManager using JMX. *Parameter and File where Port is Configured*: Not Applicable
ResourceManager Admin RPC	*Source IP*: Applications that access the ResourceManager *Destination IP*: Nodes running ResourceManager *Ports*: 8033 *Purpose*: The port that applications use to access the ResourceManager RPC *Parameter and File where Port is Configured*: `yarn.resourcemanager.admin.address` in `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/yarn-site.xml`
ResourceManager Client RPC	*Source IP*: Clients that submit YARN applications *Destination IP*: Nodes running ResourceManager *Ports*: 8032 *Purpose*: The port that clients use to access the YARN applications *Parameter and File where Port is Configured*: `yarn.resourcemanager.address` in `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/yarn-site.xml`
ResourceManager Resource Tracker RPC (for NodeManagers)	*Source IP*: Applications that access the ResourceManager *Destination IP*: Nodes running ResourceManager *Ports*: 8031 *Purpose*: The port that applications use to access the Resource Manager Tracker RPC *Parameter and File where Port is Configured*: `yarn.resourcemanager.resource-tracker.address` in `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/yarn-site.xml`
ResourceManager Scheduler RPC (for ApplicationMasters)	*Source IP*: Applications that access the ResourceManager *Destination IP*: Nodes running ResourceManager *Ports*: 8030 *Purpose*: The port on which the applications in the cluster talk to the ResourceManager. *Parameter and File where Port is Configured*: `yarn.resourcemanager.scheduler.address` in `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/yarn-site.xml`
ResourceManager Web UI (HTTP)	*Source IP*: Clients that access ResourceManager UI in a *non-secure* cluster

	Destination IP: Nodes running ResourceManager master in a non-secure cluster *Ports*: 8088 *Purpose*: ResourceManager Web UI *Parameter and File where Port is Configured*: `yarn.resourcemanager.webapp.address` in `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/yarn-site.xml`
ResourceManager Web UI (HTTPS)	*Source IP*: Clients that access ResourceManager UI in a *secure* cluster *Destination IP*: Nodes running ResourceManager master in a secure cluster *Ports*: 8090 *Purpose*: ResourceManager Web UI *Parameter and File where Port is Configured*: `yarn.resourcemanager.webapp.address` in `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/yarn-site.xml`
Shuffle HTTP	*Source IP*: Not Applicable *Destination IP*: Shuffle HTTP *Ports*: 13562 *Purpose*: The port that MapReduce Shuffle uses. Transferring the map outputs to reducer inputs in sorted form is the shuffle operation. *Parameter and File where Port is Configured*: Not Applicable
Spark Standalone Master (RPC)	*Source IP*: Not Applicable *Destination IP*: Not Applicable *Ports*: 7077 *Purpose*: The port on which to submit jobs in a Spark standalone cluster. *Parameter and File where Port is Configured*: `SPARK_MASTER_PORT` in `SPARK_HOME/conf/spark-env.sh`
Spark Standalone Master (Web UI)	*Source IP*: Nodes/clients accessing Spark services in a *non-secure* cluster *Destination IP*: Nodes running Spark services in a non-secure cluster *Ports*: 8580 *Purpose*: The port on which browsers connect to Spark master in a non-secure Spark standalone cluster. *Parameter and File where Port is Configured*: `SPARK_MASTER_WEBUI_PORT` in `SPARK_HOME/conf/spark-env.sh`
Spark Standalone Master (Web UI)	*Source IP*: Nodes/clients accessing Spark services in a *secure* cluster *Destination IP*: Nodes running Spark services in a secure cluster *Ports*: 8980 *Purpose*: The port on which browsers connect to a Spark master in a secure Spark standalone cluster.

	Parameter and File where Port is Configured: `SPARK_MASTER_WEBUI_PORT` in `SPARK_HOME/conf/spark-env.sh`
Spark Standalone Worker	*Source IP*: Not Applicable *Destination IP*: Not Applicable *Ports*: 8081 *Purpose*: The port on which browsers connect to Spark workers in a Spark standalone cluster. *Parameter and File where Port is Configured*: `SPARK_WORKER_WEBUI_PORT` in `SPARK_HOME/conf/spark-env.sh`
Spark Thrift Server (if start and stop server using Spark scripts)	*Source IP*: Not Applicable *Destination IP*: Not Applicable *Ports*: 10000 *Purpose*: The port on which JDBC clients connect to Spark Thrift server. *Parameter and File where Port is Configured*: `hive.server2.thrift.port` in `SPARK_HOME/conf/hive-site.xml`
Spark Thrift Server (if start and stop server through Warden, starting in EEP 4.0)	*Source IP*: Not Applicable *Destination IP*: Not Applicable *Ports*: 2304 *Purpose*: The port on which JDBC clients connect to Spark Thrift server. *Parameter and File where Port is Configured*: `hive.server2.thrift.port` in `SPARK_HOME/conf/hive-site.xml`
Spark History Server	*Source IP*: Clients that access Spark Job History in a *non-secure* cluster *Destination IP*: Nodes running Spark History Server in a non-secure cluster *Ports*: 18080 *Purpose*: The port on which browsers connect to a non-secure Spark history server. *Parameter and File where Port is Configured*: `spark.history.ui.port` in `SPARK_HOME/conf/spark-default.conf"`
Spark History Server	*Source IP*: Clients that access Spark Job History in a *secure* cluster *Destination IP*: Nodes running Spark History Server in a secure cluster *Ports*: 18480 *Purpose*: The port on which browsers connect to a secure Spark history server. *Parameter and File where Port is Configured*: `spark.ssl.historyServer.port` in `SPARK_HOME/conf/spark-defaults.conf (starting from Spark-2.2.1)`
Spark External Shuffle Service	*Source IP*: Not Applicable *Destination IP*: Not Applicable *Ports*: 7337

	Purpose: The port on which Spark jobs connect to External Shuffle server. *Parameter and File where Port is Configured*: `spark.shuffle.service.port` in `SPARK_HOME/conf/spark-default.conf`
Sqoop2 Server	*Source IP*: Nodes/clients accessing Sqoop2 services *Destination IP*: Nodes running Sqoop2 services *Ports*: 12000 *Purpose*: Used by Sqoop2 clients to access the Sqoop2 server *Parameter and File where Port is Configured*: Not Applicable
Tez Shuffle	*Source IP*: Not Applicable *Destination IP*: Not Applicable *Ports*: 13563 *Purpose*: Port to communicate with the Tez Shuffler. A Tez specific shuffle handler allows data to be shuffled in a way that takes advantage of the new features in Tez *Parameter and File where Port is Configured*: Not Applicable
Timeline Server	*Source IP*: Not Applicable *Destination IP*: Not Applicable *Ports*: 10200 *Purpose*: Hadoop IPC port used for internal communication in Hadoop *Parameter and File where Port is Configured*: Not Applicable
Timeline Server Web Interface (HTTP)	*Source IP*: Not Applicable *Destination IP*: Not Applicable *Ports*: 8188 *Purpose*: Non-secure web access for the Timeline Server. The Timeline Server allows storage and retrieval of an application's current and historic information in a generic fashion. *Parameter and File where Port is Configured*: Not Applicable
Timeline Server Web Interface (HTTPS)	*Source IP*: Not Applicable *Destination IP*: Not Applicable *Ports*: 8190 *Purpose*: Secure web access for the Timeline Server *Parameter and File where Port is Configured*: Not Applicable
Tomcat Port (Hive-on-Tez UI)	*Source IP*: Not Applicable *Destination IP*: Not Applicable *Ports*: 9383 *Purpose*: The non-secure port to access the Tez UI. Hive-on-Tez speeds up execution of Hive queries. *Parameter and File where Port is Configured*: Not Applicable

Tomcat SSL Port (Hive-on-Tez UI)	*Source IP*: Not Applicable *Destination IP*: Not Applicable *Ports*: 9393 *Purpose*: The secure port to access the Tez UI. *Parameter and File where Port is Configured*: Not Applicable
Web UI	*Source IP*: External web browser accessing either a *non-secure* or a *secure* cluster *Destination IP*: Nodes running the Control System Web UI in a non-secure or a secure cluster *Ports*: 8443 *Purpose*: Control System Web UI *Parameter and File where Port is Configured*: `/opt/mapr/apiserver/conf/properties.cfg`
Zeppelin	*Source IP*: Not Applicable *Destination IP*: Not Applicable *Ports*: 9995 *Purpose*: The port to connect to the Zeppelin Docker container *Parameter and File where Port is Configured*: Configurable by setting `ZEPPELIN_SSL_PORT` when running the Zeppelin Docker image
ZooKeeper	*Source IP*: Nodes running ZooKeeper services, clients executing ZooKeeper API calls *Destination IP*: Nodes running ZooKeeper services *Ports*: 5181 *Purpose*: ZooKeeper API calls *Parameter and File where Port is Configured*: • `/opt/mapr/zookeeper/zookeeper-<version>/conf/zoo.cfg` • `/opt/mapr/conf/warden.conf, /opt/mapr/conf/cldb.conf` • `/opt/mapr/hbase/hbase-<version>/conf/hbase-site.xml` • `/opt/mapr/hive/hive-<version>/conf/hive-site.xml`
ZooKeeper follower-to-leader Communication	*Source IP*: Nodes running ZooKeeper services *Destination IP*: Nodes running ZooKeeper services *Ports*: 2888 *Purpose*: ZooKeeper Server > Server Communication *Parameter and File where Port is Configured*: `/opt/mapr/zookeeper/zookeeper-<version>/conf/zoo.cfg`
ZooKeeper Leader Election	*Source IP*: Nodes running ZooKeeper services *Destination IP*: Nodes running ZooKeeper services *Ports*: 3888 *Purpose*: ZooKeeper Server > Server Communication

last-updated: Apr 11, 2022

Parameter and File where Port is Configured: `/opt/mapr/zookeeper/zookeeper-<version>/conf/zoo.cfg`

Log Files

Lists the log files for each MapR component.

The table below provides information on the log files for the components.

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
CLDB	Main	/opt/mapr/logs/cldb.log	/opt/mapr/conf/log4j.cldb.properties	By Size	100MB	9	-
	Disk Balancer	/opt/mapr/logs/cldbdiskbalancer.log	/opt/mapr/conf/log4j.cldb.properties	Daily	-	-	10 days
	Role Balancer	/opt/mapr/logs/cldbrolebalancer.log	/opt/mapr/conf/log4j.cldb.properties	Daily	-	-	10 days
	Time Skew	/opt/mapr/logs/timeskew.log	/opt/mapr/conf/log4j.cldb.properties	Daily	-	-	10 days
	MapR filesystem Summary	/opt/mapr/logs/cldbfssummary.log	/opt/mapr/conf/log4j.cldb.properties	Daily	-	-	10 days
	Audit	/opt/mapr/logs/cldbaudit.log.json	/opt/mapr/conf/log4j.cldb.properties	Daily	-	-	-
	Guts	/opt/mapr/logs/cldbguts.log	/opt/mapr/conf/log4j.properties	-	-	-	10 days
	Proxy	/opt/mapr/logs/cldbproxy.log	/opt/mapr/conf/log4j.properties	-	-	-	10 days

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
WebServer	Main	/opt/mapr/apiserver/logs/apiserver.log	/opt/mapr/apiserver/conf/properties.cfg	Every Startup	-	10	10 days
			/opt/mapr/conf/log4j.mcs.properties	Daily	-	-	
	Authentication Audit	/opt/mapr/logs/authaudit.log.json	/opt/mapr/conf/log4j.mcs.properties	Daily	-	-	-
	PAM	/opt/mapr/logs/pam.log	/opt/mapr/conf/log4j.properties	Daily	-	-	10 days
Warden	Main	/opt/mapr/logs/warden.log	/opt/mapr/initscripts/mapr-warden	Every Startup	-	10	10 days
			/opt/mapr/conf/log4j.properties	Daily	-	-	
	System Volume Initialization	/opt/mapr/logs/createsystemvolumes.log	/opt/mapr/server/createsystemvolumes.sh	-	-	-	10 days
MapR File System	CLDB Connection Initialization	/opt/mapr/logs/mfs.log-0	/opt/mapr/conf/mfs.conf	By Size	200MB per file (1GB in total)	5	-
	MapR File System	/opt/mapr/logs/mfs.log-3	/opt/mapr/conf/mfs.conf	By Size	200MB per file (1GB in total)	5	-
	MapR Database	/opt/mapr/logs/mfs.log-5	/opt/mapr/conf/mfs.conf	By Size	200MB per file (1GB in total)	5	-
	stderr on startup	/opt/mapr/logs/mfs.err	-	-	-	-	-
	stdout on startup	/opt/mapr/logs/mfs.out	-	-	-	-	10 days
	Initialization	/opt/mapr/logs/mfsinit.log	/opt/mapr/initscripts/mapr-mfs	-	-	-	10 days
	Audit Initialization	/opt/mapr/logs/initaudit.log	/opt/mapr/server/initaudit.sh	-	-	-	10 days

last-updated: Apr 11, 2022

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
NFS	Main	/opt/mapr/logs/nfsserver.log	/opt/mapr/conf/nfsserver.conf	By Size	200MB	5	-
	NFS Monitoring	/opt/mapr/logs/nfsmon.log	-	-	-	-	10 days
	Local Mount	/opt/mapr/logs/mount_local_fs.log	/opt/mapr/bin/mount_local_fs.pl	-	-	-	10 days
NFSv4	NFS Ganesha Server	/opt/mapr/logs/nfs4/nfs4server.log	/opt/mapr/conf/nfs4server.conf	By Size	-	-	-
	NFsv4Server filesystem logs	/opt/mapr/logs/nfs4/fsal.log-0, 1, 2	-	By Size	200MB	5	-
	VIP	/opt/mapr/logs/nfs4/nfs4mon.log	-	-	-	-	-
HostStats	Main	/opt/mapr/logs/hoststats.log	-	By Size	20MB	5	-
	stderr on startup	/opt/mapr/logs/hoststats.err	-	-	-	-	-
Gateway	Main	/opt/mapr/logs/gateway.log	/opt/mapr/conf/log4j.properties	By Size	256MB	20	10 days
	Initialization	/opt/mapr/logs/gatewayinit.log	/opt/mapr/initscripts/mapr-gateway	-	-	-	10 days
Loopbacknfs POSIX Client	Main	/usr/local/mapr-loopbacknfs/logs/loopbacknfs.log	/usr/local/mapr-loopbacknfs/conf/nfsserver.conf	By Size	200MB	5	-
	NFS Monitoring	/usr/local/mapr-loopbacknfs/logs/nfsmon.log	-	-	-	-	-
	Local Mount	/usr/local/mapr-loopbacknfs/logs/mount_local_fs.log	/usr/local/mapr-loopbacknfs/bin/mount_local_fs.pl	-	-	-	-

last-updated: Apr 11, 2022

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
FUSE-based POSIX Client	Basic	/opt/mapr/logs/posix-client-basic.log	/opt/mapr/conf/fuse.conf	-	-	-	-
	Platinum	/opt/mapr/logs/posix-client-platinum.log					
	PACC	/opt/mapr/logs/posix-client-basic.log					
	FUSE logs	/opt/mapr/logs/ffs.log-n (where n is between 0 and 4)		By Size	256 MB	5	-

last-updated: Apr 11, 2022

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
Tools	configure.sh	/opt/mapr/logs/configure.log	/opt/mapr/server/configure.sh	-	-	-	-
	disksetup	/opt/mapr/logs/disksetup.<uid>.log	/opt/mapr/server/disksetup	-	-	-	10 days
	config-mapr-user.sh	/opt/mapr/logs/config-mapr-user.log	/opt/mapr/server/config-mapr-user.sh	-	-	-	10 days
	prerequisitecheck.sh	/opt/mapr/logs/prerequisitecheck-<username>.log	/opt/mapr/server/prerequisitecheck.sh	-	-	-	10 days
	handle_disk_failure.sh	/opt/mapr/logs/faileddisk.log	/opt/mapr/server/handle_disk_failure.sh	-	-	-	10 days
	diskremove	/opt/mapr/logs/diskremove.<uid>.log	/opt/mapr/server/diskremove	-	-	-	10 days
	gfsck	/opt/mapr/logs/gfsck.log	/opt/mapr/conf/log4j.properties	Daily	-	-	10 days
	maprlogin	/opt/mapr/logs/maprlogin-<username>-<uid>.log	/opt/mapr/conf/log4j.properties	-	-	-	10 days
	maprexecute	/opt/mapr/logs/maprexecute.log	-	-	-	-	10 days
	mrdisk	/opt/mapr/logs/mrdisk.<uid>.log	-	-	-	-	10 days
	expandaudit	/opt/mapr/logs/expandaudit-<username>-<uid>.log	/opt/mapr/conf/log4j.properties	Daily	-	-	10 days
	expandaudit error	/opt/mapr/logs/expandaudit-<username>-<uid>-$(date +%Y%m%d_%H%M%S).errlog	-	-	-	-	10 days
	upgrade	/opt/mapr/logs/upgrade.log	/opt/mapr/server/upgrade	-	-	-	10 days

last-updated: Apr 11, 2022

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
Upgrade	Single Node Upgrade	/opt/mapr/logs/singlenodeupgrade.log	/opt/upgrade-mapr/singlenodeupgrade.sh	-	-	-	10 days
	Single Node Upgrade Summary	/opt/mapr/logs/singlenodeupgrade.log.summary	/opt/upgrade-mapr/singlenodeupgrade.sh	-	-	-	10 days
	Rolling Upgrade	/opt/mapr/logs/rollingupgrade.log	/opt/upgrade-mapr/rollingupgrade.sh	-	-	-	10 days
	Rolling Upgrade Summary	/opt/mapr/logs/rollingupgrade.log.summary	/opt/upgrade-mapr/rollingupgrade.sh	-	-	-	10 days
CentralConfig	Main	/opt/mapr/logs/pullcentralconfig.log	/opt/mapr/server/pullcentralconfig	By Size	50MB	1	10 days
			/opt/mapr/conf/log4j.properties	Daily	-	-	
	Error	/opt/mapr/logs/central_config_err_pid<pid>.log	/opt/mapr/server/pullcentralconfig	-	-	-	10 days
MapR CLI	Main	/opt/mapr/logs/maprcli-<username>-<uid>.log	/opt/mapr/conf/log4j.properties	Daily	-	-	10 days
	Volume Dump	/opt/mapr/logs/maprcli-dump-<username>-<uid>.log	/opt/mapr/bin/maprcli	By Size	512KB	-	10 days
	Temporary Volume Dump	/opt/mapr/logs/maprcli-dump-<username>-<uid>-cmd-`date +"%F-%T"`.$$.log	/opt/mapr/bin/maprcli	-	-	-	10 days
	Audit	/opt/mapr/mapr-cli-audit-log/audit.log.json	-	Weekly	-	-	-

last-updated: Apr 11, 2022

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
mapr Command	Main	/opt/mapr/logs/mapr-<username>-<hostname>.log	/opt/mapr/conf/log4j.properties	-	-	-	10 days
	dbshell	/opt/mapr/logs/maprdb-shell-<username>-<hostname>.log	/opt/mapr/conf/log4j.properties	Daily	-	-	10 days
ResourceManager	Main	/opt/mapr/hadoop/hadoop-<version>/logs/yarn-mapr-resourcemanager-<hostname>.log	/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/log4j.properties	By Size	256MB	20	10 days
	stdout on startup	/opt/mapr/hadoop/hadoop-<version>/logs/yarn-mapr-resourcemanager-<hostname>.out	/opt/mapr/hadoop/hadoop-<version>/sbin/yarn-daemon.sh	Every Startup	-	5	10 days
	RM Volume Initialization	/opt/mapr/logs/createRMVolume.log	/opt/mapr/server/createJTVolume.sh	-	-	-	10 days
MAST Gateway	Main	/opt/mapr/logs/mastgateway.log	/opt/mapr/conf/mastgateway.conf	By Size			

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
NodeManager	Main	/opt/mapr/hadoop/hadoop-<version>/logs/yarn-mapr-nodemanager-<hostname>.log	/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/log4j.properties	By Size	256MB	20	10 days
	stdout on startup	/opt/mapr/hadoop/hadoop-<version>/logs/yarn-mapr-nodemanager-<hostname>.out	/opt/mapr/hadoop/hadoop-<version>/sbin/yarn-daemon.sh	Every Startup	-	5	10 days
	NM Volume Initialization	/opt/mapr/logs/createNMVolume.<uid>.log	/opt/mapr/server/createTTVolume.sh	-	-	-	10 days
	NM Volume Initialization	/opt/mapr/logs/createNMVolume.<uid>.cmd.out	/opt/mapr/server/createTTVolume.sh	-	-	-	10 days
HistoryServer	Main	mapred-mapr-historyserver-<hostname>.log	/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/log4j.properties	By Size	256MB	20	10 days
	stdout on startup	mapred-mapr-historyserver-<hostname>.out	/opt/mapr/hadoop/hadoop-<version>/sbin/mr-jobhistory-daemon.sh	Every Startup	-	5	10 days
ZooKeeper	Main	/opt/mapr/zookeeper/zookeeper-3.4.11/logs/zookeeper.log	/opt/mapr/zookeeper/zookeeper-3.4.11/conf/log4j.properties	By Size	10MB	4	-
	stdout on startup	/opt/mapr/zookeeper/zookeeper-3.4.11/logs/zookeeper.out	/opt/mapr/zookeeper/zookeeper-3.4.11/bin/zkServer.sh	-	-	-	-
	Cleanup	/opt/mapr/zookeeper/zookeeper-3.4.11/logs/zookeepercleanup.log	/opt/mapr/zookeeper/zk_cleanup.sh	-	-	-	-

last-updated: Apr 11, 2022

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
Hive	Main	/opt/mapr/hive/hive-<version>/logs/<username>/hive.log	/opt/mapr/hive/hive-<version>/conf/hive-log4j.properties	Daily	-	-	-
	Execution	/opt/mapr/hive/hive-<version>/logs/<username>/<queryid>.log	/opt/mapr/hive/hive-<version>/conf/hive-exec-log4j.properties	-	-	-	-
	HS2 stdout on startup	/opt/mapr/hive/hive-<version>/logs/hive-mapr-hiveserver2-<hostname>.out	/opt/mapr/hive/hive-<version>/bin/ext/hiveserver2.sh	-	-	-	-
	Metastore stdout on startup	/opt/mapr/hive/hive-<version>/logs/hive-mapr-metastore-<hostname>.out	/opt/mapr/hive/hive-<version>/bin/ext/metastore.sh	-	-	-	-
	WebHCat	/opt/mapr/hive/hive-<version>/logs/<username>/webhcat/webhcat.log	/opt/mapr/hive/hive-<version>/hcatalog/etc/webhcat/webhcat-log4j.properties	Daily	-	-	-
	WebHCat stdout on startup	/opt/mapr/hive/hive-<version>/logs/<username>/webhcat/webhcat-console.log	/opt/mapr/hive/hive-<version>/hcatalog/sbin/webhcat_server.sh	-	-	-	-
	WebHCat stderr on startup	/opt/mapr/hive/hive-<version>/logs/<username>/webhcat/webhcat-console-error.log	/opt/mapr/hive/hive-<version>/hcatalog/sbin/webhcat_server.sh	-	-	-	-
	Beeline	<stderr>	/opt/mapr/hive/hive-<version>/conf/beeline-log4j.properties	-	-	-	-
	History	/tmp/<username>/hive_job_log	/opt/mapr/hive/hive-<version>/conf/	-	-	-	<tmpwatch>

last-updated: Apr 11, 2022

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
HBase	Shell	/opt/mapr/hbase/hbase-<version>/logs/hbase-<username>-shell-<hostname>.log	/opt/mapr/hbase/hbase-<version>/conf/log4j.properties	Daily	-	-	-
	REST	/opt/mapr/hbase/hbase-<version>/logs/hbase-<username>-rest-<hostname>.log	/opt/mapr/hbase/hbase-<version>/conf/log4j.properties	By Size	256MB	20	-
	REST stdout on startup	/opt/mapr/hbase/hbase-<version>/logs/hbase-<username>-rest-<hostname>.out	/opt/mapr/hbase/hbase-<version>/bin/hbase-daemon.sh	Every Startup	-	5	-
	Thrift	/opt/mapr/hbase/hbase-<version>/logs/hbase-<username>-thrift-<hostname>.log	/opt/mapr/hbase/hbase-<version>/conf/log4j.properties	By Size	256MB	20	-
	Thrift stdout on startup	/opt/mapr/hbase/hbase-<version>/logs/hbase-<username>-thrift-<hostname>.out	/opt/mapr/hbase/hbase-<version>/bin/hbase-daemon.sh	Every Startup	-	5	-
SparkHistory Server	Main	<stderr>	/opt/mapr/spark/spark-<version>/conf/log4j.properties	-	-	-	-
	stdout on startup	/opt/mapr/spark/spark-<version>/logs/spark-mapr-org.apache.spark.deploy.history.HistoryServer-1-<hostname>.out	/opt/mapr/spark/spark-<version>/sbin/spark-daemon.sh	Every Startup	-	5	-

last-updated: Apr 11, 2022

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
Impala	Server	/opt/mapr/impala/impala-<version>/logs/impalad.INFO	/opt/mapr/impala/impala-<version>/mapr/conf/env.sh	Every Startup	-	-	-
	Server	/opt/mapr/impala/impala-<version>/logs/impalad.WARNING	/opt/mapr/impala/impala-<version>/mapr/conf/env.sh	Every Startup	-	-	-
	Server	/opt/mapr/impala/impala-<version>/logs/impalad.ERROR	/opt/mapr/impala/impala-<version>/mapr/conf/env.sh	Every Startup	-	-	-
	Server stdout on startup	/opt/mapr/impala/impala-<version>/logs/impalaserver.out	/opt/mapr/impala/impala-<version>/mapr/warden/warden_helper	-	-	-	-
	State Store	/opt/mapr/impala/impala-<version>/logs/statestored.INFO	/opt/mapr/impala/impala-<version>/mapr/conf/env.sh	Every Startup	-	-	-
	State Store	/opt/mapr/impala/impala-<version>/logs/statestored.WARNING	/opt/mapr/impala/impala-<version>/mapr/conf/env.sh	Every Startup	-	-	-
	State Store	/opt/mapr/impala/impala-<version>/logs/statestored.ERROR	/opt/mapr/impala/impala-<version>/mapr/conf/env.sh	Every Startup	-	-	-
	State Store stdout on startup	/opt/mapr/impala/impala-<version>/logs/impalastore.out	/opt/mapr/impala/impala-<version>/mapr/warden/warden_helper	-	-	-	-
	Catalog	/opt/mapr/impala/impala-<version>/logs/catalogd.INFO	/opt/mapr/impala/impala-<version>/mapr/conf/env.sh	Every Startup	-	-	-
	Catalog	/opt/mapr/impala/impala-<version>/logs/catalogd.WARNING	/opt/mapr/impala/impala-<version>/mapr/conf/env.sh	Every Startup	-	-	-

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
Drill	Drillbit	/opt/mapr/drill/drill-<version>/logs/drillbit.log	/opt/mapr/drill/drill-<version>/conf/logback.xml	By Size	100MB	10	-
	Drillbit stdout on startup	/opt/mapr/drill/drill-<version>/logs/drillbit.out	/opt/mapr/drill/drill-<version>/bin/drillbit.sh	-	-	-	-
	Drillbit Query	/opt/mapr/drill/drill-<version>/logs/drillbit_queries.json	/opt/mapr/drill/drill-<version>/conf/logback.xml	By Size	100MB	10	-
	Sqlline	/opt/mapr/drill/drill-<version>/logs/sqlline.log	/opt/mapr/drill/drill-<version>/conf/logback.xml	By Size	100MB	10	-
	Sqlline Query	/opt/mapr/drill/drill-<version>/logs/sqlline_queries.json	/opt/mapr/drill/drill-<version>/conf/logback.xml	By Size	100MB	10	-
	Submitter	/opt/mapr/drill/drill-<version>/logs/submitter.log	/opt/mapr/drill/drill-<version>/conf/logback.xml	By Size	100MB	10	-
	Submitter Query	/opt/mapr/drill/drill-<version>/logs/submitter_queries.json	/opt/mapr/drill/drill-<version>/conf/logback.xml	By Size	100MB	10	-
	Dumpcat	/opt/mapr/drill/drill-<version>/logs/drill_dumpcat.log	/opt/mapr/drill/drill-<version>/conf/logback.xml	By Size	100MB	10	-
	Query Plan	/opt/mapr/drill/drill-<version>/logs/profiles/<queryid>.sys.drill	-	-	-	-	-
Flume	Main	/opt/mapr/flume/flume-<version>/logs/flume.log	/opt/mapr/flume/flume-<version>/conf/log4j.properties	By Size	100MB	10	-

last-updated: Apr 11, 2022

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
Oozie	Main	/opt/mapr/oozie/oozie-<version>/logs/oozie.log	/opt/mapr/oozie/oozie-<version>/conf/oozie-log4j.properties	Hourly	-	720	-
	JPA	/opt/mapr/oozie/oozie-<version>/logs/oozie-jpa.log	/opt/mapr/oozie/oozie-<version>/conf/oozie-log4j.properties	Daily	-	-	-
	Operations	/opt/mapr/oozie/oozie-<version>/logs/oozie-ops.log	/opt/mapr/oozie/oozie-<version>/conf/oozie-log4j.properties	Daily	-	-	-
	Instrumentation	/opt/mapr/oozie/oozie-<version>/logs/oozie-instrumentation.log	/opt/mapr/oozie/oozie-<version>/conf/oozie-log4j.properties	Daily	-	-	-
	Audit	/opt/mapr/oozie/oozie-<version>/logs/oozie-audit.log	/opt/mapr/oozie/oozie-<version>/conf/oozie-log4j.properties	Daily	-	-	-
	Tomcat	/opt/mapr/oozie/oozie-<version>/logs/catalina.<date>.log	/opt/mapr/oozie/oozie-<version>/oozie-server/conf/logging.properties	Daily	-	-	-
	Tomcat Application	/opt/mapr/oozie/oozie-<version>/logs/localhost.<date>.log	/opt/mapr/oozie/oozie-<version>/oozie-server/conf/logging.properties	Daily	-	-	-
	Tomcat Manager	/opt/mapr/oozie/oozie-<version>/logs/manager.<date>.log	/opt/mapr/oozie/oozie-<version>/oozie-server/conf/logging.properties	Daily	-	-	-
	Tomcat Host Manager	/opt/mapr/oozie/oozie-<version>/logs/host-manager.<date>.log	/opt/mapr/oozie/oozie-<version>/oozie-server/conf/logging.properties	Daily	-	-	-

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
Hue	CherryPy Server	/opt/mapr/hue/hue-<version>/logs/runcpserver.log	/opt/mapr/hue/hue-<version>/desktop/conf/log.conf	By Size	1MB	3	-
	CherryPy Server stdout	/opt/mapr/hue/hue-<version>/logs/hue-<username>-runcpserver-<hostname>.out	/opt/mapr/hue/hue-<version>/bin/hue.sh	-	-	-	-
	Livy Server	/opt/mapr/hue/hue-<version>/logs/livy_server.log	/opt/mapr/hue/hue-<version>/desktop/conf/log.conf	By Size	1MB	3	-
	Livy Server stdout	/opt/mapr/hue/hue-<version>/logs/hue-<username>-livy_server-<hostname>.out	/opt/mapr/hue/hue-<version>/bin/hue.sh	-	-	-	-
	Access	/opt/mapr/hue/hue-<version>/logs/access.log	/opt/mapr/hue/hue-<version>/desktop/conf/log.conf	By Size	1MB	3	-
	Error	/opt/mapr/hue/hue-<version>/logs/error.log	/opt/mapr/hue/hue-<version>/desktop/conf/log.conf	By Size	1MB	3	-
	Security	/opt/mapr/hue/hue-<version>/logs/secure-sh-log.out	/opt/mapr/hue/hue-<version>/bin/hue.sh	-	-	-	-

last-updated: Apr 11, 2022

Component	Type of Log	Log File	Configuration File	Rotation	Maximum File Size	# of Backups	Expiration Period
HttpFs	Main	/opt/mapr/httpfs/httpfs-<version>/logs/httpfs.log	/opt/mapr/httpfs/httpfs-<version>/etc/hadoop/httpfs-log4j.properties	Daily	-	-	-
	Audit	/opt/mapr/httpfs/httpfs-<version>/logs/httpfs-audit.log	/opt/mapr/httpfs/httpfs-<version>/etc/hadoop/httpfs-log4j.properties	Daily	-	-	-
	Tomcat	/opt/mapr/httpfs/httpfs-<version>/logs/httpfs-catalina.log	/opt/mapr/httpfs/httpfs-<version>/share/hadoop/httpfs/tomcat/conf/logging.properties	Daily	-	-	-
	Tomcat Application	/opt/mapr/httpfs/httpfs-<version>/logs/httpfs-localhost.log	/opt/mapr/httpfs/httpfs-<version>/share/hadoop/httpfs/tomcat/conf/logging.properties	Daily	-	-	-
	Tomcat Manager	/opt/mapr/httpfs/httpfs-<version>/logs/httpfs-manager.log	/opt/mapr/httpfs/httpfs-<version>/share/hadoop/httpfs/tomcat/conf/logging.properties	Daily	-	-	-
	Tomcat Host Manager	/opt/mapr/httpfs/httpfs-<version>/logs/httpfs-host-manager.log	/opt/mapr/httpfs/httpfs-<version>/share/hadoop/httpfs/tomcat/conf/logging.properties	Daily	-	-	-
	Tomcat stdout on startup	/opt/mapr/httpfs/httpfs-<version>/logs/httpfs-catalina.out	/opt/mapr/httpfs/httpfs-<version>/share/hadoop/httpfs/tomcat/bin/catalina.sh	-	-	-	-

Increasing Log Retention

To increase log retention for specific component, you can modify the configuration file and reset the value of the properties.

Increasing Log Retention for MapR File System

In the `/opt/mapr/conf/mfs.conf` file, increase the value for the `mfs.max.logfile.size.in.mb` property. The value for this property is computed using the following formula:

```
maxSizePerLogFile = maxLogSize / MAX_NUM_OF_LOG_FILES
```

Here:

- `maxLogSize` specifies the total amount of space that MapR File System log files can consume.
- `MAX_NUM_OF_LOG_FILES` specifies the total number of MapR File System log files (5 hard coded).

For example, for a value of 10 GB, there will be 5 log files (`mfs.log-3` to `mfs.log-3.4`) of 2GB each.

Increasing Log Retention for CLDB

In the `/opt/mapr/conf/log4j.cldb.properties` file, modify the value for the following properties:

- `log4j.appender.R.MaxFileSize` — specifies the size of each `cldb.log` before rolling over
- `log4j.appender.R.MaxBackupIndex` — specifies the number of `cldb.log*` before the oldest one is deleted.

For example, suppose the following configuration:

- `log4j.appender.R.MaxFileSize` = 1024MB
- `log4j.appender.R.MaxBackupIndex` = 10

The CLDB log files will grow to 10GB before they are purged.

Note: This setting does not impact audit logging.

Increasing Log Retention for Hadoop Services

You can increase log retention for ResourceManager, NodeManager, HistoryServer, and TimelineServer by modifying the following properties in the `/opt/mapr/hadoop/hadoop-<Version>/etc/hadoop/log4j.properties` file:

- `hadoop.log.maxfilesize` — specifies the size of each `<service-name>-hostname>.log` before rolling over.
- `hadoop.log.maxbackupindex` — specifies the number of `<service-name>-<hostname>.log*` before the oldest one is deleted.

For example, suppose the following configuration:

- `hadoop.log.maxfilesize` = 1024Mb
- `hadoop.log.maxbackupindex` = 10

There will be 10 GB of total service-specific logs (1GB per file) before the oldest file is purged.

Setting the Tracing Level

last-updated: Apr 11, 2022

To check all the modules and their current logging levels, run the following command:

```
maprcli trace info
```

To set the tracing level for a module, run the `maprcli trace setlevel` command. For example:

```
maprcli trace setlevel -module FuseMonitor -level DEBUG
```

Configuring Profiling for Operations

To check the amount of time it took to complete each operation (from the time of submission), enable profiling for client RPC and MapR File System operations. To enable profiling for:

- Client RPC, run the following command:

  ```
  fcdebug -s <shmid> -m ClntProfileRpc -1 DEBUG
  ```

  Tip: For more information, see `fcdebug`.

  Enabling profiling for the client RPC will allow you to determine, for each RPC, the amount of time it took to receive a response after submitting the request.

- MapR File System operations, run the following command:

  ```
  maprcli trace setlevel -module FSProfile -level debug
  ```

  Tip: For more information, see `maprcli`.

  Enabling profiling for MapR File System operations will allow you to determine the amount of time it took MapR File System to process each operation.

By default, profiling is disabled for both client RPC and MapR File System operations. Once enabled, the log for:

- Client RPC should look similar to the following:

  ```
  2016-06-16 10:58:04,6404 DEBUG ClntProfileRpc
  fs/client/fileclient/cc/client.cc:3483 Thread: 32188 Profile: CltRpcDone:
  server 10.10.100.196:5692 took 1 msec error 0 FID 2125.34.262486  Getattr

  2016-06-16 11:13:55,7480 DEBUG ClntProfileRpc
  fs/client/fileclient/cc/client.cc:3202 Thread: 32161 Profile: CltRpcDone:
  server 10.10.100.196:5692 took 1 msec error 0 FID 2125.16.2  PathWalkPlus
  path
  abc
  ```

- MapR File System should look similar to the following:

```
2016-06-19 15:51:05,0231 DEBUG FSProfile unlink.cc:2456 OP unlink:
localTm 34
elapsedTm 34 client 10.10.100.196 err 0 PFID: 1.32.131398 name
unreachableFSIdTable itype Regular

2016-06-19 15:51:11,0249 DEBUG FSProfile writev3.cc:1296 OP Write:
localTm 13
elapsedTm 13 client 10.10.100.196 err 0 FID: 2121.532.2364014 off 29234
count
1424

2016-06-19 15:51:08,1184 DEBUG FSProfile readdir.cc:505 OP ReadDir:
elapsedTm
28 client 10.10.100.196 err 0 FID: 2121.16.2 Isplus true
```

For example, to check the time it took for a read RPC, run the following command:

```
# cat /opt/mapr/logs/ffs.log* | grep -nrui "CltRpcDone" | grep -nrui "read"
| less
1009:1014:2016-06-16 11:21:51,8203 DEBUG ClntProfileRpc
fs/client/fileclient/cc/client.cc:4900 Thread: 32151 Profile: CltRpcDone:
server10.10.100.196:5660 took 0 msec for Proc Read error 0 FID
2182.32.131232
off 7143424 len 131072 name 2125.34.262486
```

Archiving CLDB Logs

The CLDB logs can be archived by setting the value for the configuration parameter, `cldb.logarchiver.enabled`, using the maprcli config save command. The value can be:

- 0 — disable
- 1 — enable

To:

- Enable archiving, run the following command:

```
maprcli config save -values '{"cldb.logarchiver.enabled":"1"}'
```

- Disable archiving, run the following command:

```
maprcli config save -values '{"cldb.logarchiver.enabled":"0"}'
```

The default value for this parameter is 2, which indicates that the CLDB log archiving is disabled; but on clusters with 50 or more MapR File System nodes, the CLDB log archiving will be automatically enabled unless the value is explicitly set to 0.

If/when archiving is enabled:

- All static CLDB log files, except the active `cldb.log` file, are periodically scanned and archived in `/var/mapr/cldblog/<hostname>` directory.

- The filename for the archived log file is autogenerated based on the date and timestamp on the first log line in the file chosen for archival.

  For example, suppose a log file with the following first line:

  ```
  2017-04-06 12:42:16,020 INFO CLDB [main]: Loading properties file : /opt/
  mapr/conf/cldb.conf
  ```

  The archived log filename in `/var/mapr/cldblog/<hostname>` directory will be:
  `2017-04-06_12.42.16.020`

Enabling Runtime Logging

To enable logging at runtime for the file client (`libfsalmapr.so` library), run the `fcdebug` utility:

```
/opt/mapr/server/tools/fcdebug -s <shmid> -m <module> [-l <level>]
/opt/mapr/server/tools/fcdebug -i -s <shmid>
```

Before running this command, make a note of the following:

- If necessary, run `maprcli trace info` to retrieve the list of modules.

  The default value for module is all.

- Level should be one of FATAL, ERROR, WARN, INFO, DEBUG.

  If level is not specified, default level is applied for the module.

- The `-i` lists the current debug level of all modules.

- The `shmid` is available in the `fsal.log-*` files when `libMapRClient` is loaded.

  For example, the first line in the log file is something similar to the following:

  ```
  2017-02-13 11:56:32,6809 ERROR FuseAPI fs/client/fileclient/cc/
  fuse_api.cc:1371
  Thread: 428 Shmid to be used by fcdebug 512720897
  ```

  This `shmid` can be used with `fcdebug`.

Note: Run `fcdebug` once for every library (every library has a separate shared memory). The `shmids` can be found in the respective `fsal` log files.

See also:

- Enabling Debug Logging for NFSv3 on page 1226
- Enable Debug Logging for NFSv4 on page 1228

Viewing Audit Logs

The following sections describe audit logs for execution of any maprcli command, REST API call, or action in Control System, and audit logs for cluster administration, filesystem, table, and stream operations.

Viewing Log Entries for Audited maprcli Command Executions
Describes where audit records of operations performed using the CLI are stored and how to view them.

The execution of any `maprcli` command on the cluster is logged in the local filesystem on the node on which the execution happened. The log file is `/opt/mapr/mapr-cli-audit-log/audit.log.json`. Auditing of CLI operations is always enabled, whether or not auditing is enabled for cluster-level operations with the `maprcli audit cluster` command.

Typical log entries provide a timestamp of the execution, the UID of the user who ran the command, the IP address from which the user ran the command, the command itself, and the status of the execution. Status codes are 0 for success and 1 for failure. The error messages field provides the reasons for failures.

Below are some typical log entries:

```
{"timestamp":
{"$date":"2015-06-15T11:45:56.434Z"},"uid":2147483632,"ipAddress":
"10.10.20.12","command":"volume info","arguments":
{"name":"mapr.opt"},"status":
1,"errors": ["Volume lookup of mapr.opt failed, No such volume"]}
{"timestamp":
{"$date":"2015-06-15T11:49:34.434Z"},"uid":2147483632,"ipAddress":
"10.10.20.12","command":"alarm add","arguments":{"baseService":"1","alarm":
"NODE_ALARM_SERVICE_GATEWAY_DOWN","service":"gateway","displayName":"Gateway
ServiceDown",
"serviceName":"GatewayService","terse":"nagwsd"},"status":1,"errors":
["Terse name of
nagwsd already exists in the system.","Alarm
NODE_ALARM_SERVICE_GATEWAY_DOWN already
exists in the system."]}
{"timestamp":
{"$date":"2015-06-15T11:49:52.598Z"},"uid":2147483632,"ipAddress":
"10.10.20.12","command":"volume create","arguments":
{"name":"mapr.hbase","path":"/hbase",
"replicationtype":"low_latency"},"status":1,"errors":["Volume Name
mapr.hbase, Already In Use"]}
```

Viewing Audit Logs for Cluster Administration
Describes where audit records of cluster administration operations are stored and how to view them.

Entries for audit logs are initially held in memory until 128 operations have been logged or 10 seconds have elapsed, whichever happens first. At that point, the new log entries are flushed to disk.

Audit logs are in JSON format, so they can be queried by Drill or processed by other third-party tools or your own scripts.

Audit logs are readable only by the `mapr` and `root` users on the cluster where the logs are located. These users can also copy and delete audit logs.

The `status` field in every log entry shows the status of the attempted operation. The status codes are taken from the Linux `errno.h` file. For a list of these codes, see Status Codes That Can Appear in Audit Logs.

Audit logs use Coordinated Universal Time (UTC) in the records of audited operations.

The cleanup of old audit log files is handled by Warden either when they are older than 10 days (the default retention time) or when they are older than the number of days set for the `log.retention.time` parameter in the `/opt/mapr/conf/warden.conf` file. To prevent Warden from removing the log files, by default, `cldbaudit*` and `authaudit*` are listed under the `log.retention.exceptions` parameter in the `warden.conf` file.

To enable Warden to automatically cleanup log files, remove `cldbaudit*` and `authaudit*` from the `log.retention.exceptions` parameter in the `warden.conf` file and, if you want a shorter cleanup time, set the value for `log.retention.time` parameter in the `warden.conf` file. The value for `log.retention.time` must be specified in milliseconds.

To disable all exceptions, comment out the `log.retention.exceptions` parameter, that is, `#log.retention.exceptions`. When this parameter is null, that is, `log.retention.exceptions=`, no files are picked for log cleanup.

Viewing Audit Logs for File System, Table, and Stream Operations
Describes where MapR File System, MapR Database, and MapR Event Store For Apache Kafka audit logs are stored and how to view them.

Operations on data-fabric file, database, and event data are captured and recorded in the audit logs. The operations take place within volumes and have effects at the level of the file system.

These audit logs are stored in a system volume created specifically to store them. This volume is created automatically during cluster installations and upgrades. Operations are logged on the nodes on which the operations are executed, which could differ from the nodes where operations are initiated. Logs are stored in the file system at `/var/mapr/local/<node_name>/audit/`. By default, only root and the cluster administrator (typically `mapr`) can read the log files. To allow other users to read the logs, set ACEs on the directory granting `readfile` (`rf`), `readdir` (`rd`), and `lookupdir` (`ld`) permissions to the users. For example:

```
~# hadoop mfs -setace -R -aces "rf:u:root|u:mapr|u:m7user1,rd:u:root|u:mapr|
u:m7user1,ld:u:root|u:mapr|u:m7user1" /var/mapr/local/sample.qa.lab/audit/
```

Note: For more information, see Enabling Volume, Directory, and File Authorizations with ACEs on page 1452.

Audit logs for operations on directories and files

Operations on directories and files, as well as the deletion of MapR Database tables, are logged in files that have this naming convention: `FSAudit.log.json-dd-mm-yyyy-<001-999>`

To see what information is recorded in typical log entries, see Example Log Entries for Audited File System Operations.

Audit logs for operations on MapR Database tables and MapR streams

All operations on MapR Database tables and streams are logged in files that have this naming convention: `DBAudit.log.json-dd-mm-yyy-<001-999>`

Operations that result from maprcli commands, REST calls, or activity in MCS are also logged in `/opt/mapr/mapr-cli-audit-log/audit.log.json` in the local file system on the nodes where the operations are processed.

To see what information is recorded in typical log entries, see Example Log Entries for Audited Operations on MapR Database Tables.

Note: Due to the way that the creation of tables is processed internally, sometimes the creation of tables is logged in `FSAudit.log.json`, rather than in `DBAudit.log.json`.

Common Features of Audit Logs for File System, Table, and Stream Operations

Entries for audit logs are initially held in memory until 128 operations have been logged or 10 seconds have elapsed, whichever happens first. At that point, the new log entries are flushed to disk, depending on the *coalesce* interval.

The coalesce interval represents the interval of time during which READ, WRITE, or GETATTR operations on one file from one client IP address and UID/GID are logged only once for a particular operation, if auditing is enabled.

For example, suppose that a client application reads a single file three times in 6 minutes, so that there is one read at 0 minutes, another at 3 minutes, and a final read at 6 minutes. If the coalesce interval is at least 6 minutes, then only the first read operation is logged. However, if the interval is between 4 minutes, then only the first and third read operations are logged. If the interval is 2 minutes, all three read operations are logged.

Now however, if the client was also writing to the file, irrespective of the coalesce interval for the read operation in the example stated previously, the write operation is logged, as it is a different operation from reading.

The default value is 60 minutes. Setting this field to a larger number helps prevent audit logs from growing quickly. To change the coalesce interval, see volume audit on page 1922.

Audit logs are in JSON format, so they can be queried by Drill or processed by other third-party tools or your own scripts.

Audit logs are readable only by the `mapr` and `root` users on the cluster where the logs are located. These users can also copy and delete audit logs.

The status field in every log entry shows the status of the attempted operation. The status codes are taken from the Linux `errno.h` file. For a list of these codes, see Status Codes That Can Appear in Audit Logs.

Audit logs use Coordinated Universal Time (UTC) in the records of audited operations.

When operations are performed on directories, files, or tables that are being audited, the full names for those objects, as well as the current volume and the name of the user performing the operation, are not immediately available to the auditing feature. What are immediately available are IDs for those objects and users. Converting IDs to names at run-time would be costly for performance. Therefore, audit logs contain file identifiers (FIDs) for directories, files, and tables; volume identifiers for volume; and user identifiers (UIDs) for users.

You can resolve identifiers into names by using the expandaudit utility. This utility creates a copy of the log files for a specified volume, and in that copy are the names of the file system objects, users, and volumes that are in the audit log records. You can then query or process the copy.

A sample of the logs is as follows:

```
{"timestamp":
{"$date":"2021-07-14T13:05:01.506Z"},"resource":"test-audit-logs","operation
":"volumeMirrorPermCheck","username":"root","uid":0,"clientip":"10.163.167.2
14","status":0}
{"timestamp":
{"$date":"2021-07-14T08:44:01.553Z"},"resource":"255","operation":"volumeLoo
kup","username":"root","uid":0,"clientip":"10.163.167.214","status":2}
```

Note: There will be an entry in the audit log for each IP address on a node. For example, suppose there is a node with multiple IP addresses. The audit log on this node may show multiple entries of the same operation, each associated with a different IP address.

Note: The number of bytes read or written is not recorded.

Example Log Entries for Audited File System Operations

When auditing of file system operations is enabled at the cluster level, volume level, and file system level, each operation on a directory or file is logged on the node on which the operation was initiated.

Typical log entries provide a timestamp of the operation, the type of operation, the UID of the user who ran the command, the IP address from which the user ran the command, identifiers of the affected resources, the volume identifier, and the status of the operation. Status codes come from the Linux `errno.h` file. For a list of these codes, see Status Codes That Can Appear in Audit Logs.

Note: Due to the way that the creation of tables is processed internally, sometimes the creation of tables is logged in `FSAudit.log.json`, rather than in `DBAudit.log.json`.

Below are some typical log entries:

```
{"timestamp":
{"$date":"2015-06-06T10:44:22.800Z"},"operation":"MKDIR","uid":0,"ipAddress"
:
```


last-updated: Apr 11, 2022

```
"10.10.104.51","parentFid":"2049.51.131248","childFid":"2049.56.131258","chi
ldName":
"ycsbTmp_1433587462796","volumeId":68048396,"status":0}
{"timestamp":
{"$date":"2015-06-06T10:44:22.823Z"},"operation":"LOOKUP","uid":0,"ipAddress
":
"10.10.105.51","srcFid":"2049.56.131258","srcName":"range0","volumeId":68048
396,"status":2}
{"timestamp":
{"$date":"2015-06-06T10:44:22.824Z"},"operation":"CREATE","uid":0,"ipAddress
":
"10.10.104.51","parentFid":"2049.56.131258","childFid":"2049.57.131260","chi
ldName":"range0"
,"volumeId":68048396,"status":0}
{"timestamp":
{"$date":"2015-06-06T10:44:22.838Z"},"operation":"WRITE","uid":0,"ipAddress"
:
"10.10.105.51","srcFid":"2049.57.131260","volumeId":68048396,"status":0}
{"timestamp":
{"$date":"2015-06-06T10:44:48.628Z"},"operation":"READ","uid":0,"ipAddress":
"10.10.105.51","srcFid":"2049.63.131272","volumeId":68048396,"status":0}
```

To convert the user IDs to usernames, file identifiers to pathnames, and volume IDs to volume names, run the expandaudit on page 2096 utility. For example, here are the same audit records after they were processed by this utility:

```
{"timestamp":"{$date=2015-06-06T10:44:22.800Z}","operation":"MKDIR","user":"
root","uid":"0","ipAddress":
"10.10.104.51","parentPath":"/ycsb1433587356934","childPath":"/
ycsb1433587356934/ycsbTmp_1433587462796",
"childName":"ycsbTmp_1433587462796","VolumeName":"mapr.cluster.root","volume
Id":"68048396","status":"0"}
{"timestamp":"{$date=2015-06-06T10:44:22.823Z}","operation":"LOOKUP","user":
"root","uid":"0","ipAddress":
"10.10.105.51","srcPath":"/ycsb1433587356934/
ycsbTmp_1433587462796","srcName":"range0","VolumeName":
"mapr.cluster.root","volumeId":"68048396","status":"2"}
{"timestamp":"{$date=2015-06-06T10:44:22.824Z}","operation":"CREATE","user":
"root","uid":"0","ipAddress":
"10.10.104.51","parentPath":"/ycsb1433587356934/
ycsbTmp_1433587462796","childPath":
"/ycsb1433587356934/ycsbTmp_1433587462796/
range0","childName":"range0","VolumeName":"mapr.cluster.root",
"volumeId":"68048396","status":"0"}
{"timestamp":"{$date=2015-06-06T10:44:22.838Z}","operation":"WRITE","user":"
root","uid":"0","ipAddress":
"10.10.105.51","srcPath":"/ycsb1433587356934/ycsbTmp_1433587462796/
range0","VolumeName":"mapr.cluster.root",
"volumeId":"68048396","status":"0"}
{"timestamp":"{$date=2015-06-06T10:44:48.628Z}","operation":"READ","user":"r
oot","uid":"0","ipAddress":
"10.10.105.51","srcPath":"/ycsb1433587356934/ycsbTmp_1433587462796/
range6","VolumeName":"mapr.cluster.root",
"volumeId":"68048396","status":"0"}
```

Example Log Entries for Audited Operations on MapR Database Binary and JSON Tables

When auditing of table operations is enabled at the cluster level, volume level, and file system level, each operation on a table is logged on the node where the operation was executed, which could differ from the node where the operation was initiated.

Typical log entries provide a timestamp of the operation, the type of operation, the UID of the user who ran the command, the IP address from which the user ran the command, identifiers of the affected resources,

and the status of the operation. Fields such as “ColumnFamily” and “Column” for some operations are also included when applicable. Row keys are not included. Status codes come from the Linux `errno.h` file. For a list of these codes, see Status Codes That Can Appear in Audit Logs.

Note: Due to the way that the creation of tables is processed internally, sometimes the creation of tables is logged in `FSAudit.log.json`, rather than in `DBAudit.log.json`.

Note: Audit logs do not display the indices of array elements when there are put or update operations on arrays that are in documents within JSON tables.

Below are some typical log entries:

```
{"timestamp":
{"$date":"2015-06-06T11:31:02.621Z"},"operation":"DB_GET","uid":0,"ipAddress
":
"10.10.105.51","volumeId":48210891,"tableFid":"2751.77.131402","status":0}
{"timestamp":
{"$date":"2015-06-06T11:31:02.623Z"},"operation":"DB_SCAN","uid":0,"ipAddres
s":
"10.10.104.51","volumeId":48210891,"tableFid":"2751.77.131402","status":0}
{"timestamp":
{"$date":"2015-06-06T11:31:02.624Z"},"operation":"DB_PUT","uid":0,"ipAddress
":
"10.10.104.51","volumeId":48210891,"columnFamily":"cf0","columnQualifier":"c
0","tableFid":
"2751.77.131402","status":0}
```

To convert the user IDs to usernames, file identifiers to pathnames, and volume IDs to volume names, run the expandaudit utility. For example, here are the same audit records after they were processed by this utility:

```
{"timestamp":
{"$date":"2015-06-06T11:31:02.621Z"},"operation":"DB_GET","uid":0,"ipAddress
":
"10.10.105.51","VolumeName":"mapr.cluster.root","volumeId":48210891,"tablePa
th":
"/ycsb1433588330006/ycsbTable0","status":0}
{"timestamp":"{$date=2015-06-06T11:03:16.721Z}","operation":"DB_SCAN","user"
:"root","uid":
"0","ipAddress":"10.10.105.51","VolumeName":"mapr.cluster.root","volumeId":"
48210891","tablePath":
"/ycsb1433588330006/ycsbTable0","status":"0"}
{"timestamp":
{"$date":"2015-06-06T11:31:02.624Z"},"operation":"DB_PUT","uid":0,"ipAddress
":
"10.10.104.51","VolumeName":"mapr.cluster.root","volumeId":48210891,"columnF
amily":"cf0",
"columnQualifier":"c0","tablePath":"/ycsb1433588330006/
ycsbTable0","status":0}
```

Managing Audit Logs for File System and Table Operations

There are three parameters that you can use to manage audit logs for file system and table operations:

- `-maxSize`
- `-retention`
- `-coalesce`

You can set the first two parameters with the maprcli audit data command. You can set the third parameter with the maprcli volume audit command.

last-updated: Apr 11, 2022

Effects of the -maxSize parameter

When you enable auditing with the audit data on page 1553`maprcli audit data` command, you can use the `-maxSize` parameter to specify the size at which an alarm is raised concerning the size of the audit volume. The alarm is displayed on the dashboard in the Control System and in the output of the alarm list on page 1545`maprcli alarm list` command. This alarm simply means that the threshold size has been reached. Audited operations are still logged to the audit volume in question.

There are three actions that you can take:

- If you decide that you want to be notified when the audit volume reaches a smaller or larger size, you can change the threshold by running the `maprcli audit data` command and changing the value of the `-maxSize` parameter.
- If you want to try preventing audit log files from growing as quickly as they are, you can change the number of identical operations that are logged within a number of minutes. Run the `maprcli audit data` command and increase the value of the `-coalesce` parameter. This parameter is described subsequently.
- If you are concerned about longer-term space requirements for storing audit log files, you can change the number of days to keep old log files before they are deleted. Run the `maprcli audit data` command and decrease the value of the `-retention` parameter. This parameter is also described below.

Effects of the -retention parameter

When you enable auditing with the `maprcli audit data` command, you can use the `-retention` parameter to specify how many days to keep old log files.

Audit logs are rotated every night at midnight UTC time . The saved audit logs are kept until the retention period expires.

For example, suppose the retention period is 30 days. The node 192.168.10.15 in the volume `/myVolume` contains 30 days of saved log files for file-system operations and the current date is March 30, 2016. The directory `/var/mapr/local/102.168.10.15/audit/` contains these log files:

```
FSAudit.log.json-30-03-2016-001
FSAudit.log.json-29-03-2016-001
FSAudit.log.json-28-03-2016-001
...
FSAudit.log.json-01-03-2016-001
```

Note: If MFS is restarted on the same day, audit logs gets rotated, and new files with convention -002, -003, and so on are created with each restart.

If there is no more disk space for new entries in audit logs, audit logging stops.

If the size of the audit log volume exceeds its quota, an alarm is raised, though logging continues. The alarm is `VOLUME_ALARM_ADVISORY_QUOTA_EXCEEDED`. You can view alarms in the Control System or by running the command `maprcli alarm list`. The default quota is 32 GB.

Effects of the -coalesce parameter

The coalesce on page 6594parameter represents the interval of time during which READ, WRITE, or GETATTR operations on one file from one client IP address and UID/GID are logged only once for a particular operation, if auditing is enabled.

For example, suppose that a client application reads a single file three times in 6 minutes, so that there is one read at 0 minutes, another at 3 minutes, and a final read at 6 minutes. If the coalesce interval is at

least 6 minutes, then only the first read operation is logged. However, if the interval is between 4 minutes, then only the first and third read operations are logged. If the interval is 2 minutes, all three read operations are logged.

Now however, if the client was also writing to the file, irrespective of the coalesce interval for the read operation in the example stated previously, the write operation is logged, as it is a different operation from reading.

The default value is 60 minutes. Setting this field to a larger number helps prevent audit logs from growing quickly.

Status Codes That Can Appear in Audit Logs

In the `status` field in entries in audit logs, numeric codes other than 0 for success can appear. For the list of possible codes, their keyword equivalents, and descriptions, refer to the standard Linux `errno-base.h` and `errno.h` files. The `authaudit.log.json` file contains HTTP status codes. See HTTP Status Codes for more information.

Viewing Application Logs

You can use logs to view the status and analyze the execution of applications in a cluster.

To view the status or to access logs for running applications, you can use the user interface associated with the YARN framework.

- For MapReduce version 2 or non-MapReduce applications, access the ResourceManager user interface from the Control System to view the status and logs for a particular application.

For completed applications, the distributed nature of YARN frameworks can make analyzing the execution of applications difficult because tasks and containers are scattered throughout the cluster. Without centralized or aggregated logging, you must manually access all the log files for a completed application by merging the log details for a particular application across multiple nodes in the cluster. With centralized or aggregated logging, you can can access all the logs for a completed application in a centralized location. However, the steps to access logs for completed applications differ based on the configured logging option.

Logging Options

In a data-fabric cluster, the logging option that you configure defines how the logs are stored and accessed.

- **Centralized logging.** The logs are written to local volumes on the file system.
- **YARN log aggregation.** The logs are written to the local file system and then the container logs from each node are aggregated and stored on the file system.
- **Local logging.** The log files for each job or application are written to the local file system. This is the default behavior for MapReduce version 2 (MRv2) applications and non-MapReduce applications.
  - For MRv2 or other applications that run on YARN, the logs are written to the following directory on the local file system: `/opt/mapr/hadoop/hadoop-2.x.x/logs/userlogs/`

The logging options that you can choose from are determined by the type of jobs or applications that you run:

Type of Job or Application	Available Logging Options
MRv2	• Centralized logging • YARN log aggregation • Local logging (default)

Type of Job or Application	Available Logging Options
YARN applications (non-MapReduce)	• YARN log aggregation • Local logging (default)

If you enable centralized logging for MRv2, the MapReduce applications will use centralized logging, while the other YARN applications in the cluster will use local logging.

Note: Select a logging option that stores the logs on the file system for the following reasons:

- Prevent job or application failures due to a lack of space on the local file system for logs.
- Prevent the loss or inaccessibility of logs due to node failure. Logs stored in a local volume are two-way replicated.

YARN Log Aggregation

The YARN Log Aggregation option aggregates and moves log files for completed applications from the local filesystem to the MapR File System. This allows users to view the entire set of logs for a particular application using the HistoryServer UI or by running the `yarn logs` command.

By default, YARN container logs are not aggregated on the MapR File System. Instead, the logs are retained for 3 hours on the local filesystem before they are deleted. To enable YARN log aggregation or to edit the configuration of YARN log aggregation, you must edit the yarn-site.xml file in the following directory: `/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop/`

Enabling YARN Log Aggregation

To enable YARN log aggregation, add or edit the following properties in `yarn-site.xml`:

- Set the value of the `yarn.log-aggregation-enable` to `true`.
- Configure the `yarn.log.server.url` property to contain the URL of the YARN HistoryServer, which should look like the following:

secure cluster	`https://<historyserver-host>:19890/jobhistory/logs`
non-secure cluster	`http://<historyserver-host>:19888/jobhistory/logs`

- Optional: Set the value of `yarn.nodemanager.remote-app-log-dir` to a location in the MapR filesystem. By default, the location is `maprfs:///tmp/logs`.
- Optional: Set the value of `yarn.nodemanager.remote-app-log-dir-suffix` to the name of the folder that should contain the logs for each user. By default, the folder name is `logs`.

On a non-secure cluster, you also need to add the following property to `/opt/mapr/hadoop/hadoop-2.x/etc/hadoop/yarn-env.sh` on the Node Manager nodes:

```
export MAPR_IMPERSONATION_ENABLED=1
```

Then restart the Node Manager services. This setting enables impersonation for Node Manager processes so that log files can be created with the correct user ownership.

Aggregated logs are owned by the user who runs the job. For example, when user admin runs a job, the logs are stored to `maprfs:///tmp/logs/admin`. If user analyst runs a job, the logs are stored to `maprfs:///tmp/logs/analyst`. If these two users do not share the same UNIX group, they will not be able to see each other's logs.

last-updated: Apr 11, 2022

Note: If centralized logging and YARN log aggregation are enabled, the logs for MapReduce version 2 applications will be managed by Centralized Logging while the logs for non-MapReduce applications will be managed by YARN log aggregation.

Enabling YARN Local-Node Log Aggregation

The steps in this procedure configure log aggregation for NodeManager processes, enabling you to store the logs on nodes (node-local volumes) where the YARN containers are launched.

To enable YARN local-node log aggregation, add or edit the following properties in the `yarn-site.xml` file:

1. Set the value of `yarn.node-local-log-aggregation.enable` to `true`.

   Note: The default setting for YARN Log Aggregation (`yarn.log-aggregation-enable`) should be removed or set to `false` in the `yarn-site.xml` file.

2. Optional: Set the value of `yarn.node-local-log-aggregation.metadata-path` to a location in the system. By default the location is `maprfs:///NM_REMOTE_APP_LOG_DIR/<user>/logsMeta`. `NM_REMOTE_APP_LOG_DIR` should match the `yarn.nodemanager.remote-app-log-dir` property.

   Note: The location should not be an absolute path (the location begins from `/`). In the MapR File System (`maprfs`), the default setting for NM_REMOTE_APP_LOG_DIR is `/tmp/logs`.

3. Optional: Set the value of `node-local-log-aggregation.metadata-filename` to the name of the metafile that should contain the information about containers for each node. By default, the file name is `containers.seq`, so if you use the default paths, the file will be stored at `/tmp/logs/${user}/logsMeta/<appId>/<nodeName>/containers.seq`.

4. Restart the NodeManager and HistoryServer services.

Aggregated logs are owned by the user who runs the job.

Different users cannot see each other's logs. For example, when the data-fabric user **`admin`** runs a job, the logs are stored in `maprfs:///var/mapr/local/<nodeNames>/mapred/nodeManager/logs/admin/<appId>`. If a user **`analyst`** runs a job, the logs are stored in `maprfs:///var/mapr/local/<nodeNames>/mapred/nodeManager/logs/analyst/<appId>`.

Viewing Logs for Completed Applications

With YARN log aggregation, you can use yarn commands or the HistoryServer UI to access logs for completed applications.

View Application Logs from the Command Line

Get the application ID and then view log files for the application.

Get the Application ID

To get the application ID for an application that is in a "running" state, you can run the following command:

```
yarn application -list
```

However, if you run `yarn application -list` after a job completes, the command will not return any information.

last-updated: Apr 11, 2022

To get the application ID for an application in any state (submitted, accepted, or running), run the following command:

```
yarn application -list -appStates ALL
```

The `yarn appication -list` command returns information similar to the following, including the application ID:

```
20/10/19 14:57:51 INFO
client.MapRZKBasedRMFailoverProxyProvi
der: Updated RM address to
node1.cluster.com/192.168.33.11:8032
Total number of applications
(application-types: [] and states:
[SUBMITTED, ACCEPTED, RUNNING]):1
Application-Id Application-Name
Application-Type User Queue State
Final-State Progress Tracking-URL
application_1603118361219_0002
QuasiMonteCarlo MAPREDUCE mapr
root.mapr ACCEPTED UNDEFINED 0% N/A
```

View Application Logs

Run the following command to view log files for an application:

```
yarn logs -applicationId
<application-ID>

//Example: yarn logs -applicationId
application_1603118361219_0002
```

Using UI to View Logs for Completed Applications

Explains how to view HistoryServer logs using the graphical interface.

You can view the logs for completed applications through the Control System and by directly accessing the HistoryServer UI.

Using the Control System to View the HistoryServer Logs for Completed Applications

1. Log on to the Control System and click **Services** to display the list of services.

   Note: The **Services** menu is not available on the Kubernetes version of the Control System.

2. Click the **History Server** link in the list of services to display the **JobHistory** page in a new tab.
3. Click the job ID link for the job you want to view the logs for.
4. Click the logs link in the **Logs** column of the **Application Master** section.

Using the HistoryServer UI to View Logs for Completed Applications

1. Open a browser and go to the following URL to open the JobHistory page:

Non-secure cluster

```
http://<IP address of HistoryServer
node>:19888
```


last-updated: Apr 11, 2022

Secure cluster	`https://<IP address of HistoryServer node>:19890`

2. Click the job ID link for the job for which you want to view the logs.
3. Click the logs link in the **Logs** column of the **Application Master** section.

Editing the Retention Settings of Aggregated Logs

By default, aggregate logs are stored on the MapRMapR Data Platform filesystem for 30 days. The retention time for aggregated logs also applies to centralized logs.

To edit the retention settings, add or edit the following properties in the *yarn-site.xml* file:

1. Set the value of `yarn.log-aggregation.retain-seconds` to set the duration that the logs are maintained. If you set a negative value for `yarn.log-aggregation.retain-seconds`, logs will not be deleted.

   Note: The duration specified by `yarn.log-aggregation.retain-seconds` starts from the time that the application starts running. Therefore, when you configure the duration, consider how long you want the log to remain in addition to the amount of time that the application will take to run. For example, if you expect most applications to take 20 seconds to run, do not set the value of this property to 20 seconds because the log might be deleted as soon as the applications completes.

2. Optionally, set the `yarn.log-aggregation.retain-check-interval-seconds` to specify how often the log retention check should be run. By default, it is one-tenth of the log retention time.

For more details about the properties that impact the YARN container logs and the aggregation option, see yarn-site.xml.

Centralized Logging

Describes the centralized logging feature of MapR.

MapR's Centralized Logging feature provides a job-centric or application-centric view of all log files generated by a MapReduce program. With centralized logging, the log files are written to local volumes in the MapR filesystem. You can run the `maprcli job linklogs` command for running or completed jobs to create a centralized log directory populated with symbolic links to all log files pertaining to the specified jobs or to the application.

Managing Centralized Logs for MapReduce Version 2 Applications

To manage centralized logs for MapReduce Version 2 applications, enable centralized logging, configure log retention, and view application logs.

Enabling Centralized Logging for MapReduce Version 2

Describes how to enable central logging for MapReduce version 2 applications.

As of MapR version 4.0.2, you can use centralized logging for MapReduce version 2 applications. However, this feature is disabled by default. In MapR version 4.0.1, centralized logging is not supported for MapReduce version 2 applications.

- Configure the `yarn.use-central-logging-for-mapreduce-only` property in the `yarn-site.xml` file to enable or disable centralized logging.

  The `yarn-site.xml` file is located in the following directory: `/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop/`.

last-updated: Apr 11, 2022

- To disable centralized logging, remove the property `yarn.use-central-logging-for-mapreduce-only` from the `yarn-site.xml` file, or set the value of `yarn.use-central-logging-for-mapreduce-only` to `false` in the `yarn-site.xml` file.
- To enable centralized logging, set the value of `yarn.use-central-logging-for-mapreduce-only` to `true` in the `yarn-site.xml` file. If you enable centralized logging while applications are running, restart all ResourceManagers. In a production cluster, restart ResoureManagers one at a time to prevent interruption to the running applications. The applications running during this process may not have centralized logging enabled.

Configuring Log Retention Time for MapReduce Version 2 Applications

Lists the parameter that controls the log retention time for MapReduce applications.

- Set the value of `yarn.log-aggregation.retain-seconds` to the number of seconds you want to retain the logs once the application starts in the `yarn-site.xml` file.

  The value defaults to 30 days. The value that you set for this property also applies to the retention of aggregated YARN logs.

  The `yarn-site.xml` file is in the following directory: `/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop`.

Viewing Logs for Completed MapReduce Version 2 Applications

With centralized logging, you can use maprcli or the HistoryServer user interface to access completed logs.

Using the Command Line to View Logs for Completed Applications

Describes how to view logs from the CLI.

1. Use the `maprcli job linklogs` command to create centralized logs for completed applications. For example, you can run the following `maprcli job linklogs` command to create centralized logs for `application_1434605941718_0001:`

   ```
   maprcli job linklogs -jobid application_1434605941718_0001 -todir /logsdir
   ```

   The centralized log directory contains symbolic links that are organized by hostname and containerID.

2. To determine where the logs are located, run the following command on the directory that contains the symlinks to the log files for a specific container:

   ```
   hadoop mfs -ls <todir>/<applicationID>/hosts/<hostName>/<containerID>
   ```

   For example, if you specified `logsdir` as the directory, you might issue a command similar to the following example. The system then displays the location of the log files:

   ```
   hadoop mfs -ls /logsdir/application_1434605941718_0001/hosts/
   qa-node178.qa.lab/container_e02_1434605941718_0001_01_000003

   Found 1 items
   lrwxrwxrwx U U U 3 root root 138 2015-06-18 05:50 0
   /logsdir/application_1434605941718_0001/hosts/qa-node178.qa.lab/
   container_e02_1434605941718_0001_01_000003
   ->
   ../../../../var/mapr/local/qa-node178.qa.lab/logs/yarn/userlogs/
   application_1434605941718_0001/container_e02_1434605941718_0001_01_000003
   p 2068.40.262432 qa-node178.qa.lab:5660 qa-node175.qa.lab:5660
   ```

   The link location appears after the arrow.

last-updated: Apr 11, 2022

3. To determine the types of log files that are available for this container and the path to each available log file, run the following command:

   ```
   hadoop fs -ls <link location>
   ```

   For example:

   ```
   hadoop fs -ls ../../../../var/mapr/local/qa-node178.qa.lab/logs/yarn/
   userlogs/application_1434605941718_0001/
   container_e02_1434605941718_0001_01_000003

   -rw-r-----   2 root root       2337 2015-06-18 05:48
   ../../../../var/mapr/local/qa-node178.qa.lab/logs/yarn/userlogs/
   application_1434605941718_0001/
   container_e02_1434605941718_0001_01_000003/syslog
   ```

   In this example, the path to the syslog is the only one that is displayed in the output. However, the stdout or stderr may also be available depending on what is generated by the application.

4. Run one of the following options to view the contents of a log file:
   a) To view the end of the log file, run `hadoop fs -tail <path to log file>`.

      ```
      hadoop fs -tail ../../../../var/mapr/local/
      qa-node178.qa.lab/logs/yarn/userlogs/application_1434605941718_0001/
      container_e02_1434605941718_0001_01_000003/syslog
      ```

   b) To view the entire log file, run `hadoop fs- cat <path to log file>`.

      ```
      hadoop fs- cat ../../../../var/mapr/local/
      qa-node178.qa.lab/logs/yarn/userlogs/application_1434605941718_0001/
      container_e02_1434605941718_0001_01_000003/syslog
      ```

Using the HistoryServer UI to View Logs for Completed Applications

Describes how to view logs using the HistoryServer interface.

You can view the logs in the HistoryServer interface.

View Logs in the HistoryServer Interface Launched Using the Control System

1. Log on to the Control System and click **Services** to display the list of services installed on the cluster.
2. Click the HistoryServer link in the list of services to open the **Job History** page in a new tab.
3. Click the Job ID link for the job for which you want to view the logs.
4. Click the logs link in the **Logs** column of the **Application Master** section.

View Logs Using the HistoryServer Interface

1. Go the URL similar to the following example, to open the Job History page:

   ```
   <IP address of HistoryServer node>:19888
   ```

2. Click the Job ID link for the job for which you want to view the logs.
3. Click the logs link in the **Logs** column of the **Application Master** section.

last-updated: Apr 11, 2022

Viewing the Service Log

Esplains how to view service logs using Kibana.

If Kibana is installed on the node, you can view the service log in the Kibana UI from the Control System. To view the log in the Kibana UI from the Control System:

1. Log in to the Control System and do one of the following:
   - Click **Services** to display the list of services installed on the cluster.
   - Go to the **Summary** tab in the service information page for the service.
2. Click  in the **Log Viewer** column to view the log for the associated service in the Kibana UI.
   See Kibana User Guide for more information.

Cluster Maintenance Schedule

Lists a sample maintenance schedule for the cluster.

A maintenance schedule shows which tasks a cluster administrator should perform daily, weekly, monthly, and quarterly, along with the initial setup tasks. This sample schedule is offered as a template that you can customize to suit your needs.

Initial Setup	Daily	Weekly	Monthly	Quarterly
Perform hardware checks	Check alarms	Check logs (cluster and job logs)	Clean up logs	Audit storage use
Check Prerequisites	Onboard new users	Check for zombie jobs	Reevaluate node and volume topology	Perform security audit
Verify Installation	Offboard users	Perform simple performance test	Check performance (jobs, I/O)	Upgrade MapR core, Hive, HCatalog
High Availability planning	Check cluster health	Verify snapshots	Verify mirrors	
Set up VIPs for NFS	Perform hardware maintenance (disks)	Remove old snapshots	Clean up data	
Benchmark and tuning		Failbacks to restore service layout	MapR File System balancing	
Set up node and volume topologies				
Capacity planning			Upgrade ecosystem components (other than Hive and HCatalog, which are done quarterly)	
Set up user permissions (ACLs, directory permissions, users, home directories)				
Set up quotas				
Set up compression				
Configure cluster queues for MapReduce jobs				

Initial Setup	Daily	Weekly	Monthly	Quarterly
Set up schedules (fair scheduler, capacity scheduler, job placement)				
Set up snapshots and mirrors				
Configure NICs				
Set up monitoring tools				
Set up security				
Create a disaster recovery plan				

Language Support for MapR Database Tables

This section lists the human languages that MapR Database tables can store, retrieve, and process.

MapR Database tables can store, retrieve, and process data in the following languages:

A

Abaza Abkhazian Achinese Acoli Adangme Adyghe Afar Afrikaans Aghem Ainu Akan Akkadian Akoose Albanian Aleut Amharic Amo Ancient Egyptian Ancient Greek Angika Arabic Aragonese Aramaic Arapaho Arawak Armenian Aromanian Assamese Assyrian Neo-Aramaic Asturian Asu Atikamekw Atsam Avaric Avestan Awadhi Aymara Azerbaijani

B

Badaga Bafia Bafut Bagheli Balinese Balkan Gagauz Turkish Balti Baluchi Bambara Bamun Bantawa Basaa Bashkir Basque Batak Batak Toba Bateri Beja Belarusian Bemba Bena Bengali Bhili Bhojpuri Bikol Bini Bislama Blin Bodo Bomu Bosnian Braj Breton Bube Buginese Buhid Bulgarian Bulu Buriat Burmese Bushi

C

Caddo Cantonese Carian Carib Catalan Cayuga Cebaara Senoufo Cebuano Central Atlas Tamazight Central Huasteca Nahuatl Central Mazahua Central Okinawan Chadian Arabic Chakma Chamorro Chechen Cherokee Cheyenne Chhattisgarhi Chiga Chinese Chinook Jargon Chipewyan Choctaw Chukot Church Slavic Chuukese Chuvash Classical Mandaic Colognian Comorian Congo Swahili Coptic Cornish Corsican Cree Creek Crimean Turkish Croatian Czech

D

Dakota Dan Dangaura Tharu Danish Dargwa Dari Dazaga Delaware Dinka Divehi Dogri Dogrib Domari Duala Dungan Dutch Dyula Dzongkha

E

Eastern Cham Eastern Frisian Eastern Gurung Eastern Huasteca Nahuatl Eastern Kayah Eastern Lawa Eastern Magar Eastern Tamang Efik Ekajuk Embu English Erzya Esperanto Estonian Etruscan Evenki Ewe Ewondo

F

Fang Fanti Faroese Fijian Filipino Finnish Fon French Friulian Fulah

G

Ga Gagauz Galician Ganda Garhwali Garo Gayo Gbaya Geez Georgian German Ghomala Gilbertese Gondi Gorontalo Gothic Grebo Greek Gronings Guajajára Guarani Guianese Creole French Gujarati Gujari Gusii Gwichʼin

last-updated: Apr 11, 2022

H

Hadothi Haida Haitian Hanunoo Hausa Hawaiian Hebrew Herero Hiligaynon Hindi Hiri Motu Hittite Hmong Ho Hopi Hungarian Hupa

I

Iban Ibibio Icelandic Igbo Iloko Inari Sami Indonesian Indus Kohistani Ingush Interlingua Inuktitut Inupiaq Irish Italian

J

Japanese Javanese Jenaama Bozo Jju Jola-Fonyi Judeo-Arabic Judeo-Persian Jumli

K

Kabardian Kabuverdianu Kabyle Kachchi Kachi Koli Kachin Kaingang Kako Kalaallisut Kalanga Kalenjin Kalmyk Kalo Finnish Romani Kamba Kanauji Kanembu Kannada Kanuri Kara-Kalpak Karachay-Balkar Karelian Kashmiri Kashubian Kathoriya Tharu Kazakh Kerinci Khaling Khasi Khmer Khmu Khowar Kikuyu Kimbundu Kinyarwanda Kita Maninkakan Kochila Tharu Kom Komering Komi Komi-Permyak Kongo Konkani Korean Koro Koro Wachi Koryak Kosraean Koyra Chiini Koyraboro Senni Kpelle Krio Kuanyama Kumyk Kurdish Kurukh Kutenai Kuy Kwasio Kyrgyz

L

Ladino Lahnda Lak Laki Lakota Lamba Lambadi Langi Lao Large Flowery Miao Latin Latvian Lepcha Lezghian Limbu Limburgish Lingala Lisu Literary Chinese Lithuanian Lombard Low German Lower Sorbian Lozi Lü Luba-Katanga Luba-Lulua Luiseno Lule Sami Lunda Luo Lushootseed Luxembourgish Luyia Lycian Lydian

M

Maba Macedonian Machame Madurese Mafa Magahi Maguindanaon Maithili Makasar Makhuwa-Meetto Makonde Malagasy Malay Malayalam Maltese Manchu Mandar Mandingo Manipuri Mansi Manx Manyika Maori Mapuche Marathi Mari Marshallese Marwari Masai Mbere Mbunga Medumba Mende Meroitic Meru Meta' Micmac Minangkabau Mirandese Mizo Mohawk Moksha Mon Mongo Mongolian Montagnais Moose Cree Morisyen Mossi Munda Mundang Mundari Myene

N

N’Ko Nama Nanai Naskapi Nauru Navajo Naxi Ndonga Neapolitan Negeri Sembilan Malay Nenets Nepali Newari Ngaju Ngambay Ngiemboon Ngomba Nias Nigerian Pidgin Niuean Nogai North Ndebele North Slavey Northeastern Thai Northern East Cree Northern Frisian Northern Sami Northern Sotho Northern Thai Norwegian Norwegian Bokmål Norwegian Nynorsk Nuer Nyamwezi Nyanja Nyankole Nyasa Tonga Nyoro Nzima

O

Occitan Ojibwa Old Irish Old Norse Old Persian Old Turkish Oriya Oromo Osage Oscan Ossetic

P

Pahlavi Palauan Pali Pampanga Pangasinan Papiamento Parkari Koli Parsi-Dari Parthian Pashto Persian Phoenician Plains Cree Pohnpeian Pökoot Polish Portuguese Prussian Punjabi Punu

Q

Quechua

R

Rajasthani Rajbanshi Rana Tharu Rangpuri Rapanui Rarotongan Rejang Réunion Creole French Riang (India) Rinconada Bikol Romanian Romansh Romany Rombo Ronga Rundi Russian Rusyn Rwa

S

Sabaean Safaliba Saho Sakha Samaritan Samaritan Aramaic Samburu Samoan Sandawe Sangir Sango Sangu Sanskrit Santali Sardinian Sasak Saurashtra Scots Scottish Gaelic Seki Selkup Sena Seneca

Serbian Serbo-Croatian Serer Shambala Shan Sherpa Shona Shor Sichuan Yi Sicilian Sidamo Siksika Sindhi Sinhala Sinte Romani Sirmauri Skolt Sami Slave Slovak Slovenian Soga Somali Soninke Sora Sorani Kurdish South Ndebele Southern Altai Southern East Cree Southern Hindko Southern Kurdish Southern Luri Southern Sami Southern Sotho Southwestern Tamang Spanish Sranan Tongo Standard Moroccan Tamazight Sukuma Sundanese Susu Swahili Swampy Cree Swati Swedish Swiss German Sylheti Syriac

T

Tabassaran Tachelhit Tae' Tagalog Tagbanwa Tahitian Tai Dam Tai Nüa Taita Tajik Tamashek Tamil Taroko Tasawaq Tatar Tausug Tavringer Romani Telugu Tereno Teso Tetum Thai Thulung Tibetan Tigre Tigrinya Timne Tiv Tlingit Tok Pisin Tokelau Tolaki Tomo Kan Dogon Tongan Tooro Tornedalen Finnish Tshangla Tsimshian Tsonga Tswana Tulu Tumbuka Turkish Turkmen Turoyo Tuvalu Tuvinian Twi Tyap

U

Uab Meto Udihe Udmurt Ugaritic Ukrainian Ulithian Umbrian Umbundu Unknown Language Upper Sorbian Urdu Uyghur Uzbek Vai Venda Vietnamese Virgin Islands Creole English Volapük Votic Vunjo

W

Wadiyara Koli Walloon Walser Waray Washo Welsh Western Cham Western Frisian Western Gurung Western Huasteca Nahuatl Western Kayah Western Lawa Western Magar Western Mari Western Tamang Wolaytta Wolof

X

Xaasongaxango Xaványte Xhosa

Y

Yangben Yao Yapese Yemba Yiddish Yoruba Yucateco

Z

Zapotec Zarma Zaza Zeeuws Zenaga Zhuang Zulu Zuni

Sample JSON File for Metering

A sample metering JSON file for an 8-node cluster with no workloads enabled.


```
{
   "id":"metering-1529939795-b82b1db90a364a0d9af5b35328db133f",
   "clusterId":"8036442972050269505",
   "collectionDate":"1529939795",
   "mapRCoreBuildVersion":"6.1.0.20180621112549.GA",

"signature":"8c9820b3eaac45c45952635a0733ba82c1f5e1d6ddbfa31602acff351bfe6f2
a",
   "version":2.0,
   "isSecure":true,
   "numberOfNodes":8,
   "storage":{
      "clusterDiskCapacityInGB":1374,
      "clusterDiskSpaceUsedInGB":68,
      "totalAmountOfDataOffloadedToColdTiersInGB":0
   },
   "nodes":[
      {
         "id":"5411072923155745779",
         "yarn":{
            "allocatedVcores":0.0,
            "availableVcores":4.0
         },
         "processes":[
            {
```

```
                "name":"warden",
                "cpuCoreInSeconds":44854
            },
            {
                "name":"mfs",
                "cpuCoreInSeconds":7386580
            },
            {
                "name":"collectd",
                "cpuCoreInSeconds":10633592
            },
            {
                "name":"zookeeper_server",
                "cpuCoreInSeconds":255249
            },
            {
                "name":"data-access-gateway",
                "cpuCoreInSeconds":890914
            },
            {
                "name":"hbase-mapr-rest",
                "cpuCoreInSeconds":33099
            },
            {
                "name":"yarn-mapr-nodemanager",
                "cpuCoreInSeconds":116368
            },
            {
                "name":"hoststats",
                "cpuCoreInSeconds":31701
            },
            {
                "name":"apiserver",
                "cpuCoreInSeconds":138405
            },
            {
                "name":"gateway",
                "cpuCoreInSeconds":44854
            },
            {
                "name":"mastgateway",
                "cpuCoreInSeconds":187132
            },
            {
                "name":"ganesha",
                "cpuCoreInSeconds":49779
            },
            {
                "name":"nfs4server",
                "cpuCoreInSeconds":49779
            },
            {
                "name":"drillbit",
                "cpuCoreInSeconds":97031
            },
            {
                "name":"cldb",
                "cpuCoreInSeconds":501441
            },
            {
                "name":"fluentd",
                "cpuCoreInSeconds":10191
            }
        ],
```

```
         "clients":[
            {
               "id":"1842047781700531199",
               "clienttype":"posixclientplatinum",
               "clienthealth":"Active"
            }
         ],
         "dsr_service_configured":false
      },
      {
         "id":"6250302233036420992",
         "yarn":{
            "numberOfCPUsAllocated":0.0,
            "numberOfCPUsAvailable":4.0
         },
         "processes":[
            {
               "name":"warden",
               "cpuCoreInSeconds":53348
            },
            {
               "name":"mfs",
               "cpuCoreInSeconds":9973216
            },
            {
               "name":"collectd",
               "cpuCoreInSeconds":10868359
            },
            {
               "name":"zookeeper_server",
               "cpuCoreInSeconds":322375
            },
            {
               "name":"data-access-gateway",
               "cpuCoreInSeconds":24619
            },
            {
               "name":"hbase-mapr-rest",
               "cpuCoreInSeconds":36581
            },
            {
               "name":"yarn-mapr-nodemanager",
               "cpuCoreInSeconds":132342
            },
            {
               "name":"hoststats",
               "cpuCoreInSeconds":36616
            },
            {
               "name":"apiserver",
               "cpuCoreInSeconds":164456
            },
            {
               "name":"gateway",
               "cpuCoreInSeconds":91140
            },
            {
               "name":"mastgateway",
               "cpuCoreInSeconds":236735
            },
            {
               "name":"opentsdb",
               "cpuCoreInSeconds":3492717
            },
```

```
            {
               "name":"ganesha",
               "cpuCoreInSeconds":68427
            },
            {
               "name":"nfs4server",
               "cpuCoreInSeconds":68427
            },
            {
               "name":"drillbit",
               "cpuCoreInSeconds":91138
            },
            {
               "name":"elasticsearch",
               "cpuCoreInSeconds":3060305
            },
            {
               "name":"cldb",
               "cpuCoreInSeconds":326018
            },
            {
               "name":"fluentd",
               "cpuCoreInSeconds":10965
            }
         ],
         "clients":[

         ],
         "dsr_service_configured":true
      },
      {
         "id":"2948891216591685686",
         "yarn":{
            "numberOfCPUsAllocated":0.0,
            "numberOfCPUsAvailable":4.0
         },
         "processes":[
            {
               "name":"warden",
               "cpuCoreInSeconds":77977
            },
            {
               "name":"mfs",
               "cpuCoreInSeconds":257182
            },
            {
               "name":"collectd",
               "cpuCoreInSeconds":12581361
            },
            {
               "name":"zookeeper_server",
               "cpuCoreInSeconds":505537
            },
            {
               "name":"data-access-gateway",
               "cpuCoreInSeconds":35647
            },
            {
               "name":"hbase-mapr-rest",
               "cpuCoreInSeconds":39213
            },
            {
               "name":"yarn-mapr-nodemanager",
               "cpuCoreInSeconds":185212
```

```
            },
            {
               "name":"hoststats",
               "cpuCoreInSeconds":101114
            },
            {
               "name":"apiserver",
               "cpuCoreInSeconds":4237000
            },
            {
               "name":"gateway",
               "cpuCoreInSeconds":903572
            },
            {
               "name":"mastgateway",
               "cpuCoreInSeconds":256831
            },
            {
               "name":"opentsdb",
               "cpuCoreInSeconds":3492194
            },
            {
               "name":"ganesha",
               "cpuCoreInSeconds":149134
            },
            {
               "name":"nfs4server",
               "cpuCoreInSeconds":149134
            },
            {
               "name":"drillbit",
               "cpuCoreInSeconds":102115
            },
            {
               "name":"elasticsearch",
               "cpuCoreInSeconds":3887653
            },
            {
               "name":"cldb",
               "cpuCoreInSeconds":110166
            },
            {
               "name":"yarn-mapr-resourcemanager",
               "cpuCoreInSeconds":347747
            },
            {
               "name":"fluentd",
               "cpuCoreInSeconds":16277
            }
         ],
         "clients":[

         ],
         "dsr_service_configured":true
      },
      {
         "id":"952332856626659546",
         "yarn":{
            "numberOfCPUsAllocated":0.0,
            "numberOfCPUsAvailable":4.0
         },
         "processes":[
            {
               "name":"warden",
```

```
            "cpuCoreInSeconds":54471
         },
         {
            "name":"mfs",
            "cpuCoreInSeconds":340097
         },
         {
            "name":"collectd",
            "cpuCoreInSeconds":11863923
         },
         {
            "name":"hbase-mapr-rest",
            "cpuCoreInSeconds":30307
         },
         {
            "name":"yarn-mapr-nodemanager",
            "cpuCoreInSeconds":150883
         },
         {
            "name":"hoststats",
            "cpuCoreInSeconds":31853
         },
         {
            "name":"mastgateway",
            "cpuCoreInSeconds":202457
         },
         {
            "name":"opentsdb",
            "cpuCoreInSeconds":3626835
         },
         {
            "name":"ganesha",
            "cpuCoreInSeconds":36490
         },
         {
            "name":"nfs4server",
            "cpuCoreInSeconds":36488
         },
         {
            "name":"drillbit",
            "cpuCoreInSeconds":76016
         },
         {
            "name":"elasticsearch",
            "cpuCoreInSeconds":2804536
         },
         {
            "name":"fluentd",
            "cpuCoreInSeconds":10227
         },
         {
            "name":"mapred-mapr-historyserver",
            "cpuCoreInSeconds":37070
         },
         {
            "name":"httpfs",
            "cpuCoreInSeconds":47311
         },
         {
            "name":"grafana",
            "cpuCoreInSeconds":14206
         },
         {
            "name":"hue",
```

```
               "cpuCoreInSeconds":4261
            },
            {
               "name":"spark-mapr-org",
               "cpuCoreInSeconds":61948
            },
            {
               "name":"kibana",
               "cpuCoreInSeconds":101479
            },
            {
               "name":"hive-mapr-hiveserver2",
               "cpuCoreInSeconds":105652
            },
            {
               "name":"hive-mapr-metastore",
               "cpuCoreInSeconds":39584
            },
            {
               "name":"oozie",
               "cpuCoreInSeconds":717478
            },
            {
               "name":"hbase-mapr-thrift",
               "cpuCoreInSeconds":32331
            }
         ],
         "clients":[

         ],
         "dsr_service_configured":false
      },
      {
         "id":"8619138289230167562",
         "yarn":{
            "numberOfCPUsAllocated":0.0,
            "numberOfCPUsAvailable":4.0
         },
         "processes":[
            {
               "name":"warden",
               "cpuCoreInSeconds":41172
            },
            {
               "name":"mfs",
               "cpuCoreInSeconds":189731
            },
            {
               "name":"collectd",
               "cpuCoreInSeconds":27735
            },
            {
               "name":"hbase-mapr-rest",
               "cpuCoreInSeconds":31519
            },
            {
               "name":"yarn-mapr-nodemanager",
               "cpuCoreInSeconds":150749
            },
            {
               "name":"hoststats",
               "cpuCoreInSeconds":34185
            },
            {
```

```
                "name":"apiserver",
                "cpuCoreInSeconds":27253
             },
             {
                "name":"mastgateway",
                "cpuCoreInSeconds":194093
             },
             {
                "name":"ganesha",
                "cpuCoreInSeconds":36815
             },
             {
                "name":"nfs4server",
                "cpuCoreInSeconds":36815
             },
             {
                "name":"drillbit",
                "cpuCoreInSeconds":85679
             },
             {
                "name":"fluentd",
                "cpuCoreInSeconds":9464
             }
          ],
          "clients": [

{ "id": "3893819064903061731", "clientType": "posixclientplatinum",
"clientHealth": "Active" },

{ "id": "6476832847333424974", "clientType": "posixclientplatinum",
"clientHealth": "Active" },

{ "id": "4713798361306456121", "clientType": "posixclientplatinum",
"clientHealth": "Active" },

{ "id": "2714377982799294814", "clientType": "posixclientplatinum",
"clientHealth": "Active" }

],
       }
   ]
}
```


Metering Data Descriptions

This table lists the metrics collected by the metering feature.

Table

JSON Field Name	Description
clusterId	Randomly generated ID created when the cluster is installed
collectionDate	Epoch timestamp for the latest metric collection
mapRCoreBuildVersion	MapR core software build version
signature	Digital signature of the metering JSON file
version	Version of the metering program used to capture this JSON result

Table (Continued)

JSON Field Name	Description
isSecure	Identifies whether the cluster is secure or not Options: `true` or `false`.
numberOfNodes	Number of nodes in the cluster
clusterDiskCapacityInGB	Total data in the cluster at that time
clusterDiskCapacityUsedInGB	Total data consumed in the cluster at that time
totalAmountOfDataOffloadedToColdTiersInGB	Data offloaded to the cloud, using Object Tiering You can display this metric using the `maprcli dashboard info` command.
cpuCoreInSeconds	CPU core hours consumed by all MapR Services across all nodes Also, CPU core-hours used by all Apache ecosystem components, as well as MapR File System, CLDB, API servers, REST gateways, replication gateways, and so on.
clients	Number of active instances at that moment
numberOfactiveUniquePlatinumPosixClients	Number of monthly active unique Platinum POSIX clients You can display this metric using the `maprcli node list -clientsonly` command.
numberOfactiveUniqueDSROrPACCInstances	Number of monthly active unique DSR instances, using Gold POSIX clients

Troubleshooting Cluster Administration

Lists the common errors and their solutions.

The following list identifies how to address several cluster administration issues:

The URL reported by YARN for tracking job details does not load.

This URL uses the output of the `hostname -f` command, which must be the fully qualified domain name (FQDN) for the node. On Ubuntu, make sure that the */etc/hostname* file is configured with the node's FQDN. On CentOS/Redhat, make sure that the */etc/sysconfig/network* file is configured with the node's FQDN, then restart the node.

The ResourceManager does not start.

If the ResourceManager does not come up, check the following:

- Check that you supplied the correct ResourceManager hostname or IP address in the `-RM` parameter when running configure.sh on each node at installation time. If you are not sure, you can re-run configure.sh to correct the problem.
- Do not specify a ResourceManager port with the hostname or IP address in the `-RM` parameter; there is no `<port>` option.

last-updated: Apr 11, 2022

- Make sure that you specified the same ResourceManager hostname or IP address on all nodes when running configure.sh.
- For more information about what might be causing a problem, check the ResourceManager logs:*/opt/mapr/hadoop/hadoop-<version>/logs*

The NodeManager does not start.

If the NodeManager does not come up, check the following:

- Make sure that the fileserver role is installed on the node by looking in the */opt/mapr/roles* directory.
- Make sure that the fileserver service is running, using either the service list command or the MapR Control System.
- For more information about what might be causing a problem, check the NodeManager logs:*/opt/mapr/hadoop/hadoop-2.3.0/logs*

Job history is not available.

If job history is not recorded, check the following:

- Make sure that the HistoryServer role is installed on the desired node by looking in the */opt/mapr/roles* directory. Note that only one node in the cluster can have the HistoryServer role.
- Make sure the HistoryServer is running on the desired node, using either the service list command or the MapR Control System.
- Check that you supplied the correct HistoryServer hostname or IP address in the `-HS` parameter when running configure.sh on each node at installation time. If you are not sure, you can re-run configure.sh to correct the problem.

Submitted applications do not show up in the ResourceManager.

If you submit an application and it does not appear in the ResourceManager, check the following:

- Make sure the application is running in YARN and not as a local application (check for *app_local* or *job_local* in the application output).
- Check the class path on which the application was invoked, and make sure that */opt/mapr/hadoop/hadoop-<version>/etc/hadoop* includes the class paths.
- Make sure that you are running the correct version of Hadoop. Example:

```
ls -l /usr/bin/hadooplrwxrwxrwx 1
root root 40
   Mar
4 11:38 /usr/bin/hadoop -> /opt/
mapr/hadoop/hadoop-2.3.0/bin/hadoop
```

The application throws a ClassNotFound exception at job submission time.

If you submit an application and it throws the ClassNotFound exception, check the following:

last-updated: Apr 11, 2022

- Check that the application jar is correctly packaged with the required class.
- Make sure that you are running the correct version of Hadoop. Example:

```
ls -l /usr/bin/hadooplrwxrwxrwx 1
root root 40
   Mar
4 11:38 /usr/bin/hadoop -> /opt/
mapr/hadoop/hadoop-2.3.0/bin/hadoop
```

I want to move the HistoryServer and ResourceManager to different nodes.

If you have installed the HistoryServer and the ResourceManager on the same node (when you initially ran configure.sh you did not specify the `-HS` parameter, or you specified the same IP address or hostname for both the `-RM` and `-HS` parameters), you can use configure.sh to move one or both services to different nodes. Make sure to specify both the `-HS` parameter and the `-RM` parameter, because if you only specify the `-RM` parameter the HistoryServer will move to the ResourceManager node.

Timing issue prevents services from starting on a secure cluster.

If your cluster has security features enabled, you may encounter a timing issue that results in services failing to start during initial configuration with the `-F` option. (This issue does not arise if you are bringing up a cluster that has already been installed and configured.)

When you run the configure.sh script with the `-F` option, the ZooKeeper and Warden services start up on the primary node first, then as other nodes are installed, services are automatically started on those nodes. However, because of this timing issue, Warden may fail to communicate with ZooKeeper, and the cluster may fail to come up.

If you encounter this problem, do not use the `-F` option. Instead, stop all ZooKeeper and Warden services on all nodes, then start the ZooKeeper services on all of the ZooKeeper nodes (that is, the nodes where the ZooKeeper packages are installed). Finally, start the Warden services on all nodes.

How to find a node's serverid.

Some `maprcli` commands take an argument `serverid`, which is an unique identifier for each node in a cluster. This id is also sometimes referred to as the *node id*.

To find the `serverid`, use the maprcli node list command, which lists information about all nodes in a cluster. The `id` field is the value to use for `serverid`.

For example:

```
maprcli node list -columns hostname,id
   id                   hostname
ip
   4800813424089433352  node-28.lab
10.10.20.28
   6881304915421260685  node-29.lab
10.10.20.29
   4760082258256890484  node-31.lab
```

```
10.10.20.31
   8350853798092330580  node-32.lab
10.10.20.32
   2618757635770228881  node-33.lab
10.10.20.33
```

You can also get this listing as a JSON object by using the `-json` option. For example:

```
maprcli node list -columns
id,hostname -json
   {
        "timestamp":1358537735777,
        "status":"OK",
        "total":5,
        "data":[
                {

"id":"4800813424089433352",

"ip":"10.10.20.28",

"hostname":"node-28.lab"
                },
                {

"id":"6881304915421260685",

"ip":"10.10.20.29",

"hostname":"node-29.lab"
                },
                {

"id":"4760082258256890484",

"ip":"10.10.20.31",

"hostname":"node-31.lab"
                },
                {

"id":"8350853798092330580",

"ip":"10.10.20.32",

"hostname":"node-32.lab"
                },
                {

"id":"2618757635770228881",

"ip":"10.10.20.33",

"hostname":"node-33.lab"
                }
        ]
   }
```

Error 'mv Failed to rename maprfs...' when moving files across volumes.

Prior to version 2.1, you cannot move files across volume boundaries in the MapR Platform. You can

move files within a volume using the `hadoop fs -mv` command, but attempting to move files to a different volume results in an error of the form "`mv: Failed to rename maprfs://<source path> to <destination path>`".

As a workaround, you can copy the file(s) from source volume to destination volume, and then remove the source files.

The example below shows the failure occurring. In this example directories `/a` and `/b` are mount-points for two distinct volumes.

```
hadoop fs -ls /
   Found 2 items
   drwxrwxrwx   - root root
0 2011-12-02 15:14 /a
   drwxrwxrwx   - root root
0 2011-12-02 15:09 /b

hadoop fs -put testfile /a
hadoop fs -ls /a
   Found 1 items
   -rwxrwxrwx   3 root root
2048000 2011-12-02 15:18 /a/testfile

root@node1:~# hadoop fs -mv /a/
testfile /b
   mv: Failed to rename maprfs://
10.10.80.71:7222/a/testfile to /b
```

The following example shows the work-around, moving a file `/a/testfile` to directory `/b`, and then removing the source file.

```
hadoop fs -cp /a/testfile /b/testfile
hadoop fs -ls /b
   Found 1 items
   -rwxrwxrwx   3 root root
2048000 2011-12-02 15:19 /b/testfile

hadoop fs -rmr /a/testfile
   Deleted maprfs://
10.10.80.71:7222/a/testfile

hadoop fs -ls /a
```

This workaround is only necessary if `/a` and `/b` correspond to different volumes.

'ERROR com.mapr.baseutils.cldbutils.CLDBRpcCommonUtils' in cldb.log, caused by mixed-case cluster name in mapr-clusters.conf.

MapR cluster names are case sensitive. However, some versions of MapR v1.2.x have a bug in which the cluster names specified in `/opt/mapr/conf/mapr-clusters.conf` are not treated as case sensitive. If you have a cluster with a mixed-case name, after upgrading from v1.2 to v2.0+, you may experience CLDB errors (in particular for mirror volumes) which generate messages like the following in `cldb.log`:

```
2012-07-31 04:43:50,716 ERROR
com.mapr.baseutils.cldbutils.CLDBRpcCo
```

```
mmonUtils
[VolumeMirrorThread]: Unable to reach
cluster with name: qacluster1.2.9. No
entry found in file /conf/
mapr-clusters.conf for cluster
qacluster1.2.9.
Failing the CLDB RPC with status 133
```

(The path given in this message is relative to `/opt/mapr/`, which might be misleading.)

As a work-around after upgrading, to continue working with mirror volumes created in v1.2, duplicate any lines with upper-case letters in `mapr-clusters.conf`, converting all letters to lower case.

Mirror volumes created in v2.0+ do not exhibit this behavior.

MapR Control System does not display on Internet Explorer.

The MapR Control System supports Internet Explorer version 9 and above. In IE9, **Compatibility View** under the **Tools** menu must be turned off, or else the user interface will not display correctly.

Unable to kill a job using the Metrics UI.

The following error displays when the root user tries to kill a job using the Metrics UI: `Failed to get Job information for job_x, Error: mapr is not allowed to impersonate root`

To resolve this issue, add the following properties to core-site.xml in directory `/opt/mapr/hadoop/hadoop-0.20.2/etc/` :

```
<property>
<name>hadoop.proxyuser.mapr.groups</
name>
<value>*</value>
<description>Allow the superuser mapr
to impersonate any member of any
group</description>
</property>


<property>
<name>hadoop.proxyuser.mapr.hosts</
name>
<value>*</value>
<description>The superuser can
connect from any host to impersonate a
user</description>
</property>
```

YARN logs are deleted before the application completes.

The duration that YARN container logs are maintained starts from the time that the application starts running.

When YARN container logs are not aggregated, the YARN container logs are retained for 3 hours on each node. To update the duration, edit the value of `yarn.nodemanager.log.retain-seconds` in the yarn-site.xml file.

When YARN container log aggregation is enabled, by default, the aggregated logs are not deleted. However, this setting can be overridden in yarn-site.xml

	file. To update the duration, edit the value of `log-aggregation.retain-seconds` in the yarn-site.xml file. You must consider how long you want the log to remain past the amount of time that the application will take to run. For example, if you expect most applications to take 20 seconds to run, do not set the value of this property to 20 seconds because the log may be deleted before the applications competes.
YARN applications fail because /tmp subdirectories have been deleted.	Some RHEL and CentOS platforms include the `tmpwatch` service by default. This service cleans up the `/tmp` directory on a regular basis. However, this operation causes the deletion of directories that are needed for applications to run (for example, `nm-local-dir` for YARN). The running NodeManager process does not re-create these missing directories, causing applications to fail.
Jobs fail when the timeline server is down	The timeline server for the Hive-on-Tez user interface does not support high availability. Jobs fail when the resource manager cannot connect to the timeline server. However, you can change the `yarn.timeline-service.client.best-effort` property to TRUE in the yarn-site.xml file to allow applications to run successfully even when the timeline server is down.

Best Practices for Backing Up MapR Information

Lists the best practices and performance considerations to follow when backing up MapR information.

To back up configuration information and data from your MapR cluster, you must install the appropriate Linux backup client from your backup software provider on your servers in your MapR cluster. Your backup client user must have the proper filesystem, and volume permissions. For details on how to configure MapR volume permissions see Creating Volume-level ACLs on page 1447 and Managing Access Controls on page 1445.

Backup Configuration Data

By default, all installation files on the cluster, for each server in the cluster, are stored in a single directory on each server in the MapR cluster. To ensure that you backup all the configuration files, MapR supported applications, as well as log files, back up the `/opt/mapr` directory for all servers in the cluster.

Note that the `/opt/mapr` location includes all log files. Log files can add a significant amount of data to your backup environment, so evaluate if they are needed for your business continuity requirements. To backup just the configuration files for the cluster, backup the `/opt/mapr/conf` directory from all servers in the cluster.

Backup Volume Data

MapR's recommended way to backup and restore data, is to enable and configure snapshots and volume mirroring for your data, to another MapR cluster. This step ensures that your business continuity and disaster recovery needs are met.

See the following links for setting up Snapshots, Mirroring, Table and Streams replications.

- Snapshots: Managing Snapshots on page 947
- Mirrors: Mirror Volumes on page 459

last-updated: Apr 11, 2022

- MapR-DB Table Replication: Managing Table Replication on page 1065
- MapR-ES Streams Replication: Stream Replication on page 656

If you do not have a secondary cluster to mirror your data, back up your volumes by specifying the following path in your Linux backup agent: `/mapr/cluster_name/` - For example: `/mapr/my.cluster.com/`.

Performance Considerations When Backing Large Data Sets

You could run into bandwidth and performance limitations when you specify only one path to your MapR cluster, where your data in your volumes is stored on only one Linux host agent. The bottleneck can occur due to the size of that data you are backing up (large file sizes), or due to the number of files you have in your directory structure (millions of files in one directory).

To mitigate performance issues, break up the volumes across multiple Linux backup agents, with specific mount paths. For example:

```
MapR Linux Host 1 (hostname1):
          /mapr/my.cluster.com/volume1
          /mapr/my.cluster.com/volume2
          /mapr/my.cluster.com/volume3
```

```
MapR Linux Host 2 (hostname2):
          /mapr/my.cluster.com/volume4
          /mapr/my.cluster.com/volume5
          /mapr/my.cluster.com/volume6
```

```
MapR Linux Host 3 (hostname3):
          /mapr/my.cluster.com/volume7
          /mapr/my.cluster.com/volume8
          /mapr/my.cluster.com/volume9
```

Preserve Metadata About the Volumes

To preserve metadata such as permissions and *ACE* rules, run a pre-script process as the `mapr` user, in your backup agent. For example in your pre-script configuration for your host agent for your cluster, you would run:

```
maprcli volume dump create
        -name volume1 -dumpfile volume1_fulldump1 -e statefile1
```

Some backup software may need "stderr" or "stdout" codes to run pre or post processing scripts within their product. In that case, you may need to write a bash script to dump the file to a location of your choice, and ensure that your backup agent is configured to backup that directory. Consult your backup software provider's documentation. For information on creating volume dumps, see Create and Maintain Volume Dump File on page 1961.

6.1 Development

This section contains information related to application development for Ezmeral ecosystem components and MapR Data Platform products, including Filesystem, Database (Key-Value and JSON), and Event Streams.

Application Development Process

Before you start developing applications on the MapR Data Platform platform, consider how you will get the data into the platform, the storage format of the data, the type of processing or modeling that is required, and how the data will be accessed.

At a high-level, building an application comes down to the following steps:

1. Step 1: Select a Data Storage Format on page 2359
2. Step 2: Write Data to MapR Data Platform on page 2362
3. Step 3: Explore Ways to Work With the Data on page 2362
4. Step 4: Set Up the Development Environment on page 2365
5. Step 5: Build the Application on page 2374

Step 1: Select a Data Storage Format

Consider the data format options and determine how you want to use to store your data.

Keep in mind that a single application can access data from a variety of data formats. The following data formats are available.

MapR XD Distributed File and Object Store

MapR XD Distributed File and Object Store is a random read-write distributed filesystem that allows applications to concurrently read and write directly to files. This data store is great for storing and scanning large data sets of historical data, and for sharing files between various services and applications. Any node with access to the filesystem can access files on it.

Consider the following examples:

- Write large amounts of user click-stream data for a web site in a simple directory structure based on the date, and then process that data using tools like Spark, Drill, Hive or another MapReduce application.
- Store various types of images, audio files, and video files in one shared directory so that web or mobile applications can render the content as required.
- Share configuration files or internationalized resources among various applications by storing these files in a shared directory.
- Simplify the deployment of new applications by adding java libraries (.jar files) to a shared directory and then including the directory in the classpath of one or more applications.

last-updated: Apr 11, 2022

- Store the Docker files and images in a shared location which can be accessed by various servers. This provides a single, shared location from which users can launch containers.

When you store large data sets, use a file format in which the data can be consumed efficiently. For example, Parquet, ORC, sequence files are good for storing and scanning. Parquet is great for storing data on the filesystem because it stores data in columnar format, which can be partitioned. Parquet also works well for use cases where you query the data with Drill or process the data with Spark applications. Note that you can use CSV or JSON formats, but they scanning these formats is less efficient.

For more information about the MapR File System, see File System on page 452.

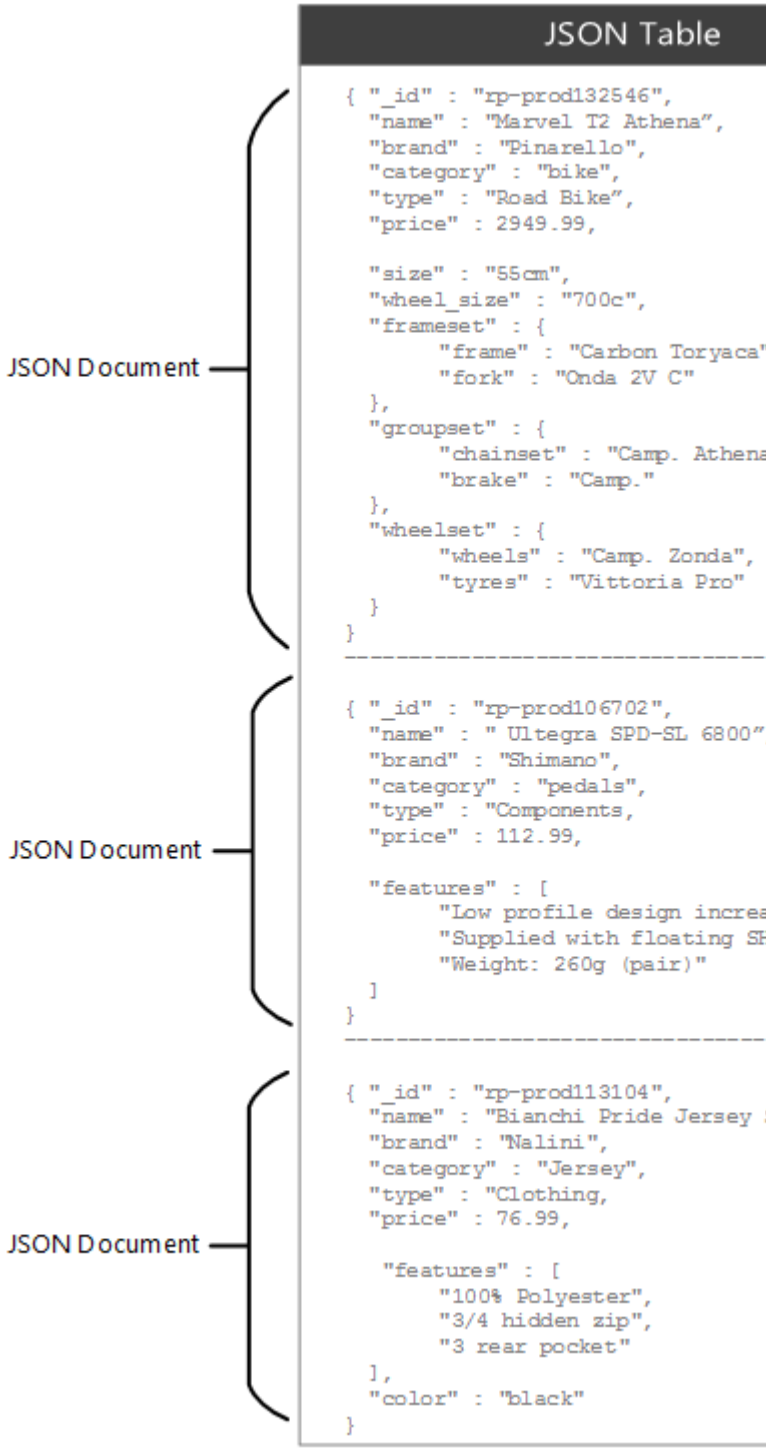
MapR Database

MapR Database is an enterprise-grade, high performance, NoSQL database management system that supports both binary and JSON tables. Consider using MapR Database tables when you want to query and organize large amounts data. It also integrates with Drill, Apache Spark, Hive and other MapReduce tools to provide applications the ability to scan or query large data sets in an efficient, distributed way.

MapR Database provides the following features:

- **A flexible schema.** Each row or document can have its own set of attributes.
- **Efficient random access.** Applications can quickly access one or more records using a row key, document ID, or a conditional queries.
- **Easy and efficient data mutation.** Applications can insert, update, and delete rows or documents.

MapR Database Binary Tables	MapR Database binary tables consist of rows that are identified by primary keys and row data is identified by key/value pairs. MapR Database tables are similar to HBase tables in that MapR Database does not determine or store the datatype of each value in the table. But, MapR Database tables perform operations more efficiently than HBase table. You might want to use binary tables when you want to create or use an existing HBase application. However, on the Converged Data Platform, JSON tables are usually preferred due to their flexibility.
MapR Database JSON Tables	A MapR Database JSON tables provide a flexible, powerful schema that you can customize based on the data that you want to represent. Each row in a JSON table corresponds to an JSON document with an unique _id and each JSON document can have a different set of columns. MapR Database JSON tables determine the datatype of each value based on the type of data written to the document. The following example lists three JSON documents from a single JSON table. Note that the attributes associated with each document varies.



For more information, see MapR Database on page 496.

MapR Event Store For Apache Kafka

MapR Event Store For Apache Kafka is a publish/subscribe messaging solution that uses the Apache Kafka API. MapR Event Store For Apache Kafka writes events as messages in a topic and topics are part of a stream. Producer applications can publish events to a stream and consumer applications can read all or a subset of the messages in a stream. By default, messages are stored in a topic for 7 days and then automatically purged. However, you can shorten or extend the time-to-live (ttl) for messages in a stream based on your use case.

For more information, see MapR Event Store For Apache Kafka on page 627.

Step 2: Write Data to MapR Data Platform

Depending on your use case, move existing data onto the MapR Data Platform platform or write data directly to the platform.

You can write batch data or streaming data to MapR Data Platform. Batch data refers to data that is already in a data-store while streaming data refers to the continuous flow of real-time messages that have yet to be written to a data-store. Streaming data is generally processed as it is received while batch data is processed after a set of data is written to the datastore. There are many ways to write batch and streaming data to the platform, the following sections provide a few examples.

Write Batch Data to the Platform

You can use an NFS client, hadoop command, or ecosystem components to write batch data to MapR File System. Basic POSIX filesystem operations can be used to move data to MapR File System. For example, you can use NFS clients, POSIX clients, or applications that utilize libraries such as java.io to access the filesystem. Hadoop commands and hdfs APIs can be used to add or update files on the MapR File System. For example, you can use the hadoop `distcp` command to copy data from HDFS to MapR File System. Hadoop Ecosystem components, such as Apache Flume, can also be used to push log files to MapR File System.

You can also write, update, or delete batch data to MapR Database tables. Applications can use the OJAI API to write to JSON tables or the HBase API to write to binary tables.

Write Streaming Data to the Platform

Write streaming event data as messages in HPE Ezmeral Data Fabric Event Data Streams topics using Kafka APIs or a REST client application. C, Java, or Python applications can produce messages to one or more topics in event streams. Additionally, applications written in any language can use the REST Proxy to produce messages to one or more topics in an event stream. For example, a financial service application, written in Java, could produce messages about stock market activity to an event stream topic.

Step 3: Explore Ways to Work With the Data

Once the data is in the MapR Data Platform platform, explore the various features and components available on the platform and determine your path. You may want to access data in its initial format or perform some data modeling or processing prior to accessing the data.

The following sections provide some examples to help you determine which approach will work for your particular use case.

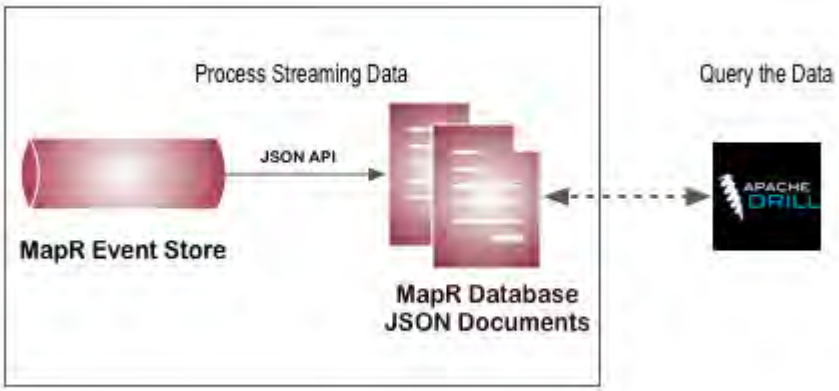
Process Data

When developing applications that ingest data, consider if the data requires some processing before the data can be consumed or stored.

Consider the following scenarios:

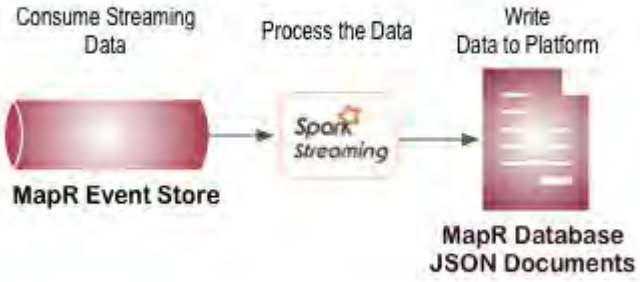
- **Process the Data Before Querying the Data**

  To efficiently query data, you may want to convert the data into a different format. For example, if you want to use Drill to query event data, you can convert streaming data from topics to a JSON table to enable more efficient querying. To do this, ingest event data using MapR Event Store For Apache Kafka, use the MapR Event Store For Apache Kafka API to read the data from topics and the JSON API to store the data in a JSON table, then use Drill to query the JSON tables.



- **Process Data before Storing the Data**

  You may also want to process the data based on business needs to perform some pre-processing before long term storage. For example, you can consume streaming data from a MapR Event Store For Apache Kafka topic with a Spark Streaming application which performs calculations or adds additional data before storing the data in a different data-store such as a MapR Database JSON table.



- **Perform Calculations as the Data is Stored**

  You may want to modify a single row in a table and then incrementally aggregate data. For example, you can use MapR Database tables to store large amounts of customer information or product catalog data and then read and write to a subset of that data. Then, modify a single row in a table to incrementally aggregate data. For example, to aggregate the number of clicks on a page, you can have a row key for each date and page. Internally, you can design the table to increment based on timestamp. The following example shows a row of data for the info page on 2017-02-22:

```
2017-01-22-info.html => key
Total : 1230 +1
H00 :   100
M1:1
M2: 5
...
M30 : 10
...
H20 : 50 +1
M1:1
...
M25: 1
```

- **Process Large Sets of Data**

  There are also many methods to process files in their initial state. To process large sets of data on the MapR XD Distributed File and Object Store, it is common to use Spark or MapReduce applications. MapReduce applications perform parallel,distributed processing of data in batches and are therefore a great way to process large datasets. Spark applications can be used to iteratively process large sets of data with machine learning algorithms. For an example of using a machine learning algorithm with a Spark application, see Building a Recommendation Engine with Spark.

Access Data

There are many use cases for why you might want to access data and many methods to access the data. Operational applications or E-commerce services may want to access data on the MapR Data Platform platform to provide customers a view of transactional data. Business users may want to view user profile data or submit queries through a BI tool to visually analyze the data.

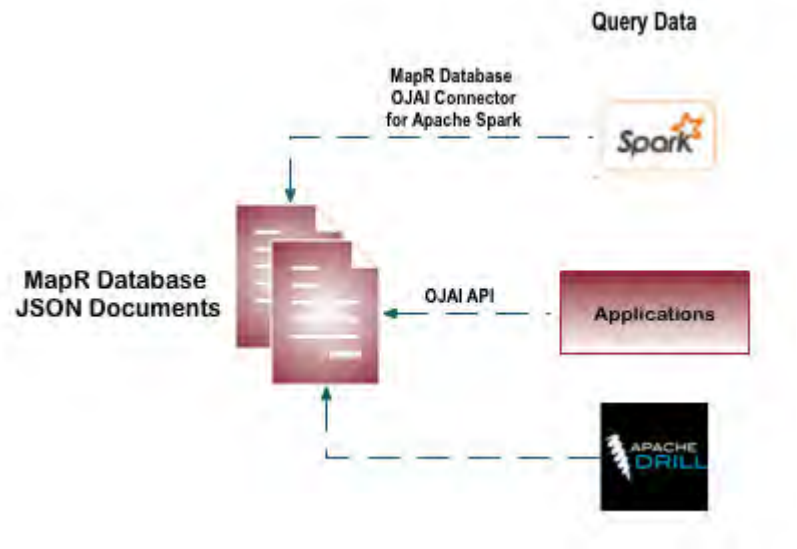
The following sections will provide some examples for how to access the data so that you can envision that will work for your use case.

- **Access Data in MapR XD Distributed File and Object Store**

  The most common way to access data in the MapR XD Distributed File and Object Store is via a NFS mount point that is remote or local to the cluster. You can use HDFS commands as well but they are generally only used for migrating hadoop applications to the MapR Data Platform platform. If you require high throughput, security, and scalability, consider installing the MapR POSIX client as this provides a more efficient way to access data in the MapR XD Distributed File and Object Store. You can also query the data directly using Drill.

- **Access Data in MapR Database**

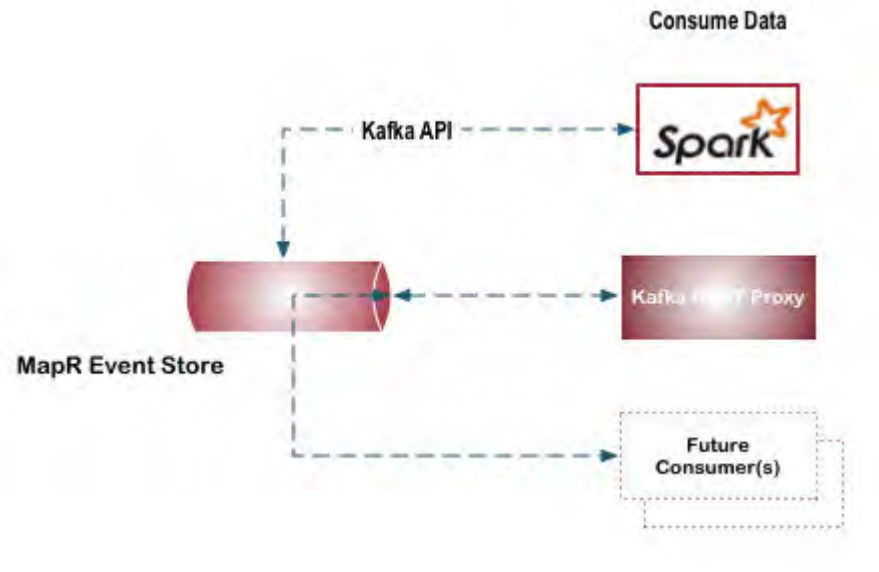
  The methods that you can use to access MapR Database table data differs based on the table type. You can access MapR Database binary table data with the HBase shell and applications that use the HBase API. You can access MapR Database JSON table data with the mapr dbshell, and java applications that use the OJAI API. You can also use Drill or Spark to query MapR Database binary and JSON table data directly.



- **Access Data in MapR Event Store For Apache Kafka**

  Data in MapR Event Store For Apache Kafka can be accessed by one or more stream consumers and the number of consumers can change over time depending on business needs.

  Similar to the various ways you can write data to topics a stream, data in stream topics can be accessed by applications that utilize the Kafka API or a REST interface. You can also use Spark to query streams for new messages at a given interval and access any new messages that are available.



  MapR Event Store For Apache Kafka provides flexibility to add new consumers without making changes to the producer application. For example, you have a MapR Event Store For Apache Kafka producer that writes all twitter feeds to a stream. Today, this stream is accessed by a single consumer application that provides access to twitter feeds with content related to IOT. The next week, there may be a request check for how many tweets originate from a specific account. Providing access to different data in an existing stream can be achieved by creating a new consumer which reads from the same stream.

Step 4: Set Up the Development Environment

Before you start building the application, figure out how your the application will connect to the cluster and what the library dependencies and installation requirements are.

last-updated: Apr 11, 2022

Applications are often run on edge nodes which are nodes that are not part of the cluster. Setting up an edge node so that it can be used to develop and run applications consists of the following steps:

- Selecting a client that you will use to connect to the cluster.
- Installing additional clients required for your use case.
- Determining the application dependencies.

Note: The dependencies to build and run applications differ based on your use case and the various types of data or tools that are part of an application.

The following sub-topics include the various methods to connect to the cluster, and minimal requirements to build and run applications.

Connect to the Cluster

Application are often run on nodes that are not part of the MapR Data Platform cluster. There are many methods to connect to a MapR cluster; this section briefly describes each option.

MapR Data Platform Client	The MapR Data Platform client includes the libraries and utilities required on an edge node to perform the following: connect to the cluster, submit MapReduce applications, submit YARN applications, run hadoop fs commands, and run hadoop mfs commands. However, to run applications that access data from MapR Database or MapR Event Store For Apache Kafka, you must configure additional dependencies. For more information about the MapR Data Platform client, see MapR Client on page 388 and How MapR clients Connect to the cluster on page 746. **Note:** Although it is not recommended, you can include the MapR File System JAR file in the application instead of installing the MapR Data Platform client. However, there are caveats and specific requirements to make this work. For information, see Using the File System JAR to Connect to the Cluster on page 2367.
MapR Data Platform Persistent Application Client Container (PACC)	The MapR Data Platform Persistent Application Client Container (PACC) is a Docker-based container image that includes a container-optimized MapR Data Platform client. The PACC provides seamless access to MapR Data Platform Converged Data Platform services, including MapR File System, MapR Database, and MapR Event Store For Apache Kafka. The PACC includes the POSIX client, the MapR Data Platform client, and the libraries required to build MapR Database and MapR Event Store For Apache Kafka applications. For more information, see About the MapR Persistent Application Client Container (PACC) on page 403.
MapR Data Platform POSIX Clients	MapR Data Platform POSIX clients enable app servers, web servers, and other client nodes and applications to read and write directly and securely to the filesystem. For more information about the POSIX clients, see POSIX Clients on page 492 and MapR POSIX Clients on page 399.
MapR Data Platform NFS Clients	You can mount the cluster itself via NFS so that your applications can read and write data directly. For more information, see Managing the MapR NFS Service on page 1176.

last-updated: Apr 11, 2022

Using the File System JAR to Connect to the Cluster

The MapR File System JAR file includes the data-fabric client libraries required to connect to the cluster. While this is strongly discouraged, application developers can bundle the MapR File System JAR file in data-fabric file system, MapR Database, and MapR Event Store For Apache Kafka applications instead of installing the data-fabric client on the edge node (node that runs the application). Applications should not bundle the MapR File System JAR file unless the application meets certain requirements.

Important: When bundling the MapR File System JAR file, if there is a binary mismatch between the bundled JAR file and the version that the cluster expects, this can result in failures. In release 5.2.2 and later, the system detects the mismatch and prevents the application from starting. In releases earlier than 5.2.2, nodes running applications may run out of memory or shut down unexpectedly.

Requirements

You can bundle the MapR File System JAR (`maprfs-<version>-mapr.jar`) with applications that meet all of the following requirements:

- The application communicates directly with the MapR File System, MapR Database, or MapR Event Store For Apache Kafka
- The application does not run as a MapReduce or YARN job/application on the cluster.
- The application does not include MapR File System JARs on the local machine in its classpath.
- The application accesses a cluster that is not secure.

Configuring the Cluster Connection

When you include the MapR File System JAR in an application instead of installing the data-fabric client on the edge node, you must create and configure a `mapr-clusters.conf` file on node that runs the application.

1. Set a `MAPR_HOME` environment variable to a location such as `/opt/mapr`.
2. Create the `mapr-clusters.conf` file in the `$MAPR_HOME/conf` directory.
3. Configure the `mapr-clusters.conf` file with the cluster name and the list of CLDB nodes.

   For example, the `mapr-clusters.conf` on an edge node would contain the following content if it was connecting to a cluster named `my.cluster` with CLDB nodes on centos765, centos234, and centos123:

   ```
   my.cluster secure=false centos765 centos234 centos123
   ```

   For more information about how to configure `mapr-clusters.conf`, see mapr-clusters.conf on page 2200.

For more information about how the data-fabric client connects to the cluster, see How MapR clients Connect to the cluster on page 746.

Using Maven to Include MapR File System JAR as a Dependency

If you use Maven to bundle the MapR File System JAR file with an application and you plan to run the application on a data-fabric cluster where a patch has been applied, ensure that you specify both a system scope and a local system path to the file.

For example, to bundle the MapR File System JAR file, the `pom.xml` file may include the following:

```
...
 <groupId>com.mapr.hadoop</groupId>
        <artifactId>maprfs</artifactId>
        <version>${mapr.core.version}</version>
        <scope>system</scope>
        <systemPath>/opt/mapr/lib/maprfs-5.2.0-mapr.jar</systemPath>
...
```

By default, the data-fabric Maven repository includes JAR files from https://repository.mapr.com/maven/. This default Maven repository includes JAR files associated with the GA packages for each data-fabric release. Therefore, when a patch has been applied to the cluster, failure to specify a system scope may result in errors due to a binary mismatch between the MapR File System JAR files used by the application and the cluster.

Known Issues

Nodes running applications with a bundled MapR File System JAR file may run out of memory or shut down unexpectedly in the following scenarios:

The version of the MapR File System JAR included in the application differs from the version that is available on the cluster.	This may occur when a patch was applied to some, but not all the nodes in the cluster. It can also occur when Maven is bundling the GA version of the JAR file when the cluster expects a newer, patched version.
Two versions of the JAR are available on the node.	For YARN applications, the NodeManager nodes that run the tasks or containers store local versions of the dependencies included with the application. In this scenario, since both the cluster's MapR File System JAR and the version included in the application are available on the node, it is unknown which JAR will be used when processing the application.

MapR Database JSON Application Requirements

The following tables include the minimal node requirements for building and running MapR Database JSON table applications.

Java Applications

Node Requirements	Method(s) to Meet Requirement
A connection to the MapR cluster.	Select one of the following options: • Install and configure the MapR client. • Install the PACC and run an application container. • Use the MapR File System JAR to connect to the cluster. For more information, see Connect to the Cluster on page 2366.
The OJAI Query Service is installed	To use secondary indexes, you may need to enable this service. See Preparing Clusters for Querying using Secondary Indexes on JSON Tables on page 1089 for more information.

Node Requirements	Method(s) to Meet Requirement
MapR Database libraries are configured as a dependency.	When you build an application, use the Maven Repository to determine the dependencies. The POM file should include the MapR Repository and the MapR OJAI Driver project. When you run the application, provide the dependencies in the application's classpath. For more information, see Compiling and Running Java OJAI Applications on page 2666
Other items to consider.	If an ecosystem component, such as Spark, runs or integrates with the application, you may need to include additional dependencies in the POM file. For example, to use the MapR Database OJAI Connector for Apache Spark, see Configuring the MapR Database OJAI Connector for Apache Spark on page 4052

MapR Database Binary Application Requirements

The following tables include the minimal node requirements for building and running MapR Database binary table applications.

Java Applications

Node Requirement	Method(s) to Meet Requirement
A connection to the MapR cluster.	Select one of the following options: • Install and configure the MapR client. • Install the PACC and run an application container. • Use the MapR File System JAR to connect to the cluster. For more information, see Connect to the Cluster on page 2366.
The HBase client is installed.	Install the HBase client. The HBase client is as part of the EEP installation. For more information, see 6.1 Installation on page 107. **Note:** The HBase client is include when you install PACC.
HBase client library files are configured as an application dependency.	When you build the application, use the Maven Repository to determine the dependencies. The POM file should include the MapR Repository and the HBase client dependency. When you run an application, include the `hbase classpath` script and additional dependencies in the application's classpath. For more information, see Compiling and Running MapR Database Binary Applications on page 2478
Other Items	If an ecosystem component, such as Spark, runs or integrates with the application, you may need to include additional dependencies in the POM file.

C Applications

Node Requirement	Method(s) to Meet Requirement
A connection to the MapR cluster.	Select one of the following options: • Install and configure the MapR client. • Install the PACC and run an application container. • Use the MapR File System JAR to connect to the cluster. For more information, see Connect to the Cluster on page 2366.
The HBase Client is installed.	Install the HBase client. The HBase client is as part of the EEP installation. For more information, see 6.1 Installation on page 107. **Note:** The HBase client is include when you install PACC.
The libMapRClient library and libjvm shared libraries are in the application's library search path and the `libMapRClient` header files are in this directory:`/opt/mapr/include/hbase`	For more information, see Building and Launching C Applications on page 2472.

MapR Event Store For Apache Kafka Application Requirements

The following tables include the minimal node requirements for building and running MapR Event Store For Apache Kafka consumer and producer applications.

Java Applications

Node Requirement	Method(s) to Meet Requirement
A connection to the MapR cluster.	Select one of the following options: • Install and configure the MapR client. • Install the PACC and run an application container. • Use the MapR File System JAR to connect to the cluster. For more information, see Connect to the Cluster on page 2366.
The MapR Streams Java Client is installed.	Install the MapR Event Store For Apache Kafka Java Client. The MapR Event Store For Apache Kafka Java client (mapr-kafka) is available as part of the EEP installation. For more information, see 6.1 Installation on page 107. **Note:** The MapR Streams Java Client is include when you install PACC.

last-updated: Apr 11, 2022

Node Requirement	Method(s) to Meet Requirement
MapR Streams Java client and the MapR Streams project library files are configured as an application dependency.	When you build the application, use the Maven Repository to determine the dependencies. The POM file should include the MapR Repository, the MapR Event Store For Apache Kafka Java Client dependency, and the MapR Event Store For Apache Kafka project dependency. When you run an application, include the dependencies in the application's classpath. For more information, see Compiling and Running MapR Event Store For Apache Kafka Java Apps on page 2777
Other Items	If an ecosystem component, such as Spark, runs or integrates with the application, you may need to include additional dependencies in the POM file.

MapR File System Application Requirements

The following tables include the minimal node requirements for building and running MapR File System applications.

Java Applications

Node Requirement	Method(s) to Meet Requirement
A connection to the MapR cluster.	Select one of the following options: • Install and configure the MapR client. • Install the PACC and run an application container. • Use the MapR File System JAR to connect to the cluster. For more information, see Connect to the Cluster on page 2366.

Node Requirement	Method(s) to Meet Requirement

Include hadoop-common libraries as a dependency.	When you compile the application, use the Maven Repository to determine the dependencies.The POM file should include the MapR Repository and the hadoop-common dependency:
	
	`<repositories>`
	`<repository> <id>mapr-releases</id>`
	`<url>https://repository.mapr.com/maven/</url>`
	`<snapshots><enabled>false</enabled></snapshots> <releases><enabled>true</enabled></releases>`
	`</repository>`
	`</repositories>`
	
	`<dependencies>`
	`<dependency>`
	`<groupId>org.apache.hadoop</groupId>`
	`<artifactId>hadoop-common</artifactId>`
	`<version>${hadoop.version}</version>`
	`</dependency>`
	`</dependencies>`
	
	When you run the application, include the following in the application's classpath: `` `hadoop classpath` ``
	
	For more information, see Compiling and Running a Java Application on page 2436.

Note:

When you develop a Java application, you can use a dependency management tool such as Maven to compile your application. However, it is recommended that do the following instead:

1. Compile the Java application without including dependencies
2. Specify the required classpath when you submit the application to the cluster

If you choose to bundle the JAR file, and there is a mismatch between the bundled JAR file and the version that your MapR cluster expects, this can result in failures. The failures differ depending on the version of MapR you are using. For more information, see Using the File System JAR to Connect to the Cluster on page 2367.

C Applications

Node Requirement	Method(s) to Meet Requirement
A connection to the MapR cluster.	Select one of the following options: • Install and configure the MapR client. • Install the PACC and run an application container. For more information, see Connect to the Cluster on page 2366.

Node Requirement	Method(s) to Meet Requirement
Include the libhdfs libraries and MapR libraries when you compile the application.	The MapR libraries are available in the following location: `/opt/mapr/lib` Link to the libhdfs libraries in the following location: `MAPR_HOME/hadoop/hadoop-2.x/` For more information, see Compiling and Running C Applications on File System Clients on page 2379

YARN Application Requirements

The following tables include the minimal node requirements for building and running YARN applications.

Node Requirement	Method(s) to Meet Requirement
A connection to the MapR cluster.	Select one of the following options: • Install and configure the MapR client. • Install the PACC and run an application container. For more information, see Connect to the Cluster on page 2366.

Hadoop libraries are configured as an application dependency.	When you compile the application, use the Maven Repository to determine the dependencies. The POM file should include the MapR Repository and the hadoop-common dependency:
	`<repositories>`
	`<repository>`
	`<id>mapr-releases</id> <url>https://repository.mapr.com/maven/</url>`
	`<snapshots><enabled>false</enabled></snapshots>`
	`<releases><enabled>true</enabled></releases>`
	`</repository>`
	`</repositories>`
	
	`<dependencies>`
	`<dependency>`
	`<groupId>org.apache.hadoop</groupId>`
	`<artifactId>hadoop-common</artifactId>`
	`<version>${hadoop.version}</version>`
	`</dependency>`
	`</dependencies>`
	When you run the application, include the following in the application's classpath: `` `hadoop classpath` ``
	Note: Based on how you submit the application, the classpath locations and requirements differ. See External Applications and Classpath on page 3030 and Classpath Construction on page 3031.

last-updated: Apr 11, 2022

Node Requirement	Method(s) to Meet Requirement
Other Items	• If an ecosystem component, such as Spark, runs or integrates with the application, you may need to include additional dependencies in the POM file. • Any third-party library that is required by a MapReduce program must be accessible to this node and the data node that processes the job or application. For more information, see Managing Third-Party Libraries on page 3031.

Step 5: Build the Application

Start building an application! This section lists a few of the sample applications available in MapR Data Platform.

Note: When building and running your application, you may want to use a classpath for printing to standard output. The following classpaths are available:

mapr classpath	Prints CLASSPATH to standard output. This classpath could be used to build your application classpath to runMapR Data Platform applications for YARN and other components.
mapr clientclasspath	Prints CLASSPATH for MapR Database clients to standard output. The clientclasspath could be used to build the classpath to run your MapR Database (binary and JSON) and MapR Event Store For Apache Kafka applications.

MapR Database JSON

Build Your Java Application on page 2519	This section shows how to build a Java application that uses OJAI and accesses MapR Database JSON tables.
Managing JSON Tables on page 2522	This section describes how to create, list, and delete JSON tables as well as set permissions and manage column families using either the MapR Data Platform Java API library or the MapR Database Shell commands.
Creating JSON Documents in Java OJAI on page 2545	This section provides several Java examples of creating a document.
Examples: Inserting JSON Documents on page 2554	This section shows how to insert a document or data into a document store (MapR Database JSON table) with Java and dbshell.
Examples: Querying JSON Documents on page 2624	This section provides several examples of querying documents.
MapR Database JSON MapReduce: Sample App on page 2715	This sample Java application extends the Apache Hadoop MapReduce framework to read records (JSON documents) from a JSON table and inserts new documents into another JSON table. This API library allows you to write your own MapReduce applications to write data from one JSON table to another.

last-updated: Apr 11, 2022

MapR Database Binary

C API Examples on page 2455	This section provides examples using the MapR Database libMapRClient C API to operate on MapR Database binary tables.
MapR Database Sample C Application on page 2459	This section provides additional sample C applications that accesses and performs operations on MapR Database binary tables.

MapR Event Store For Apache Kafka Streams

Sample Java Consumer on page 2721	This sample Java consumer application iterates through the returned records, extracts the value of each message, and prints the value to standard output.
Sample Java Producer on page 2722	This sample Java producer application publishes messages to a stream topic.
Developing a MapR Event Store For Apache Kafka C Application on page 2797	These sample C applications publish messages to a stream topic and consumes the messages.
Developing MapR Event Store For Apache Kafka Python Applications on page 2999	These sample Python applications publish messages to a stream topic and consumes the messages.

MapR File System

Sample Applications on page 2440	This sample Java application demonstrates how to set, get, modify, and delete ACEs on files using the Java APIs.
hdfs_write_revised.c on page 2384	This C application demonstrates how to write to files by using the hdfsWrite() and hdfsPwrite() APIs.
hdfs_read_revised.c on page 2391	This C application demonstrates how to read from files by using the hdfsRead() and hdfsPread() APIs.
hdfs_connect_as_user.c on page 2398	This C application demonstrates how to create and write to files impersonating another user by using the hdfsConnectAsUser() API.

Github Repo

You can find additional sample code on the MapR Demos Github repository.

MapR XD and Apps

The following sections provide information about accessing the MapR XD with C and Java applications.

Copying Data from Apache Hadoop to a MapR Cluster

Describes the procedure to copy data from an Apache Hadoop to a MapR cluster.

You can use the hdfs protocol, webhdfs protocol, or NFS to copy data from Apache Hadoop to a MapR cluster.

The following table describes these methods:

last-updated: Apr 11, 2022

Method	Description
hdfs:// protocol	You can use the `hadoop distcp` command with the `hdfs://` protocol to copy data from a HDFS cluster into a MapR cluster if the HDFS cluster and the MapR cluster use the same RPC protocol version. For all other scenarios, use the `webhdfs://` protocol or NFS gateway to copy data to a MapR cluster.
webhdfs:// protocol	You can use the `hadoop distcp` command with the `webhdfs://` protocol to copy data from a HDFS cluster into a MapR cluster.
NFS	You can mount a MapR cluster to a HDFS cluster using NFS mount and then use the `hadoop distcp` command to copy data between the two clusters.

Refer to the following sections for information about how to copy data from Hadoop to a MapR cluster:

Copy Data Using the hdfs:// Protocol
Describes the procedure to copy data from a HDFS cluster to a MapR cluster using the *hdfs://* protocol.

Before you can copy data from an HDFS cluster to a MapR cluster using the *hdfs://* protocol, you must configure the MapR cluster to access the HDFS cluster. To do this, complete the steps listed in Configuring a MapR Cluster to Access an HDFS Cluster for the security scenario that best describes your HDFS and MapR clusters, and then complete the steps listed under Verifying Access to an HDFS Cluster.

You also need the following information:

- `<NameNode>` - the IP address or hostname of the NameNode in the HDFS cluster
- `<NameNode Port>` - the port for connecting to the NameNode in the HDFS cluster
- `<HDFS path>` - the path to the HDFS directory from which you plan to copy data
- `<MapRFilesystem path>` - the path in the MapR cluster to which you plan to copy HDFS data
- `<file>` - a file in the HDFS path

To copy data from HDFS to MapR filesystem using the `hdfs://` protocol, complete the following steps:

1. Run the following Hadoop command to determine if the MapR cluster can read the contents of a file in a specified directory on the HDFS cluster:

   ```
   hadoop fs -cat <NameNode>:<NameNode port>/<HDFS path>/<file>
   ```

   Example

   ```
   hadoop fs -cat hdfs://nn1:8020/user/sara/contents.xml
   ```

2. If the MapR cluster can read the contents of the file, run the `distcp` command to copy the data from the HDFS cluster to the MapR cluster:

   ```
   hadoop distcp hdfs://<NameNode>:<NameNode Port>/<HDFS path> maprfs://
   <MapRFilesystem path>
   ```

   Example

   ```
   hadoop distcp hdfs://nn1:8020/user/sara maprfs:///user/sara
   ```

Copying Data Using the webhdfs:// Protocol

Describes how to copy data from a HDFS cluster to a MapR cluster using the *webhdfs://* protocol.

Before you can copy data from an HDFS cluster to a MapR cluster using the *webhdfs://* protocol, you must configure the MapR cluster to access the HDFS cluster. To do this, complete the steps listed in Configuring a MapR Cluster to Access an HDFS Cluster for the security scenario that best describes your HDFS and MapR clusters, and then complete the steps listed under Verifying Access to an HDFS Cluster.

The HDFS cluster must have WebHDFS enabled. Verify that the following parameter exists in the `hdfs-site.xml` file and that the value is set to `true`.

```
<property>
<name>dfs.webhdfs.enabled</name>
<value>true</value>
</property>
```

You also need the following information:

- `<NameNode>` - the IP address or hostname of the NameNode in the HDFS cluster
- `<NameNode HTTP Port>` - the HTTP port on the NameNode in the HDFS cluster
- `<HDFS path>` - the path to the HDFS directory from which you plan to copy data
- `<MapR filesystem path>` - the path in the MapR cluster to which you plan to copy HDFS data

To copy data from the HDFS to the MapR filesystem using the `webhdfs://` protocol, complete the following step:

Run the following command from a node in the MapR cluster:

```
hadoop distcp webhdfs://<NameNode>:<NameNode HTTP Port>/<HDFS path>
maprfs:///<MapR filesystem path>
```

Example

```
hadoop distcp webhdfs://nn2:50070/user/sara maprfs:///user/sara
```

Note: The triple slashes in `maprfs:///...` are required.

Copying Data Using NFS

Describes how to copy files from one MapR cluster to another using NFS.

If NFS is installed on the MapR cluster, you can mount the MapR cluster to the HDFS cluster and then copy files from one cluster to the other using `hadoop distcp`. If you do not have NFS installed and a mount point configured, see Accessing Data with NFS v3 on page 1183 and Managing the MapR NFS Service on page 1176.

To perform a copy using distcp via NFS, you need the following information:

- `<MapR NFS Server>` - the IP address or hostname of the NFS server in the MapR cluster
- `<maprfs_nfs_mount>` - the NFS export mount point configured on the MapR cluster; default is `/mapr`
- `<hdfs_nfs_mount>` - the NFS mount point configured on the HDFS cluster
- `<NameNode>` - the IP address or hostname of the NameNode in the HDFS cluster
- `<NameNode Port>` - the port on the NameNode in the HDFS cluster
- `<HDFS path>` - the path to the HDFS directory from which you plan to copy data

last-updated: Apr 11, 2022

- `<MapR filesystem path>` - the path in the MapR cluster to which you plan to copy HDFS data

To copy data from HDFS to the MapR filesystem using NFS, complete the following steps:

1. **Mount HDFS.**

   Issue the following command to mount the MapR cluster to the HDFS NFS mount point:

   ```
   mount <MapR NFS Server>:/<maprfs_nfs_mount> /<hdfs_nfs_mount>
   ```

   Example

   ```
   mount 10.10.100.175:/mapr /hdfsmount
   ```

2. **Copy data.**

   a. Issue the following command to copy data from the HDFS cluster to the MapR cluster:

   ```
   hadoop distcp hdfs://<NameNode>:<NameNode Port>/<HDFS path> file:///
   <hdfs_nfs_mount>/<MapR filesystem path>
   ```

   Example

   ```
   hadoop distcp hdfs://nn1:8020/user/sara/file.txt file:///hdfsmount/
   user/sara
   ```

   b. Issue the following command from the MapR cluster to verify that the file was copied to the MapR cluster:

   ```
   hadoop fs -ls /<MapR filesystem path>
   ```

   Example

   ```
   hadoop fs -ls /user/sara
   ```

Accessing the File System with C Applications

MapR Data Platform provides a modified version of `libhdfs` that supports access to the data-fabric file system. You can develop applications with C that read files, write to files, change file permissions and file ownership, create and delete files and directories, rename files, and change the access and modification times of files and directories.

`libMapRClient` supports and makes modifications to hadoop-2.x version of `libhdfs`. The API reference notes which APIs are supported by hadoop-2.x.

`libMapRClient`'s version of `libhdfs` contains the following changes and additions:

- There are no calls to a JVM, so applications run faster and more efficiently.
- Changes to APIs
  - *hadoop-2.x:* Support for `hdfsBuilder` structures for connections to HDFS is limited. Some of the parameters are ignored.
  - *hadoop-2.x:* `hdfsGetDefaultBlockSize()`: If the file system that the client is connected to is an instance of MapR File System, the returned value is 256 MB, regardless of the actual setting.
  - *hadoop-2.x:* `hdfsCreateDirectory()`: The parameters for buffer size, replication, and block size are ignored for connections to the data-fabric file system.

  - *hadoop-2.x:* `hdfsGetDefaultBlockSizeAtPath()`: If the file system that the client is connected to is an instance of MapR File System, the returned value is 256 MB, regardless of the actual setting.
  - *hadoop-2.x:* `hdfsOpenFile()`: The parameters for buffer size and replication are ignored for connections to the data-fabric file system.
- APIs that are unique to `libMapRClient` for hadoop-2.x
  - `hdfsCreateDirectory2()`
  - `hdfsGetNameContainerSizeBytes()`
  - `hdfsOpenFile2()`
  - `hdfsSetRpcTimeout()`
  - `hdfsSetThreads()`

Installing and Configuring MapR File System C Clients

Install the `mapr-client` package on the nodes on which you plan to build and run client applications. This package installs the `libMapRClient` library. See Setting Up the Client.

Note: The package `mapr-core` contains the files that are in the mapr-client package. If you have installed the `mapr-core` package, you do not need to install the `mapr-client` package.

The modified versions of `libhdfs` are installed in this directory:

`MAPR_HOME/hadoop/hadoop-2.x/`

Compiling and Running C Applications on File System Clients

MapR File System exposes the HDFS API; if you already have a client program built to use `libhdfs`, you do not have to relink your program just to access the MapR file system. However, re-linking to the MapR-specific shared library `libMapRClient.so` will give you better performance on MapR file system, because it does not make any Java calls to access the file system (unlike `libhdfs.so`).

The script below sets environment variables to necessary values and compiles one of the sample applications. Use this script as an example for building and launching your own applications.

When you set HADOOP_HOME for your own client applications, set it to the path for the version of `libhdfs` that your application uses. The path is:

`MAPR_HOME/hadoop/hadoop-2.x/`

Also, set the path to your application in the `gcc` command, of course.

This script assumes that `MAPR_HOME` is set to the default value of `/opt/mapr`.

```
#!/bin/bash
#Setup environment
export HADOOP_HOME=${MAPR_HOME}/hadoop/hadoop-2.7.0/
export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:${MAPR_HOME}/lib/native/
export LD_RUN_PATH=${LD_RUN_PATH}:${MAPR_HOME}/lib
GCC_OPTS="-Wl,--allow-shlib-undefined -I. -I{$HADOOP_HOME}/include/"

#Compile and Link
gcc ${GCC_OPTS} ${HADOOP_HOME}src/c++/libhdfs/hdfs_read.c -o hdfs_read -L$
{MAPR_HOME}/lib -lMapRClient
```

```
#Launch the application
./hdfs_read
```

Note:

- The compiled `libMapRClient` is statically linked to the following third-party libraries:
  - Crypto++: `libcryptoapp.a` (v5.6.2)
  - Protobuf: `libprotobuf-lite.a` (v2.5.0)
- If a client application connects to the local fileserver, before launching the application you can set the MAPR_CLIENT_SHMEM environment variable to control how much of the local system's memory should be devoted to the resources and buffers used for communication between the client application and the local fileserver. By default, the size of the shared memory is 20 MB. If you want to change this value, specify it as a number of pages. For example, to set the shared memory at 128 MB, multiply 128 by 10242 bytes and then divide the product by 8192 bytes. In this case, the value would be 16384 pages.

Overview of the File System C APIs in libMapRClient

Although you can use the file system C APIs to perform other tasks, the most common use of the file system C APIs is to write to and read from files.

Review the following information for an overview of the steps required to use MapR File System C APIs in libMapRClient:

1. Create a connection to the MapR file system running on a MapR cluster. For information about connections, see Establishing Connections to Filesystems.
2. Create or open a file. You can create explicitly or by calling one of the `hdfsOpen()` APIs and specifying a file that doesn't exist. Opening a file sets the current offset to 0, the first byte in the file. You can move file offset explicitly with the `hdfsSeek()` API. Writes done by `hdfsWrite()` and reads done by `hdfsRead()` increment the offset by the number of bytes written or read.

   Use `hdfsTell()` to find out what the current offset is.

   For information about how to specify the location of a file to create or open, see Specifying Paths to Files and Directories.
3. Write to or read from the file. When you write to a file, you pass a buffer that contains the data that you want to write. Writes can be done from the default offset, which is 0 when the file is opened, appended to the end of the file, or done from an offset that you specify. Write buffers are flushed to the server periodically. For more information about writes, seeWriting to Files.

   When you read from a file, you pass a pointer to a buffer for storing the data that is read. Reads can be done from the default offset or from an offset that you specify. For more information about reads, see Reading from Files.
4. Close the file.

   Closing a file implicitly flushes any remaining write buffers to the server. It also frees resources that are associated with the file.
5. Disconnect from the file system.

For more detailed information about these steps, see:

Establishing Connections to the File System

The APIs for establishing connections to the MapR File System and returning file system handles are:

- hadoop-2.x: `hdfsConnect()`
- hadoop-2.x: `hdfsConnectAsUser()`

  Note: This API ignores the impersonation request and therefore is equivalent to hdfsConnect().

- hadoop-2.x: `hdfsConnectNewInstance()`

The API `hdfsConnectAsNewUserInstance()` is not supported for connections to MapR File System fileservers.

These APIs behave in the same way:

- If "`default`" is specified for the `host` parameter, the APIs connect to the first cluster listed in the file `MAPR_HOME/conf/mapr-clusters.conf`. (`MAPR_HOME` defaults to /opt/mapr.)
- If a hostname or IP address is specified for the `host` parameter:
  1. Look in `MAPR_HOME/conf/mapr-clusters.conf` on the client node to match the specified hostname or IP address to a CLDB host and port.
  2. If they find a match, they try to connect to the cluster and all standard features for connections to MapR clusters are available. These features include high availability across CLDBs and secure connections.
  3. If they do not find a match or if they cannot locate a `mapr-clusters.conf` file, they try to connect to the CLDB host specified in the call to create the connection. However, the standard features for connections to MapR clusters are not available. For example, if the cluster is secured, the connection will fail.

It is possible to have more than one open connection at a time. For each connection, simply return the file system handle to a different instance of `hdfsFS`, as in this example:

```
    //Connect to Cluster 1 (picked up from /opt/mapr/conf/
mapr-clusters.conf)
        hdfsFS fs1 = hdfsConnectNewInstance("default", 7222);
    //Connect to Cluster 2
        hdfsFS fs2 = hdfsConnectNewInstance("n1c", 7222);
    //Connect to Cluster 3
        hdfsFS fs3 = hdfsConnectNewInstance("n1d", 7222);
```

You can then obtain file handles for files in each connected cluster, as in this example. For each cluster, this example code calls `hdfsOpenFile()`, passing in the handle to the file system, the absolute path to a file (and the file is created before being opened, if it doesn't already exist) and a file-access flag that specifies to open the file in write-only mode. This mode truncates existing files to offset 0, deleting their content.

Ignore the last three parameters for this example. `hdfsOpenFile()` returns a handle to the file or an error message, if the open operation fails.

```
     //Create files for write operations on all clusters
        const char* writePath = "/tmp/write-file1.txt";
        hdfsFile writeFile1 = hdfsOpenFile(fs1, writePath, O_WRONLY, 0, 0,
0);
        if (!writeFile1) {
        fprintf(stderr, "Failed to open %s for writing on Cluster 1!\n",
writePath);
            exit(-2);
     }
        hdfsFile writeFile2 = hdfsOpenFile(fs2, writePath, O_WRONLY, 0, 0,
```


last-updated: Apr 11, 2022

```
0);
        if (!writeFile2) {
        fprintf(stderr, "Failed to open %s for writing on Cluster 2!\n",
writePath);
            exit(-2);
     }
         hdfsFile writeFile3 = hdfsOpenFile(fs3, writePath, O_WRONLY, 0, 0,
0);
     if (!writeFile3) {
        fprintf(stderr, "Failed to open %s for writing on Cluster 3!\n",
writePath);
            exit(-2);
     }
         fprintf(stderr, "Opened %s for writing successfully on all 3
clusters...\n", writePath);
```

After working with the files, close them and disconnect from the file systems, as in this example:

```
// Close all files
if (writeFile1)
        hdfsCloseFile(fs1, writeFile1);
if (writeFile2)
        hdfsCloseFile(fs2, writeFile2);
if (writeFile3)
        hdfsCloseFile(fs3, writeFile3);

// Disconnect from all clusters
hdfsDisconnect(fs1);
hdfsDisconnect(fs3);
hdfsDisconnect(fs3);
```

Specifying Paths to Files and Directories

Many of the APIs require clients to pass a path to a file or directory. You can specify absolute paths or relative paths. Absolute paths must begin with a forward slash. Relative paths are relative to the working directory, which you can set by calling `hdfsSetWorkingDirectory()` and find out by calling `hdfsGetWorkingDirectory()`. Any path that does not begin with a forward slash is considered to be relative to the working directory.

The maximum length of paths is 4096 bytes.

You cannot specify paths to a cluster other than the cluster for the current connection. All paths are local to the cluster connected to. You can, however, explicitly connect to multiple clusters, as described in Establishing Connections to the File System on page 2380.

Writing to Files

There are two APIs for writing to files: `hdfsWrite()` and `hdfsPwrite()`. With both APIs, you pass a buffer that contains the data to write. You also pass the length of the buffer in bytes. There maximum length of the buffer is the maximum size of the datatype that is used to specify the buffer length. The datatype is a custom datatype: `tSize`, a signed 32-bit integer.

Both APIs return the number of bytes that were written. Flushes to the server happen automatically at intervals during a write operation. After a write operation is finished, either call `hdfsFlush()` explicitly or call `hdfsFlush()` implicitly by calling `hdfsCloseFile()` to be sure that any data remaining in the write buffer is flushed.

For an example of both APIs in action, see `hdfs_write_revised.c`.

Note: The `core-site.xml` flags:

- `fs.mapr.flush.unaligned` default setting (`false`) enables flushes to the server in 8K boundaries. Unaligned flushes can happen only if idle flusher (`fs.mapr.write.idleflush.timeout`) is triggered. If this behavior is not desired, set the value for `fs.mapr.flush.unaligned` to `true`, which will enable flushing of unaligned write buffers (so that even small writes can be flushed on every subsequent write call).
- `fs.mapr.write.idleflush.timeout` automatically flushes the buffer, by default, after 3 seconds for all the open files. This can be disabled by setting the value to `0`. If value is specified, buffer is flushed automatically between *n* to *n*+1 seconds. For example, if value is 3 seconds, the write buffer is not cached after 4 seconds.

See also: Default core Parameters.

Using hdfsWrite()

When a file is opened in write-only mode or read-write mode, the file is truncated from offset 0, effectively deleting the content of the file. Therefore, the initial write to the file begins at offset 0. You can start subsequent writes anywhere in the file after first calling `hdfsSeek()` to move to the desired offset. After a write operation, the offset is located at the last written byte.

If the file is opened in append mode, data is appended to the end of the file only.

If a call to `hdfsSeek()` moves the offset past the end of the file before a call to `hdfsWrite()`, the result is a hole in the file between the previous end of the file and the offset at which the write begins.

You can obtain the size of a file in bytes by calling `hdfsGetPathInfo`.()

On error, pending write buffers are flushed to the server.

Using hdfsPwrite()

Whereas `hdfsWrite()` increments the current offset by the amount of bytes returned by the API (except in case of error), `hdfsPwrite()` does not change the value of current offset. If the current offset before the call to `hdfsPwrite()` is 0 and you specify the offset 10 for the write operation, after the write the current offset remains 0.

If a call to `hdfsPwrite()` specifies an offset that is past the end of the file, the result is a hole in the file between the previous end of the file and the offset at which the write begins.

You can obtain the size of a file in bytes by calling `hdfsGetPathInfo()`.

On error, pending write buffers are flushed to the server.

Reading from Files

There are two APIs for reading from files: `hdfsRead()` and `hdfsPread()`. With both `hdfsRead()` and `hdfsPRead()`, you pass a pointer to a buffer for the runtime to read bytes into and the length of the buffer. There maximum length of the buffer is the maximum size of the datatype that is used to specify the buffer length. The datatype is a custom datatype: `tSize`, a signed 32-bit integer.

Both functions return the number of bytes that are actually read.

For an example of both APIs in action, see `hdfs_read_revised.c`.

Using hdfsRead()

Whenever you open a file, the file pointer is placed at offset 0. If you want to start reading at an offset other than 0, call `hdfsSeek()` to move the file pointer forward to that offset before you call `hdfsRead()`.

last-updated: Apr 11, 2022

When you call `hdfsSeek()`, you specify the offset as a value of type `tOffset`, which is a fixed-width, signed 64-byte integer type for storing offsets. `tOffset` is defined in `hdfs.h`.

If a file is already open and you are not sure what the current offset is, you can find out by calling `hdfsTell()`.

After `hdfsRead()` finishes a read operation, the current offset is set to the last byte read plus one.

Using hdfsPread()

With `hdfsPread()`, you specify the offset at which you want to start reading, so you don't first have to call `hdfsSeek()` to move to that offset.

However, the offset that you specify does not change the current offset in the file. After `hdfsPread()` finishes the read operation, the current offset is not set to the last byte read plus one. Instead, the current offset remains as it was before the read operation.

Sample Applications

The following applications demonstrate how to write to and read from files using the APIs:

hdfs_write_revised.c

Sample Application

This application demonstrates how to write to files by using the APIs `hdfsWrite()` and `hdfsPwrite()`: `hdfs_write_revised.c`

Before running this application:

- Ensure that you have access either to a cluster running MapR File System.
- Ensure that a text-based file that you have access to exists on the cluster. Note the path to the file and the size of the file in bytes.
- The content of the file will be deleted before the first write is performed by the application.
- Decide on the length in bytes of a string to write to the file.

To build and run it, download it from this page to a MapR client or to a system with the `mapr-core` package installed. Then, modify the `run.sh` script in Building and Running C Applications on MapR File System Clients to point to this sample application. Run the script and then run the application.

The application includes these header files:

- `stdio.h`
- `hdfs.h`
- `errno.h`
- `fcntl.h`

The APIs are defined in `hdfs.h`. The file `fcntl.h` defines the file-access flags.

The application performs the actions that are described in the following sections.

Takes a filename, file size, and buffer size as input	When you launch the application, provide the path and name of the file, the size of the file, and the number of bytes to write. `hdfs_write <filename> <filesize> <buffersize>`

Sets an RPC timeout

`hdfsSetRpcTimeout()` is specific to the `libMapRClient` version of `libhdfs` and takes a value that is specified in seconds. The default is 99 seconds. If you change this value, set it either to 0 (which eliminates timeouts) or to a value greater than 30.

```
int err = hdfsSetRpcTimeout(30);
if (err) {
  fprintf(stderr, "Failed to set rpc
timeout!\n");
  exit(-1);
}
```

Connects to a filesystem, using an API that is supported in the hadoop-2.x version of libhdfs

The application tries to connect to the first MapR File System cluster that is specified in the `mapr-clusters.conf` file in the `MAPR_HOME/conf` directory on the client. After connecting to the filesystem, the application returns a handle to the filesystem.

```
hdfsFS fs = hdfsConnect("default", 0);
if (!fs) {
  fprintf(stderr, "Oops! Failed to
connect to hdfs!\n");
  exit(-1);
}
```

Stores the values of the arguments

The application stores the values of the arguments in a character array and in two variables of type `tSize`. This datatype is defined in `hdfs.h` and is a fixed-width, signed 32-byte integer type for storing the size of data for read or write operations.

```
const char* rfile = argv[1];
tSize fileSize = strtoul(argv[2],
NULL, 10);
tSize bufferSize = strtoul(argv[3],
NULL, 10);
```

Opens the file that you specified

The application opens the specified file, passing the following values to the `hdfsOpenFile()` function:

- The handle to the filesystem
- The name of the file, which you supplied when you launched the application.
- A flag to indicate the mode in which to open the file. In this case, the flag is `O_WRONLY`. This flag creates the file if the file does not exist and truncates the file if the file does exist. If the file existed and you wanted to preserve the content of the file, you would specify `O_WRONLY | O_APPEND` for flag. These flags are defined in the header file `fcntl.h`.
- The default chunk size for the directory in which the file is either located or will be created. This value is specified by the 0 in the last parameter.

Although there are two other parameters in the `hdfsOpenFile()` function – the fourth and fifth, the `libMapRClient` version of `libhdfs` ignores them.

```
hdfsFile writeFile = hdfsOpenFile(fs,
rfile, O_WRONLY, 0, 0, 0);
if (!writeFile) {
   fprintf(stderr, "Failed to open %s
for writing!\n", rfile);
   exit(-2);
}
```

Creates a buffer of the size that you specified and populates the buffer

At this point that the application, creates a string to populate the buffer. This is the data that the application will write.

```
char* buffer = malloc(sizeof(char) *
bufferSize);
if(buffer == NULL) {
  fprintf(stderr, "Failed to allocate
memory!\n");
  return -2;
}
int i;
  for (i=0; i<bufferSize; i++) {
  buffer[i] = 'a' + i%26;
}
```

Writes an entire file with hdfsWrite()

The application calls the function `writeLength()`:

```
int ret = writeLength(fs, writeFile,
buffer, bufferSize, fileSize);
if (ret < 0) {
  goto done;
}
```

This function writes the content of the buffer to the file, starting at offset 0.

```
int
writeLength(hdfsFS fs, hdfsFile
writeFile, char *buffer, tSize
bufferSize, tSize writeSize)
{
  tSize writeBytes = 0;
  tSize ret = 0;
  uint64_t totalWrite = 0;
  if (fs == NULL || writeFile == NULL
|| buffer == NULL) {
    return -1;
  }
  if (writeSize == 0) {
    return 0;
  }
  for
(writeBytes=0; writeBytes<writeSize;
writeBytes+=bufferSize) {
    ret = hdfsWrite(fs, writeFile,
(void*)buffer, bufferSize);
    if (ret > 0) {
```

```
      totalWrite += ret;
    } else {
      fprintf(stderr, "hdfsWrite
failed with error %d \n", errno);
      hdfsCloseFile(fs, writeFile);
      return -1;
    }
  }
  return 0;
}
```

Seeks an offset and writes from that offset with hdfsWrite()

The application next calls the function `writeAtOffse()`:

```
  tSize writeBytes =
writeAtOffset(fs, writeFile, 0,
buffer, bufferSize);
if (writeBytes < 0) {
  goto done;
}
```

This function writes the content of the buffer to the file, starting at the specified offset. If the file already exists, the file is first truncated to this offset before the write operation begins. In this case, the specified offset is 0.

The difference between this function and the previous function is that, before writing, it calls `hdfsSeek()` to move to the specified offset in the file.

```
tSize
writeAtOffset(hdfsFS fs, hdfsFile
writeFile, tOffset offset,
              char *buffer, tSize
bufferSize)
{
  tSize ret = 0;
  if (fs == NULL || writeFile == NULL
|| buffer == NULL) {
    return -1;
  }
  ret = hdfsSeek(fs, writeFile,
offset);
  if (!ret) {
    //hdfsWrite will return -1 if
ret != number of bytes asked to
    //be written.
    ret = hdfsWrite(fs, writeFile,
buffer, bufferSize);
    if (ret < 0) {
      fprintf(stderr, "hdfsWrite
failed with error %d \n", errno);
    }
  } else {
    fprintf(stderr, "hdfsSeek failed
with error %d \n", errno);
  }
  if (ret < 0) {
    //hdfsWrite does a flush in case
of an error, explicit flush
    //is not required.
    hdfsCloseFile(fs, writeFile);
```

```
  }
  //Current offset within the file
will be positioned at (offset +
writeBytes)th byte.
  return ret;
}
```

Performs a positional write with hdfsPwrite()

The application next calls the function `positionalWrite()`:

```
writeBytes = positionalWrite(fs,
writeFile, 20, buffer, bufferSize);
if (writeBytes < 0) {
  goto done;
}
```

This function writes the content of the buffer to the file, starting at the offset that you specify.

```
tSize
positionalWrite(hdfsFS fs, hdfsFile
writeFile, tOffset offset,
               char *buffer, tSize
bufferSize)
{
  tSize writeBytes = 0;
  if (fs == NULL || writeFile == NULL
|| buffer == NULL) {
    return -1;
  }
  writeBytes = hdfsPwrite(fs,
writeFile, offset, buffer,
bufferSize);
  if (writeBytes < 0) {
    fprintf(stderr, "hdfsPwrite
failed with error %d \n", errno);
    hdfsCloseFile(fs, writeFile);
  }
  //Current offset within the file
will not be advanced if hdfsPwrite is
used
  return writeBytes;
}
```

Closes the file

```
hdfsCloseFile(fs, writeFile);
```

Frees the buffer

```
free(buffer);
```

Disconnects from the filesystem

```
hdfsDisconnect(fs);
```

Example hdfs_write_revised.c File

```
/**
 * Licensed to the Apache Software Foundation (ASF) under one
 * or more contributor license agreements.  See the NOTICE file
 * distributed with this work for additional information
```

```
 * regarding copyright ownership.  The ASF licenses this file
 * to you under the Apache License, Version 2.0 (the
 * "License"); you may not use this file except in compliance
 * with the License.  You may obtain a copy of the License at
 *
 *     http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

#include <stdio.h>
#include "hdfs.h"
#include <errno.h>
#include <fcntl.h>

tSize
writeAtOffset(hdfsFS fs, hdfsFile writeFile, tOffset offset,
              char *buffer, tSize bufferSize)
{
  tSize ret = 0;

  if (fs == NULL || writeFile == NULL || buffer == NULL) {
    return -1;
  }

  ret = hdfsSeek(fs, writeFile, offset);
  if (!ret) {
    //hdfsWrite will return -1 if ret != number of bytes asked to
    //be written.
    ret = hdfsWrite(fs, writeFile, buffer, bufferSize);
    if (ret < 0) {
      fprintf(stderr, "hdfsWrite failed with error %d \n", errno);
    }
  } else {
    fprintf(stderr, "hdfsSeek failed with error %d \n", errno);
  }

  if (ret < 0) {
    //hdfsWrite does a flush in case of an error, explicit flush
    //is not required.
    hdfsCloseFile(fs, writeFile);
  }

  //Current offset within the file will be positioned at (offset +
writeBytes)th byte.
  return ret;
}

tSize
positionalWrite(hdfsFS fs, hdfsFile writeFile, tOffset offset,
              char *buffer, tSize bufferSize)
{
  tSize writeBytes = 0;

  if (fs == NULL || writeFile == NULL || buffer == NULL) {
    return -1;
  }

  writeBytes = hdfsPwrite(fs, writeFile, offset, buffer, bufferSize);
  if (writeBytes < 0) {
```

```
    fprintf(stderr, "hdfsPwrite failed with error %d \n", errno);
    hdfsCloseFile(fs, writeFile);
  }

  //Current offset within the file will not be advanced if hdfsPwrite is
used
  return writeBytes;
}

int
writeLength(hdfsFS fs, hdfsFile writeFile, char *buffer, tSize bufferSize,
tSize writeSize)
{
  tSize writeBytes = 0;
  tSize ret = 0;
  uint64_t totalWrite = 0;

  if (fs == NULL || writeFile == NULL || buffer == NULL) {
    return -1;
  }

  if (writeSize == 0) {
    return 0;
  }

  for (writeBytes=0; writeBytes<writeSize; writeBytes+=bufferSize) {
    ret = hdfsWrite(fs, writeFile, (void*)buffer, bufferSize);
    if (ret > 0) {
      totalWrite += ret;
    } else {
      fprintf(stderr, "hdfsWrite failed with error %d \n", errno);
      hdfsCloseFile(fs, writeFile);
      return -1;
    }
  }

  return 0;
}

int
main(int argc, char **argv)
{
  if (argc != 4) {
    fprintf(stderr, "Usage: hdfs_write <filename> <filesize>
<buffersize>\n");
    exit(-1);
  }

  int err = hdfsSetRpcTimeout(30);
  if (err) {
    fprintf(stderr, "Oops! Failed to set rpc timeout!\n");
    exit(-1);
  }

  hdfsFS fs = hdfsConnect("default", 0);
  if (!fs) {
    fprintf(stderr, "Oops! Failed to connect to hdfs!\n");
    exit(-1);
  }

  const char* rfile = argv[1];
  tSize fileSize = strtoul(argv[2], NULL, 10);
  tSize bufferSize = strtoul(argv[3], NULL, 10);
```


last-updated: Apr 11, 2022

```
  //O_WRONLY creates the file if the file doesn't exist.
  //O_WRONLY truncates the file if the file exists.
  //O_WRONLY | O_APPEND will preserve the contents of the file if the file
exists.
  hdfsFile writeFile = hdfsOpenFile(fs, rfile, O_WRONLY, 0, 0, 0);
  if (!writeFile) {
    fprintf(stderr, "Failed to open %s for writing!\n", rfile);
    exit(-2);
  }

  char* buffer = malloc(sizeof(char) * bufferSize);
  if(buffer == NULL) {
    fprintf(stderr, "Failed to allocate memory!\n");
    return -2;
  }

  int i;
  for (i=0; i<bufferSize; i++) {
    buffer[i] = 'a' + i%26;
  }

  //Write entire file from the beginning
  int ret = writeLength(fs, writeFile, buffer, bufferSize, fileSize);
  if (ret < 0) {
    goto done;
  }

  //Write file at a particular offset
  //In this case, we are writing from offset 0
  tSize writeBytes = writeAtOffset(fs, writeFile, 0, buffer, bufferSize);
  if (writeBytes < 0) {
    goto done;
  }

  //Write file at a particular offset using positional write
  //In this case, write from offset 20
  writeBytes = positionalWrite(fs, writeFile, 20, buffer, bufferSize);
  if (writeBytes < 0) {
    goto done;
  }

  hdfsCloseFile(fs, writeFile);
done:
  free(buffer);
  hdfsDisconnect(fs);

  return 0;
}

/**
 * vim: ts=4: sw=4: et:
 */
```

hdfs_read_revised.c

Sample Application

This application demonstrates how to read from files by using the APIs `hdfsRead()` and `hdfsPread()`: `hdfs_read_revised.c`

Before running this application:

- Ensure that you have access to a cluster running MapR File System.

- Ensure that a text-based file that you have access to exists on the cluster. Note the path to the file.
- Decide on the number of bytes to read from the file.

To build and run it, download it from this page and copy it to a MapR client. Then, modify the `run.sh` script in Building and Running C Applications on MapR File System Clients to point to this sample application. Run the script and then run the application.

The application includes these header files:

- `stdio.h`
- `hdfs.h`
- `errno.h`
- `fcntl.h`

The APIs are defined in `hdfs.h`. The file `fcntl.h` defines the file-access flags.

The application performs the actions that are described in the following sections.

Takes a filename and buffer size as input

After compiling the application, type the following command to launch the application and pass in the path and name of the file, as well as the size of the buffer to read data into:

```
hdfs_read <filename> <buffer size>
```

Sets an RPC timeout

`hdfsSetRpcTimeout()` is specific to the `libMapRClient` version of `libhdfs` and takes a value that is specified in seconds. The default is 99 seconds. If you change this value, set it either to 0 (which eliminates timeouts) or to a value greater than 30.

```
int err = hdfsSetRpcTimeout(30);
if (err) {
  fprintf(stderr, "Failed to set rpc
timeout!\n");
  exit(-1);
}
```

Connects to a filesystem, using an API that is supported in the hadoop-2.x version of libhdfs

The application tries to connect to the first MapR File System cluster that is specified in the `mapr-clusters.conf` file in the `MAPR_HOME/conf` directory on the client. After connecting to the filesystem, the application returns a handle to the filesystem.

```
hdfsFS fs = hdfsConnect("default", 0);
if (!fs) {
  fprintf(stderr, "Oops! Failed to
connect to hdfs!\n");
  exit(-1);
}
```

Stores the values of the arguments

The application stores them in a character array and in a variable of type `tSize`. This datatype is defined in `hdfs.h` and is a fixed-width, signed 32-byte integer

type for storing the size of data for read or write operations.

```
const char* rfile = argv[1];
tSize bufferSize = strtoul(argv[2],
NULL, 10);
```

Opens the file that you specified

The application opens the specified file, passing the following values to the `hdfsOpenFile()` function:

- The handle to the filesystem
- The name of the file, which you supplied when you launched the application.
- A flag to indicate the mode in which to open the file. In this case, the flag is `O_RDONLY`, which specifies read-only mode.
- The default chunk size for the directory in which the file is either located or will be created. This value is specified by the 0 in the last parameter.

Although there are two other parameters in the `hdfsOpenFile()` function – the fourth and fifth, the `libMapRClient` version of `libhdfs`ignores them.

```
hdfsFile readFile = hdfsOpenFile(fs,
rfile, O_RDONLY, 0, 0, 0);
if (!readFile) {
  fprintf(stderr, "Failed to open %s
for reading!\n", rfile);
  exit(-2);
}
```

Creates a buffer of the size that you specified

This is the buffer that the application will read data into.

```
  char* buffer = malloc(sizeof(char)
* bufferSize);
if(buffer == NULL) {
  fprintf(stderr, "Failed to allocate
memory!\n");
  return -2;
}
```

Reads an entire file with hdfsRead

The application calls the function `readEntireFile()`:

```
//Read entire file from the beginning
int ret = readEntireFile(fs,
readFile, buffer, bufferSize);
if (ret < 0) {
  goto done;
}
```

This function uses a WHILE loop. In each loop iteration, the function reads an amount of data that is equal to the size of the buffer. When the amount of bytes read is less than the size of the buffer, the end of the file has been reached and the function breaks the

loop. The number of bytes read is added to a total in each iteration.

```
int
readEntireFile(hdfsFS fs, hdfsFile
readFile, char *buffer, tSize
bufferSize)
{
  tSize readBytes = bufferSize;
  uint64_t totalRead = 0;
  if (fs == NULL || readFile == NULL
|| buffer == NULL) {
    return -1;
  }
  while (readBytes == bufferSize) {
    readBytes = hdfsRead(fs,
readFile, (void*)buffer, bufferSize);
    if (readBytes > 0) {
      totalRead += readBytes;
    } else {
      if (readBytes < 0) {
        fprintf(stderr, "hdfsRead
failed with error %d \n", errno);
        hdfsCloseFile(fs, readFile);
        return -1;
      }
      break;
    }
  }
  return 0;
}
```

Seeks an offset and reads from that offset with hdfsRead()

The application next calls the function `readAtOffset()`, passing in 0 as the offset from which to start reading the file.

```
//Read file at a particular offset
//In this case, we are reading from
offset 0
tSize readBytes = readAtOffset(fs,
readFile, 0, buffer, bufferSize);
if (readBytes < 0) {
  goto done;
}
```

This function calls `hdfsSeek()` to move to the specified offset in the file.

If the seek is successful, the function reads from that offset until the buffer is full. The function then returns the number of bytes that were read.

If the seek or the read is not successful (meaning `hdfsSeek()` or `hdfsRead()` returned -1), the function closes the file and returns -1.

The offset in the file is the next byte after the end of the data that was read.

```
tSize
readFromOffset(hdfsFS fs, hdfsFile
readFile, tOffset offset,
               char *buffer, tSize
```

```
bufferSize)
{
  tSize ret = 0;
  if (fs == NULL || readFile == NULL
|| buffer == NULL) {
    return -1;
  }
  ret = hdfsSeek(fs, readFile,
offset);
  if (!ret) {
    ret = hdfsRead(fs, readFile,
buffer, bufferSize);
    if (ret < 0) {
      fprintf(stderr, "hdfsRead
failed with error %d \n", errno);
    }
  } else {
    fprintf(stderr, "hdfsSeek failed
with error %d \n", errno);
  }
  if (ret < 0) {
    hdfsCloseFile(fs, readFile);
  }
  //Current offset within the file
will be positioned at (offset +
readBytes)th byte.
  return ret;
}
```

Performs a positional read with hdfsPread()

The application calls `positionalRead()`, passing 100 as the offset from which to start the read.

```
  readBytes = positionalRead(fs,
readFile, 100, buffer, bufferSize);
if (readBytes < 0) {
  goto done;
}
```

The function reads data into the buffer, starting at offset 100, without first calling `hdfsSeek()` to move the offset to that position. The offset is not moved to 100 before the read begins. The offset stays where it is, the read begins at offset 100, and (after the read) the offset remains where it was before the read. The offset in the file is ignored by the positional read.

```
tSize
positionalRead(hdfsFS fs, hdfsFile
readFile, tOffset offset,
               char *buffer, tSize
bufferSize)
{
  tSize readBytes = 0;
  if (fs == NULL || readFile == NULL
|| buffer == NULL) {
    return -1;
  }
  readBytes = hdfsPread(fs, readFile,
offset, buffer, bufferSize);
  if (readBytes < 0) {
    fprintf(stderr, "hdfsPread failed
```

```
with error %d \n", errno);
    hdfsCloseFile(fs, readFile);
  }
  //Current offset within the file
will not be advanced if hdfsPread is
used
  return readBytes;
}
```

Closes the file

```
hdfsCloseFile(fs, readFile);
```

Frees the buffer

```
free(buffer);
```

Disconnects from the filesystem

```
hdfsDisconnect(fs);
```

Example hdfs_read_revised.c File

```
/**
 * Licensed to the Apache Software Foundation (ASF) under one
 * or more contributor license agreements.  See the NOTICE file
 * distributed with this work for additional information
 * regarding copyright ownership.  The ASF licenses this file
 * to you under the Apache License, Version 2.0 (the
 * "License"); you may not use this file except in compliance
 * with the License.  You may obtain a copy of the License at
 *
 *     http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

#include <stdio.h>
#include "hdfs.h"
#include <errno.h>
#include <fcntl.h>

tSize
readFromOffset(hdfsFS fs, hdfsFile readFile, tOffset offset,
               char *buffer, tSize bufferSize)
{
  tSize ret = 0;

  if (fs == NULL || readFile == NULL || buffer == NULL) {
    return -1;
  }

  ret = hdfsSeek(fs, readFile, offset);
  if (!ret) {
    ret = hdfsRead(fs, readFile, buffer, bufferSize);
    if (ret < 0) {
      fprintf(stderr, "hdfsRead failed with error %d \n", errno);
    }
  } else {
```

```
    fprintf(stderr, "hdfsSeek failed with error %d \n", errno);
  }

  if (ret < 0) {
    hdfsCloseFile(fs, readFile);
  }

  //Current offset within the file will be positioned at (offset +
readBytes)th byte.
  return ret;
}

tSize
positionalRead(hdfsFS fs, hdfsFile readFile, tOffset offset,
               char *buffer, tSize bufferSize)
{
  tSize readBytes = 0;

  if (fs == NULL || readFile == NULL || buffer == NULL) {
    return -1;
  }

  readBytes = hdfsPread(fs, readFile, offset, buffer, bufferSize);
  if (readBytes < 0) {
    fprintf(stderr, "hdfsPread failed with error %d \n", errno);
    hdfsCloseFile(fs, readFile);
  }

  //Current offset within the file will not be advanced if hdfsPread is used
  return readBytes;
}

int
readEntireFile(hdfsFS fs, hdfsFile readFile, char *buffer, tSize bufferSize)
{
  tSize readBytes = bufferSize;
  uint64_t totalRead = 0;

  if (fs == NULL || readFile == NULL || buffer == NULL) {
    return -1;
  }

  while (readBytes == bufferSize) {
    readBytes = hdfsRead(fs, readFile, (void*)buffer, bufferSize);
    if (readBytes > 0) {
      totalRead += readBytes;
    } else {
      if (readBytes < 0) {
        fprintf(stderr, "hdfsRead failed with error %d \n", errno);
        hdfsCloseFile(fs, readFile);
        return -1;
      }
      break;
    }
  }

  return 0;
}

int
main(int argc, char **argv)
{
  if (argc != 3) {
    fprintf(stderr, "Usage: hdfs_read <filename> <buffersize>\n");
```

```
    exit(-1);
  }

  int err = hdfsSetRpcTimeout(30);
  if (err) {
    fprintf(stderr, "Failed to set rpc timeout!\n");
    exit(-1);
  }

  hdfsFS fs = hdfsConnect("default", 0);
  if (!fs) {
    fprintf(stderr, "Failed to connect to hdfs!\n");
    exit(-1);
  }

  const char* rfile = argv[1];
  tSize bufferSize = strtoul(argv[2], NULL, 10);

  hdfsFile readFile = hdfsOpenFile(fs, rfile, O_RDONLY, 0, 0, 0);
  if (!readFile) {
    fprintf(stderr, "Failed to open %s for reading!\n", rfile);
    exit(-2);
  }

  char* buffer = malloc(sizeof(char) * bufferSize);
  if(buffer == NULL) {
    fprintf(stderr, "Failed to allocate memory!\n");
    return -2;
  }

  //Read entire file from the beginning
  int ret = readEntireFile(fs, readFile, buffer, bufferSize);
  if (ret < 0) {
    goto done;
  }

  //Read file at a particular offset
  //In this case, we are reading from offset 0
  tSize readBytes = readFromOffset(fs, readFile, 0, buffer, bufferSize);
  if (readBytes < 0) {
    goto done;
  }

  //Read file at a particular offset using positional read
  //In this case, read from offset 100
  readBytes = positionalRead(fs, readFile, 100, buffer, bufferSize);
  if (readBytes < 0) {
    goto done;
  }

  hdfsCloseFile(fs, readFile);
done:
  free(buffer);
  hdfsDisconnect(fs);

  return 0;
}

/**
 * vim: ts=4: sw=4: et:
 */
```

hdfs_connect_as_user.c

last-updated: Apr 11, 2022

Sample Application

This application demonstrates how to create and write to files impersonating another user by using the API `hdfsConnectAsUser()`.

Before running this application, ensure that you have access to a cluster running MapR filesystem.

To build and run it, download it from this page and copy it to a MapR client or to a system with the mapr-core package installed. Then, modify the `run.sh` script in Building and Running C Applications on MapR File System Clients to point to this sample application. Run the script and then run the application.

The application includes these header files:

- `stdio.h`
- `hdfs.h`
- `stdlib.h`
- `string.h`

Note: The impersonation APIs are defined in `hdfs.h`.

The application performs the actions that are described in the following sections.

Takes two usernames and a hostname as input

After compiling the application, type the following command to launch the application and pass in the two usernames (to impersonate) and host name:

```
hdfs_connect_as_user <username1>
<username2> <hostname>if (argc < 4) {
  fprintf (stderr, "Provide two
usernames to impersonate and the host
name\n");
  printf ("USAGE: ./
hdfs_connect_as_user mapruser1
mapruser2 10.10.xx.xxx\n");
  exit(EXIT_FAILURE);
}
```

Stores the values of the arguments

The application stores the values of the arguments in character arrays. The application uses the port, 7222, to connect to the given host, and uses character arrays for user directory and file path.

```
char *impersonate_user1 = argv[1];
char *impersonate_user2 = argv[2];
char *host_addr = argv[3];
int port_num = 7222;
char user1_dir[100];
char writePath[100];
int ret_val;
```

Populates the directory path and file path

The application creates a default path for the user directory and file.

```
sprintf(user1_dir, "/tmp/%s_dir",
impersonate_user1);
sprintf(writePath, "%s/test_file",
user1_dir);
```


last-updated: Apr 11, 2022

Sets an RPC timeout

The `hdfsSetRpcTimeout()` is specific to the libMapRClient version of `libhdfs` and takes a value that is specified in seconds. The default is 99 seconds. If you change this value, set it either to 0 (which eliminates timeouts) or to a value greater than 30.

```
int err = hdfsSetRpcTimeout(30);
if (err) {
  fprintf(stderr, "Failed to set rpc
timeout!\n");
  exit(-1);
}
```

Connects to the filesystem as the impersonated user

The application connects to the filesystem as the impersonated user (<username1>) using `hdfsConnectAsUser()`. If successful, this operation returns a handle to the filesystem.

```
printf("Impersonate user: %s\n",
impersonate_user1);
printf("Connecting using
hdfsConnectAsUser() as user %s\n",
impersonate_user1);
hdfsFS fs_handle =
hdfsConnectAsUser(host_addr,
port_num, impersonate_user1);
if (fs_handle == NULL) {
    printf("hdfsConnectAsUser()
failed.\n");
    exit(EXIT_FAILURE);
}
```

Creates a directory as the impersonated user

The application creates a directory under `/tmp` as the impersonated user (<username1>).

```
printf("User1: Create a directory :
%s\n", user1_dir);
ret_val =
hdfsCreateDirectory(fs_handle,
user1_dir);
if (ret_val != EXIT_SUCCESS) {
  printf("hdfsCreateDirectory()
failed.\n");
  exit(EXIT_FAILURE);
}
```

Creates and opens a file

The application creates a file and opens the file, passing the following values to the `hdfsOpenFile()` function:

- The handle to the filesystem.
- A flag to indicate the mode in which to open the file. In this case, the flag is `O_WRONLY|O_CREAT`. This flag creates the file and opens it for writing.

For more details, see `hdfsOpenFile()` documentation.

```
printf("User1: Create and write to
the file as user1 : %s\n", writePath);
```

```
hdfsFile writeFile =
hdfsOpenFile(fs_handle, writePath,
O_WRONLY|O_CREAT, 0, 0, 0);
if(!writeFile) {
  printf("hdfsOpenFile() failed.\n");
  exit(EXIT_FAILURE);
}
```

Writes to the open file

The application writes to the open file as the impersonated user (<username1>).

```
char* buffer = "Hello, from user 1!";
tSize num_written_bytes =
hdfsWrite(fs_handle, writeFile,
(void*)buffer, strlen(buffer)+1);
if (hdfsFlush(fs_handle, writeFile)) {
  printf("failed to flush %s\n",
writePath);
  exit(EXIT_FAILURE);
}
```

Closes the file

The application closes the file after successfully writing to the file as the impersonated user (<username1>).

```
printf("User1: Close file %s.\n",
writePath);
hdfsCloseFile(fs_handle, writeFile);
```

Connects to the filesystem as the impersonating user

The application connects to the filesystem as the second user (<username2>) using `hdfsConnectAsUser()` and returns a handle to the filesystem.

```
printf("Impersonate user: %s\n",
impersonate_user2);
printf("Connecting using
hdfsConnectAsUser() as user %s\n",
impersonate_user2);
hdfsFS fs_handle2 =
hdfsConnectAsUser(host_addr,
port_num, impersonate_user2);
if (fs_handle2 == NULL) {
  printf("hdfsConnectAsUser()
failed.\n");
  exit(EXIT_FAILURE);
}
```

Tries to write to the file as the impersonating user

The application tries to open the file created by the impersonated user (<username1>) and write to the file as the impersonating user (<username2>). This operation fails as the impersonating user (<username2>) is denied access to the file created by the impersonated user (<username1>) and the application returns error EACCES.

```
printf("User2: Try opening file
created by user1 for writing : %s\n",
writePath);
hdfsFile writeFile2 =
hdfsOpenFile(fs_handle2, writePath,
```


last-updated: Apr 11, 2022

```
O_WRONLY, 0, 0, 0);
int errNum = errno;
if(writeFile2) {
    printf("User2: hdfsOpenFile()
should have failed for %s.\n",
impersonate_user2);
    exit(EXIT_FAILURE);
} else {
    if (errNum == EACCES) {
        printf("User2: As expected
hdfsOpenFile() with EACCES.\n");
    } else {
        printf("User2: hdfsOpenFile()
failed with errno:%d expected %d.\n",
errNum, EACCES);
        exit(EXIT_FAILURE);
    }
}
```

Deletes a directory

The application deletes a directory as the impersonated user (<username1>) using the filesystem handle created for this user.

```
printf("Delete directory : %s\n",
user1_dir);
ret_val = hdfsDelete(fs_handle,
user1_dir, 1);
if (ret_val != EXIT_SUCCESS) {
  printf("hdfsDelete() failed.\n");
  exit(EXIT_FAILURE);
}
```

Disconnects the impersonating user

The application disconnect the impersonating user (<username2>) from the filesystem.

```
printf("Disconnect the impersonation
user2.\n");
ret_val = hdfsDisconnect(fs_handle2);
if (ret_val != EXIT_SUCCESS) {
  printf("hdfsDisconnect()
failed.\n");
  exit(EXIT_FAILURE);
}
```

Disconnects the impersonated user

The application disconnect the impersonated user (<username1>) from the filesystem.

```
printf("Disconnect the impersonation
user1.\n");
ret_val = hdfsDisconnect(fs_handle);
if (ret_val != EXIT_SUCCESS) {
  printf("hdfsDisconnect()
failed.\n");
  exit(EXIT_FAILURE);
}
```

Example hdfs_connect_as_user.c File

```
/**
 * Licensed to the Apache Software Foundation (ASF) under one
 * or more contributor license agreements.  See the NOTICE file
 * distributed with this work for additional information
 * regarding copyright ownership.  The ASF licenses this file
 * to you under the Apache License, Version 2.0 (the
 * "License"); you may not use this file except in compliance
 * with the License.  You may obtain a copy of the License at
 *
 *     http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "hdfs.h"

int main(int argc, char *argv[]) {
    if (argc < 4) {
    fprintf (stderr, "Provide two usernames to impersonate and the host
name\n");
    printf ("USAGE: ./connectAsUser mapruser1 mapruser2 10.10.xx.xxx\n");
    exit(EXIT_FAILURE);
    }

    char *impersonate_user1 = argv[1];
    char *impersonate_user2 = argv[2];
    char *host_addr = argv[3];
    int port_num = 7222;
    char user1_dir[100];
    char writePath[100];
    int ret_val;

    /* Populate the directory patha nd file path */
    sprintf(user1_dir, "/tmp/%s_dir", impersonate_user1);
    sprintf(writePath, "%s/test_file", user1_dir);

    /* Impersonate as user1 using hdfsConnectAsUser(). */
    printf("Impersonate user: %s\n", impersonate_user1);
    printf("Connecting using hdfsConnectAsUser() as user %s\n",
impersonate_user1);
    hdfsFS fs_handle = hdfsConnectAsUser(host_addr, port_num,
impersonate_user1);
    if (fs_handle == NULL) {
        printf("hdfsConnectAsUser() failed.\n");
    exit(EXIT_FAILURE);
    }

    /* Create a directory under /tmp. This is done as the impersonated
user1. */
    printf("User1: Create a directory : %s\n", user1_dir);
    ret_val = hdfsCreateDirectory(fs_handle, user1_dir);
    if (ret_val != EXIT_SUCCESS) {
        printf("hdfsCreateDirectory() failed.\n");
        exit(EXIT_FAILURE);
```


last-updated: Apr 11, 2022

```
    }

    /*
     * Create and write a file using the filesystem
     * handle from impersonate_user1.
     */
    printf("User1: Create and write to the file as user1 : %s\n",
writePath);
    hdfsFile writeFile = hdfsOpenFile(fs_handle, writePath, O_WRONLY|
O_CREAT, 0, 0, 0);
    if(!writeFile) {
        printf("hdfsOpenFile() failed.\n");
        exit(EXIT_FAILURE);
    }

    char* buffer = "Hello, from user 1!";
    tSize num_written_bytes = hdfsWrite(fs_handle, writeFile,
(void*)buffer, strlen(buffer)+1);
    if (hdfsFlush(fs_handle, writeFile)) {
    printf("failed to flush %s\n", writePath);
    exit(EXIT_FAILURE);
    }
    printf("User1: Close file %s.\n", writePath);
    hdfsCloseFile(fs_handle, writeFile);

    /*
     * Impersonate as user 2 and try to write the file create by user1.
     * Writing to the file created by user1 will be denied with error
EACCES.
     */
    printf("Impersonate user: %s\n", impersonate_user2);
    printf("Connecting using hdfsConnectAsUser() as user %s\n",
impersonate_user2);
    hdfsFS fs_handle2 = hdfsConnectAsUser(host_addr, port_num,
impersonate_user2);
    if (fs_handle2 == NULL) {
        printf("hdfsConnectAsUser() failed.\n");
        exit(EXIT_FAILURE);
    }

    printf("User2: Try opening file created by user1 for writing : %s\n",
writePath);
    hdfsFile writeFile2 = hdfsOpenFile(fs_handle2, writePath, O_WRONLY, 0,
0, 0);
    int errNum = errno;
    if(writeFile2) {
        printf("User2: hdfsOpenFile() should have failed for %s.\n",
impersonate_user2);
        exit(EXIT_FAILURE);
    } else {
    if (errNum == EACCES) {
        printf("User2: As expected hdfsOpenFile() with EACCES.\n");
    } else {
        printf("User2: hdfsOpenFile() failed with errno:%d expected %d.\n",
errNum, EACCES);
        exit(EXIT_FAILURE);
    }
   }

    /* Delete the directory. This is done using the fliesystem handle
creatd for user1. */
    printf("Delete directory : %s\n", user1_dir);
    ret_val = hdfsDelete(fs_handle, user1_dir, 1);
    if (ret_val != EXIT_SUCCESS) {
```

```
    printf("hdfsDelete() failed.\n");
    exit(EXIT_FAILURE);
    }

    /* Disconnect the impersonation user1 */
    printf("Disconnect the impersonation user1.\n");
    ret_val = hdfsDisconnect(fs_handle);
    if (ret_val != EXIT_SUCCESS) {
    printf("hdfsDisconnect() failed.\n");
    exit(EXIT_FAILURE);
    }

    /* Disconnect the impersonation user2 */
    printf("Disconnect the impersonation user2.\n");
    ret_val = hdfsDisconnect(fs_handle2);
    if (ret_val != EXIT_SUCCESS) {
        printf("hdfsDisconnect() failed.\n");
        exit(EXIT_FAILURE);
    }
    exit(EXIT_SUCCESS);
}
```

Reference for the MapR File System C APIs

The following sections describe the custom datatypes, structures, and APIs in the `libMapRClient` version of `libhdfs`:

Type Definitions

`libhdfs` defines the following custom data types, which are supported by `libMapRClient`:

tObjectKind

An enumeration, the values of which are ‘F’ for file and ‘D’ for directory. Used to specify whether an object is a file or directory.

tOffset

A signed 64-bit integer that is used to specify an offset within a file and the size of a file.

tPort

An unsigned 16-bit integer that is used to specify a port to use in connections to filesystems.

tSize

A signed 32-bit integer that is used to specify the size of data in bytes to read or write.

tTime

A data type of `time_t` that is used to specify a time in seconds.

Structures

`libhdfs` defines these structures, which are supported by `libMapRClient`.

hdfsBuilder

Supported by libhdfs for hadoop-2.x

This structure can be passed to `hdfsBuilderConnect()` for creating connections to MapR File System clusters. In the `libMapRClient`, four of the parameters are ignored. `forceNewInstance` is ignored, though the header file does not indicate this.

```
struct hdfsBuilder {
    int forceNewInstance;
    const char *nn;
    tPort port;
    const char *kerbTicketCachePath; // Ignored
    const char *userName;            // Ignored
    struct hdfsBuilderConfOpt *opts; // Ignored
};
```

Parameters

`nn`

Specifies the CLDB node to connect to when `hdfsBuilderConnect()` is called. This value is set by `hdfsBuilderSetNameNode()`.

- If `default` is specified for the `host` parameter, `hdfsBuilderConnect()` will connect to the first cluster listed in the file `MAPR_HOME/conf/mapr-clusters.conf`. (`MAPR_HOME` defaults to `/opt/mapr`.)
- If a hostname or IP address is specified for the `host` parameter, `hdfsBuilderConnect()`, look in `MAPR_HOME/conf/mapr-clusters.conf` on the client node to match the specified hostname or IP address to a CLDB host and port.
  - If they find a match, they try to connect to the cluster and all standard features for connections to MapR clusters are available. These features include high availability across CLDBs and secure connections.
  - If they do not find a match or if they cannot locate a `mapr-clusters.conf` file, they try to connect to the CLDB host specified in the call to create the connection. However, the standard features for connections to MapR clusters are not available. For example, if the cluster is secured, the connection will fail.

`port`

Specifies the port to connect to on the CLDB node. This value is set by `hdfsBuilderSetNameNodePort()`.

hdfsFileInfo

Supported by libhdfs for hadoop-2.x

This structure is returned by `hdfsGetPathInfo()` and deleted by `hdfsFreeFileInfo()`. It contains information about the file or directory that is specified in the call to `hdfsGetPathInfo()`.

`ParameterstObjectKind mKind`

Specifies whether the object is a file or directory.

`char *mName`

Specifies the name of the object.

`tTime mLastMod`

Specifies the epoch time in milliseconds of the last modification to the object.

`tOffset mSize`

Specifies the size of the object in bytes.

`short mReplication`

Specifies the count of replicas of the object.

`tOffset mBlockSize`

Specifies the block size for the object.

`char *mOwner`

Specifies the owner of the object.

`char *mGroup`

Specifies the group that is associated with the object.

`short mPermissions`

Specifies the permissions on the object.

`tTime mLastAccess`

Specifies the epoch time in milliseconds at which the object was created.

APIs

The following sections provide information about the hdfs APIs:

hdfsAvailable()

Supported by `libMapRClient` for hadoop-2.x

Returns the number of bytes that can be read from an input stream without blocking. This number is simply the size of the file in bytes.

Signature

```
int hdfsAvailable(hdfsFS fs, hdfsFile file)
```

Parameters

Parameter	Description
fs	The handle of the filesystem where the file is located. Obtain this handle with one of the `hdfsConnect()` APIs.
file	The file handle. Obtain this handle with one of the `hdfsOpenFile()` APIs.

Return Value

Returns the size of the file in bytes, -1 on error.

Check errno for error codes and meanings.

errno is set to EINVAL if the arguments provided are invalid.

hdfsBuilderConnect()

Supported by `libMapRClient` for hadoop-2.x

Connects to a MapR filesystem using the parameters that are specified in an `hdfsBuilder` structure.

Signature

```
hdfsFS hdfsBuilderConnect(struct hdfsBuilder *bld)
```


last-updated: Apr 11, 2022

Parameters

Parameter	Description
bld	The builder to use for the connection. This value cannot be NULL.

Return Value

Returns the handle to the filesystem or NULL on error.

hdfsBuilderSetForceNewInstance()

Not supported for MapR File System

This API is ignored.

hdfsBuilderSetKerbTicketCachePath()

Not supported for MapR File System

This API is ignored.

hdfsBuilderSetNameNode()

Supported by `libMapRClient` for hadoop-2.x

Specifies a CLDB node for an `hdfsBuilder` structure.

Signature

```
void hdfsBuilderSetNameNode(struct hdfsBuilder *bld, const char *nn)
```

Parameters

Parameter	Description
bld	An `hdfsBuilder` structure. This value cannot be NULL.
nn	The hostname or IP address of a name node. Use NULL to connect to the local filesystem. Use `default` to connect to the first MapR File System cluster that is listed in the `MAPR_HOME/conf/mapr-clusters.conf` file on the client.

hdfsBuilderSetNameNodePort()

Supported by `libMapRClient` for hadoop-2.x

Sets the port in an `hdfsBuilder` structure.

Signature

```
void hdfsBuilderSetNameNodePort(struct hdfsBuilder *bld, tPort port)
```

Parameters

Parameter	Description
bld	An `hdfsBuilder` structure. This value cannot be NULL.
port	The port to use for connections. If the CLDB node is set to NULL or `default`, use 0.

last-updated: Apr 11, 2022

hdfsBuilderSetUserName()

Not supported for MapR File System

This API is ignored.

hdfsChmod()

Supported by `libMapRClient` for hadoop-2.x

Changes permissions on a file or directory in the manner of the `chmod` command.

Signature

```
int hdfsChmod(hdfsFS fs, const char* path, short mode)
```

Parameters

Parameter	Description
fs	The handle to the filesystem. Obtain this handle with one of the `hdfsConnect()` APIs.
path	The path to the file or directory.
mode	The bitmask for the new permissions.

Return Value

Returns 0 on success, -1 on error.

Check errno for error codes and meanings.

errno is set to EINVAL if the input arguments are invalid.

errno is set to EPERM if the process does not have enough privileges to perform the operation.

hdfsChown()

Supported by `libMaprClient` for hadoop-2.x

Changes ownership of a file or directory in the manner of the `chown` command.

Attention:

- To permit a client to resolve user or group from a server, set the `fs.mapr.server.resolve.user` parameter to `true` in `core-site.xml`, for both secure and non secure clusters. Setting this is essential when the client does not belong to the same domain as the mapr cluster nodes, and does not have any knowledge of users present in that domain.
- To permit CLDB to resolve user or group, set the `cldb.security.resolve.user` configuration parameter to `1` on a non-secure cluster as follows:

  ```
  maprcli config save -values {"cldb.security.resolve.user":1}
  ```

  You do not have to set this parameter for a secure cluster, as it is already set to `1`.

Signature

```
int hdfsChown(hdfsFS fs, const char* path, const char *owner, const char
*group)
```

Parameters

Parameter	Description
fs	The handle to the filesystem. Obtain this handle with one of the `hdfsConnect()` APIs.
path	The path to the file or directory
owner	The user to own the file or directory. Set to NULL to keep the owner as is.
group	The group to own the file or directory. Set to NULL to keep the owner as is.

Return Value

Returns 0 on success, -1 on error.

Check errno for error codes and meanings.

errno is set to EINVAL if the input arguments are invalid.

errno is set to EPERM if the process does not have enough privileges to perform the operation.

hdfsCloseFile()

Supported by `libMapRClient` for hadoop-2.x

Closes an open file. Flushes all pending write buffers for the file and releases resources that are associated with the file.

Signature

```
int hdfsCloseFile(hdfsFS fs, hdfsFile file)
```

Parameters

Parameter	Description
fs	The handle of the filesystem where the file is located. Obtain this handle with one of the `hdfsConnect()` APIs.
file	The file handle. Obtain this handle with one of the `hdfsOpenFile()` APIs.

Return Value

Returns 0 on success, -1 on error.

Check errno for error codes and meanings.

hdfsConnect()

Supported by `libMapRClient` for hadoop-2.x

Connects to a MapR File System cluster.

If a connection to the cluster in which the remote host is located already exists, the functions return a handle to this existing connection.

If a connection to the cluster does not already exist, the functions return a handle to a new connection instance.

Note that if `default` is used for the `host` parameter, this means connect to the first cluster listed in the `MAPR_HOME/conf/mapr-clusters.conf` file on the client.

For more information about connections, see Establishing Connections to MapR File System.

Signature

```
hdfsFS hdfsConnect(const char* host, tPort port)
```

Parameters

Parameter	Description
host	A string containing either a hostname or an IP address of a CLDB node of a MapR File System cluster. To connect to the first MapR File System cluster that is specified in `MAPR_HOME/conf/mapr-clusters.conf` on the client, pass the value `default` and use the port number 0. This parameter does not accept NULL as a value.
port	The port on which the host is listening.

Return Value

Returns a handle to the connected filesystem, or NULL on error.

Check errno for error codes and meanings.

hdfsConnectAsUser()

Supported by `libMapRClient` for hadoop-2.x

Connects to a MapR File System cluster as specified user.

If a connection to the cluster in which the remote host is located already exists, the functions return a handle to this existing connection.

If a connection to the cluster does not already exist, the functions return a unique handle to a connection instance for each user.

Note: If `default` is used for the host parameter, this means connect to the first cluster listed in the `MAPR_HOME/conf/mapr-clusters.conf` file on the client.

For more information about connections, see Establishing Connections to MapR File System.

Signature

```
hdfsFS hdfsConnectAsUser(const char* host, tPort port, const char* user)
```


last-updated: Apr 11, 2022

Parameters

Parameter	Description
`host`	A string containing either a hostname or an IP address of a CLDB node of a MapR File System cluster. To connect to the first MapR File System cluster that is specified in `MAPR_HOME/conf/mapr-clusters.conf` on the client, pass the value default and use the port number 0. This parameter does not accept NULL as a value.
`port`	The port on which the host is listening.
`user`	The user connected to the cluster.

Return Value

Returns a handle to the connected filesystem, or NULL on error.

hdfsConnectAsUid()
Connects to a MapR File System cluster as specified user ID.

Supported by `libMapRClient` for hadoop-2.x

Attention: You need to install a patch for version 6.1 to use this command.

If a connection to the cluster in which the remote host is located already exists, the function returns a handle to this existing connection.

If a connection to the cluster does not already exist, the function returns a unique handle to a connection instance for the specified user ID.

Note: To connect to the first cluster listed in the `MAPR_HOME/conf/mapr-clusters.conf` file on the client, use `default` as the value for the `host` parameter.

For more information about connections, see Establishing Connections to the File System on page 2380.

Signature

```
hdfsFS hdfsConnectAsUid(const char* host, tPort port, uid_t uid)
```

Parameters

Parameter	Description
`host`	A string containing either a hostname or an IP address of a CLDB node of a MapR File System cluster. To connect to the first MapR File System cluster that is specified in `MAPR_HOME/conf/mapr-clusters.conf` on the client, pass the value default and use the port number 0. This parameter does not accept NULL as a value.
`port`	The port on which the host is listening.
`user ID`	The ID of the user connected to the cluster.

Return Value

Returns a handle to the connected filesystem, or NULL on error.

hdfsConnectAsUserNewInstance()

Supported by libMapRClient for hadoop-2.x

Connects to a MapR File System cluster as specified user.

The function returns a handle to a new connection instance.

Note: If default is used for the host parameter, this means connect to the first cluster listed in the `MAPR_HOME/conf/mapr-clusters.conf` file on the client.

For more information about connections, see Establishing Connections to MapR File System.

Signature

```
hdfsFS hdfsConnectAsUserNewInstance(const char* host, tPort port, const
char* user)
```

Parameters

Parameter	Description
`host`	A string containing either a hostname or an IP address of a CLDB node of a MapR File System cluster. To connect to the first MapR File System cluster that is specified in `MAPR_HOME/conf/mapr-clusters.conf` on the client, pass the value default and use the port number 0. This parameter does not accept NULL as a value.
`port`	The port on which the host is listening.

Return Value

Returns a handle to the connected filesystem, or NULL on error.

hdfsConnectNewInstance()

Supported by `libMapRClient` for hadoop-2.x

Connects to a MapR File System cluster.

The function returns a handle to a new connection instance.

Note: If `default` is used for the host parameter, this means connect to the first cluster listed in the `MAPR_HOME/conf/mapr-clusters.conf` file on the client.

For more information about connections, see Establishing Connections to MapR File System.

Signature

```
hdfsFS hdfsConnectNewInstance(const char* host, tPort port)
```


last-updated: Apr 11, 2022

Parameters

Parameter	Description
`host`	A string containing either a hostname or an IP address of a CLDB node of a MapR File System cluster. To connect to the first MapR File System cluster that is specified in `MAPR_HOME/conf/mapr-clusters.conf` on the client, pass the value default and use the port number 0. This parameter does not accept NULL as a value.
`port`	The port on which the host is listening.
`user`	The user connected to the cluster.

Return Value

Returns a handle to the connected filesystem, or NULL on error.

hdfsCopy()

This API is not supported.

hdfsCreateDirectory()

Supported by `libMapRClient` for hadoop-2.x

Creates a file or directory at the specified path. Intermediate directories in the path that do not exist are created.

Signature

```
int hdfsCreateDirectory(hdfsFS fs, const char* path)
```

Parameters

Parameter	Description
fs	The handle of the filesystem in which to create the file or directory. Obtain this handle with one of the `hdfsConnect()` APIs.
path	The path of the directory.

Return Value

Returns 0 on success, -1 on error.

Check errno for error codes and meanings.

errno is set to EINVAL if the input arguments are not valid, EEXIST if the directory already exists, or EACCES if the parent directory does not allow the user write permission.

hdfsCreateDirectory2()

Supported by `libMapRClient` for hadoop-2.x

Makes the given file and all non-existent parents into directories. Stores the size of the name container in a location that you pass a pointer into, so that you can keep track of this size. The size is in bytes.

Keeping track of the size of the name container is useful when you are creating files that are less than or equal to 64 KB. When the size of all of the such files together for one name container exceeds 64 GB,

operations on the name container can become inefficient. If the size of a name container reaches 64 GB, you can switch to a new or different volume.

Signature

```
int hdfsCreateDirectory2(hdfsFS fs, const char* path, tSize
*nameSizeInBytes)
```

Parameters

Parameter	Description
fs	The handle of the filesystem in which to create the file or directory. Obtain this handle with one of the `hdfsConnect()` APIs.
path	The path of the directory.
nameSizeInBytes	A pointer to a memory buffer that can store the size in bytes of the name container.

Return Value

Returns 0 on success, -1 on error.

Check errno for error codes and meanings.

hdfsDelete()

Supported by `libMapRClient` for hadoop-2.x

Deletes the specified directory or file.

Signature

```
int hdfsDelete(hdfsFS fs, const char* path, int recursive)
```

Parameters

Parameter	Description
fs	The handle of the filesystem where the file or directory to delete is located. Obtain this handle with one of the `hdfsConnect()` APIs.
path	The path of the file.
recursive	A value of 0 deletes the specified directory, if the directory is empty. If the directory is not empty, an error is returned. A non-zero value deletes the specified directory and all of its subdirectories. If the specified object is a file, not a directory, this parameter is ignored.

Return Value

Returns 0 on success, -1 on error.

Check errno for error codes and meanings. Some of the key errors are ESTALE, EACCES, and EPERM.

last-updated: Apr 11, 2022

hdfsDisconnect()

Supported by `libMapRClient` for hadoop-2.x

Disconnects from the specified filesystem.

Even if there is an error, the resources that are associated with the filesystem handle are freed.

Signature

```
int hdfsDisconnect(hdfsFS fs)
```

Parameter

Parameter	Description
fs	The handle of the filesystem to disconnect from. Obtain this handle with one of the `hdfsConnect()` APIs.

Return Value

Returns 0 on success, -1 on error.

Check errno for error codes and meanings.

hdfsExists()

Supported by `libMapRClient` for hadoop-2.x

Checks whether a given directory or file exists on the filesystem.

Signature

```
int hdfsExists(hdfsFS fs, const char* path)
```

Parameters

Parameter	Description
fs	The filesystem handle. Obtain this handle with one of the `hdfsConnect()` APIs.
path	The directory or file to check the existence of.

Return Value

Returns 0 if the directory or file exists, -1 on error.

Check errno for error codes and meanings.

hdfsExists2()

Supported by `libMapRClient` for hadoop-2.x

Checks the filesystem directly (avoiding a client cache) to determine whether a given file or directory exists.

Signature

```
int hdfsExists2(hdfsFS fs, const char* path)
```

Parameters

Parameter	Description
fs	The filesystem handle. Obtain this handle with one of the `hdfsConnect()` APIs.
path	The directory or file to check the existence of.

Return Value

Returns 0 if the directory or file exists, -1 on error.

Check errno for error codes and meanings.

hdfsFileFreeReadStatistics()

This API is not supported.

hdfsFileGetReadStatistics()

This API is not supported.

hdfsFileIsOpenForRead()

This API is not supported.

hdfsFileIsOpenForWrite()

This API is not supported.

hdfsFlush()

Supported by `libMapRClient` for hadoop-2.x

Flushes the write buffer for the specified file to the server

Signature

```
int hdfsFlush(hdfsFS fs, hdfsFile file)
```

Parameters

Parameter	Description
fs	The filesystem handle. Obtain this handle with one of the `hdfsConnect()` APIs.
file	The file handle. Obtain this handle with one of the `hdfsOpenFile()` APIs.

Return Value

Returns 0 on success, -1 on error.

Check errno for error codes and meanings.

errno is set to EINVAL if the input arguments are invalid.

hdfsFreeBuilder()

Supported by `libMapRClient` for hadoop-2.x

Frees the memory that was used by an `hdfsBuilder` structure and its parameter values.

Signature

```
void hdfsFreeBuilder(struct hdfsBuilder *bld)
```

Parameters

Parameter	Description
bld	An `hdfsBuilder` structure. The value cannot be NULL.

hdfsFreeFileInfo()

Supported by `libMapRClient` for hadoop-2.x

Frees up the array of `hdfsFileInfo` structures that is returned by `hdfsListDirectory()`, including allocated fields.

Signature

```
void hdfsFreeFileInfo(hdfsFileInfo *hdfsInfo, int numEntries)
```

Parameters

Parameter	Description
hdfsInfo	The array of dynamically-allocated `hdfsFileInfo` structures.
numEntries	The size of the array.

hdfsFreeHosts()

Supported by `libMapRClient` for hadoop-2.x

Frees an array that was returned by `hdfsGetHosts()`.

Signature

```
void hdfsFreeHosts(char ***blockHosts)
```

Parameters

Parameter	Description
blockHosts	The two-dimensional array that was returned by `hdfsGetHosts()`.

hdfsGetCapacity()

Supported by `libMapRClient` for hadoop-2.x

Returns the capacity in bytes of the connected filesystem.

Signature

```
tOffset hdfsGetCapacity(hdfsFS fs)
```

Parameters

Parameter	Description
fs	The filesystem handle. Obtain this handle by calling one of the `hdfsConnect()` APIs.

Return Value

Returns the capacity in bytes of the connected filesystem, -1 on error.

Check errno for error codes and meanings.

errno can be set to EINVAL in case of error.

hdfsGetDefaultBlockSize()

Supported by libMapRClient for hadoop-2.x

Gets the default size of blocks for the connected filesystem.

Signature

```
tOffset hdfsGetDefaultBlockSize(hdfsFS fs)
```

Parameters

Parameter	Description
fs	The handle of filesystem. Obtain this handle with one of the `hdfsConnect()` APIs.

Return Value

Returns 256 MB.

Returns -1 on error.

Check errno for error codes and meanings.

hdfsGetDefaultBlockSizeAtPath()

Supported by `libMapRClient` for hadoop-2.x

Gets the block size of a file at the specified path.

Signature

```
tOffset hdfsGetDefaultBlockSizeAtPath(hdfsFS fs, const char *path)
```

Parameters

Parameter	Description
fs	The handle of the filesystem. Obtain this handle with one of the `hdfsConnect()` APIs.
path	The location and name of the file.

Return Value

Returns 256 MB.

last-updated: Apr 11, 2022

Check errno for error codes and meanings.

hdfsGetHosts()

Supported by `libMapRClient` for hadoop-2.x

Gets hostnames where a particular block, as determined by the offset and block size, is stored. Due to replication, a single block could be present on multiple hosts.

This function can be useful for understanding the performance implications of file access, and to validate or verify changes to the replication factor.

Signature

```
char*** hdfsGetHosts(hdfsFS fs, const char* path, tOffset start, tOffset
length)
```

Parameters

Parameter	Description
fs	The handle of the filesystem where the file is located. Obtain this handle with one of the `hdfsConnect()` APIs.
path	The path of the file.
start	The start of the block.
length	The length of the block.

Return Value

If successful, returns a dynamically-allocated two-dimensional array of hostnames. The last element in the array is NULL.

Returns NULL on error.

Check errno for error codes and meanings.

hdfsGetNameContainerSizeBytes()

Unique to libMapRClient

Get the size of the container hosting the path.

Keeping track of the size of the name container is useful when you are creating files that are less than or equal to 64 KB. When the size of all of the such files together for one name container exceeds 64 GB, operations on the name container can become inefficient. If the size of a name container reaches 64 GB, you can switch to a new or different volume.

Signature

```
int tSize hdfsGetNameContainerSizeBytes(hdfsFS fs, const char *path)
```

Parameters

Parameter	Description
path	Path of the file or directory residing on the container.

Return Value

Returns size of the container on success; -1 on error.

Check errno for error codes and meanings.

errno is set to EINVAL if the input arguments are invalid.

hdfsGetPathInfo()

Supported by libMapRClient for hadoop-2.x

Returns a dynamically-allocated `hdfsFileInfo` structure that contains information about the given path.

Call `hdfsFreeFileInfo()` when the structure is no longer needed.

See `hdfsFileInfo()` for information about the information that this object contains.

Signature

```
hdfsFileInfo * hdfsGetPathInfo(hdfsFS fs, const char* path)
```

Parameters

Parameter	Description
fs	The filesystem handle. Obtain this by calling one of the `hdfsConnect()` APIs.
path	The path of the file.

Return Value

Returns a dynamically-allocated `hdfsFileInfo` structure on success, and NULL on error.

errno is set to EINVAL for invalid arguments and to EACCES for invalid access.

hdfsGetUsed()

Supported by libMapRClient for hadoop-2.x

Returns the total number of bytes bytes that are being used by all of the files in the filesystem.

Signature

```
tOffset hdfsGetUsed(hdfsFS fs)
```

Parameters

Parameter	Description
fs	The filesystem handle.

Return Value

Returns the total size in bytes or -1 on error.

Check errno for error codes and meanings.

hdfsGetWorkingDirectory()

Supported by libMapRClient for hadoop-2.x

last-updated: Apr 11, 2022

Gets the current working directory for the filesystem. Before calling this method, the application must have called `hdfsSetWorkingDirectory()`.

Signature

```
char* hdfsGetWorkingDirectory(hdfsFS fs, char *buffer, size_t bufferSize)
```

Parameters

Parameter	Description
fs	The filesystem handle.
buffer	The buffer in which to copy path of current working directory.
bufferSize	The length of user-buffer.

Return Value

Returns the buffer on success, NULL on error.

errno is set to EINVAL for invalid arguments.

hdfsGetXattr()

Supported by libMapRClient for hadoop-2.x

Gets extended attribute values from a file.

Signature

```
int hdfsGetXattr(hdfsFS fs, const char* path, const char *name, char
*value,  size_t size);
```

Parameters

Parameter	Description
`fs`	The handle of the filesystem where the file is located. Obtain this handle with one of the `hdfsConnect()` APIs.
`name`	The name of the extended attribute.
`path`	The path to the file.
`size`	The size of the buffer.
`value`	The value for the extended attribute that is read from the system and written to the buffer.

Return Value

Returns:

- Current size of the value of the extended attribute on success
- -1 on error

  Check `errno` for error codes and meanings.

hdfsListDirectory()

Supported by `libMapRClient` for hadoop-2.x

Gets list of files and directories for a given path. Returns the information in a dynamically allocated array of `hdfsFileInfo` structures.

`hdfsFreeFileInfo()` should be called to deallocate memory when this structure is no longer needed.

This method is the equivalent of the `ls -l` command.

Signature

```
hdfsFileInfo *hdfsListDirectory(hdfsFS fs, const char* path, int
*numEntries)
```

Parameters

Parameter	Description
fs	The handle of the filesystem. Obtain this handle with one of the `hdfsConnect()` APIs.
path	The path of the directory.
numEntries	Set to the number of files/directories in path. Cannot be 0 or NULL.

Return Value

Returns a dynamically-allocated array of `hdfsFileInfo` structures on success and NULL on error.

Check errno for error codes and meanings.

hdfsMove()

This API is not supported.

hdfsNewBuilder()

Supported by `libMapRClient` for hadoop-2.x

Returns an `hdfsBuilder` structure. You can set values for its parameters and then pass it to `hdfsBuilderConnect()`.

Signature

```
struct hdfsBuilder *hdfsNewBuilder(void)
```

Return Value

Returns a new `hdfsBuilder` structure.

Returns ENOMEM if unable to allocate memory for a new `hdfsBuilder` structure.

hdfsOpenFile()

Supported by `libMapRClient` for hadoop-2.x

Opens a file in the specified mode. Creates the file and intermediate directories if they do not exist.

Requires a valid filesystem handle, which one of the `hdfsConnect()` APIs can provide.

Before the call to `hdfsOpenFile()`, `hdfsExists()` can check that the file exists, if a check is needed.

last-updated: Apr 11, 2022

After finishing work on a file, call `hdfsCloseFile()` to free the memory that is associated with the file.

Signature

```
hdfsFile hdfsOpenFile(hdfsFS fs, const char* path, int flags, int
bufferSize, short replication, tSize blocksize)
```

Parameters

Parameter	Description
fs	The handle of the filesystem where the file is located. Obtain this handle with one of the `hdfsConnect()` APIs.
path	The full path to the file.
flags	One of the following values. These flags are included in the fcntl.h header file. **O_RDONLY** — Opens the file in read-only mode with the current offset at 0. **O_RDWR** — Opens the file in read-write mode. If the file already exists, it is truncated to offset 0, effectively deleting the content of the file to offset 0. **O_RDWR | O_APPEND** — Opens the file in read-write mode with the current offset at 0. Writing to the file with `hdfsWrite()` appends the written data to the end of the file. Data written with `hdfsPwrite()` is not appended, but written starting at the offset specified in the call to that API. **O_WRONLY** — Opens the file in write-only mode. If the file already exists, it is truncated to offset 0, effectively deleting the content of the file. **O_WRONLY | O_APPEND** — Opens the file in write-only mode with the current offset at 0. Writing to the file with `hdfsWrite()` appends the written data to the end of the file. Data written with `hdfsPwrite()` is not appended, but written starting at the offset specified in the call to that API.
bufferSize	*Ignored for files on MapR filesystem*
replication	*Ignored for files on MapR filesystem*

Parameter	Description
blocksize	The size of chunks for the file in bytes. Specify 0 if you want to use the value that is specified for the `fs.mapr.block.size` parameter in the `/opt/mapr/hadoop/hadoop-2.x/etc/hadoop/core-site.xml` file on the client (if the client is using the libMapRClient version of hadoop-2.x). If this parameter is not set in `core-site.xml`, the default value is taken from the directory's `.dfs_attrributes` file.

Return Value

Returns the handle to the open file or NULL on error.

Check errno for error codes and meanings.

hdfsOpenFile2()

Supported by `libMapRClient` for hadoop-2.x

Opens a file in a given mode. Creates the file if the file does not exist.

If `hdfsOpenFile2()` creates a file, it stores the size of the name container in a location that you pass a pointer into, so that you can keep track of this size. The size is in bytes.

Keeping track of the size of the name container is useful when you are creating files that are less than or equal to 64 KB. When the size of all of the such files together for one name container exceeds 64 GB, operations on the name container can become inefficient. If the size of a name container reaches 64 GB, you can switch to a new or different volume.

Before the call to `hdfsOpenFile2()`, `hdfsExists()` can check that the file exists, if a check is needed.

After finishing work on a file, call `hdfsCloseFile()` to free the resources that are associated with the file.

Signature

```
hdfsFile hdfsOpenFile2(hdfsFS fs, const char* path, int flags, int
bufferSize, short replication, tSize blocksize, tSize *nameSizeInBytes)
```

Parameters

Parameter	Description
fs	The handle of the filesystem where the file is located. Obtain this handle with one of the `hdfsConnect()` APIs.
path	The full path to the file.

Parameter	Description
flags	One of the following values. These flags are included in the fcntl.h header file. **O_RDONLY** — Opens the file in read-only mode with the current offset at 0. **O_RDWR** — Opens the file in read-write mode. If the file already exists, it is truncated to offset 0, effectively deleting the content of the file. **O_RDWR** | **O_APPEND** — Opens the file in read-write mode with the current offset at 0. Writing to the file with `hdfsWrite()` appends the written data to the end of the file. Data written with `hdfsPwrite()` is not appended, but written starting at the offset specified in the call to that API. **O_WRONLY** — Opens the file in write-only mode. If the file already exists, it is truncated to offset 0, effectively deleting the content of the file. **O_WRONLY** | **O_APPEND** — Opens the file in write-only mode with the current offset at 0. Writing to the file with `hdfsWrite()` appends the written data to the end of the file. Data written with `hdfsPwrite()` is not appended, but written starting at the offset specified in the call to that API.
bufferSize	*Ignored for files on MapR filesystem*
replication	*Ignored for files on MapR filesystem*
blocksize	The size of chunks for the file in bytes. Use 0 if you want to use the value that is specified for the `fs.mapr.block.size` parameter in the `/opt/mapr/hadoop/hadoop-2.x/etc/hadoop/core-site.xml` file on the client (if the client is using the libMapRClient version of hadoop-2.x). If this parameter is not set in `core-site.xml`, the default value is taken from the directory's `.dfs_attrributes` file.
nameSizeInBytes	A pointer to a memory buffer that can store the size in bytes of the name container. The value is returned only if `hdfsOpenFile2()` creates the specified file because the file does not already exist.

last-updated: Apr 11, 2022

Return Value

Returns the handle to the open file or NULL on error.

Check errno for error codes and meanings.

hdfsPread()

Supported by libMapRClient for hadoop-2.x

Reads an open file from a specified offset.

Whereas `hdfsRead()` increments the current offset in the file by the number of bytes that are read, `hdfsPread()` does not change the current offset. For example, if the current offset is 0 and `hdfsPread()` starts reading from offset 100, after the read the current offset is still 0.

Signature

```
tSize  hdfsPread(hdfsFS fs, hdfsFile file, tOffset position, void* buffer,
tSize length)
```

Parameters

Parameter	Description
fs	The handle of the filesystem where the file is located. Obtain this handle with one of the `hdfsConnect()` APIs.
file	The file handle. Obtain this handle with one of the `hdfsOpenFile()` APIs.
position	Offset from which to read.
buffer	The buffer to copy read bytes into.
length	The length of the buffer. The maximum size of `tSize` is the maximum buffer length.

Return Value

Returns the number of bytes actually read, which can be less than than the length of the buffer if the end of the file is reached during the read. Returns -1 on error.

On error, errno is set to one of the following values:

- EACCES if the access permissions are violated.
- ESTALE if the file doesn't exist on the server.
- EINVAL if the arguments are invalid or if the file type doesn't support read operations.

To recover from errors, close the file by calling `hdfsCloseFile()`.

hdfsPwrite()

Supported by libMapRClient for hadoop-2.x

Writes starting at a specified position in an open file.

Whereas `hdfsWrite()` increments the current offset by the amount of bytes returned by the API (except in case of error), `hdfsPwrite()` does not change the value of current offset. If the current offset before the call to `hdfsPwrite()` is 0 and you specify the offset 10 for the write operation, after the write the current offset remains 0.

last-updated: Apr 11, 2022

If a call to `hdfsPwrite()` specifies an offset that is past the end of the file, the result is a hole in the file between the previous end of the file and the offset at which the write begins.

You can obtain the size of a file in bytes by calling `hdfsGetPathInfo()`.

Flushes to the server happen automatically at intervals during a write operation. After a write operation is finished, either call `hdfsFlush()` explicitly or call `hdfsFlush()` implicitly by calling `hdfsCloseFile()` to be sure that any data remaining in the write buffer is flushed.

On error, pending write buffers are flushed to the server.

Signature

```
tSize hdfsPwrite(hdfsFS fs, hdfsFile file, tOffset position, const void*
buffer, tSize length)
```

Parameters

Parameter	Description
fs	The handle of the filesystem where the file is located. Obtain this handle with one of the `hdfsConnect()` APIs.
file	The handle of the file. Obtain this handle with one of the `hdfsOpenFile()` APIs.
position	The offset at which to start writing.
buffer	The data to write.
length	The number of bytes to write. The maximum length is the maximum size of `tSize`.

Return Value

Returns the number of bytes written, -1 on error.

Check errno for error codes and meanings.

hdfsRead()

Supported by libMapRClient for hadoop-2.x

Reads data from the current offset in an open file. After the read, the current offset is incremented by the number of bytes read.

To read from a specific offset, first call `hdfsSeek()` to move to that offset in the file. Then, call `hdfsRead()`.

Alternatively, call `hdfsPread()`, specifying an offset in the call. `hdfsPread()` does not increment the current offset in the file. The offset that you specify in the call is used only for the read.

Signature

```
tSize hdfsRead(hdfsFS fs, hdfsFile file, void* buffer, tSize length)
```

Parameters

Parameter	Description
fs	The filesystem handle. Filesystem handle can be obtained using one of the `hdfsConnect()` APIs.

Parameter	Description
file	The file handle. Obtain this handle with one of the `hdfsOpenFile()` APIs.
buffer	The buffer to copy bytes into during the read.
length	The length of the buffer. The maximum length of the buffer is the maximum size of `tSize`.

Returned Value

Returns the number of bytes actually read, which can be less than than the length of the buffer if the end of the file is reached during the read. Returns -1 on error.

Check errno for error codes and meanings.

hdfsReadStatisticsGetRemoteBytesRead()

This API is not supported.

hdfsRename()

Supported by libMapRCient for hadoop-2.x

Renames the specified file. For information about the format to use for paths, see Specifying Paths to Files and Directories.

Signature

```
int hdfsRename(hdfsFS fs, const char* oldPath, const char* newPath)
```

Parameters

Parameter	Description
fs	The handle of the filesystem where the file is located. Obtain this handle with one of the `hdfsConnect()` APIs.
oldPath	The path of the source file.
newPath	The path of the destination file.

Return Value

Returns 0 on success, -1 on error.

Check errno for error codes and meanings.

hdfsSeek()

Supported by `libMapRClient` for hadoop-2.x

Moves the current offset to another offset in the specified file.

Signature

```
int hdfsSeek(hdfsFS fs, hdfsFile file, tOffset desiredPos)
```


last-updated: Apr 11, 2022

Parameters

Parameter	Description
fs	The handle for the filesystem where the file is located. Obtain this handle with one of the `hdfsConnect()` APIs.
file	The handle to the file. Obtain this handle with one of the `hdfsConnect()` APIs.
desiredPos	The offset to move forward to.

Return Value

Returns 0 on success, -1 on error.

Check errno for error codes and meanings.

hdfsSetTicketAndKeyFile()
Dynamically loads a ticket file

Supported by `libMapRClient` for hadoop-2.x

Use this API to dynamically load a ticket file to connect to newly added clusters and nodes, without restarting your application.

Signature

```
int hdfsSetTicketAndKeyFile(const char *fname)
```

Parameters

Parameter	Description
fname	The name of the ticket file to reload.

Return Value

Returns 0 on success, -1 on error.

Check errno for error codes and meanings.

hdfsSetReplication()

This API is not supported.

hdfsSetRpcTimeout()

Unique to libMapRClient

Sets the RPC timeout in seconds. Before creating a connection to a MapR File System cluster, you can set an RPC timeout for your connections to CLDB nodes and fileservers, passing the number of seconds for the timeout as an integer.

Signature

```
int hdfsSetRpcTimeout(int seconds)
```

Parameters

Parameter	Description
seconds	The time in seconds to wait before timing out. The default is 99 seconds. If you change the value, set it either to 0 or to greater than 30 seconds. If RPC timeout is set to 0, remote procedure calls will continue to be retried until they are successful.

Return Value

Returns 0 on success, -1 on error.

Check errno for error codes and meanings.

hdfsSetThreads()

Unique to libMapRClient

Configures the number of threads for flushing write buffers. This number is specific to individual clients. The default number is 8.

The number of threads must be positive. If it isn't, EINVAL is returned.

Signature

```
int hdfsSetThreads(int threads)
```

Parameters

Parameter	Description
threads	The number of threads for flushing write buffers.

Return Value

Returns the current number of flush threads for the client, -1 on error. Check errno for error codes and meanings.

hdfsSetWorkingDirectory()

Supported by libMapRClient for hadoop-2.x

Set the working directory. All relative paths will be resolved relative to it.

For example, if you call this API to set the working directory to `/mycluster/myvolume` and subsequently call `hdfsOpenFile()` with the path `/temp/tmp.txt`, the full path to the file to open is assumed to be `/mycluster/myvolume/temp/tmp.txt`.

Signature

```
int hdfsSetWorkingDirectory(hdfsFS fs, const char* path)
```

Parameters

Parameter	Description
fs	The handle of the filesystem where the directory is located. Obtain this handle with one of the `hdfsConnect()` APIs.
path	The path of the new working directory.

Return Value

Returns 0 on success, -1 on error.

Check errno for error codes and meanings.

errno is set to EINVAL for invalid arguments.

hdfsSetXattr()

Supported by libMapRClient for hadoop-2.x

Sets extended attribute on a file.

Signature

```
int hdfsSetXattr(hdfsFS fs, const char* path, const char *name, int
nameLen, char *value, int valueLen);
```

Parameters

Parameter	Description
`fs`	The handle of the filesystem where the file is located. Obtain this handle with one of the `hdfsConnect()` APIs.
`name`	The name of the extended attribute.
`nameLen`	The length of the name of the extended attribute.
`path`	The path to the file.
`value`	The value for the extended attribute.
`valueLen`	The length of the value of the extended attribute.

Return Value

Returns `0` on success, `-1` on error.

Check `errno` for error codes and meanings.

hdfsTell()

Supported by libMapRClient for hadoop-2.x

Gets the current offset in the file in bytes.

Signature

```
tOffset hdfsTell(hdfsFS fs, hdfsFile file)
```

Parameters

Parameter	Description
fs	The handle of the filesystem where the file is located. Obtain this handle with one of the `hdfsConnect()` APIs.
file	The file handle. Obtain this handle with one of the `hdfsOpenFile()` APIs.

Return Value

Returns the current offset in bytes, or -1 on error.

Check errno for error codes and meanings.

hdfsUtime()
Changes the access and modification times of a file or directory.

Supported by libMapRClient for hadoop-2.x

Signature

```
int hdfsUtime(hdfsFS fs, const char* path, tTime mtime, tTime )
```

Parameters

Parameter	Description
fs	The handle of the filesystem where the file or directory is located. Obtain this handle with one of the `hdfsConnect()` APIs.
path	The path to the file or directory.
mtime	The new modification time or 0 (if you want to set only the access time) in seconds.

Return Value

Returns 0 on success, -1 on error.

errno is set to EINVAL if the input arguments are invalid.

errno is set to EPERM if the process does not have enough privileges to perform the operation.

hdfsWrite()

Supported by libMapRClient for hadoop-2.x

Writes to the specified open file.

If the file is opened in write-only mode, writes start at offset 0 because write-only mode causes the content of the file to be truncated when the file is opened.

If the file is opened in append mode, data is appended to the end of the file.

If there are concurrent writes that start at the same offset, only the last write to finish persists.

If a call to `hdfsSeek()` moves the offset past the end of the file before a call to `hdfsWrite()`, the result is a hole in the file between the previous end of the file and the offset at which the write begins.

You can obtain the size of a file in bytes by calling `hdfsGetPathInfo()`.

Flushes to the server happen automatically at intervals during a write operation. After a write operation is finished, either call `hdfsFlush()` explicitly or call `hdfsFlush()` implicitly by calling `hdfsCloseFile()` to be sure that any data remaining in the write buffer is flushed.

On error, pending write buffers are flushed to the server.

Signature

```
tsize hdfsWrite(hdfsFS fs, hdfsFile file, const void* buffer, tSize length)
```

Parameters

Parameter	Description
fs	The handle of the filesystem where the file is located. Obtain this handle with one of the `hdfsConnect()` APIs.
file	The handle of the file. Obtain this handle with one of the `hdfsOpenFile()` APIs.
buffer	The buffer containing the data to be written.
length	The number of bytes to write. This value cannot be zero. The maximum length of the buffer is the maximum size of the `tSize` data type.

Return Value

Returns the number of bytes written, -1 on error.

Check errno for error codes and meanings.

Accessing MapR XD Distributed File and Object Store in Java Applications

As a high-performance filesystem, portions of the MapR XD Distributed File and Object Store file client are based on a native maprfs library. When developing an application, specifying dependence on the JAR file that includes the maprfs library enables you to build applications without having to manage platform-specific dependencies.

The following sections describe how to access the MapR XD Distributed File and Object Store in a Java program.

Writing a Java Application

In your Java application, you will use a Configuration object to interface with the filesystem. When you instantiate a Configuration object, it is created with values from Hadoop configuration files.

If the program is built with JAR files from the MapR Data Platform installation, the Hadoop 1 configuration files are in the `$MAPR_HOME/hadoop/hadoop-<version>/conf` directory, and the Hadoop 2 configuration files are in the `$HADOOP_HOME/etc/hadoop` directory. This Hadoop configuration directory is in the hadoop classpath that you include when you compile and run the Java program.

If the program is built through maven using mapr maven artifacts, the default Hadoop configuration files are included in the maven artifacts. The user needs to programmatically update the Hadoop configuration to match the Hadoop configuration files on the MapR cluster.

Sample Code

The following sample code shows how to interface with MapR filesystem using Java. The example creates a

last-updated: Apr 11, 2022

directory, writes a file, then reads the contents of the file.

```
/* Copyright (c) 2009 & onwards. MapR
Tech, Inc., All rights reserved */

//package com.mapr.fs;

import java.net.*;
import org.apache.hadoop.fs.*;
import org.apache.hadoop.conf.*;

/**
* Assumes mapr installed in /opt/mapr
*
* Compilation:
* javac -cp $(hadoop classpath)
MapRTest.java
*
* Run:
* java -cp .:$(hadoop classpath)
MapRTest /test
*/
public class MapRTest
{
 public static void main(String
args[]) throws Exception {
        byte buf[] = new
byte[ 65*1024];
        int ac = 0;
        if (args.length != 1) {

System.out.println("usage: MapRTest
pathname");
        return;
        }

        // maprfs:/// -> uses
the first entry in /opt/mapr/conf/
mapr-clusters.conf
        // maprfs:///mapr/
my.cluster.com/
        // /mapr/my.cluster.com/

        // String uri = "maprfs:///";
        String dirname = args[ac++];

        Configuration conf = new
Configuration();

        //FileSystem fs =
FileSystem.get(URI.create(uri),
conf); // if wanting to use a
different cluster
        FileSystem fs =
FileSystem.get(conf);

        Path dirpath = new
Path( dirname + "/dir");
        Path wfilepath = new
Path( dirname + "/file.w");
        //Path rfilepath = new
```

```
Path( dirname + "/file.r");
        Path rfilepath = wfilepath;


        // try mkdir
        boolean res =
fs.mkdirs( dirpath);
        if (!res) {

System.out.println("mkdir failed,
path: " + dirpath);
        return;
        }

        System.out.println( "mkdir( "
+ dirpath + ") went ok, now writing
file");

        // create wfile
        FSDataOutputStream ostr =
fs.create( wfilepath,
                true, // overwrite
                512, // buffersize
                (short) 1, //
replication
                (long)
(64*1024*1024) // chunksize
                );
        ostr.write(buf);
        ostr.close();

        System.out.println( "write( "
+ wfilepath + ") went ok");

        // read rfile
        System.out.println( "reading
file: " + rfilepath);
        FSDataInputStream istr =
fs.open( rfilepath);
        int bb = istr.readInt();
        istr.close();
        System.out.println( "Read
ok");
        }
}
```

Compiling and Running a Java Application

You can compile and run the Java application using JAR files from the mapr maven repository or from the Data Fabric installation.

Using JARs from the Maven Repository

Maven artifacts from version 2.1.2 onward are published to https://repository.mapr.com/maven/. When compiling for MapR Data Platform core version 6.1, add the following dependency to the `pom.xml` file for your project:

```
<dependency>
   <groupId>org.apache.hadoop</
groupId>
  <artifactId>hadoop-common</
```

```
artifactId>
  <version>2.7.0-mapr-1808</version>
</dependency>
```

This dependency adds the dependencies from the mapr maven repository the next time you do a `mvn clean install`. The JAR that includes the `maprfs` library is a dependency for the `hadoop-common` artifact.

For a complete list of artifacts and further details, see Maven Artifacts for MapR on page 4155.

Using JARs from the MapR Data Platform Installation

The `maprfs` library is included in the hadoop classpath. Add the hadoop classpath to the JAVA classpath when you compile and run the Java application.

- To compile the sample code, use the following command:

```
javac -cp $(hadoop classpath)
MapRTest.java
```

- To run the sample code, use the following command:

```
java -cp .:$(hadoop classpath)
MapRTest /test
```

Loading the MapR Data Platform Native Library

By default, the root class loader will load the native library to allow all children to see and access it. If the native library is loaded by a child class, other classes will not be able to access the library. To allow applications and associated child classes to access the symbols and variables in the native library, we recommend loading the native library via the root loader.

The loading of the native library via the root class loader is accomplished by injecting code into the root loader. If MapR Data Platform runs on top of applications (such as Tomcat) where it does not have access to the root class loader, the native library will not be loaded. Child classes that try to access the symbols under the assumption that the root class loader successfully loaded the native library will fail.

The parameter `-Dmapr.library.flatclass`, when specified with Java, disables the injection of code via the root class loader, thus disabling the loading of the native library using the root class loader. Instead, the application trying to access the symbols can load the native library themselves. However, since the native library can be loaded only once and can only be seen by the application loading it, ensure that only one application within the JVM attempts to load and access the native library.

Garbage Collection in MapR Data Platform

The garbage collection (GC) algorithms in Java provide opportunities for performance optimizations for your application. Java provides the following GC algorithms:

- *Serial* GC. This algorithm is typically used in client-style applications that don't require low pause times. Specify `-XX:+UseSerialGC` to use this algorithm.
- *Parallel* GC, which is optimized to maximize throughput. Specify `-XX:+UseParNewGC` to use this algorithm.

last-updated: Apr 11, 2022

- *Mostly-Concurrent* or *Concurrent Mark-Sweep* GC, which is optimized to minimize latency. Specify `-XX:+UseConcMarkSweepGC` to use this algorithm.
- *Garbage First* GC, a new GC algorithm intended to replace Concurrent Mark-Sweep GC. Specify `-XX:+UseG1GC` to use this algorithm.

Consider testing your application with different GC algorithms to determine their effects on performance.

Flags for GC Debugging

Set the following flags in Java to log the GC algorithm's behavior for later analysis:

```
-verbose:gc
-Xloggc:<filename>
-XX:+PrintGCDetails
-XX:+PrintGCDateStamps
-XX:+PrintTenuringDistribution
-XX:+PrintGCApplicationConcurrentTime
-XX:+PrintGCApplicationStoppedTime
```

For more information, see the Java Garbage Collection Tuning document or the Java Garbage Collection links.

Converting fid and volid

The following MapR File System APIs are available in `com.mapr.fs.MapRFileSystem` for converting fid to file path and volid to volume name:

- `public String getMountPathFidCached(String fidStr) throws IOException`
- `public String getVolumeNameCached(int volId) throws IOException`
- `public String getVolumeName(int volId) throws IOException`
- `public String getMountPathFid(String fidStr) throws IOException`

Converting fid to File Path

The `getMountPathFid(string)` and `getMountPathFidCached(string)` APIs can be used for converting file ID to the full path to the file. The `getMountPathFid()` API makes a call to CLDB and MapR File System to get the file path from the fid. Because this API does not cache or store this information locally, it might make repeated requests to CLDB and MapR File System for the same fid and this might result in many RPCs to both CLDB and MapR File System. The `getMountPathFidCached()` API makes a call the CLDB and MapR File System one time and stores the information locally in the shared library of the client. For subsequent calls, it uses the locally stored information to retrieve the file path from the fid. However, if there are many files in the volume, there might still be a large number of calls to CLDB and MapR File System to determine the file path for each fid in the volume. The caching is useful if the API attempts to determine the file path for the same fid repeatedly. The cache is purged after 15 seconds. If the file name changes before the cache is purged, you will see the old name for the file until the cache expires. You can use these APIs to convert the fid to the file path.

For example, the sample consumer application and the sample uncached consumer application for consuming

last-updated: Apr 11, 2022

audit logs as stream messages use these methods as shown below.

- **Sample Cached Consumer**

```
{
     String token =
st1.nextToken();
     /* If the field has fid,
expand it using Cached API */
     if (token.endsWith("Fid")) {
       String lfidStr =
st1.nextToken();
       String path= null;
       try {
           path =
fs.getMountPathFidCached(lfidStr);
// Expand FID to path
       } catch (IOException e){
     }
     lfidPath =
"\"FidPath\":\""+path+"\",";
     // System.out.println("\nPAth
for fid " + lfidStr +  "is " +
path);
}
```

- **Sample Uncached Consumer**

```
{
     String token =
st1.nextToken();
     if (token.endsWith("Fid")) {
         String lfidStr =
st1.nextToken();
         String path= null;
         try {
           path =
fs.getMountPathFid(lfidStr);//
Expand FID to path
         } catch (IOException e){
         }
     lfidPath =
"\"FidPath\":\""+path+"\",";
     // System.out.println("\nPAth
for fid " + lfidStr +  "is " +
path);
}
```

Converting volid to Volume Name

The `getVolumeName()` and `getVolumeNameCached()` APIs can be used for converting volume IDs to volume name. The `getVolumeName()` API makes a call to the CLDB every time to get the volume name from the volid and this may result in too many RPCs to CLDB. The `getVolumeNameCached()` API makes a call to the CLDB one time and stores the information locally in the shared library of the client. For subsequent calls, it uses the locally stored information to retrieve the volume name from the volid. The cache is purged after

last-updated: Apr 11, 2022

15 seconds. You can use these APIs to convert the volid to volume name.

For example, the sample consumer application and the sample uncached consumer application for consuming audit logs as stream messages uses these methods as shown below.

- **Sample Cached Consumer**

```
if (token.endsWith("volumeId")) {
       String volid =
st1.nextToken();
       String name= null;
       try {
         int volumeId =
Integer.parseInt(volid);
           // Cached API to
convert volume Id to volume Name
           name =
fs.getVolumeNameCached(volumeId);
         }
       catch (IOException e){
       }
       lvolName =
"\"VolumeName\":\""+name+"\",";
       //
System.out.println("\nVolume Name
for volid " + volid +  "is " +
name);
}
```

- **Sample Uncached Consumer**

```
if (token.endsWith("volumeId")) {
       String volid =
st1.nextToken();
       String name= null;
       try {
         int volumeId =
Integer.parseInt(volid);
           // API to convert
volume Id to volume Name
           name =
fs.getVolumeName(volumeId);
         }
       catch (IOException e){
       }
       lvolName =
"\"VolumeName\":\""+name+"\",";
       //
System.out.println("\nVolume Name
for volid " + volid +  "is " +
name);
}
```

Sample Applications
Demonstrates how to set ACEs using the Java APIs.

Sample Application

The sample application demonstrates how to set, get, modify, and delete ACES on files using the Java APIs.

Before running this application, verify that you have access to a cluster running MapR File System. To build and run this application:

1. Set the classpath:

```
export CLASSPATH=`hadoop classpath`
```

2. Compile the java file:

```
javac FileAceTest.java
```

3. Run the final `FileAceTest.class` file.

The application imports the following libraries:

- `java.io.*`
- `java.net.*`
- `java.util.*`

The application performs the actions described in the following sections.

Connects to a filesystem

The application tries to connect to the first MapR File System cluster that is specified in the `mapr-clusters.conf` file in the `$MAPR_HOME/conf` directory on the client. After connecting to the filesystem, the application returns a handle to the filesystem.

```
Configuration conf = new
Configuration();
FileSystem fs = FileSystem.get(conf);
//MapRFileSystem fs =
getMapRFileSystem();
```

Creates a new directory and a new file in the directory

The application creates a new directory and a file in the directory.

```
Path testDir = new Path(rootDir +
"FileAceTest");
mkDir(fs, testDir);

Path testFile = new Path(testDir + "/
testFile");
createFile(fs, testFile);
```

Sets ACEs on the new directory and the file

The application then sets ACEs to grant m7user1 user or the root user permissions to list the contents of the directory, which is required for the user to write and/or execute files in the directory. In addition, the

application sets ACEs on the file in the directory to grant the m7user1 user read access on the file.

```
MapRFileAce ace = new
MapRFileAce(MapRFileAce.AccessType.REA
DFILE);
ace.setBooleanExpression("u:m7user1");
aces.add(ace);
ace = new
MapRFileAce(MapRFileAce.AccessType.REA
DDIR);
ace.setBooleanExpression("u:m7user1|
u:root");
aces.add(ace);
((MapRFileSystem)fs).setAces(testDir,
aces);
((MapRFileSystem)fs).setAces(testFile,
 aces);
```

Verifies the ACEs set on the directory and file

The application then prints the ACEs set on the directory and file.

```
List<MapRFileAce> newDirAces =
((MapRFileSystem)fs).getAces(testDir);
System.out.println("Path: " +
testDir);
for (int i = 0; i <
newDirAces.size(); ++i) {

System.out.println(newDirAces.get(i).g
etAccessType() + ": " +

newDirAces.get(i).getBooleanExpressio
n());
}

List<MapRFileAce> newFileAces =
((MapRFileSystem)fs).getAces(testFile)
;
System.out.println("Path: " +
testFile);
for (int i = 0; i <
newFileAces.size(); ++i) {

System.out.println(newFileAces.get(i).
getAccessType() + ": " +

newFileAces.get(i).getBooleanExpressio
n());
}
```

Modifies the ACEs on the directory and file

The application modifies the ACEs on the directory to grant m7user2 user also permissions to list the contents of the directory, which is required for the users to write and/or execute files in the directory. It also modifies the ACEs on the file to grant write access on the file to m7user2 user. Please note that when modifying ACEs on the file to grant write access

to m7user2 user, the application does not change read access, which was granted to m7user1 user.

```
aces = new ArrayList<MapRFileAce>();
ace = new
MapRFileAce(MapRFileAce.AccessType.REA
DDIR);
ace.setBooleanExpression("u:m7user1|
u:root|u:m7user2");
aces.add(ace);
ace = new
MapRFileAce(MapRFileAce.AccessType.WRI
TEFILE);
ace.setBooleanExpression("u:m7user2");
aces.add(ace);
((MapRFileSystem)fs).modifyAces(testDi
r, aces);
((MapRFileSystem)fs).modifyAces(testFi
le, aces);
```

Verifies the changes to ACEs on the directory and file

The application prints the changes in ACEs on the directory and the file.

```
newDirAces =
((MapRFileSystem)fs).getAces(testDir);
System.out.println("Path: " +
testDir);
for (int i = 0; i <
newDirAces.size(); ++i) {

System.out.println(newDirAces.get(i).g
etAccessType() + ": " +

newDirAces.get(i).getBooleanExpressio
n());
}

newFileAces =
((MapRFileSystem)fs).getAces(testFile)
;
System.out.println("Path: " +
testFile);
for (int i = 0; i <
newFileAces.size(); ++i) {

System.out.println(newFileAces.get(i).
getAccessType() + ": " +

newFileAces.get(i).getBooleanExpressio
n());
}
```

Deletes ACEs on directory and file

The application deletes all ACEs on the directory and the file.

```
((MapRFileSystem)fs).deleteAces(testDi
r);
((MapRFileSystem)fs).deleteAces(testFi
le);
```

Verifies the ACEs on the directory and the file

The application prints the ACEs on the directory and the file after they are deleted.

```
newDirAces =
((MapRFileSystem)fs).getAces(testDir);
System.out.println("Path: " +
testDir);
if (newDirAces == null ||
newDirAces.size() == 0)
  System.out.println("AceCount: 0");
else
  System.out.println("AceCount: " +
newDirAces.size());
System.out.println("");
newFileAces =
((MapRFileSystem)fs).getAces(testFile)
;
System.out.println("Path: " +
testFile);
if (newFileAces == null ||
newFileAces.size() == 0)
  System.out.println("AceCount: 0");
else
  System.out.println("AceCount: " +
newFileAces.size());
```

Example FileAceTest.java File

```
import java.io.*;
import java.net.*;
import java.util.*;

import org.apache.hadoop.fs.*;
import org.apache.hadoop.conf.*;

import com.mapr.fs.MapRFileAce;
import com.mapr.fs.MapRFileSystem;

public class FileAceTest {
  public FileSystem fs = null;

  public static void mkDir(FileSystem fs, Path path) throws IOException {
    boolean res = fs.mkdirs(path);
    if (!res) {
      throw new IOException("mkdir failed, path: " + path);
    }
  }

  public static void createFile(FileSystem fs, Path path) throws Exception {
    byte buf[] = new byte[1024];

    FSDataOutputStream ostr = fs.create(path,
                                        true, // overwrite
                                        512, // buffersize
                                        (short) 1, // replication
                                        (long)(64*1024*1024) // chunksize
                                       );
    ostr.write(buf);
    ostr.close();
  }
```

```
  public static void rmR(FileSystem fs, Path path) throws IOException {
    boolean res = fs.delete(path, true /*recursive*/);
    if (!res) {
      throw new IOException("rmR failed, path: " + path);
    }
  }

  public static void main(String args[]) throws Exception {
    String rootDir = "maprfs:///";

    Configuration conf = new Configuration();
    FileSystem fs = FileSystem.get(conf);

    Path testDir = new Path(rootDir + "FileAceTest");
    mkDir(fs, testDir);

    Path testFile = new Path(testDir + "/testFile");
    createFile(fs, testFile);

    ArrayList<MapRFileAce> aces = new ArrayList<MapRFileAce>();

    // Set
    System.out.println("SETTING ACES");
    MapRFileAce ace = new MapRFileAce(MapRFileAce.AccessType.READFILE);
    ace.setBooleanExpression("u:m7user1");
    aces.add(ace);
    ace = new MapRFileAce(MapRFileAce.AccessType.READDIR);
    ace.setBooleanExpression("u:m7user1|u:root");
    aces.add(ace);
    ((MapRFileSystem)fs).setAces(testDir, aces);
    ((MapRFileSystem)fs).setAces(testFile, aces);

    // Get
    System.out.println("GETTING ACES");
    List<MapRFileAce> newDirAces = ((MapRFileSystem)fs).getAces(testDir);
    System.out.println("Path: " + testDir);
    for (int i = 0; i < newDirAces.size(); ++i) {
      System.out.println(newDirAces.get(i).getAccessType() + ": " +
                         newDirAces.get(i).getBooleanExpression());
    }
    System.out.println("");
    List<MapRFileAce> newFileAces = ((MapRFileSystem)fs).getAces(testFile);
    System.out.println("Path: " + testFile);
    for (int i = 0; i < newFileAces.size(); ++i) {
      System.out.println(newFileAces.get(i).getAccessType() + ": " +
                         newFileAces.get(i).getBooleanExpression());
    }

    // Modify
    System.out.println("MODIFYING ACES");
    aces = new ArrayList<MapRFileAce>();
    ace = new MapRFileAce(MapRFileAce.AccessType.READDIR);
    ace.setBooleanExpression("u:m7user1|u:root|u:m7user2");
    aces.add(ace);
    ace = new MapRFileAce(MapRFileAce.AccessType.WRITEFILE);
    ace.setBooleanExpression("u:m7user2");
    aces.add(ace);
    ((MapRFileSystem)fs).modifyAces(testDir, aces);
    ((MapRFileSystem)fs).modifyAces(testFile, aces);

    // Get
    System.out.println("GETTING ACES");
    newDirAces = ((MapRFileSystem)fs).getAces(testDir);
    System.out.println("Path: " + testDir);
```


last-updated: Apr 11, 2022

```
    for (int i = 0; i < newDirAces.size(); ++i) {
      System.out.println(newDirAces.get(i).getAccessType() + ": " +
                         newDirAces.get(i).getBooleanExpression());
    }
    System.out.println("");
    newFileAces = ((MapRFileSystem)fs).getAces(testFile);
    System.out.println("Path: " + testFile);
    for (int i = 0; i < newFileAces.size(); ++i) {
      System.out.println(newFileAces.get(i).getAccessType() + ": " +
                         newFileAces.get(i).getBooleanExpression());
    }

    // Delete
    System.out.println("DELETING ACES");
    ((MapRFileSystem)fs).deleteAces(testDir);
    ((MapRFileSystem)fs).deleteAces(testFile);

    // Get
    System.out.println("GETTING ACES");
    newDirAces = ((MapRFileSystem)fs).getAces(testDir);
    System.out.println("Path: " + testDir);
    if (newDirAces == null || newDirAces.size() == 0)
      System.out.println("AceCount: 0");
    else
      System.out.println("AceCount: " + newDirAces.size());
    System.out.println("");
    newFileAces = ((MapRFileSystem)fs).getAces(testFile);
    System.out.println("Path: " + testFile);
    if (newFileAces == null || newFileAces.size() == 0)
      System.out.println("AceCount: 0");
    else
      System.out.println("AceCount: " + newFileAces.size());

    // Get
    System.out.println("GETTING ACES");
    newDirAces = ((MapRFileSystem)fs).getAces(testDir);
    System.out.println("Path: " + testDir);
    for (int i = 0; i < newDirAces.size(); ++i) {
      System.out.println(newDirAces.get(i).getAccessType() + ": " +
                         newDirAces.get(i).getBooleanExpression());
    }
    System.out.println("");
    newFileAces = ((MapRFileSystem)fs).getAces(testFile);
    System.out.println("Path: " + testFile);
    for (int i = 0; i < newFileAces.size(); ++i) {
      System.out.println(newFileAces.get(i).getAccessType() + ": " +
                         newFileAces.get(i).getBooleanExpression());
    }

    // Remove path
    rmR(fs, testDir);
  }
}
```

Troubleshooting

My application that includes maprfs-0.1.jar is now missing dependencies and fails to link.

As of version 2.1.2 , the contents of `maprfs-0.1.jar` are now in two JAR files:

- `maprfs-<version>.jar`

- `maprfs-jni-<version>.jar`

The `<version>` refers to the version of the distribution. For example, if you have an existing application written for `maprfs-0.1.jar` and you update it to load `maprfs-2.1.2.jar`, you must also include `maprfs-jni-2.1.2.jar`. This change was made to enable loading on distributed class-loader environments that use the `maprfs` libraries to access the filesystem from multiple contexts.

These JAR files are installed in the `/opt/mapr/hadoop/hadoop<version>/lib/` directory, or can be accessed via the Maven Central Repository.

MapR Database and Apps

This section contains information about developing client applications for JSON and key-value tables.

Why use MapR Database?

From a developer's point-of-view, MapR Database provides the following capabilities:

- **Extreme scale for CRUD operations**: Enabled by the integration of MapR Database with the MapR Data Platform Filesystem, CRUD operations are extremely fast and efficient.
- **Flexible data model**: MapR Database can be used as both a document database and a column-oriented database. So if the content structure changes, the applications do not need to be re-written.
- **Rich query**: Integration with Apache Drill on page 3185 for MapR Database provides a low-latency distributed query engine for large-scale datasets, including structured and semi-structured/nested data.
- **Integration with Apache Spark**: MapR Database provides MapR Database Connectors for Apache Spark on page 4050 that allow you to access MapR Database tables through Spark applications.
- **Strong data consistency**: Consistently fast response with strong data consistency with row/document level ACID transactions and in-memory database options for faster speeds.

MapR Database JSON provides additional benefits:

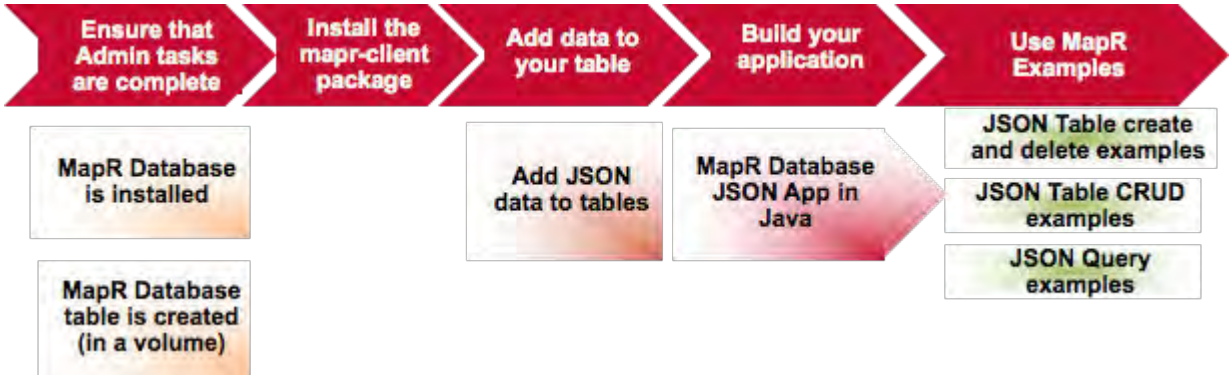
- **High performance via Secondary Indexes on page 544**: No memory copying. No need to retrieve the full document to make updates due the log-based database architecture. No application changes needed to leverage secondary indexes for efficient query execution.
- **Easy application development**: JSON constructs such as maps, arrays, and data types are supported natively.
- **Language-specific client APIs:** Java, Python, and Node.js client APIs

Note: MapR Data Platform has a universal namespace which means that the same namespace is used for files, tables, and streams. A universal namespace streamlines application development for different operations.

How Do I Get Started with MapR Database JSON?

The following diagram illustrates an end-to-end flow associated with getting started with MapR Database JSON.

last-updated: Apr 11, 2022



1. Ensure that the administrative tasks are complete as described in the introduction to MapR Database Administration.
2. Describes how to install the MapR Client package. This package allows you to run applications from your client machine.
3. This topics describes the methods available for loading OJAI documents into JSON tables.
4. This topic describes how to use mapr dbshell to create JSON tables and add JSON document to JSON tables.
5. This topic provides information on how to develop Java client applications for MapR Database JSON tables.
6. This topic provides examples creating, listing, and deleting MapR Database JSON tables.
7. This topic provides examples performing CRUD operations on JSON documents in MapR Database JSON tables.
8. Installing MapR describes how to install MapR software and Ecosystem components. You can install manually, with the MapR Installer, or with the MapR Installer Stanza.
9. This topic describes the different methods for creating tables and provides examples.
10. This topic provides examples querying JSON documents in MapR Database JSON tables.

Note: *This flow is not the only way to get started!*

You can also run through end-to-end examples using preconfigured, single node MapR Data Platform clusters. The following table describes two options:

MapR Sandbox	The MapR Sandbox is a virtual machine (VM) that runs a single node MapR cluster. Sandbox Tutorial for JSON on page 2516 describes how to use that VM with MapR Database JSON to do the following: • Install and configure MapR Database in the VM • Run a sample Java OJAI application • Use MapR Database Shell to perform simple MapR Database JSON operations
Container for Developers	MapR Container for Developers on page 96 is a Docker container that runs a single node cluster. Using MapR Database JSON: Getting Started, you can do the following:

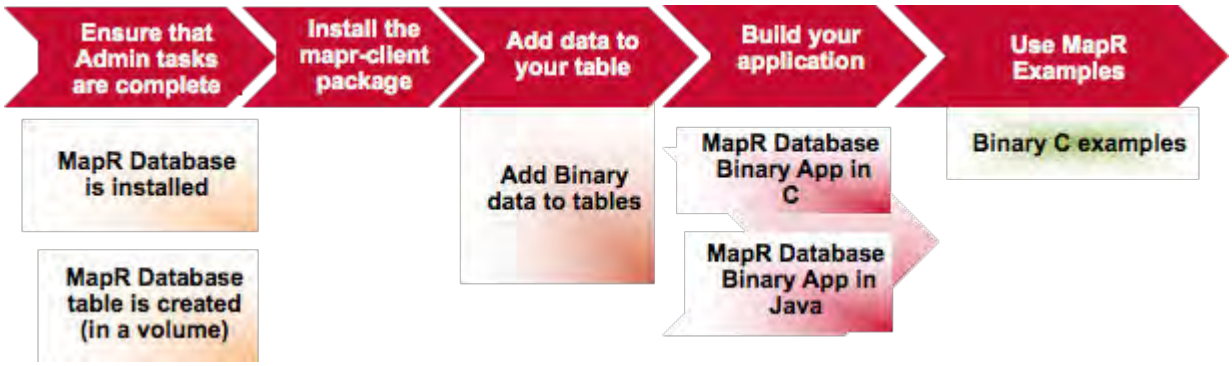
- Set up the Docker container
- Import data into MapR Database JSON tables
- Use MapR Database Shell to query, insert, and update JSON documents in MapR Database JSON tables
- Use Drill to query MapR Database JSON tables
- Run sample Java OJAI applications
- Run a sample Spark application using the MapR Database OJAI Connector for Apache Spark

Useful MapR Database JSON Developer Resources

Getting Started and Examples	Tools, Utilities, and Applications	General (Blogs, etc)	API Details
Managing JSON Tables on page 2522 - Examples creating, listing, and deleting MapR Database JSON tables	maprcli and REST API Syntax on page 1522	Data Modeling Guidelines for NoSQL JSON Document Databases	MapR Database JSON Client API **Note:** Beginning with core version 6.0, the MapR Database `Table` interface in the MapR Database JSON Client API is deprecated and replaced by the `DocumentStore` interface in the OJAI API library. See the next row for details on that API.
Managing JSON Documents on page 2542 - Examples performing CRUD operations on JSON documents in MapR Database JSON tables	Utilities for MapR Database JSON Tables on page 5312	App development with OJAI	Java OJAI Client API
Querying JSON Documents on page 2579 - Examples querying JSON documents in MapR Database JSON tables	Apache Drill on page 3185	How to Build Applications on a NoSQL Document Database and Perform Analytics in Place	Node.js OJAI Client API
Getting Started with the MapR Database JSON REST API on page 2697	Understanding the MapR Database OJAI Connector for Spark on page 4050		Python OJAI Client API
Getting Started with the Node.js OJAI Client on page 2673			
Getting Started with the Python OJAI Client on page 2678			
Tutorials - Instructions and code to build a sample web application using MapR Database JSON			

How Do I Get Started with MapR Database Binary?

The following diagram illustrates an end-to-end flow associated with getting started with MapR Database Binary.



1. Ensure that the administrative tasks are complete as described in the introduction to MapR Database Administration.
2. Describes how to install the MapR Client package. This package allows you to run applications from your client machine.
3. This topic describes how to bulk load data into binary tables.
4. This topic provides information on creating C applications for MapR Database binary tables.
5. This topic provides information on creating Java application for MapR Database binary tables.
6. This topics provides a step-by-step C application example that performs CRUD operations on a MapR Database binary tables.
7. Installing MapR describes how to install MapR software and Ecosystem components. You can install manually, with the MapR Installer, or with the MapR Installer Stanza.
8. This topic describes the different methods for creating tables and provides examples.
9. This topic provides information for developing client applications for MapR Database Binary tables.

Useful MapR Database Binary Developer Resources

Getting Started and Examples	Tools, Utilities, and Applications	General (Blogs, etc)
MapR Database Sample C Application on page 2459 - C application example for binary tables	maprcli and REST API Syntax on page 1522	High Performance C APIS on MapR Database
	Utilities for MapR Database Binary Tables on page 5329	
	Apache Drill on page 3185	
	MapR Database Binary Connector for Apache Spark on page 4101	

Installing the mapr-client Package

The `mapr-client` package must be installed on each node where you will be building and running your applications. This package installs all of the MapR Libraries needed for application development regardless of programming language or type of MapR Database table (binary or JSON).

last-updated: Apr 11, 2022

Note: The package `mapr-core` contains the files that are in the `mapr-client` package. If you have installed the `mapr-core` package, you do not need to install the `mapr-client` package.

Complete the following steps to install the `mapr-client` package from a repository:

1. Configure the repository to point to `https://package.mapr.hpe.com/releases/<release version>/<operating system>`

   For example, if your VM has a CentOS operating system, edit the **/etc/yum.repos.d/mapr_core.repo** file and add the location.

2. Based on your operating system, run one of the following commands to install the package:
   - On Red Hat /Centos: `yum install mapr-client`
   - On Ubuntu: `apt-get install mapr-client`
   - On SLES: `zypper install mapr-client`

Passing the MapR Database Table Path

This topic describes the methods for passing a MapR Database table name. Binary table names can be passed by either specifying the table path in the API or by setting the table path in the **core-site.xml** file. JSON table names are passed by specifying the table path in the API.

Specifying the Table Path in the API

With this method, you provide the complete path to the table using the following format: `/mapr/<cluster>/<volume>/<table>`

Note: This format is independent of the programming language. This method is used for both MapR Database binary and JSON tables.

For example, if you were adding a new column family to a table and you had the following information:

- Cluster name: `newyork`
- Volume name: `vol1`
- Table name: `table1`

The table path would be `/mapr/newyork/vol1/table1`

Specifying the Table Path in the core-site.xml File

With this method, you specify the table path with `hbase.table.namespace.mappings` property in the **/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/core-site.xml** file.

Note: This method is specific to MapR Database binary tables only. This method is independent of the programming language.

In the following example, all tables that you create and access via the API, will be in the `/tables_dir1` directory. If you specified the table name `table1` in the API, the full path to the table would be `/tables_dir1/table1`.

```
<property>
   <name>hbase.table.namespace.mappings</name>
   <value>*:/tables_dir1</value>
</property>
```


last-updated: Apr 11, 2022

Tuning Parameters for Client Apps

Though tuning client applications is generally not necessary, MapR does offer tuning parameters to change the behavior of client-side caching.

Client application cache pending puts in buffers that are unique to each tablet (*region*, in HBase terminology). Individual put buffers are flushed when they are full or idle. As a result of this architecture and behavior, client applications tend to send RPCs of 128KB when flushing puts to disk, which results in better performance than flushing a single global buffer would, as that could result in a large number of small RPCs.

You can change the values of parameters that affect how put buffers are flushed. Set values for them in the `hbase-site.xml` file or `core-site.xml` file in your MapR installation. If a non-default value for a parameter is set in both files, the value in the `hbase-site.xml` file is used.

db.mapr.putbuffer.threshold.mb	Specifies the size of the cumulative put buffer for all tablets in the client application. When this threshold is reached, the put buffer that is most full is flushed to its tablet. The default value is 32MB. Increasing this value can improve performance when an application performs operations on very large tables or on a very large number of tables.
db.mapr.putbuffer.threshold.sec	Specifies the number of seconds that MapR Database should wait before flushing an idle put buffer. The default value is 3. This parameter has no effect if automatic flushing is enabled.
fs.mapr.tabletlru.size.kb	Specifies the size of the metadata cache for all tables in a client application.. The metadata for each tablet is 128 bytes. The default value of this parameter is 512KB, which allows for the caching of the metadata of 4,096 tablets. When this metadata cache is full, any operation on a tablet for which the metadata is not cached requires an RPC to fetch that tablet's metadata. Moreover, caching the newly retrieved metadata removes from the cache the metadata of a different tablet. Increasing this value can improve performance when an application performs operations on very large tables or on a very large number of tables.
fs.mapr.threads	Specifies the number of threads to use when flushing put buffers. Each thread makes synchronous RPCs when flushing. The default value is 64.

Developing Applications for Binary Tables

MapR Database provides a C API, `libMapRClient` and partially supports the Apache HBase 1.1 Java APIs for performing operation on MapR Database binary tables.

The MapR Database C API, `libMapRClient`, runs more efficiently on MapR Database and performs faster against MapR Database tables than the open source library of C APIs, `libhbase`, that is used to create and access Apache HBase tables. The `libMapRClient` header files are in this directory: **/opt/mapr/include/hbase**

MapR Database also supports all of the Apache HBase 1.1 Java APIs, except where noted in this documentation. For a number of critical Java APIs, for filters, and for comparators, this documentation explicitly lists what is supported, rather than what is not supported.

You can easily port existing applications that use the open-source version of `libhbase` or the HBase Java APIs to use MapR Database binary tables.

Current Limitations

- Custom HBase filters are not supported.
- HBase co-processors are not supported.

Note: Filters used with Scan operations support regular expressions. When you filter scans on MapR Database tables, you can use regular expressions that comprise the Perl Compatible Regular Expressions (PCRE) library as well as a subset of the regular expressions that are supported in java.util.regex.pattern. See HBase Java Regular Expressions Support on page 2506 for a list of supported regular expressions.

Creating C Apps - Binary Tables

MapR provides a library of C APIs – `libMapRClient` – for performing operations on MapR Database binary tables.

The MapR Database `libMapRClient` C API library is MapR's extension of the `libhbase` C API library. The `libMapRClient` header files are in this directory: **/opt/mapr/include/hbase**

`libMapRClient` uses the following conventions:

- All data types are prefixed with 'hb_'.
- All exported functions are annotated with HBASE_API, prefixed with 'hb_' and named using the following convention: `'hb_<subject>_<operation>_[<object>|<property>]'`
- All asynchronous APIs take a callback which is triggered when a request completes. This callback can be triggered in the caller's thread or in another thread. To avoid any potential deadlock or starvation, applications should not block in the callback routine.
- All callbacks take a void pointer for application developers to supply their own data. This void pointer is passed when callback is triggered.

Note: No explicit batching is supported for asynchronous APIs.

Warning: It is the responsibility of applications to free up all backing data buffers. However, for asynchronous APIs, applications must wait before freeing buffers until after receiving callbacks or manipulating results. For better performance of asynchronous APIs, `libMapRClient` does not copy data buffers that are allocated for mutations, gets, and scans. These buffers hold table names, name space identifiers, row keys, column-family names, and column names or qualifiers. Instead, libMaprClient temporarily takes ownership of the buffers and references them with pointers until the callback is triggered. Therefore, applications should not free memory buffers before receiving callbacks for mutations. Applications also should not free memory buffers before receiving results for gets and scans. If applications must read results, the applications should not free memory buffers until the results are destroyed.

last-updated: Apr 11, 2022

Note: When one of these asynchronous APIs is invoked, a work item is created and queued for processing on the client:

- `hb_client_destroy()`
- `hb_get_send()`
- `hb_mutation_send()`
- `hb_scanner_destroy()`
- `hb_scanner_next()`

The work item is picked up as soon as possible by a thread in a thread pool.

Client applications can often call these asynchronous APIs faster than the work items are processed. To ensure that the queue of work items does not grow without bound, the configuration parameter `fs.mapr.pool.queue.max_size` is set by default to 10,000. You can modify this parameter in the `/opt/mapr/conf/dbclient.conf` file for a client.

Whenever the number of work items in the queue reaches this limit, `libMapRClient` returns the `ENOBUFS` error for each asynchronous call. Client applications are expected to handle this error, and can try the call again later.

libMapRClient C APIs

This section provides the MapR Database `libMapRClient` C API library. This library is MapR's extension of the `libhbase` C API library. The `libMapRClient` header files are in the directory: **/opt/mapr/include/hbase**.

The `libMapRClient` API implements functions in addition to the functions in the `libhbase` API.

Note: Your applications need include only the `hbase.h` header file. The header files are provided for display purposes.

admin.h	Describes the APIs for Apache HBase table administration operations such as creating and enabling tables, checking if tables exist, and deleting tables, to name a few .
client.h	Describes the APIs for Apache HBase client side operations such as creating and terminating client connections, and flushing buffered client-side writes to Apache HBase.
coldesc.h	Describes the APIs for performing operations such as creation, deletion, and setting the maximum and minimum number of cell versions to be retained for each Apache HBase column family.
connection.h	`libMapRClient` includes a function in the `connection.h` header file: `hb_connection_create_as_user()`. This function provides support for impersonation, so that you can connect to a MapR cluster and access MapR Database tables by using a specific username. The user that is passed with the `hb_connection_create_as_user()` API must have permissions on the tables that the application accesses. For example, to read from a table, the user must have the `readperm` permission. To write to a table, the user must have the `writeperm` permission.

last-updated: Apr 11, 2022

	See Enabling Table and Stream Authorizations with ACEs on page 1461. For `hb_connection_create()` and `hb_connection_create_as_user()`, the standard C APIs for Apache HBase require a list of ZooKeeper nodes. For MapR Database, this list is interpreted as a list of CLDB nodes. The `zk_root` parameter is ignored. If `zk_quorum` is NULL, then the connection is created to the default cluster that is listed in the `mapr-clusters.conf` file.
get.h	Describes the APIs to query and fetch data from Apache HBase tables.
hbase.h	Describes the APIs and data structures of a C client for Apache HBase.
log.h	Describes the APIs to manage Apache HBase logs.
macros.h	Defines internal macros that Apache HBase uses for its operations.
multiget.h	Describes the APIs to queue and manage multiple GET requests to fetch data from Apache HBase tables.
mutations.h	Describes the APIs for row and column mutations on Apache HBase tables.
result.h	Describes the buffers for internal temporary storage of results.
scanner.h	Describes the APIs for the client side scanner to scan and request rows from the Apache HBase server.
types.h	Defines the data types and error codes that Apache HBase uses.

C API Examples

This section provides examples using the MapR Database `libMapRClient` C API to operate on MapR Database binary tables.

The MapR Database `libMapRClient` C API library is MapR's extension of the `libhbase` C API library. The libMapRClient header files are in this directory: `/opt/mapr/include/hbase`

Filtering SCAN Operation Results Example

This example shows filtering on the results of a SCAN operation.

```
for (uint32_t i = 0; i < num_filters; ++i) {
    hb_scanner_t scanner = NULL;
    hb_scanner_create(client, &scanner);
    hb_scanner_set_table(scanner, table_name, table_name_len);
    hb_scanner_set_num_max_rows(scanner, 3);  // maximum 3 rows at a time
    hb_scanner_set_num_versions(scanner, 10); // up to 10 versions of the
cell
    hb_scanner_set_filter(scanner, (byte_t *)filters[i],
strlen(filters[i]));
    hb_scanner_next(scanner, scan_callback, NULL); // dispatch the call
    wait_for_scan();
 }
```

This example uses the following array of filters:

```
static char filters[][200] = {"RandomRowFilter(0.5)",
       "ColumnCountGetFilter(2)",
       "ColumnPaginationFilter(1)",
       "ColumnPrefixFilter('column-a')",
       "FamilyFilter(=,'binaryprefix:f')",
       "PrefixFilter('row_') AND QualifierFilter(<,'binaryprefix:g')",
       "SKIP TimestampsFilter(1392222222222)",
       "WHILE ValueFilter(=,'binaryprefix:cell2_value_v1')",
       "FuzzyRowFilter('row00','00001')",
       "TimestampsFilter(1430937732000,1431024132000)"};
       }
```

Note: For more information about suppport for HBase Java Filters by the MapR Database C API, see HBase Java Filters Support on page 2473

Filtering GET Operation Results Example
This example shows filtering on the results of a GET operation.

```
{
    bytebuffer rowKey = bytebuffer_strcpy("row_with_two_cells");
    hb_get_t get = NULL;
    hb_get_create(rowKey->buffer, rowKey->length, &get);
    hb_get_add_column(get, FAMILIES[0], 1, NULL, 0);
    hb_get_add_column(get, FAMILIES[1], 1, NULL, 0);
    hb_get_set_table(get, table_name, table_name_len);
    hb_get_set_num_versions(get, 10); // up to ten versions of each column
    hb_get_set_filter(get, (byte_t *)filters[9], strlen(filters[9]));
    get_done = false;
    hb_get_send(client, get, get_callback, rowKey);
    wait_for_get();
 }
```

This example uses the following array of filters:

```
static char filters[][200] = {"RandomRowFilter(0.5)",
       "ColumnCountGetFilter(2)",
       "ColumnPaginationFilter(1)",
       "ColumnPrefixFilter('column-a')",
       "FamilyFilter(=,'binaryprefix:f')",
       "PrefixFilter('row_') AND QualifierFilter(<,'binaryprefix:g')",
       "SKIP TimestampsFilter(1392222222222)",
       "WHILE ValueFilter(=,'binaryprefix:cell2_value_v1')",
       "FuzzyRowFilter('row00','00001')",
       "TimestampsFilter(1430937732000,1431024132000)"};
       }
```

Note: For more information about suppport for HBase Java Filters by the MapR Database C API, see HBase Java Filters Support on page 2473

Impersonation Example
This sample application demonstrates the capabilities of the new C API for impersonation.

The sample also shows how to use the API in your own programs. The application is located in the `/opt/mapr/examples/interactive` directory.

Prerequisite for compiling and running this sample application

Install the `mapr-client` package on the node where you will build the application. See Installing the MapR Client on page 389. If the `mapr-core` package is already installed, you do not need to install the `mapr-client` package.

Compiling this sample application

To compile and run this sample application, set the `MAPR_IMPERSONATION_ENABLED` environment variable to `true` and then read the instructions in the `README` file in the `/opt/mapr/examples/interactive` directory.

Though the application links against `libjvm`, a Java virtual machine (JVM) is not spawned. However, the `libMapRClient` does have Java dependencies.

Set Time Range Example
This example scans MapR Database binary tables and sets the time range.

```
int32_t scanWithTimeranges( std::string table_name, int32_t num_versions,
uint64_t max_num_rows, std::string start_row_key,
                std::string end_row_key, int64_t min_ts, int64_t max_ts,
std::string name_space, std::string column_name) {
        hb_scanner_t scanner = NULL;
        int32_t retCode;

        scan_num_rows = 0;
        scan_cell_count = 0;
        scan_done = false;
        // scanner object create
        hb_scanner_create(client, &scanner);

        scan_data_t *scan_data = (scan_data_t *) calloc(1,
sizeof(scan_data_t));

        // set the table to scan
        scan_data->table_name_ = bytebuffer_printf("%s",
table_name.c_str());
        hb_scanner_set_table(scanner, (char *)
scan_data->table_name_->buffer, scan_data->table_name_->length));

        // start and end row - optional
        scan_data->start_row_key_ = bytebuffer_printf("%s",
start_row_key.c_str());
        hb_scanner_set_start_row(scanner, scan_data->start_row_key_->buffer,
                        scan_data->start_row_key_->length);

        scan_data->end_row_key_ = bytebuffer_printf("%s",
end_row_key.c_str());
        hb_scanner_set_end_row(scanner, scan_data->end_row_key_->buffer,
                        scan_data->end_row_key_->length);

        // add columns
        bytebuffer cfName   = bytebuffer_printf("%s",
data_qualifier[0].c_str());
        bytebuffer columnName   = bytebuffer_printf("%s",
data_qualifier[1].c_str());
        retCode=hb_scanner_add_column(scanner,(byte_t
*)cfName->buffer,cfName->length,columnName->buffer,columnName->length);

        // set versions
        hb_scanner_set_num_versions(scanner, num_versions);

        // set timerange
```


last-updated: Apr 11, 2022

```
        hb_scanner_set_timerange(scanner, min_ts, max_ts);

        // scan data
        retCode = hb_scanner_next(scanner, scan_callback, scan_data);
        return retCode;
}
```

Set Time Stamp Example
This example scans MapR Database binary tables and sets the time stamp.

```
int32_t scanWithTimestamp( std::string table_name, int32_t num_versions,
uint64_t max_num_rows, std::string start_row_key,
                std::string end_row_key, int64_t ts, std::string
name_space, std::string column_name) {
        hb_scanner_t scanner = NULL;
        int32_t retCode;

        scan_num_rows = 0;
        scan_cell_count = 0;
        scan_done = false;

        // scanner object create
        hb_scanner_create(client, &scanner);

        scan_data_t *scan_data = (scan_data_t *) calloc(1,
sizeof(scan_data_t));

        // set the table to scan
        scan_data->table_name_ = bytebuffer_printf("%s",
table_name.c_str());
        hb_scanner_set_table(scanner, (char *)
scan_data->table_name_->buffer, scan_data->table_name_->length));

        // start and end row - optional
        scan_data->start_row_key_ = bytebuffer_printf("%s",
start_row_key.c_str());
        hb_scanner_set_start_row(scanner, scan_data->start_row_key_->buffer,
                        scan_data->start_row_key_->length);

        scan_data->end_row_key_ = bytebuffer_printf("%s",
end_row_key.c_str());
        hb_scanner_set_end_row(scanner, scan_data->end_row_key_->buffer,
                        scan_data->end_row_key_->length);

        // add columns
        bytebuffer cfName   = bytebuffer_printf("%s",
data_qualifier[0].c_str());
        bytebuffer columnName   = bytebuffer_printf("%s",
data_qualifier[1].c_str());
        retCode=hb_scanner_add_column(scanner,(byte_t
*)cfName->buffer,cfName->length,columnName->buffer,columnName->length);

        // set versions
        hb_scanner_set_num_versions(scanner, num_versions);

        // set timestamp
        hb_scanner_set_timestamp(scanner, ts);

        // scan data
        retCode = hb_scanner_next(scanner, scan_callback, scan_data);
        return retCode;
}
```

Delete Specific Cells Example
This example deletes specific cells that correspond to a specific timestamp.

```
nt32_t deleteRowExactTS(std::string table_name, std::string row_key,
        std::vector<std::string> data, int64_t ts) {

        int32_t retCode = 0;
        // initialize delete object
        hb_delete_t del = NULL;

        row_data_t *row_data = (row_data_t *) calloc(1, sizeof(row_data_t));
        row_data->key = bytebuffer_printf("%s", row_key.c_str());
        row_data->tablename = bytebuffer_printf("%s",table_name.c_str());

        // create delete object
        hb_delete_create(row_data->key->buffer, row_data->key->length,
&del);

        cell_data_t *prevCell=NULL;
        // add cells that needs to be deleted
        for(int t=0; t < (int)data.size(); t++) {
             vector<string> data_qualifier = split(data.at(t),':');
             cell_data_t *cell_data= new_cell_data();
             if(t==0)
                row_data->first_cell = cell_data;
             else
                prevCell->next_cell = cell_data;

             cell_data->columnFamily =
bytebuffer_printf("%s",data_qualifier[0].c_str());
             cell_data->columnName =
bytebuffer_printf("%s",data_qualifier[1].c_str());

             //add column, fam/column/ts to delete the exact version
             retCode=hb_delete_add_column_exact(del,
cell_data->columnFamily->buffer, cell_data->columnFamily->length,

cell_data->columnName->buffer,cell_data->columnName->length,ts);
             prevCell = cell_data;
        }

        // set the table name in delete mutation object
        hb_mutation_set_table(del, (const char
*)row_data->tablename->buffer,row_data->tablename->length);

        //send the delete request
        retCode = hb_mutation_send(client, del, delete_callback, row_data);
        return retCode;
}
```

MapR Database Sample C Application
MapR provides a sample C application that accesses and performs operations on MapR Database binary tables. This section describes the various operation performed by the application.

The sample C application is located in the `/opt/mapr/examples/sample` directory.

Prerequisites

In order to compile and run the sample application, install the `mapr-client` package on the node where you will build the application. See Impersonation Example on page 2456. If the `mapr-core` package is already installed, you do not need to install the `mapr-client` package.

Compiling

To compile and run the sample application, read the instructions in the `README` file in the `/opt/mapr/examples/sample` directory.

Set the log level and specify the log stream

APIs used

These two APIs are defined in the header file `log.h`:

- `hb_log_set_level()`: Sets the log output level. The levels are defined in the header file types.h.
- `hb_log_set_stream()`: Sets the location of the log output. By default, log messages are sent to stderr.

Code

```
hb_log_set_level(HBASE_LOG_LEVEL_DEBUG); // defaults to INFO
const char *logFilePath = getenv("HBASE_LOG_FILE");
if (logFilePath != NULL) {
  FILE* logFile = fopen(logFilePath, "a");
  if (!logFile) {
    retCode = errno;
    fprintf(stderr, "Unable to open log file \"%s\"", logFilePath);
    perror(NULL);
    goto cleanup;
  }
  hb_log_set_stream(logFile); // defaults to stderr
}
```

Log levels are specified in the header file `types.h`.

```
/**
 *  Log levels
 */
typedef enum {
  HBASE_LOG_LEVEL_INVALID = 0,
  HBASE_LOG_LEVEL_FATAL   = 1,
  HBASE_LOG_LEVEL_ERROR   = 2,
  HBASE_LOG_LEVEL_WARN    = 3,
  HBASE_LOG_LEVEL_INFO    = 4,
  HBASE_LOG_LEVEL_DEBUG   = 5,
  HBASE_LOG_LEVEL_TRACE   = 6
} HBaseLogLevel;
```

Create a connection

API used: `hb_connection_create()`

Use this function to connect to the MapR cluster.

This API takes three parameters:

```
const char *zk_ensemble,     /* [in] NULL terminated, comma separated
                                     *   string of CLDB servers. e.g.
                                     *   "<server1[:port]>,...".
const char *zk_root,              /* [in] Ignored for MapR-DB. */
hb_connection_t *connection_ptr); /* [out] pointer to hb_connection_t */
```

There are two methods by which you can use the `zk_ensemble` parameter to determine how the MapR client connection locates the MapR cluster to connect to:

Set `zk_ensemble` to NULL to connect to the default cluster that is defined in the `mapr-clusters.conf` file.	MapR recommends this method, which uses the configuration information that is listed for the cluster in the `mapr-clusters.conf` file.
Set `zk_ensemble` to a string that includes hostnames or IP addresses.	With this method, the client application can connect to a non-default cluster explicitly. The client application searches through the `mapr-clusters.conf` file to find a cluster entry with a matching hostname/IP address. The first entry that is found to contain a matching hostname[:port]/IP address[:port} is used for the connection, as is the configuration information for that entry. If none of the hostnames or IP addresses specified for `zk_ensemble` are located in entries in `mapr-clusters.conf`, or if `mapr-clusters.conf` does not exist, the client application tries to connect to the first specified hostname[:port]/IP address[:port]. If the client application cannot make a connection, it moves to the next specified hostname[:port]/IP address[:port].  **Warning:** Because no `mapr-clusters.conf` file is involved, no additional configuration information is used for connections. For example, connections made in this way cannot be secure because no security parameters are provided.

Examples

```
//connect to default cluster specified in mapr-clusters.conf (preferred)
if ((err = hb_connection_create(NULL,
                                NULL,
                                &connection)) != 0) {
  HBASE_LOG_ERROR("Could not create MapR-DB connection : errorCode = %d.",
err);
  goto cleanup;
}

//Connect directly to cluster with these specified IP addresses.
//Typically this means there is no mapr-clusters.conf and security not used.
if ((err = hb_connection_create("192.168.1.1:7222,192.168.1.2:7222",
                                NULL /* ignored */,
                                &connection)) != 0) {
  HBASE_LOG_ERROR("Could not create MapR-DB connection : errorCode = %d.",
err);
  goto cleanup;
}
```

Code in the sample application

```
if ((retCode = hb_connection_create(zk_ensemble,
                                    zk_root_znode,
                                    &connection)) != 0) {
  HBASE_LOG_ERROR("Could not create HBase connection : errorCode = %d.",
retCode);
  goto cleanup;
}
```


last-updated: Apr 11, 2022

This API is defined in the header file `connection.h`:

Create a table

The sample application creates a table by calling a function named ensureTable.

APIs used

The definition of this function uses these APIs:

APIs defined in the header file `admin.h`.

- `hb_admin_table_create()`: Creates a table. Returns 0 on success or an error code.
- `hb_admin_table_delete()`: Deletes a table. Returns 0 on success or an error code.
- `hb_admin_destroy()`: Disconnects the hb_admin object, releasing any internal objects or connections that were created in the background.
- `hb_admin_table_disable()`: Disables an HBase table. Returns 0 on success or an error code. Only sets a flag in memory to say that the table is disabled.

  Note: Tables never need to be disabled or enabled in MapR Database, which has no notion of disabling or enabling tables. However, applications ported from HBase to MapR Database will attempt to disable and enable tables. By placing a flag in memory, MapR Database allows those applications to proceed without error when performing admin functions on tables.
- `hb_admin_table_enable()`: Enables an HBase table. Returns 0 on success or an error code. As with hb_admin_table_enable, this function sets a flag in memory and performs no other operation.
- `hb_admin_table_enabled()`: Checks whether an HBase table is enabled. Returns 0 if the table is enabled. If the table is disabled, returns either the error message HBASE_TABLE_DISABLED or an error code, if an error occurs. In MapR Database, this API only checks for the existence of an in-memory flag that indicates whether to consider a table as disabled or enabled.
- `hb_admin_table_exists()`: Checks whether a table exists. Returns 0 on success or an error code.

APIs defined in the header file `coldesc.h`.

- `hb_coldesc_create()`: Creates a column-family descriptor. Returns a handle to an hb_columndesc object or NULL, if unsuccessful.
- `hb_coldesc_set_maxversions()`: Sets the maximum number of cell versions to be retained for the column family. The default is 3.
- `hb_coldesc_set_minversions()`: Sets the minimum number of cell versions to be retained for the column family. The default is 0.
- `hb_coldesc_set_ttl()`: Sets the time-to-live value in seconds for data in column cells. The default is forever.

- `hb_coldesc_set_inmemory()`: Boolean. Determines whether preference is given to values of this column family for storage with row keys. Because row keys are cached in memory in preference to row data, column-family data that is stored inline with the row keys is also cached in memory.

  For all column families in a table together, up to 200 bytes of row data will be stored inline with each row key. Storing data inline with a row key might speed retrieval of the data from a column family because disk access can often be avoided. For each column family, up to 32 bytes can be stored inline with each row key even if its `inmemory` parameter is set to `false`, but preference will be given to column families where this parameter is set to `true`. A column family can have more than 32 bytes stored inline if its `inmemory` parameter is set to `true`.

  If the total number of bytes for all column families together exceeds 200 for a row, then preference for inclusion within the inline storage for that row is given to column families that have the `inmemory` parameter set to `true`. All of the data for a column family will be stored in-line with the row key, or none will be. If the contents in a column family for a particular row are larger than the maximum number of bytes that are allowed to be stored for that column family, no data at all will be stored in-line for that column family.

  The default value for the `inmemory` parameter is `false`.

- `hb_coldesc_destroy()`: Releases resources that are held by a column-family descriptor.

Sequence of steps in the ensureTable function

1. Create an admin handle from the connection.
2. Using the admin handle, check whether the specified table exists.
3. If the table exists, delete it.
4. Specify columns and column families for the table.
5. Create the table.
6. Check whether the table is enabled. If it isn't enabled, enable it.
7. Disable the table and then enable it again.
8. Destroy the column descriptors.
9. Destroy the `hb_admin` structure.

Code for the ensureTable function at line 325

```
static int
ensureTable(hb_connection_t connection, const char *table_name) {
  int32_t retCode = 0;
  hb_admin_t admin = NULL;

  if ((retCode = hb_admin_create(connection, &admin)) != 0) {
    HBASE_LOG_ERROR("Could not create HBase admin : errorCode = %d.",
retCode);
    goto cleanup;
  }

  if ((retCode = hb_admin_table_exists(admin, NULL, table_name)) == 0) {
    HBASE_LOG_INFO("Table '%s' exists, deleting...", table_name);
    if ((retCode = hb_admin_table_delete(admin, NULL, table_name)) != 0) {
      HBASE_LOG_ERROR("Could not delete table %s[%d].", table_name,
retCode);
```

```
      goto cleanup;
    }
  } else if (retCode != ENOENT) {
    HBASE_LOG_ERROR("Error while checking if the table exists: errorCode =
%d.", retCode);
    goto cleanup;
  }

  hb_coldesc_create(FAMILIES[0], 1, &HCD[0]);
  hb_coldesc_set_maxversions(HCD[0], 2);
  hb_coldesc_set_minversions(HCD[0], 1);
  hb_coldesc_set_ttl(HCD[0], 2147480000);
  hb_coldesc_set_inmemory(HCD[0], 1);

  hb_coldesc_create(FAMILIES[1], 1, &HCD[1]);

  HBASE_LOG_INFO("Creating table '%s'...", table_name);
  if ((retCode = hb_admin_table_create(admin, NULL, table_name, HCD, 2)) ==
0) {
    HBASE_LOG_INFO("Table '%s' created, verifying if enabled.", table_name);
    retCode = hb_admin_table_enabled(admin, NULL, table_name);
    CHECK_API_ERROR(retCode,
        "Table '%s' is %senabled, result %d.", table_name, retCode?"not
":"");
    retCode = hb_admin_table_disable(admin, NULL, table_name);
    CHECK_API_ERROR(retCode,
        "Attempted to disable table '%s', result %d.", table_name);
    retCode = hb_admin_table_disable(admin, NULL, table_name);
    CHECK_API_ERROR(retCode,
        "Attempted to disable table '%s' again, result %d.", table_name);
    retCode = hb_admin_table_enable(admin, NULL, table_name);
    CHECK_API_ERROR(retCode,
        "Attempted to enable table '%s', result %d.", table_name);
    retCode = hb_admin_table_enable(admin, NULL, table_name);
    CHECK_API_ERROR(retCode,
        "Attempted to enable table '%s' again, result %d.", table_name);
  }
  hb_coldesc_destroy(HCD[0]);
  hb_coldesc_destroy(HCD[1]);

cleanup:
  if (admin) {
    hb_admin_destroy(admin, NULL, NULL);
  }
  return retCode;
}
```

Code to call the ensureTable function

```
if ((retCode = ensureTable(connection, table_name)) != 0) {
  HBASE_LOG_ERROR("Failed to ensure table %s : errorCode = %d", table_name,
retCode);
  goto cleanup;
}
```

Create a client

API used

`hb_client_create()`: Initializes a handle to `hb_client_t` object that can be passed to other APIs. You need to use this method only once per cluster. The returned handle is thread-safe. This API is defined in the `client.h` header file.

Code

```
if ((retCode = hb_client_create(connection, &client)) != 0) {
  HBASE_LOG_ERROR("Could not connect to HBase cluster : errorCode = %d.",
retCode);
  goto cleanup;
}
```

Asynchronously put ten rows of one cell each

APIs used in this operation

The first 6 APIs are defined in the header file `mutations.h`:

- `hb_put_create()`: Creates a structure for the put operation and returns its handle.
- `hb_mutation_set_table()`: Sets the name of the table for the put operation.
- `hb_mutation_set_bufferable()`: Sets whether or not the RPC call for the put operation can be buffered on the client side.
- `hb_put_add_cell()`: Adds a cell to the put structure. The row key of the cell must be the same as the row key of the put structure.
- `hb_mutation_send()`: Queues the put operation for sending to the server. Mutations are not performed atomically and can be batched in a non-deterministic way on either the client side or the server side. Any buffer attached to a mutation object (put or delete) must not be altered until the callback has been received.

The last API is defined in the header file `client.h`:

- `hb_client_flush()`: Flushes any buffered client-side write operations to the server. The callback is invoked after everything that was buffered at the time of the call is flushed. Invocation of the callback is a guarantee that all outstanding RPC calls are complete.

Sequence of steps in this code extract

1. Create a row object named row_data.
2. Create a put object.
3. Specify the name of the table.
4. Set whether or not the RPC call for the put operation can be buffered on the client side.
5. Create cell data.
6. Create a cell.
7. Add the cell to the row.

last-updated: Apr 11, 2022

8. Queue the put.
9. After following the steps above 10 times, flush the puts to the server.
10. Wait for the RPC calls to complete.

Code

```
// let's send a batch of 10 puts with single cell asynchronously
outstanding_puts_count += num_puts;
for (int i = 0; i < num_puts; ++i) {
  row_data_t *row_data = (row_data_t *) calloc(1, sizeof(row_data_t));
  row_data->key   = bytebuffer_printf("%s%02d", rowkey_prefix, i);
  hb_put_create(row_data->key->buffer, row_data->key->length, &put);
  hb_mutation_set_table(put, table_name, table_name_len);
  hb_mutation_set_durability(put, DURABILITY_SKIP_WAL);
  hb_mutation_set_bufferable(put, false);

  cell_data_t *cell_data = new_cell_data();
  row_data->first_cell = cell_data;
  cell_data->value = bytebuffer_printf("%s%02d", value_prefix, i);

  hb_cell_t *cell = (hb_cell_t*) calloc(1, sizeof(hb_cell_t));
  cell_data->hb_cell = cell;

  cell->row = row_data->key->buffer;
  cell->row_len = row_data->key->length;
  cell->family = FAMILIES[rand() % 2];
  cell->family_len = 1;
  cell->qualifier = column_a->buffer;
  cell->qualifier_len = column_a->length;
  cell->value = cell_data->value->buffer;
  cell->value_len = cell_data->value->length;
  cell->ts = HBASE_LATEST_TIMESTAMP;

  hb_put_add_cell(put, cell);
  HBASE_LOG_INFO("Sending row with row key : '%.*s'.",
                 cell->row_len, cell->row);
  hb_mutation_send(client, put, put_callback, row_data);
}
hb_client_flush(client, client_flush_callback, NULL);
wait_for_flush();

wait_for_puts(); // outside the loop, wait for 10 puts to complete
```

Asynchronously put two cells in a single row

APIs used

These APIs are defined in the header file `mutations.h`:

- `hb_put_create()`: Creates a structure for the put operation and returns its handle.
- `hb_mutation_set_table()`: Sets the table name for the mutation.
- `hb_put_add_cell()`: Adds a cell to the put structure. The row key of the cell must be the same as the row key of the put structure.
- `hb_mutation_send()`: Queues the put operation for sending to the server. Mutations are not performed atomically and can be batched in a non-deterministic way on either the client side or the server side. Any buffer attached to a mutation object (put or delete) must not be altered until the callback has been received.

Sequence of steps

1. Create a row object named row_data.
2. Create a put object.
3. Specify the name of the table.
4. Create cell data for the first cell.
5. Create the first cell.
6. Add the data to the first cell.
7. Add the first cell to the row.
8. Create cell data for the second cell.
9. Create the second cell.
10. Add the data to the second cell.
11. Add the second cell to the row.
12. Queue the put.
13. Wait 3 seconds, flush the queue, and wait for the RPC calls to complete.

`wait_for_puts()` without `hb_client_flush()`: Waits for all outstanding put requests to be flushed. If hb_client_flush() is not called before wait_for_puts(), it can take up to three seconds for outstanding put requests to be flushed.

Code

```
// now, let's put two cells in a single row
outstanding_puts_count++;
{
  row_data_t *row_data = (row_data_t *) calloc(1, sizeof(row_data_t));
  row_data->key = bytebuffer_printf("row_with_two_cells");
  hb_put_create(row_data->key->buffer, row_data->key->length, &put);
  hb_mutation_set_table(put, table_name, table_name_len);
  hb_mutation_set_durability(put, DURABILITY_SYNC_WAL);

  // first cell
  cell_data_t *cell1_data = new_cell_data();
  row_data->first_cell = cell1_data;
  cell1_data->value = bytebuffer_printf("cell1_value_v1");

  hb_cell_t *cell1 = (hb_cell_t*) calloc(1, sizeof(hb_cell_t));
  cell1_data->hb_cell = cell1;

  cell1->row = row_data->key->buffer;
  cell1->row_len = row_data->key->length;
  cell1->family = FAMILIES[0];
  cell1->family_len = 1;
  cell1->qualifier = column_a->buffer;
  cell1->qualifier_len = column_a->length;
  cell1->value = cell1_data->value->buffer;
  cell1->value_len = cell1_data->value->length;
  cell1->ts = 1391111111111L;
  hb_put_add_cell(put, cell1);
```


last-updated: Apr 11, 2022

```
    // second cell
    cell_data_t *cell2_data = new_cell_data();
    cell1_data->next_cell = cell2_data;
    cell2_data->value = bytebuffer_printf("cell2_value_v1");

    hb_cell_t *cell2 = (hb_cell_t*) calloc(1, sizeof(hb_cell_t));
    cell2_data->hb_cell = cell2;

    cell2->row = row_data->key->buffer;
    cell2->row_len = row_data->key->length;
    cell2->family = FAMILIES[1];
    cell2->family_len = 1;
    cell2->qualifier = column_b->buffer;
    cell2->qualifier_len = column_b->length;
    cell2->value = cell2_data->value->buffer;
    cell2->value_len = cell2_data->value->length;
    cell2->ts = 1391111111111L;
    hb_put_add_cell(put, cell2);

    HBASE_LOG_INFO("Sending row with row key : '%.*s'.",
                   cell1->row_len, cell1->row);
    hb_mutation_send(client, put, put_callback, row_data);
    wait_for_puts();
  }
```

Asynchronously put a second version in one column of one row

APIs used

These APIs are defined in the header file mutations.h:

- `hb_put_create()`: Creates a structure for the put operation and returns its handle.
- `hb_mutation_set_table()`: Sets the table name for the mutation.
- `hb_put_add_cell()`: Adds a cell to the put structure. The row key of the cell must be the same as the row key of the put structure.
- `hb_mutation_send()`: Queues the put operation for sending to the server. Mutations are not performed atomically and can be batched in a non-deterministic way on either the client side or the server side. Any buffer attached to a mutation object (put or delete) must not be altered until the callback has been received.

Sequence of steps

1. Create a row object named row_data.
2. Create a put object.
3. Specify the name of the table.
4. Create cell data for the first cell.
5. Create the first cell.
6. Add the data to the cell, using a later timestamp than in the previous operation.
7. Add the first cell to the row.
8. Queue the put.

9. Wait 3 seconds, flush the queue, and wait for the RPC calls to complete.

Code

```
  // now, let's put second version in one column
  outstanding_puts_count++;
  {
    row_data_t *row_data = (row_data_t *) calloc(1, sizeof(row_data_t));
    row_data->key = bytebuffer_printf("row_with_two_cells");
    hb_put_create(row_data->key->buffer, row_data->key->length, &put);
    hb_mutation_set_table(put, table_name, table_name_len);
    hb_mutation_set_durability(put, DURABILITY_SYNC_WAL);

    // first cell
    cell_data_t *cell1_data = new_cell_data();
    row_data->first_cell = cell1_data;
    cell1_data->value = bytebuffer_printf("cell1_value_v2");

    hb_cell_t *cell1 = (hb_cell_t*) calloc(1, sizeof(hb_cell_t));
    cell1_data->hb_cell = cell1;

    cell1->row = row_data->key->buffer;
    cell1->row_len = row_data->key->length;
    cell1->family = FAMILIES[0];
    cell1->family_len = 1;
    cell1->qualifier = column_a->buffer;
    cell1->qualifier_len = column_a->length;
    cell1->value = cell1_data->value->buffer;
    cell1->value_len = cell1_data->value->length;
    cell1->ts = 1392222222222L;
    hb_put_add_cell(put, cell1);

    HBASE_LOG_INFO("Sending row with row key : '%.*s'.",
                   cell1->row_len, cell1->row);
    hb_mutation_send(client, put, put_callback, row_data);
    wait_for_puts();
  }
```

Scan the entire table

APIs used

These APIs and more are defined in the header file `scanner.h`:

- `hb_scanner_create()`: Creates a client side row scanner. The returned scanner is not thread safe. No RPC will be invoked until the call to fetch the next set of rows is made. You can set the various attributes of this scanner until that point. @returns 0 on success, non-zero error code in case of failure.
- `hb_scanner_next()`: Request the next set of results from the server. You can set the maximum number of rows returned by this call using hb_scanner_set_num_max_rows().
- `hb_scanner_num_max_rows()`: Sets the maximum number of rows to scan per call to hb_scanner_next().
- `hb_scanner_num_versions()`: Sets the maximum versions of a column to fetch.
- `hb_scanner_set_table()`: Sets the name of the table to scan.

Sequence of steps

1. Create the scanner object.

2. Set the name of the table to scan.
3. Set the number of rows to scan at a time.
4. Set the number of versions to scan.
5. Request the next set of results from the server.
6. `wait_for_scan()`: wait for the rpc call to complete.

Code

```
  // now, scan the entire table
  {
    hb_scanner_t scanner = NULL;
    hb_scanner_create(client, &scanner);
    hb_scanner_set_table(scanner, table_name, table_name_len);
    hb_scanner_set_num_max_rows(scanner, 3);  // maximum 3 rows at a time
    hb_scanner_set_num_versions(scanner, 10); // up to 10 versions of the
cell
    hb_scanner_next(scanner, scan_callback, NULL); // dispatch the call
    wait_for_scan();
  }
```

Fetch a row that has two cells

APIs used

- `hb_get_add_column()`: Adds a column family and optionally a column qualifier to an hb_get_t object.
- `hb_get_create()`: Creates an `hb_get_t` object and populates the handle `get_ptr`.
- `hb_get_send()`: Queues the get request. The callback specified by `cb` is called on completion. Any buffers attached to the get object can be reclaimed only after the callback is received.
- `hb_get_set_num_versions()`: Sets maximum number of latest values of each column to be fetched. This API is optional.
- `hb_get_set_table()`: Sets the name of the table to get data from.

Sequence of steps

1. Create a row object named `rowKey`.
2. Create a get object.
3. Specify the column families and optional column qualifiers to get values from.
4. Specify the name of the table.
5. Specify the number of versions of the column values to get.
6. Queue the get request.
7. Wait for the get to complete.

Code

```
// fetch a row with row-key="row_with_two_cells"
{
  bytebuffer rowKey = bytebuffer_strcpy("row_with_two_cells");
  hb_get_t get = NULL;
  hb_get_create(rowKey->buffer, rowKey->length, &get);
  hb_get_add_column(get, FAMILIES[0], 1, NULL, 0);
  hb_get_add_column(get, FAMILIES[1], 1, NULL, 0);
  hb_get_set_table(get, table_name, table_name_len);
  hb_get_set_num_versions(get, 10); // up to ten versions of each column
  get_done = false;
  hb_get_send(client, get, get_callback, rowKey);
  wait_for_get();
}
```

Delete a specific version of a column in the row that was fetched

APIs used

The APIs that are used are defined in the header file `mutations.h`.

- `hb_delete_add_column()`: Set the column criteria for hb_delete_t object. Set the qualifier to NULL to delete all columns of a family. Only the cells with timestamp less than or equal to the specified timestamp are deleted. Set the timestamp to `INT64_MAX` to delete all versions of the column. This API is optional for deletes.
- `hb_delete_create()`: Creates a structure for delete operation and return its handle.
- `hb_mutation_set_table()`: Sets the table name for the mutation.
- `hb_mutation_send()`: Queues the put operation for sending to the server. Mutations are not performed atomically and can be batched in a non-deterministic way on either the client side or the server side. Any buffer attached to a mutation object (put or delete) must not be altered until the callback has been received.

Sequence of steps

1. Create a delete object for a row with a particular row key.
2. Add a column to the delete object.
3. Specify the name of the table from which to delete the cell version.
4. Queue the delete.
5. Wait three seconds for the delete to be flushed, then wait for the RPC call to complete.

Code

```
// delete a specific version of a column
{
  bytebuffer rowKey = bytebuffer_strcpy("row_with_two_cells");
  hb_delete_t del = NULL;
  hb_delete_create(rowKey->buffer, rowKey->length, &del);
  hb_delete_add_column(del, FAMILIES[0], 1,
      column_a->buffer, column_a->length, 1391111111112L);
  hb_mutation_set_table(del, table_name, table_name_len);
  delete_done = false;
  hb_mutation_send(client, del, delete_callback, rowKey);
```

```
    wait_for_delete();
  }
```

Destroy the client and the connection

APIs used

- `hb_client_destroy()`: Cleans up `hb_client_t` handle and releases any held resources. The callback is called after the connections are closed, but just before the client is freed. This API is defined in the header file `client.h`.
- `hb_connection_destroy()`: Destroys the connection and frees all resources allocated at creation time. This API is defined in the header file `connection.h`.

Sequence of steps

1. Destroy the client.
2. Destroy the connection.

Code

```
  if (client) {
    HBASE_LOG_INFO("Disconnecting client.");
    hb_client_destroy(client, client_disconnection_callback, NULL);
    wait_client_disconnection();
  }
  if (connection) {
    hb_connection_destroy(connection);
  }
```

Building and Launching C Applications

This topic describes basic setup for building and launching C application using the MapR `libMapRClient` C API library

Prerequisites

The MapR Database `libMapRClient` C API library is MapR's extension of the `libhbase` C API library). The libMapRClient header files are in this directory: **/opt/mapr/include/hbase**

- Verify that the `mapr-client` package is installed on the node. The `mapr-client` package must be installed on each node that builds an application. The `libMapRClient` header files are in this directory: `/opt/mapr/include/hbase`.
- Verify that both the `libMapRClient` library and `libjvm` shared libraries are in the application's library search path.

Building Applications

When building applications that use `libMapRClient`, run this command:

```
gcc -o <application_name> <source_file> -I/opt/mapr/include/hbase -L/opt/
mapr/lib/ -lMapRClient -L/usr/lib/jvm/java-7-sun/jre/lib/amd64/server -ljvm
```

For example, the following command builds the `hello_hbase` application with the `hello_hbase.c` source code:

```
gcc -o hello_hbase hello_hbase.c -I/opt/mapr/include/hbase -L/opt/
mapr/lib/ -lMapRClient -L/usr/lib/jvm/java-7-sun/jre/lib/amd64/server -ljvm
```

Note:

- The compiled libMapRClient is statically linked to the following third-party libraries: Crypto++: libcryptoapp.a (v5.6.2)Protobuf: libprotobuf-lite.a (v2.5.0)
- The `libMapRClient` library has dependencies on `libjvm`, though a JVM is not instantiated. In general, the `libjvm` library is located within the JDK/JRE installation directory.

Launching Applications

Before launching an application, set this value for the environment variable `LD_LIBRARY_PATH`:

```
/opt/mapr/lib:/usr/lib/jvm/java-6-openjdk-amd64/jre/lib/amd64/server
```

If the client is on Windows, append the following directories to the `PATH` environment variable:

- `$MAPR_HOME/lib`
- `$JAVA_HOME/bin/server`

If the application uses the `hb_connection_create_as_user` API for impersonation, set the `MAPR_IMPERSONATION_ENABLED` environment variable to `true`.

What To Do Next

Launch the application!

HBase Java Filters Support
The hb_get_set_filter() and hb_scanner_set_filter() C APIs are used to filter the results of GET and SCAN operations. These APIs are in the **get.h** and **scanner.h** header files.

They both take filters that are passed as strings, as well as the length of these strings. MapR Database parses the strings to construct filters.

Their signatures are:

```
int32_t hb_get_set_filter(hb_get_t get, const byte_t *filter, const int32_t
filterLen);
int32_t hb_scanner_set_filter(hb_scanner_t scanner, const byte_t *filter,
const int32_t filterLen);
```

For examples, see Filtering GET Operation Results Example on page 2456 and Filtering SCAN Operation Results Example on page 2455.

Filter Format and Arguments
Filters are specified in the Thrift Filter Language and are in this format: `FilterName (argument, argument,... , argument)`. Arguments that represent strings are enclosed in single quotation marks (‘). Arguments that represent booleans, integers, or comparison operators (<, <=, =, !=, >, >=) are not enclosed in single quotation marks.

Binary Operators

You can combine filters by using the binary operators `AND` and `OR`. For example, `PrefixFilter ('Row') AND PageFilter (1) AND FirstKeyOnlyFilter ()` returns all key-value pairs that match the following conditions:

- The row containing the key-value must start with the prefix "Row".
- The key-value must be located in the first row of the table.
- The key-value must be the first key-value pair in the row.

For another example, `(RowFilter (=, 'binary:Row 1') AND TimeStampsFilter (74689, 89734)) OR ColumnRangeFilter ('abc', true, 'xyz', false))` returns all key-value pairs that

Match both of the following conditions:

- The key-value is in a row for which the row key is "Row 1".
- The key-value has a timestamp of either 74689 or 89734.

Or match this condition:

- The key-value is located in a column that is lexicographically greater than or equal to "abc" and less than "xyz".

Unary Operators

You can also use the following unary operators with filters:

- SKIP

  For a particular row, if any of the key-values don't pass the filter condition, the entire row is skipped. For example, `SKIP ValueFilter (0)` omits rows in which any values are not 0.

- WHILE

  Rows are tested in order against the filter condition. Rows that meed the condition are included in the result set. When a row fails to meet the condition, filter processing stops and no more rows are tested.

Evaluation of Filters

When filters are combined with the binary operators, unary operators, or both, they are evaluated according to these rules:

1. First, evaluate the contents of parentheses.
2. Next, evaluate filters that use unary operators. Both SKIP and WHILE operators have the same precedence.
3. Finally, evaluate filters that use the binary operators. AND has higher precedence than OR.

For example, in a filter of the form `Filter1 AND Filter2 OR Filter`, `Filter1 AND Filter2` is evaluated and the result is *X*. Then, *X* `OR Filter3` is evaluated.

For another example, a filter of the form `Filter1 AND SKIP Filter2 OR Filter3` is evaluated in these steps:

1. Evaluate `SKIP Filter2` with the result being *X*.
2. Evaluate `Filter1 AND` *X* with the result being *Y*.
3. Evaluate *Y* `OR Filter3`.

Compare Operators and Comparators
This topic describesthe compare operators and comparators for comparison filters.

The comparison filters DependentColumnFilter, FamilyFilter, QualifierFilter, RowFilter, and ValueFilter use the following syntax:

```
filter(<compareOperator>, <comparatorType:Value>)
```

Compare Operators

The following compare operators are supported: <,<=, =, !=, >, >=

Comparators

There are four comparators:

Comparator	Description
BinaryComparator	This comparator lexicographically compares against the specified byte array . Values are byte arrays. For example, `binary:abc` matches values that are lexicographically greater than "abc".
BinaryPrefixComparator	This comparator lexicographically compares against a specified byte array. It only compares up to the length of this byte array. Values are byte arrays. For example, `binaryprefix:abc` matches values in which the first 3 characters are lexicographically equal to "abc"
RegexStringComparator	This comparator compares against the specified byte array using the given regular expression. You can use only the = and != compare operators with this comparator. Values are regular expressions. For example, `regexstring:ab*yz` matches values that begin with "ab" and end with "yz".
SubStringComparator	This comparator tests whether the given substring appears in a specified byte array. The comparison is case insensitive. You can use only the = and != compare operators with this comparator. Values are strings. For example, `substring:abc123` matches values that contains the substring "abc123".

Supported Filters

Filter	Format	Description
ColumnCountGetFilter	`ColumnCountGetFilter(`*x*`)`	Returns the first x columns in a row. Used for GET operations.

Filter	Format	Description
ColumnPaginationFilter	`ColumnPaginationFilter(x,y)`	Returns the first x columns after the number y of columns that is specified for the offset.
ColumnPrefixFilter	`ColumnPrefixFilter('prefix')`	Returns only those key-values in columns that have names that start with the specified prefix. The column prefix must be of the form “qualifier”.
ColumnRangeFilter	`ColumnRangFilter('minColumn','maxColumn',boolean,boolean)`	Returns only those key-values that are in columns that have names between minColumn and maxColumn. For example, if minColumn is 'an', and maxColumn is 'be', the filter returns key-values from columns named 'ana', 'bad', but not from columns named 'bed' or'eye' If minColumn is null, there is no lower bound. If maxColumn is null, there is no upper bound. This filter also takes two boolean variables to indicate whether to include the minColumn and maxColumn.
DependentColumnFilter	`DependentColumnFilter('family','qualifier')`	Tries to locate the specified column in each row and returns all key-values that have the same timestamp in that column. If a row does not contain the specified column, none of the key-values in that row are returned.
FamilyFilter	`FamilyFilter(compareOperator,'comparator:value')`	Filters by column family. If the comparison returns true, the filter returns all of the key-values in the matching column family.
FirstKeyOnlyFilter	`FirstKeyOnlyFilter()`	Returns the first key-value from each row.
FirstKeyValueMatchingQualifiersFilter	`FirstKeyValueMatchingQualifier('qualifier_1', 'qualifier_2',...'qualifier_n')`	Serially compares each qualifier in a row with the given qualifiers. If the current qualifier matches any of the given qualifiers, the filter stops and includes the current row (up to the current qualifier) in the result set.

Filter	Format	Description
FuzzyRowFilter	`FuzzyRowFilter('rowkey','fuzzy_info')`	Filters data based on fuzzy row key. Performs fast-forwards during scanning. It takes pairs (row key, fuzzy info) to match row keys. Where fuzzy info is a byte array with 0 or 1 as its values: 0 - means that this byte in provided row key is fixed, i.e. row key's byte at same position must match 1 - means that this byte in provided row key is NOT fixed, i.e. row key's byte at this position can be different from the one in provided row key Example: Let's assume row key format is userId_actionId_year_month. Length of userId is fixed and is 4, length of actionId is 2 and year and month are 4 and 2 bytes long respectively. Let's assume that we need to fetch all users that performed certain action (encoded as "99") in Jan of any year. Then the pair (row key, fuzzy info) would be the following: row key = "????_99_????_01" (one can use any value instead of "?") fuzzy info = "\x01\x01\x01\x01\x00\x00\x00\x00\x01\x01\x01\x01\x00\x00\x00" I.e. fuzzy info tells the matching mask is "????_99_????_01", where at ? can be any value.
InclusiveStopFilter	`InclusiveStopFilter('rowKey')`	Returns all key-values that are in the rows up to and including the specified row that has the specified row key.
KeyOnlyFilter	`KeyOnlyFilter()`	Returns the key component of each key-value.
MultipleColumnPrefixFilter	`MultipleColumnPrefixFilter('prefix_1','prefix_2',...,'prefix_n')`	Returns the key-values from columns that have names that begin with any of the specified prefixes.
PageFilter	`PageFilter(pageSize)`	Returns the number of rows that is equivalent to the specified page size..
PrefixFilter	`PrefixFilter('rowKey_prefix')`	Returns the key-values from a row that has a key which starts with the specified row-key prefix.
QualifierFilter	`QualifierFilter(compareOperator,'comparator:value')`	Filters by column. If the comparison returns true, the filter returns all of the key-values in the matching column.
RandomRowFilter	`RandomRowFilter(probability)`	Filters by probability. For example, RandomRowFilter(0.25) means that there is a 1 in 4 chance that the filter will pick the first row, a 1 in 4 chance that the filter will pick the next row, and so on until all rows in the table have been processed in this way.

last-updated: Apr 11, 2022

Filter	Format	Description
RowFilter	`RowFilter(compareOperator,'comparator:value')`	Filters by row key. If the comparison returns true, the filter returns all of the key-values in the matching row.
SingleColumnValueExcludeFilter	`SingleColumnValueExcludeFilter('columnFamily','qualifier', compareOperator,'comparator:value')`	This filter takes the same arguments and behaves the same as SingleColumnValueFilter: however, if the column is found and the condition passes, all of the columns of the row will be returned except for the tested column value.
SingleColumnValueFilter	`SingleColumnValueFilter('columnFamily','qualifier', compareOperator,'comparator:value')`	This filter takes a column family, a qualifier, a compare operator and a comparator. If the specified column is not found: all of the columns of that row will be emitted. If the column is found and the comparison with the comparator returns true, all of the columns of the row will be emitted. If the condition fails, the row will not be returned.
SkipFilter	`SKIP filter`	See the description of the SKIP unary operator above .
TimeStampsFilter	`TimeStampsFilter('timestamp_1','timestamp_2',..., 'timestamp_n')`	Returns the key-values that have timestamps that match any of the specified timestamps.
ValueFilter	`ValueFilter(compareOperator,'comparator:value')`	Filters by key-value. If the comparison returns true, the filter returns the matching key-value.
WhileMatchFilter	`WHILE filter`	See the description of the WHILE unary operator above .

Creating Java Apps - Binary Tables

This topics describes the supported Apache HBase Java APIs used for CRUD operations on MapR Database binary tables.

MapR Database supports all of the Apache HBase 1.0 Java APIs, except where noted in this documentation. For a number of critical Java APIs, for filters, and for comparators, this documentation explicitly lists what is supported, rather than what is not supported.

Code written for Apache HBase can be easily ported to use MapR Database binary tables.

MapR Database binary tables do not support low-level HBase API calls that are used to manipulate the state of an Apache HBase cluster. HBase API calls that are not supported by MapR Database tables report successful completion to allow legacy code written for Apache HBase to continue executing, but do not perform any actual operations.

Note: For the list of supported HBase v0.98 APIs, refer to the HBase Java API Support on page 2480.

Compiling and Running MapR Database Binary Applications

For applications that use the MapR Database Java API for binary tables, use Maven to compile and determine the application's dependencies. Then, when you run the application, specify those dependencies in the application's classpath.

last-updated: Apr 11, 2022

Compile and Determine Dependencies

1. Add MapR's maven repository to the list of repositories in your application's `pom.xml`, if it is not there already:

```
<repository>
      <id>mapr-releases</id>
      <url>https://repository.mapr.com/maven/</url>
      <snapshots><enabled>true</enabled></snapshots>
      <releases><enabled>true</enabled></releases>
</repository>
```

2. Add a dependency to the HBase client project:

```
<dependencies>
<dependency>
  <groupId>org.apache.hbase</groupId>
  <artifactId>hbase-client</artifactId>
  <version><version selected from the repository></version>
</dependency>
<dependencies>
```

Note:

- The hbase-client version mentioned above is an example. The actual version that your application requires is based on the current EEP and MapR version that you are running.The file versions are listed in the following location: https://repository.mapr.com/nexus/content/groups/mapr-public/org/apache/hbase/hbase-client/ Ensure that the version tags include reference to m7-<maprversion>. Otherwise, your applications will not be able to access MapR Database binary tables.
- If your application uses both the MapR Database JSON and MapR Database Binary APIs, you may encounter a conflict in the netty library used by each API. To avoid this, exclude the netty library from your `hbase-client` dependency as follows:

```
<dependency>
   <groupId>org.apache.hbase</groupId>
   <artifactId>hbase-client</artifactId>
   <version><version selected from the repository></version>
   <exclusions>
     <exclusion>
        <artifactId>netty-all</artifactId>
           <groupId>io.netty</groupId>
     </exclusion>
   </exclusions>
</dependency>
```

3. Use Maven to compile the application and resolve dependencies. For example, you can run `mvn clean package`.

Running Applications

When you develop a Java application, you can use a dependency management tool such as Maven to compile your application. However, it is recommended that do the following instead:

1. Compile the Java application without including dependencies

2. Specify the required classpath when you submit the application to the cluster

If you choose to bundle the JAR file, and there is a mismatch between the bundled JAR file and the version that your MapR cluster expects, this can result in failures. The failures differ depending on the version of MapR you are using. For more information, see Using the File System JAR to Connect to the Cluster on page 2367.

If the cluster is secure, the node must also have a mapr ticket configured for the user that runs the application.

You can use the following command to launch MapR Database binary applications:

```
java -cp <classpath>:. -Djava.library.path=/opt/mapr/lib <main class JAR>
<command line arguments>
```

Note: The classpath should include items that are part of the 'hbase classpath' script plus any other required dependencies.

HBase Java API Support
This topic describes the methods in the Apache HBase Java API library that are supported for MapR Database tables.
Admin Method Support
This topic lists the methods that the MapR Database supports in the HBase interface `Admin`.

The MapR Database tables support the following HBase methods, except where noted.

`abort(String why, Throwable e)`	*Modifier and Type*: `void` *Purpose*: Aborts the server or client *Supported*: No
`addColumn(TableName tableName, HColumnDescriptor column)`	*Modifier and Type*: `void` *Purpose*: Adds a column to an existing table *Supported*: Yes
`assign(byte[] regionName)`	*Modifier and Type*: `void` *Purpose*: Assigns a region *Supported*: No
`balancer()`	*Modifier and Type*: `boolean` *Purpose*: Invokes the balancer **Note:** The MapR Database does not require balancing. Therefore, the method is ignored. *Supported*: No
`cloneSnapshot(byte[] snapshotName, TableName tableName)`	*Modifier and Type*: `void` *Purpose*: Creates a new table by cloning the snapshot content *Supported*: No
`cloneSnapshot(String snapshotName, TableName tableName)`	*Modifier and Type*: `void` *Purpose*: Creates a new table by cloning the snapshot content *Supported*: No
`close()`	*Modifier and Type*: `void` *Purpose*: Releases any resources held *Supported*: Yes

closeRegion(byte[] regionname, String serverName)	*Modifier and Type*: `void` *Purpose*: Closes a region *Supported*: No
closeRegion(ServerName sn, HRegionInfo hri)	*Modifier and Type*: `void` *Purpose*: Closes a region *Supported*: No
closeRegion(String regionname, String serverName)	*Modifier and Type*: `void` *Purpose*: Closes a region *Supported*: No
closeRegionWithEncodedRegionName(String encodedRegionName, String serverName)	*Modifier and Type*: `boolean` *Purpose*: For expert administrators. *Supported*: No
compact(TableName tableName, byte[] columnFamily)	*Modifier and Type*: `void` *Purpose*: Compacts a column family within a table *Supported*: No
compact(TableName tableName)	*Modifier and Type*: `void` *Purpose*: Compacts a table *Supported*: No
compactRegion(byte[] regionName, byte[] columnFamily)	*Modifier and Type*: `void` *Purpose*: Compacts a column family within a region *Supported*: No
compactRegion(byte[] regionName)	*Modifier and Type*: `void` *Purpose*: Compacts an individual region *Supported*: No
compactRegionServer(ServerName sn, boolean major)	*Modifier and Type*: `void` *Purpose*: Compacts all regions on the region server *Supported*: No
coprocessorService()	*Modifier and Type*: `CoprocessorRpcChannel` *Purpose*: Creates and returns a RpcChannel instance connected to the active master *Supported*: No
coprocessorService(ServerName sn)	*Modifier and Type*: `CoprocessorRpcChannel` *Purpose*: Creates and returns a RpcChannel instance connected to the active master *Supported*: No
createNamespace(NamespaceDescriptor descriptor)	*Modifier and Type*: `void` *Purpose*: Creates a new namespace *Supported*: No
createTable(HTableDescriptor desc, byte[] startKey, byte[] endKey, int numRegions)	*Modifier and Type*: `void` *Purpose*: Creates a new table with the specified number of regions *Supported*: Yes
createTable(HTableDescriptor desc, byte[][] splitKeys)	*Modifier and Type*: `void`

last-updated: Apr 11, 2022

	Purpose: Creates a new table with an initial set of empty regions defined by the specified split keys *Supported*: Yes
`createTable(HTableDescriptor desc)`	*Modifier and Type*: `void` *Purpose*: Creates a new table *Supported*: Yes
`createTableAsync(HTableDescriptor desc, byte[][] splitKeys)`	*Modifier and Type*: `void` *Purpose*: Creates a new table but does not block and wait for it to come online  **Note:** The MapR Database treats this method as synchronous and returns null when the table is created. *Supported*: Yes
`deleteColumn(String tableName, String columnName)`	*Modifier and Type*: `void` *Purpose*: Deletes a column from a table *Supported*: Yes
`deleteColumn(TableName tableName, byte[] columnName)`	*Modifier and Type*: `void` *Purpose*: Deletes a column from a table *Supported*: Yes
`deleteNamespace(String name)`	*Modifier and Type*: `void` *Purpose*: Deletes an existing namespace. *Supported*: No
`deleteSnapshot(byte[] snapshotName)`	*Modifier and Type*: `void` *Purpose*: Deletes an existing snapshot *Supported*: No
`deleteSnapshot(String snapshotName)`	*Modifier and Type*: `void` *Purpose*: Deletes an existing snapshot *Supported*: No
`deleteSnapshots(Pattern pattern)`	*Modifier and Type*: `void` *Purpose*: Deletes existing snapshots whose names match the specified pattern *Supported*: No
`deleteSnapshots(String regex)`	*Modifier and Type*: `void` *Purpose*: Deletes existing snapshots whose names match the specified pattern *Supported*: No
`deleteTable(String tableName)`	*Modifier and Type*: `void` *Purpose*: Deletes a table *Supported*: Yes
`deleteTable(byte[] tableName)`	*Modifier and Type*: `void` *Purpose*: Deletes a table *Supported*: Yes
`deleteTable(TableName tableName)`	*Modifier and Type*: `void` *Purpose*: Deletes a table

	Supported: Yes
deleteTables(Pattern pattern)	*Modifier and Type*: `HTableDescriptor[]` *Purpose*: Delete tables that match the passed-in pattern, and waits on completion *Supported*: Yes
deleteTables(String regex)	*Modifier and Type*: `HTableDescriptor[]` *Purpose*: Delete tables that match the passed-in regular expression, and waits on completion *Supported*: Yes
disableTable (String tableName)	*Modifier and Type*: `void` *Purpose*: Disables table, and waits for completion **Note:** The MapR Database does not require disabling of tables. Therefore, although it supports these methods, it only flags the table as disabled, and does not perform any other operation. *Supported*: Yes
disableTable(TableName tableName)	*Modifier and Type*: `void` *Purpose*: Disables table, and waits for completion **Note:** The MapR Database does not require disabling of tables. Therefore, although it supports these methods, it only flags the table as disabled, and does not perform any other operation. *Supported*: Yes
disableTableAsync (String tableName)	*Modifier and Type*: `void` *Purpose*: Disables the table, but does not block and wait for it be completely disabled *Supported*: Yes
disableTableAsync(TableName tableName)	*Modifier and Type*: `void` *Purpose*: Disables the table, but does not block and wait for it be completely disabled *Supported*: Yes
disableTables(Pattern pattern)	*Modifier and Type*: `HTableDescriptor[]` *Purpose*: Disables tables that match the passed-in pattern, and waits on completion *Supported*: Yes
disableTables(String regex)	*Modifier and Type*: `HTableDescriptor[]` *Purpose*: Disable tables that match the passed-in regular expression, and waits on completion *Supported*: Yes
enableCatalogJanitor(boolean enable)	*Modifier and Type*: `boolean` *Purpose*: Enables/Disables the catalog janitor *Supported*: No
enableTable(String tableName)	*Modifier and Type*: `void` *Purpose*: Enables a table

last-updated: Apr 11, 2022

	Note: The MapR Database does not require enabling of tables. Therefore, although it supports this method, it only flags the table as enabled, and does not perform any other operation. *Supported*: Yes
enableTable(bye[] tableName)	*Modifier and Type*: `void` *Purpose*: Enables a table **Note:** The MapR Database does not require enabling of tables. Therefore, although it supports this method, it only flags the table as enabled, and does not perform any other operation. *Supported*: Yes
enableTable(TableName tableName)	*Modifier and Type*: `void` *Purpose*: Enables a table **Note:** The MapR Database does not require enabling of tables. Therefore, although it supports this method, it only flags the table as enabled, and does not perform any other operation. *Supported*: Yes
enableTableAsync (String tableName)	*Modifier and Type*: `void` *Purpose*: Enables the table, but does not block and wait for it be completely enabled *Supported*: Yes
enableTableAsync(TableName tableName)	*Modifier and Type*: `void` *Purpose*: Enables the table, but does not block and wait for it be completely enabled *Supported*: Yes
enableTables(Pattern pattern)	*Modifier and Type*: `HTableDescriptor[]` *Purpose*: Enable tables that match the passed-in pattern, and waits on completion *Supported*: Yes
enableTables(String regex)	*Modifier and Type*: `HTableDescriptor[]` *Purpose*: Enable tables that match the passed-in regular expression. and waits on completion *Supported*: Yes
execProcedure(String signature, String instance, Map<String,String> props)	*Modifier and Type*: `void` *Purpose*: Executes a distributed procedure on a cluster *Supported*: No
execProcedureWithRet(String signature, String instance, Map<String,String> props)	*Modifier and Type*: `>byte[]` *Purpose*: Executes a distributed procedure on a cluster. *Supported*: No
flush(TableName tableName)	*Modifier and Type*: `>void` *Purpose*: Flushes a table

	Supported: Yes
flush(byte[] tableNameOrRegionName)	*Modifier and Type*: `>void` *Purpose*: Flushes a table *Supported*: Yes
flush(String tableNameOrRegionName)	*Modifier and Type*: `>void` *Purpose*: Flushes a table *Supported*: Yes
flushRegion(byte[] regionName)	*Modifier and Type*: `>void` *Purpose*: Flushes an individual region *Supported*: Yes
getAlterStatus(byte[] tableName)	*Modifier and Type*: `Pair<Integer,Integer>` *Purpose*: Gets the status of the alter command - indicates the number of regions that have received the updated schema Asynchronous operation  **Note:** The MapR Database always returns (0,0). *Supported*: No
getAlterStatus(TableName tableName)	*Modifier and Type*: `Pair<Integer,Integer>` *Purpose*: Gets the status of the alter command - indicates the number of regions that have received the updated schema Asynchronous operation  **Note:** The MapR Database always returns (0,0). *Supported*: No
getClusterStatus()	*Modifier and Type*: `ClusterStatus` *Purpose*: Gets the status of the cluster *Supported*: No
getCompactionState(String tableNameOrRegionName)	*Modifier and Type*: `CompactionState` *Purpose*: Get the current compaction state of a table *Supported*: No
getCompactionState(TableName tableName)	*Modifier and Type*: `org.apache.hadoop.hbase.protobuf.generated.AdminProtos.GetRegionInfoResponse.CompactionState` *Purpose*: Gets the current compaction state of a table *Supported*: No
getCompactionStateForRegion(byte[] regionName)	*Modifier and Type*: `org.apache.hadoop.hbase.protobuf.generated.AdminProtos.GetRegionInfoResponse.CompactionState` *Purpose*: Gets the current compaction state of a region *Supported*: No
getConfiguration()	*Modifier and Type*: `org.apache.hadoop.conf.Configuration` *Purpose*: Gets the configuration used by the instance *Supported*: Yes

last-updated: Apr 11, 2022

getConnection()	*Modifier and Type*: `Connection` *Purpose*: Gets the connection used by this object *Supported*: Yes
getMasterCoprocessors()	*Modifier and Type*: `String[]` *Purpose*: Helper delegate to getClusterStatus().getMasterCoprocessors() *Supported*: No
getMasterInfoPort()	*Modifier and Type*: `int` *Purpose*: Gets the information port of the current master, if one is available *Supported*: No
getNamespaceDescriptor(String name)	*Modifier and Type*: `NamespaceDescriptor` *Purpose*: Gets a namespace descriptor by name *Supported*: No
getOnlineRegions(ServerName sn)	*Modifier and Type*: `List<HRegionInfo>` *Purpose*: Gets all the online regions on a region server *Supported*: No
getOperationTimeout()	*Modifier and Type*: `int` *Purpose*: The MapR Database never uses the timeout value **Note:** If you use the v1.1 API with the MapR Database, the method is ignored. *Supported*: No
getQuotaRetriever(QuotaFilter filter)	*Modifier and Type*: `QuotaRetriever` *Purpose*: Returns a QuotaRetriever to list the quotas based on the filter *Supported*: No
getTableDescriptor (byte[] tableName)	*Modifier and Type*: `HTableDescriptor` *Purpose*: Gets the table descriptor *Supported*: Yes
getTableDescriptor(TableName tableName)	*Modifier and Type*: `HTableDescriptor` *Purpose*: Gets the table descriptor *Supported*: Yes
getTableDescriptors(List<String> names)	*Modifier and Type*: `HTableDescriptor[]` *Purpose*: Gets table descriptors *Supported*: Yes
getTableDescriptorsByTableName(List<TableName> tableNames)	*Modifier and Type*: `HTableDescriptor[]` *Purpose*: Gets table descriptors *Supported*: Yes
getTableRegions (byte[] tableName)	*Modifier and Type*: `List<HRegionInfo>` *Purpose*: Gets the regions of a given table *Supported*: Yes
getTableRegions(TableName tableName)	*Modifier and Type*: `List<HRegionInfo>` *Purpose*: Gets the regions of a given table

last-updated: Apr 11, 2022

	Supported: Yes
isAborted()	*Modifier and Type*: `boolean` *Purpose*: Queries on the catalog janitor state. The MapR Database always returns false. *Supported*: No
isCatalogJanitorEnabled()	*Modifier and Type*: `boolean` *Purpose*: Queries on the catalog janitor state. The MapR Database always returns false. *Supported*: No
isProcedureFinished(String signature, String instance, Map<String,String> props)	*Modifier and Type*: `boolean` *Purpose*: Checks the current state of the specified procedure *Supported*: No
isSnapshotFinished(org.apache.hadoop.hbase.protobuf.generated.HBaseProtos.SnapshotDescription snapshot)	*Modifier and Type*: `boolean` *Purpose*: Checks the current state of the passed snapshot *Supported*: No
isTableAvailable(String tableName)	*Modifier and Type*: `boolean` *Purpose*: Checks if all regions of the table are available *Supported*: Yes
isTableAvailable(TableName tableName, byte[][] splitKeys)	*Modifier and Type*: `boolean` *Purpose*: Checks if the table has been created with the specified number of splitkeys that was used while creating the given table *Supported*: No
isTableAvailable(TableName tableName)>	*Modifier and Type*: `boolean` *Purpose*: Returns true if all regions of the table are available *Supported*: Yes
isTableDisabled(String tableName)	*Modifier and Type*: `boolean` *Purpose*: Checks if the table is offline  **Note:** Although the MapR Database supports this method, it only checks the flag. *Supported*: Yes
isTableDisabled(TableName tableName)	*Modifier and Type*: `boolean` *Purpose*: Checks if the table is offline  **Note:** Although the MapR Database supports this method, it only checks the flag. *Supported*: Yes
isTableEnabled(String tableName)	*Modifier and Type*: `boolean` *Purpose*: Checks if the table is online  **Note:** Although the MapR Database supports this method, it only checks the flag, and does not change the flag.

last-updated: Apr 11, 2022

	Supported: Yes
isTableEnabled(TableName tableName)	*Modifier and Type*: `boolean` *Purpose*: Checks if the table is online **Note:** Although the MapR Database supports this method, it only checks the flag, and does not change the flag. *Supported*: Yes
listNamespaceDescriptors()	*Modifier and Type*: `NamespaceDescriptor[]` *Purpose*: Lists available namespace descriptors *Supported*: No
listSnapshots()	*Modifier and Type*: `List<org.apache.hadoop.hbase.protobuf.generated.HBaseProtos.SnapshotDescription>` *Purpose*: Lists completed snapshots *Supported*: No
listSnapshots(Pattern pattern)	*Modifier and Type*: `List<org.apache.hadoop.hbase.protobuf.generated.HBaseProtos.SnapshotDescription>` *Purpose*: Lists all the completed snapshots that match the given pattern *Supported*: No
listSnapshots(String regex)	*Modifier and Type*: `List<org.apache.hadoop.hbase.protobuf.generated.HBaseProtos.SnapshotDescription>` *Purpose*: Lists all the completed snapshots that match the given regular expression *Supported*: No
listTableDescriptorsByNamespace(String name)	*Modifier and Type*: `HTableDescriptor[]` *Purpose*: Gets the list of table descriptors by namespace *Supported*: No
listTableNames()	*Modifier and Type*: `TableName[]` *Purpose*: Lists all the names of userspace tables *Supported*: Yes
listTableNames(Pattern pattern, boolean includeSysTables)	*Modifier and Type*: `TableName[]` *Purpose*: Lists all the names of userspace tables that match the specified pattern *Supported*: Yes
listTableNames(Pattern pattern)	*Modifier and Type*: `TableName[]` *Purpose*: Lists all the names of userspace tables that match the specified pattern *Supported*: Yes
listTableNames(String regex, boolean includeSysTables)	*Modifier and Type*: `TableName[]` *Purpose*: Lists all the names of userspace tables that match the specified regular expression *Supported*: Yes
listTableNames(String regex)	*Modifier and Type*: `TableName[]`

last-updated: Apr 11, 2022

	Purpose: Lists all the names of userspace tables that match the specified regular expression *Supported*: Yes
listTableNamesByNamespace(String name)	*Modifier and Type*: `TableName[]` *Purpose*: Gets the list of table names by namespace *Supported*: Yes
listTables()	*Modifier and Type*: `HTableDescriptor[]` *Purpose*: Lists all the userspace tables *Supported*: Yes
listTables(Pattern pattern, boolean includeSysTables)	*Modifier and Type*: `HTableDescriptor[]` *Purpose*: Lists all the tables that match the given pattern.  **Note:** The MapR Database does not have system tables and therefore, the boolean value is ignored. *Supported*: Yes
listTables(Pattern pattern)	*Modifier and Type*: `HTableDescriptor[]` *Purpose*: Lists all the userspace tables matching the given pattern. *Supported*: Yes
listTables(String regex, boolean includeSysTables)	*Modifier and Type*: `HTableDescriptor[]` *Purpose*: Lists all the tables matching the given pattern.  **Note:** The MapR Database does not have system tables and therefore, the boolean value is ignored. *Supported*: Yes
listTables(String regex)	*Modifier and Type*: `HTableDescriptor[]` *Purpose*: Lists all the userspace tables that match the given regular expression. *Supported*: Yes
majorCompact(byte[] tableNameOrRegionName)	*Modifier and Type*: `void` *Purpose*: Compacts a table, or an individual region. *Supported*: Deprecated
majorCompact(String tableNameOrRegionName)	*Modifier and Type*: `void` *Purpose*: Compacts a table, or an individual region. *Supported*: Deprecated
majorCompact(byte[] tableNameOrRegionName, byte[] columnFamily)	*Modifier and Type*: `void` *Purpose*: Compacts a column family within a table, or a region *Supported*: Deprecated
majorCompact(String tableNameOrRegionName, String columnFamily)	*Modifier and Type*: `void` *Purpose*: Compact a column family within a table, or a region *Supported*: Deprecated

last-updated: Apr 11, 2022

majorCompactRegion(byte[] regionName, byte[] columnFamily)	*Modifier and Type*: `void` *Purpose*: Compacts a column family within a region *Supported*: No
majorCompactRegion(byte[] regionName)	*Modifier and Type*: `void` *Purpose*: Compacts a table, or an individual region *Supported*: No
mergeRegions(byte[] nameOfRegionA, byte[] nameOfRegionB, boolean forcible)	*Modifier and Type*: `void` *Purpose*: Merges two regions *Supported*: No
modifyColumn(TableName tableName, HColumnDescriptor columnFamily)	*Modifier and Type*: `void` *Purpose*: Modifies an existing column family on a table *Supported*: Yes
modifyColumn(TableName tableName, HColumnDescriptor columnFamily)	*Modifier and Type*: `void` *Purpose*: Modifies an existing column family on a table *Supported*: Yes
modifyNamespace(NamespaceDescriptor descriptor)	*Modifier and Type*: `void` *Purpose*: Modifies an existing namespace *Supported*: No
modifyTable (byte[] tableName, HTableDescriptor htd)	*Modifier and Type*: `void` *Purpose*: Modifies an existing table *Supported*: Yes
modifyTable(final String tableName, final HTableDescriptor htd)	*Modifier and Type*: `void` *Purpose*: Modifies an existing table *Supported*: Yes
modifyTable(TableName tableName, HTableDescriptor htd)	*Modifier and Type*: `void` *Purpose*: Modifies an existing table. This method is the more IRB friendly version. *Supported*: Yes
move(byte[] encodedRegionName, byte[] destServerName)	*Modifier and Type*: `void` *Purpose*: Moves the region to the destination *Supported*: No
offline(byte[] regionName)	*Modifier and Type*: `void` *Purpose*: Offlines specified region from master's in-memory state *Supported*: No
restoreSnapshot(byte[] snapshotName, boolean takeFailSafeSnapshot)	*Modifier and Type*: `void` *Purpose*: Restores the specified snapshot on the original table *Supported*: No
restoreSnapshot(byte[] snapshotName)	*Modifier and Type*: `void` *Purpose*: Restores the specified snapshot on the original table *Supported*: No

last-updated: Apr 11, 2022

restoreSnapshot(String snapshotName, boolean takeFailSafeSnapshot)	*Modifier and Type*: `void` *Purpose*: Restores the specified snapshot on the original table *Supported*: No
restoreSnapshot(String snapshotName)	*Modifier and Type*: `void` *Purpose*: Restores the specified snapshot on the original table *Supported*: No
rollWALWriter(ServerName serverName)	*Modifier and Type*: `void` *Purpose*: Rolls the log writer. *Supported*: No
runCatalogScan()	*Modifier and Type*: `int` *Purpose*: Requests a scan of the catalog table *Supported*: No
setBalancerRunning(boolean on, boolean synchronous)	*Modifier and Type*: `boolean` *Purpose*: Turns the load balancer on or off **Note:** The MapR Database does not require balancing. Therefore, the method just sets a flag. There is no impact on the MapR table. *Supported*: No
setQuota(QuotaSettings quota)	*Modifier and Type*: `void` *Purpose*: Applies the new quota settings *Supported*: No
shutdown()	*Modifier and Type*: `void` *Purpose*: Shuts down the HBase cluster *Supported*: No
snapshot(byte[] snapshotName, TableName tableName)	*Modifier and Type*: `void` *Purpose*: Creates a timestamp consistent snapshot for the given table *Supported*: No
snapshot(org.apache.hadoop.hbase.protobuf.generated.HBaseProtos.SnapshotDescription snapshot)	*Modifier and Type*: `void` *Purpose*: Takes a snapshot, and waits for the server to complete that snapshot (blocking). *Supported*: No
snapshot(String snapshotName, TableName tableName, org.apache.hadoop.hbase.protobuf.generated.HBaseProtos.SnapshotDescription.Type type)	*Modifier and Type*: `void` *Purpose*: Creates a typed snapshot of the table *Supported*: No
snapshot(String snapshotName, TableName tableName)	*Modifier and Type*: `void` *Purpose*: Takes a snapshot for the given table *Supported*: No
split(byte[] tableNameOrRegionName)	*Modifier and Type*: `void` *Purpose*: Splits a table

	Note: The tableNameOrRegionName parameter has a different format when used with MapR tables than with Apache HBase tables. With MapR tables, specify both the table path and the FID as a comma-separated list. *Supported*: Yes
split(TableName tableName, byte[] splitPoint)	*Modifier and Type*: `void` *Purpose*: Splits a table. **Note:** The tableNameOrRegionName parameter has a different format when used with MapR tables than with Apache HBase tables. With MapR tables, specify both the table path and the FID as a comma-separated list. *Supported*: No
split(TableName tableName)	*Modifier and Type*: `void` *Purpose*: Splits a table. **Note:** The tableNameOrRegionName parameter has a different format when used with MapR tables than with Apache HBase tables. With MapR tables, specify both the table path and the FID as a comma-separated list. *Supported*: Yes
splitRegion(byte[] regionName, byte[] splitPoint)	*Modifier and Type*: `void` *Purpose*: Splits an individual region. *Supported*: No
splitRegion(byte[] regionName)	*Modifier and Type*: `void` *Purpose*: Splits an individual region *Supported*: Yes
stopMaster()	*Modifier and Type*: `void` *Purpose*: Shuts down the current HBase master only *Supported*: No
stopRegionServer(String hostnamePort)	*Modifier and Type*: `void` *Purpose*: Stops the designated region server *Supported*: No
tableExists(TableName tableName)	*Modifier and Type*: `boolean` *Purpose*: Returns whether a table with the specified name exists *Supported*: Yes
tableExists(byte[] tableName)	*Modifier and Type*: `boolean` *Purpose*: Returns whether a table with the specified name exists *Supported*: Yes
tableExists(String tableName)	*Modifier and Type*: `boolean` *Purpose*: Returns whether a table with the specified name exists *Supported*: Yes

last-updated: Apr 11, 2022

takeSnapshotAsync(org.apache.hadoop.hbase.protobuf.generated.HBaseProtos.SnapshotDescription snapshot)	*Modifier and Type*: `org.apache.hadoop.hbase.protobuf.generated.MasterProtos.SnapshotResponse` *Purpose*: Takes a snapshot without waiting for the server to complete that snapshot (asynchronous). Ensure that you take only a single snapshot at a time, or the results may be undefined. *Supported*: No
truncateTable(TableName tableName, boolean preserveSplits)	*Modifier and Type*: `void` *Purpose*: Truncates a table *Supported*: Yes
unassign(byte[] regionName, boolean force)	*Modifier and Type*: `void` *Purpose*: Unassigns a region from the current hosting region server *Supported*: No
updateConfiguration()	*Modifier and Type*: `void` *Purpose*: Updates the configuration, and triggers an online configuration change on all the region servers *Supported*: No
updateConfiguration(ServerName server)	*Modifier and Type*: `void` *Purpose*: Updates the configuration, and triggers an online configuration change on all the region servers *Supported*: No

BufferedMutator Method Support

This table indicates which methods MapR Database supports in the HBase interface `BufferedMutator`.

The following HBase methods are supported with MapR Database tables, except where noted.

Method Name	Modifier and Type	Description	Supported?
`close()`	void	Performs a flush() and releases any resources held.	Yes
`flush()`	void	Executes all the buffered, asynchronous Mutation operations and waits until they are done.	Yes
`getConfiguration()`	org.apache.hadoop.conf.Configuration	Returns the Configuration object used by this instance.	Yes
`getName()`	TableName	Gets the fully qualified table name instance of the table that this BufferedMutator writes to.	Yes
`getWriteBuffer()`	long	Get the internal write buffer.	No
`getWriteBufferSize()`	long	Returns the maximum size in bytes of the write buffer for this HTable.	No
`mutate(List<? extends Mutation> mutations)`	void	Send some Mutation objects to the table.	Yes

last-updated: Apr 11, 2022

Method Name	Modifier and Type	Description	Supported?
`mutate(Mutation mutation)`	void	Sends a Mutation to the table.	Yes
`setWriteBufferSize()`	long	Sets the maximum size (in bytes) of the write buffer for this HTable	No

Connection Method Support
This table indicates which methods MapR Database supports in the HBase interface `Connection`.

The following HBase methods are supported with MapR Database tables, except where noted. For full details about this interface, see `Interface Connection`.

Method Name	Modifier and Type	Description	Supported?
`close()`	void	Close this connection.	Yes
`getAdmin()`	Admin	Retrieve an Admin implementation to administer an HBase cluster.	Yes
`getBufferedMutator(BufferedMutatorParams params)`	BufferedMutator	Retrieve a BufferedMutator for performing client-side buffering of writes.	Yes
`getBufferedMutator(TableName tableName)`	BufferedMutator		Yes
`getConfiguration()`	org.apache.hadoop.conf.Configuration	Retrieve the Configuration object used by this connection.	Yes
`getRegionLocator(TableName tableName)`	RegionLocator	Retrieve a RegionLocator implementation to inspect region information on a table.	Yes
`getTable(TableName tableName)`	Table	Retrieve a Table implementation for accessing a table.	Yes
`getTable(TableName tableName, ExecutorService pool)`	Table		Yes
`isClosed()`	boolean	Returns whether the connection is closed or not.	Yes

ConnectionFactory Method Support
This table indicates which methods MapR Database supports in the HBase class `ConnectionFactory`.

The following HBase methods are supported with MapR Database tables, except where noted. For full details about this class, see the `ConnectionFactory` class in the Client package..

Method Name	Modifier and Type	Description	Supported?
`createConnection()`	static Connection	Create a new Connection instance using default HBaseConfiguration.	Yes
`createConnection(org.apache.hadoop.conf.Configuration conf)`	static Connection	Create a new Connection instance using the passed conf instance.	Yes

Method Name	Modifier and Type	Description	Supported?
`createConnection(org.apache.hadoop.conf.Configuration conf, ExecutorService pool)`	static Connection	Create a new Connection instance using the passed conf instance.	Yes
`createConnection(org.apache.hadoop.conf.Configuration conf, ExecutorService pool, User user)`	static Connection	Create a new Connection instance using the passed conf instance.	Yes
`createConnection(org.apache.hadoop.conf.Configuration conf, User user)`	static Connection	Create a new Connection instance using the passed conf instance.	Yes

RegionLocator Method Support
This table indicates which methods MapR Database supports in the HBase interface `RegionLocator`.

The following HBase methods are supported with MapR Database tables, except where noted. For full details about this interface, see `Interface RegionLocator` interface in the Client package.

Method Name	Modifier and Type	Description	Supported?
`close()`	void	Close this connection.	Yes
`getAllRegionLocations()`	List<HRegionLocation>	Retrieves all of the regions associated with this table.	Yes
`getConfiguration()`	Configuration	Retrieve the Configuration Object used by this RegionLocator.	Yes
`getEndKeys()`	byte[][]	Gets the ending row key for every region in the currently open table.	Yes
`getName()`	TableName	Gets the fully qualified table name instance of this table.	Yes
`getRegionLocation(byte[] row)`	HRegionLocation	Finds the region on which the given row is being served.	Yes
`getRegionLocation(byte[] row, boolean reload)`	HRegionLocation		Yes
`getStartEndKeys()`	Pair<byte[][],byte[][]>	Gets the starting and ending row keys for every region in the currently open table.	Yes
`getStartKeys()`	byte[][]	Gets the starting row key for every region in the currently open table.	Yes

Table Method Support
This table indicates which methods MapR Database supports in the HBase interface `Table`.

The following HBase methods are supported with MapR Database tables, except where noted. For full details about this interface, see `Interface Table` in the Client package.

last-updated: Apr 11, 2022

Method Name	Modifier and Type	Description	Supported?
`append(Append append)`	Result	Appends values to one or more columns within a single row.	Yes
`batch(List<? extends Row> actions)`	Object[]	A batch call on Deletes, Gets, Puts, Increments, Appends and RowMutations.	Deprecated
`batch(List<? extends Row> actions, Object[] results)`	void	Method that does a batch call on Deletes, Gets, Puts, Increments and Appends.	Yes
`batchCallback(List<? extends Row> actions, org.apache.hadoop.hbase.client.coprocessor.Batch.Callback<R> callback)`	Object[]	Method that does a batch call on Deletes, Gets, Puts, Increments and Appends with a callback.	YesFootnote.
`batchCallback(List<? extends Row> actions, Object[] results, org.apache.hadoop.hbase.client.coprocessor.Batch.Callback<R> callback)`	<R> void	Same as batch(List, Object[]), but with a callback. MapR Database ignores the callback.	YesFootnote.
`batchCoprocessorService(com.google.protobuf.Descriptors.MethodDescriptor methodDescriptor, com.google.protobuf.Message request, byte[] startKey, byte[] endKey, R responsePrototype, org.apache.hadoop.hbase.client.coprocessor.Batch.Callback<R> callback)`	<R extends com.google.protobuf.Message> void	Creates an instance of the given Service subclass for each table region spanning the range from the startKey row to endKey row (inclusive), all the invocations to the same region server will be batched into one call.	No
`batchCoprocessorService(com.google.protobuf.Descriptors.MethodDescriptor methodDescriptor, com.google.protobuf.Message request, byte[] startKey, byte[] endKey, R responsePrototype)`	<R extends com.google.protobuf.Message> Map<byte[],R>		No

[1] Not fully supported. When used with MapR Database, the callback is ignored.
[2] Not fully supported. When used with MapR Database, the durability is ignored.

Method Name	Modifier and Type	Description	Supported?
`checkAndDelete(byte[] row, byte[] family, byte[] qualifier, byte[] value, Delete delete)`	boolean	Atomically checks if a row/family/qualifier value matches the expected value.	Yes
`checkAndDelete(byte[] row, byte[] family, byte[] qualifier, CompareFilter.CompareOp compareOp, byte[] value, Delete delete)`	boolean		Yes
`checkAndMutate(byte[] row, byte[] family, byte[] qualifier, CompareFilter.CompareOp compareOp, byte[] value, RowMutations mutation)`	boolean		Yes
`checkAndPut(byte[] row, byte[] family, byte[] qualifier, byte[] value, Put put)`	boolean		Yes
`checkAndPut(byte[] row, byte[] family, byte[] qualifier, CompareFilter.CompareOp compareOp, byte[] value, Put put)`	boolean		Yes
`close()`	void	Releases any resources held or pending changes in internal buffers.	Yes
`coprocessorService(byte[] row)`	CoprocessorRpcChannel	Creates and returns a RpcChannel instance connected to the table region containing the specified row.	No

last-updated: Apr 11, 2022

Method Name	Modifier and Type	Description	Supported?
`coprocessorService(Class<T> service, byte[] startKey, byte[] endKey, org.apache.hadoop.hbase.client.coprocessor.Batch.Call<T,R> callable, org.apache.hadoop.hbase.client.coprocessor.Batch.Callback<R> callback)`	<T extends com.google.protobuf.Service,R> void	Creates an instance of the given Service subclass for each table region spanning the range from the startKey row to endKey row (inclusive), and invokes the passed Batch.Call.call(T) method with each Service instance.	No
`coprocessorService(Class<T> service, byte[] startKey, byte[] endKey, org.apache.hadoop.hbase.client.coprocessor.Batch.Call<T,R> callable)`	<T extends com.google.protobuf.Service,R> Map<byte[],R>		No
`delete(Delete delete)`	void	Deletes the specified cells/row.	Yes
`delete(List<Delete> deletes)`	void	Deletes the specified cells/rows in bulk.	Yes
`exists(Get get)`	boolean	Test for the existence of columns in the table, as specified by the Get.	Yes
`existsAll(List<Get> gets)`	boolean[]	Test for the existence of columns in the table, as specified by the Gets, in batch.	Yes
`get(Get get)`	Result	Extracts certain cells from a given row.	Yes
`get(List<Get> gets)`	Result[]	Extracts certain cells from the given rows, in batch.	Yes
`getConfiguration()`	org.apache.hadoop.conf.Configuration	Returns the Configuration object used by this instance.	Yes
`getName()`	TableName	Gets the fully qualified table name instance of this table.	Yes
`getScanner(byte[] family, byte[] qualifier)`	ResultScanner	Gets a scanner on the current table for the given family and qualifier.	Yes
`getScanner(byte[] family)`	ResultScanner	Gets a scanner on the current table for the given family.	Yes
`getScanner(Scan scan)`	ResultScanner	Returns a scanner on the current table as specified by the Scan object.	Yes
`getTableDescriptor()`	HTableDescriptor	Gets the table descriptor for this table.	Yes
`getWriteBufferSize()`	long	Returns the maximum size in bytes of the write buffer for this HTable.	No

last-updated: Apr 11, 2022

Method Name	Modifier and Type	Description	Supported?
`increment(Increment increment)`	Result	Increments one or more columns within a single row.	Yes
`incrementColumnValue(byte[] row, byte[] family, byte[] qualifier, long amount, Durability durability)`	long	Atomically increments a column value.	YesFootnote.
`incrementColumnValue(byte[] row, byte[] family, byte[] qualifier, long amount)`	long	See incrementColumnValue(byte[], byte[], byte[], long, Durability).	Yes
`mutateRow(RowMutations rm)`	void	Performs multiple mutations atomically on a single row.	Yes
`put(List<Put> puts)`	void	Puts some data in the table, in batch.	Yes
`put(Put put)`	void	Puts some data in the table.	Yes
`setWriteBufferSize(long writeBufferSize)`	void	Sets the size of the buffer in bytes.	No

HColumnDescriptor and HTableDescriptor Support
This section describes the supported fields in the HColumnDescriptor and the HTableDescriptor classes.

MapR Database supports all of the methods that are in these classes. However, it supports only a subset of their fields.

HColumnDescriptor Class

Field	Description
BLOCKSIZE	Size of blocks in files stored to the filesytem (hfiles).
BLOOMFILTER	Whether or not to use bloomfilters.
COMPRESSION	Compression type.
IN_MEMORY	Whether to serve from memory or not.
MIN_VERSIONS	Minimum number of versions to keep.
NAME	Name of the column family.
TTL	Time to live of cell contents.
VERSIONS	Number of versions to keep.

HTableDescriptor Class

Field	Description
AUTOSPLIT	Specifies whether to split the table into regions automatically as the table grows. The average size of each region is determined by the`regionsizemb` parameter. The default value is `true`.

Field	Description
BULKLOAD	Boolean. Specifies whether to perform a full bulk load of the table. The default is `false`. For more information, see Bulk Loading and MapR Tables.

last-updated: Apr 11, 2022

Field	Description
DELETE_TTL	Used for multi-master replication. Normally, delete operations are purged after the affected table cells are updated. Whereas the result of an update is saved in a table until another change overwrites or deletes it, the result of a delete is not saved. In multi-master replication, this difference can lead to tables being unsynchronized. Example Suppose that you have set up multi-master replication between table `customers` in the cluster `sanfrancisco` and table `customers` in the cluster `newyork`. Client applications then make these two changes: **1.** On `/mapr/sanfrancisco/customers`, put row A at 10:00:00 AM. **2.** On `/mapr/newyork/customers`, delete row A at 10:00:01 AM. On `/mapr/sanfrancisco/customers`, the order of operations is: **1.** Put row A with a timestamp of 10:00:00 AM **2.** Delete row A with a timestamp of 10:00:01 AM (This operation is repllicated from `/mapr/newyork/customers`.) On `/mapr/newyork/customers`, the order of operations is: **1.** Delete row A with a timestamp of 10:00:01 AM **2.** Put row A with a timestamp of 10:00:00 AM (This operation is replicated from `/mapr/sanfrancisco/customers`.) Now, though the put happened on `/mapr/sanfrancisco/customers` at 10:00:00 AM, the put reaches `/mapr/newyork/customers` several seconds after that. Suppose that the actual time that the put arrives at `/mapr/newyork/customers` is 10:00:03 AM. To ensure that both tables stay synchronized, `/mapr/newyork/customers` should preserve the delete until after the put is replicated. Then, the delete can be applied after the put. Therefore, the time-to-live for the delete should be at least long enough for the put to arrive at `/mapr/newyork/customers`. In this case, the time-to-live should be at least 3 seconds. In general, the time-to-live for deletes should be greater than the amount of time that it takes replicated operations to reach replicas. By default, the value is 24 hours. For example, suppose (to extend the scenario above) that you pause replication during weekdays and resume it on weekends. The put takes place on Monday morning `/mapr/sanfrancisco/customers` at 10:00:00 AM and the delete takes place at `/mapr/newyork/customers` at 10:00:01 AM. Replication does not resume until 12:00:00 AM Saturday morning. Given the volume of operations to be replicated and the potential for network problems, it is possible that these operations will not be replicated until Sunday. In this scenario, a value of 7 days for DELETE_TTL (7 multiplied by 24 hours) should provide sufficient margin.

Field	Description
NAME	Name of the table.

Support for HBase Java Filters Support

MapR Database supports the following Java filters, which work identically to their Apache HBase versions. See the Apache HBase API for more information, specifically, the Apache HBase Filter package.

Filter	Description
`ColumnCountGetFilter`	Simple filter that returns first N columns on row only. This filter was written to test filters in Get and as soon as it gets its quota of columns, `filterAllRemaining()` returns true. This makes this filter unsuitable as a Scan filter.
`ColumnPaginationFilter`	A filter, based on the `ColumnCountGetFilter`, takes two arguments: limit and offset. This filter can be used for row-based indexing, where references to other tables are stored across many columns, in order to efficient lookups and paginated results for end users. Only most recent versions are considered for pagination.
`ColumnPrefixFilter`	This filter is used for selecting only those keys with columns that matches a particular prefix. For example, if prefix is 'an', it will pass keys with columns like 'and', 'anti' but not keys with columns like 'ball', 'act'.
`ColumnRangeFilter`	This filter is used for selecting only those keys with columns that are between minColumn to maxColumn. For example, if minColumn is 'an', and maxColumn is 'be', it will pass keys with columns like 'ana', 'bad', but not keys with columns like 'bed', 'eye' If minColumn is null, there is no lower bound. If maxColumn is null, there is no upper bound. minColumnInclusive and maxColumnInclusive specify if the ranges are inclusive or not.
`DependentColumnFilter`	A filter for adding inter-column timestamp matching Only cells with a correspondingly timestamped entry in the target column will be retained Not compatible with Scan.setBatch as operations need full rows for correct filtering
`FamilyFilter`	This filter is used to filter based on the column family. It takes an operator (equal, greater, not equal, etc) and a byte [] comparator for the column family portion of a key. This filter can be wrapped with `WhileMatchFilter` and `SkipFilter` to add more control. Multiple filters can be combined using `FilterList` . If an already known column family is looked for, use `Get.addFamily(byte[])` directly rather than a filter.

last-updated: Apr 11, 2022

Filter	Description
`FilterList`	Implementation of `Filter` that represents an ordered List of Filters which will be evaluated with a specified boolean operator `FilterList.Operator.MUST_PASS_ALL` (`AND`) or `FilterList.Operator.MUST_PASS_ONE` (`OR`). Since you can use Filter Lists as children of Filter Lists, you can create a hierarchy of filters to be evaluated. `FilterList.Operator.MUST_PASS_ALL` evaluates lazily: evaluation stops as soon as one filter does not include the KeyValue. `FilterList.Operator.MUST_PASS_ONE` evaluates non-lazily: all filters are always evaluated. Defaults to `FilterList.Operator.MUST_PASS_ALL` .
`FirstKeyOnlyFilter`	A filter that will only return the first KV from each row. This filter can be used to more efficiently perform row count operations.
`FirstKeyValueMatchingQualifiersFilter`	The filter looks for the given columns in KeyValue. Once there is a match for any one of the columns, it returns ReturnCode.NEXT_ROW for remaining KeyValues in the row. **Note**: It may emit KVs which do not have the given columns in them, if these KVs happen to occur before a KV which does have a match. Given this caveat, this filter is only useful for special cases like `RowCounter` .
`FuzzyRowFilter`	Filters data based on fuzzy row key. Performs fast-forwards during scanning. It takes pairs (row key, fuzzy info) to match row keys. Where fuzzy info is a byte array with 0 or 1 as its values: • 0 - means that this byte in provided row key is fixed, i.e. row key's byte at same position must match • 1 - means that this byte in provided row key is NOT fixed, i.e. row key's byte at this position can be different from the one in provided row key Example: Let's assume row key format is userId_actionId_year_month. Length of userId is fixed and is 4, length of actionId is 2 and year and month are 4 and 2 bytes long respectively. Let's assume that we need to fetch all users that performed certain action (encoded as "99") in Jan of any year. Then the pair (row key, fuzzy info) would be the following: row key = "????_99_????_01" (one can use any value instead of "?") fuzzy info = "\x01\x01\x01\x01\x00\x00\x00\x00\x01\x01\x01\x01\x00\x00\x00" I.e. fuzzy info tells the matching mask is "????_99_????_01", where at ? can be any value.
`InclusiveStopFilter`	A Filter that stops after the given row. There is no "RowStopFilter" because the Scan spec allows you to specify a stop row. Use this filter to include the stop row, eg: [A,Z].
`KeyOnlyFilter`	A filter that will only return the key component of each KV (the value will be rewritten as empty). This filter can be used to grab all of the keys without having to also grab the values.

last-updated: Apr 11, 2022

Filter	Description
`MultipleColumnPrefixFilter`	This filter is used for selecting only those keys with columns that matches a particular prefix. For example, if prefix is 'an', it will pass keys will columns like 'and', 'anti' but not keys with columns like 'ball', 'act'.
`PageFilter`	Implementation of Filter interface that limits results to a specific page size. It terminates scanning once the number of filter-passed rows is > the given page size. Note that this filter cannot guarantee that the number of results returned to a client are <= page size. This is because the filter is applied separately on different region servers. It does however optimize the scan of individual HRegions by making sure that the page size is never exceeded locally.
`PrefixFilter`	Pass results that have same row prefix.
`QualifierFilter`	This filter is used to filter based on the column qualifier. It takes an operator (equal, greater, not equal, etc) and a byte [] comparator for the column qualifier portion of a key. This filter can be wrapped with `WhileMatchFilter` and `SkipFilter` to add more control. Multiple filters can be combined using `FilterList` . If an already known column qualifier is looked for, use `Get.addColumn(byte[], byte[])` directly rather than a filter.
`RandomRowFilter`	A filter that includes rows based on a chance.
`RowFilter`	This filter is used to filter based on the key. It takes an operator (equal, greater, not equal, etc) and a byte [] comparator for the row, and column qualifier portions of a key. This filter can be wrapped with `WhileMatchFilter` to add more control. Multiple filters can be combined using `FilterList`. If an already known row range needs to be scanned, use `CellScanner` start and stop rows directly rather than a filter.
`SingleColumnValueExcludeFilter`	A `Filter` that checks a single column value, but does not emit the tested column. This will enable a performance boost over `SingleColumnValueFilter` , if the tested column value is not actually needed as input (besides for the filtering itself).

Filter	Description
`SingleColumnValueFilter`	This filter is used to filter cells based on value. It takes a `CompareFilter.CompareOp` operator (equal, greater, not equal, etc), and either a byte [] value or a ByteArrayComparable. If we have a byte [] value then we just do a lexicographic compare. For example, if passed value is 'b' and cell has 'a' and the compare operator is LESS, then we will filter out this cell (return true). If this is not sufficient (eg you want to deserialize a long and then compare it to a fixed long value), then you can pass in your own comparator instead. You must also specify a family and qualifier. Only the value of this column will be tested. When using this filter on a `CellScanner` with specified inputs, the column to be tested should also be added as input (otherwise the filter will regard the column as missing). To prevent the entire row from being emitted if the column is not found on a row, use `setFilterIfMissing(boolean)` . Otherwise, if the column is found, the entire row will be emitted only if the value passes. If the value fails, the row will be filtered out. In order to test values of previous versions (timestamps), set `setLatestVersionOnly(boolean)` to false. The default is true, meaning that only the latest version's value is tested and all previous versions are ignored. To filter based on the value of all scanned columns, use `ValueFilter`.
`SkipFilter`	A wrapper filter that filters an entire row if any of the Cell checks do not pass. For example, if all columns in a row represent weights of different things, with the values being the actual weights, and we want to filter out the entire row if any of its weights are zero. In this case, we want to prevent rows from being emitted if a single key is filtered. Combine this filter with a `ValueFilter`: `scan.setFilter(new SkipFilter(new ValueFilter(CompareOp.NOT_EQUAL, new BinaryComparator(Bytes.toBytes(0))));` Any row which contained a column whose value was 0 will be filtered out (since ValueFilter will not pass that Cell). Without this filter, the other non-zero valued columns in the row would still be emitted.
`TimestampsFilter`	Filter that returns only cells whose timestamp (version) is in the specified list of timestamps (versions). Note: Use of this filter overrides any time range/time stamp options specified using `Get.setTimeRange(long, long)`, `Scan.setTimeRange(long, long)`, or `Scan.setTimeStamp(long)`. See the Apache HBase API, Client package for detailed information.

last-updated: Apr 11, 2022

Filter	Description
`ValueFilter`	This filter is used to filter based on column value. It takes an operator (equal, greater, not equal, etc) and a byte [] comparator for the cell value. This filter can be wrapped with `WhileMatchFilter` and `SkipFilter` to add more control. Multiple filters can be combined using `FilterList`. To test the value of a single qualifier when scanning multiple qualifiers, use `SingleColumnValueFilter`.
`WhileMatchFilter`	A wrapper filter that returns true from `filterAllRemaining()` as soon as the wrapped filters `Filter.filterRowKey(byte[], int, int)`, `Filter.filterKeyValue(org.apache.hadoop.hbase.Cell)`, `Filter.filterRow()` or `Filter.filterAllRemaining()` methods returns true.

HBase Java Regular Expressions Support

This topics defines the subset of Java regular expressions that are supported for MapR Database tables.

Filters used with Scan operations support regular expressions. When you filter scans on MapR Database tables, you can use regular expressions that comprise the Perl-Compatible Regular Expressions library, as well as a subset of the regular expressions that are supported in `java.util.regex.pattern`.

Characters

Pattern	Description
x	The character x
\\	The backslash character
\0n	The character with octal value 0n (0 <= n <= 7)
\0nn	The character with octal value 0nn (0 <= n <= 7)
\xhh	The character with hexadecimal value 0xhh
\t	The tab character ('\u0009')
\n	The newline (line feed) character ('\u000A')
\r	The carriage-return character ('\u000D')
\f	The form-feed character ('\u000C')
\a	The alert (bell) character ('\u0007')
\e	The escape character ('\u001B')
\cx	The control character corresponding to x

Character Classes

Pattern	Description
[abc]	a, b, or c (simple class)
[Supported Regular Expressions in MapR Tables^abc]	Any character except a, b, or c (negation)
[a-zA-Z]	a through z or A through Z, inclusive (range)

Predefined Character Classes

Pattern	Description
.	Any character (may or may not match line terminators)
\d	A digit: [0-9]
\D	A non-digit: [Supported Regular Expressions in MapR Tables^0-9]
\s	A whitespace character: [\t\n\x0B\f\r]
\S	A non-whitespace character: [Supported Regular Expressions in MapR Tables^\s]
\w	A word character: [a-zA-Z_0-9]
\W	A non-word character: [Supported Regular Expressions in MapR Tables^\w]

Classes for Unicode Blocks and Categories

Pattern	Description
\p{Lu}	An uppercase letter (simple category)
\p{Sc}	A currency symbol

Boundaries

Pattern	Description
^	The beginning of a line
$	The end of a line
\b	A word boundary
\B	A non-word boundary
\A	The beginning of the input
\G	The end of the previous match
\Z	The end of the input but for the final terminator, if any

last-updated: Apr 11, 2022

Pattern	Description
\z	The end of the input

Greedy Quantifiers

Pattern	Description
X?	X, once or not at all
X*	X, zero or more times
X+	X, one or more times
X{n}	X, exactly n times
X{n,}	X, at least n times
X{n,m}	X, at least n but not more than m times

Reluctant Quantifiers

Pattern	Description
X??	X, once or not at all
X*?	X, zero or more times
X+?	X, one or more times
X{n}?	X, exactly n times
X{n,}?	X, at least n times
X{n,m}?	X, at least n but not more than m times

Possessive Quantifiers

Pattern	Description
X?+	X, once or not at all
X*+	X, zero or more times
X++	X, one or more times
X{n}+	X, exactly n times
X{n,}+	X, at least n times
X{n,m}+	X, at least n but not more than m times

Logical Operators

Pattern	Description
XY	X followed by Y
X|Y	Either X or Y
(X)	X, as a capturing group

Back References

Pattern	Description
\n	Whatever the nth capturing group matches

Quotation

Pattern	Description
\	Nothing, but quotes the following character
\Q	Nothing, but quotes all characters until \E
\E	Nothing, but ends quoting started by \Q

Special Constructs

Pattern	Description
(?:X)	X, as a non-capturing group
(?=X)	X, via zero-width positive lookahead
(?!X)	X, via zero-width negative lookahead
(?<=X)	X, via zero-width positive lookbehind
(?<!X)	X, via zero-width negative lookbehind
(?>X)	X, as an independent, non-capturing group

HBase Java Comparators Support

MapR Database supports the following Java filters, which work identically to their Apache HBase versions. See the Apache HBase API for more information, specifically, the Apache HBase Filter package.

Comparator	Description
`>BinaryComparator`	A binary comparator which lexicographically compares against the specified byte array using `Bytes.compareTo(byte[], byte[])`.
`BinaryPrefixComparator`	A comparator which compares against a specified byte array, but only compares up to the length of this byte array. For the rest it is similar to BinaryComparator.

last-updated: Apr 11, 2022

Comparator	Description
`BitComparator`	A bit comparator which performs the specified bitwise operation on each of the bytes with the specified byte array. Then returns whether the result is non-zero.
`NullComparator`	A binary comparator which lexicographically compares against the specified byte array using `Bytes.compareTo(byte[], byte[])`.
`RegexStringComparator`	This comparator is for use with `CompareFilter` implementations, such as `RowFilter` , `QualifierFilter` , and `ValueFilter` , for filtering based on the value of a given column. Use it to test if a given regular expression matches a cell value in the column. Only EQUAL or NOT_EQUAL comparisons are valid with this comparator. For example: `ValueFilter vf = new ValueFilter(CompareOp.EQUAL, new RegexStringComparator(// v4 IP address "(((25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\\.){3,3}" + "(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?))(\\/[0-9]+)?" + "|" + // v6 IP address "((([\\dA-Fa-f]{1,4}:){7}[\\dA-Fa-f]{1,4})(:([\\d]{1,3}.)" + "{3}[\\d]{1,3})?)(\\/[0-9]+)?"));`
`SubstringComparator`	This comparator is for use with SingleColumnValueFilter, for filtering based on the value of a given column. Use it to test if a given substring appears in a cell value in the column. The comparison is case insensitive. Only EQUAL or NOT_EQUAL tests are valid with this comparator. For example: `SingleColumnValueFilter scvf = new SingleColumnValueFilter("col", CompareOp.EQUAL, new SubstringComparator("substr"));`

Unsupported HBase Java Methods

This topic identifies the HBase Java methods that are not supported for MapR Database tables. Attempts to call any of these methods results in an `UnsupportedOperationException` exception.

Methods for ACLs for cells:

- `Put.setACL(String user, org.apache.hadoop.hbase.security.access.Permission perms)`
- `Append.setACL(Map<String,org.apache.hadoop.hbase.security.access.Permission> perms)`

- `Delete.setACL(Map<String,org.apache.hadoop.hbase.security.access.Permission> perms)`
- `Increment.setACL(Map<String,org.apache.hadoop.hbase.security.access.Permission> perms)`

Methods for cell visibility:

- `Put.setCellVisibility(org.apache.hadoop.hbase.security.visibility.CellVisibility expression)`
- `Append.setCellVisibility(org.apache.hadoop.hbase.security.visibility.CellVisibility expression)`
- `Delete.setCellVisibility(org.apache.hadoop.hbase.security.visibility.CellVisibility expression)`
- `Increment.setCellVisibility(org.apache.hadoop.hbase.security.visibility.CellVisibility expression)`

Methods for time-to-live for cell values:

- `Put.setTTL(long ttl)`
- `Append.setTTL(long ttl)`
- `Delete.setTTL(long ttl)`
- `Increment.setTTL(long ttl)`

Other methods

- `Delete.deleteFamilyVersion(byte[] family, long timestamp)`
- `Scan.setReversed(boolean reversed)`
- `Scan.setBatch()`
- `Scan.setCaching()`
- `Scanner.next(int nbRows)`

Impersonation via HBase REST Gateway

Impersonation enables access to tables via user IDs other than the user that runs the Gateway.

You can enable user impersonation to access MapR Database tables via the HBase REST Gateway. This feature is not supported in earlier combinations of MapR and HBase packages.

Impersonation is only supported if the REST Gateway is running as the mapr user (MAPR_USER).

You can enable impersonation via the Gateway on both non-secure and secure MapR clusters (secured using either MapR SASL or Kerberos). MapR Database does not support gateway impersonation using a Thrift interface.

Note: Impersonation for HBase REST Gateway is enabled by default on secure clusters.

last-updated: Apr 11, 2022

Enabling Impersonation on a Non-Secure Cluster

To enable impersonation on a non-secure cluster, follow these steps:

1. Install the `mapr-hbase` package on a cluster that is running version 4.0.2 or later. This package contains all of the HBase binaries. For installation details, see HBase Client and MapR Database.
2. Enable simple authentication via the REST Gateway by appending the following property to the `hbase-site.xml` file (`/opt/mapr/hbase/hbase<hbase_version>/conf/hbase-site.xml`): The simple authentication protocol is a Hadoop pseudo authenticator that serves as an example and is part of the hadoop-common package.

   ```
   <property>
   <name>hbase.rest.authentication.type</name>
   <value>simple</value>
   </property>
   ```

3. Set the following environment variable to enable impersonation:

   ```
   export MAPR_IMPERSONATION_ENABLED=1
   ```

4. Start the REST Gateway server as the MAPR_USER.

   ```
   /opt/mapr/hbase/hbase<hbase_version>/bin/hbase-daemon.sh start rest -p
   port
   ```

Using Custom Authentication

This section describes how to use Hadoop pseudo authentication for custom authentication.

Editing the `hbase-site.xml` file is the procedure to follow if you want to use Hadoop pseudo authentication. Alternatively, you can write and use your own authenticator to substitute for the "simple" configuration by implementing the Hadoop AuthenticationHandler interface. If you are using custom authentication, place your authenticator jar in the following directory:

```
/opt/mapr/hbase/hbase-<hbase_version>/lib/
```

Then start the REST Gateway.

Mapping to HBase Table Namespaces

This section describes mapping table namespaces between Apache HBase tables and MapR Database binary tables.

The MapR implementations of the HBase Java API and libhbase differentiate between Apache HBase tables and MapR Database tables according to table names. In certain cases, such as migrating code from Apache HBase tables to MapR Database tables, users need to force the API they are using to access a MapR Database table, even though the table name could map to an Apache HBase table. The `hbase.table.namespace.mappings` property allows you to map Apache HBase table names to MapR Database tables. This property is typically set in the configuration file `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/core-site.xml`.

In general, if a table name includes a slash (`/`), the name is assumed to be a path to a MapR Database table, because slash is not a valid character for Apache HBase table names. In the case of "flat" table names without a slash, namespace conflict is possible, and you might need to use table mappings.

Table Mapping Naming Conventions

A table mapping takes the form *name*`:`*map* , where *name* is the table name to redirect and *map* is the modification made to the name. The value in *name* can be a literal string or contain the * wildcard. When mapping a name with a wild card, the mapping is treated as a directory. Requests to tables with names that match the wild card are sent to the directory in the mapping.

When mapping a name that is a literal string, you can choose from two different behaviors:

- End the mapping with a slash to indicate that this mapping is to a directory. For example, the mapping `mytable1:/user/aaa/` sends requests for table `mytable1` to the full path `/user/aaa/mytable1`.
- End the mapping without a slash, which creates an alias and treats the mapping as a full path. For example, the mapping `mytable1:/user/aaa` sends requests for table `mytable1` to the full path `/user/aaa`.

Mappings and Table Listing Behaviors

When you use the `list` command without specifying a directory, the command's behavior depends on two factors:

- Whether a table mapping exists
- Whether Apache HBase is installed and running

Here are three different scenarios and the resulting `list` command behavior for each.

- There is a table mapping for *, as in `*:/tables`. In this case, the `list` command lists the tables in the mapped directory.
- There is no mapping for *, and Apache HBase is installed and running. In this case, the `list` command lists the HBase tables.
- There is no mapping for *, and Apache HBase is not installed or is not running.
  - For HBase 0.98.12, the shell will try to connect to an HBase cluster but it will return an error instead.
  - For HBase 1.1 or above, if the `mapr.hbase.default.db` property in the hbase-site.xml is set to `hbase`, the list command will return an error stating that HBase is not available. If the `mapr.hbase.default.db` property is set to `maprdb`, list command will list the MapR Database tables under the user's home directory.

Example 1: Map all HBase tables to MapR Database tables in a directory

In this example, any flat table name `foo` is treated as a MapR Database table in the directory `/tables_dir/foo`.

```
<property>
        <name>hbase.table.namespace.mappings</name>
        <value>*:/tables_dir</value>
        </property>
```

Example 2: Map specific Apache HBase tables to specific MapR Database tables

In this example, the Apache HBase table name `mytable1` is treated as a MapR Database table at `/user/aaa/mytable1`. The Apache Hbase table name `mytable2` is treated as a MapR Database table

last-updated: Apr 11, 2022

at `/user/bbb/mytable2`. All other Apache HBase table names are treated as stock Apache HBase tables.

```
<property>
        <name>hbase.table.namespace.mappings</name>
        <value>mytable1:/user/aaa/,mytable2:/user/bbb/</value>
        </property>
```

Example 3: Combination of specific table names and wildcards

Mappings are evaluated in order. In this example, the flat table name `mytable1` is treated as a MapR Database table at `/user/aaa/mytable1`. The flat table name `mytable2` is treated as a MapR Database table at `/user/bbb/mytable2`. Any other flat table name `foo` is treated as a MapR Database table at `/tables_dir/foo`.

```
<property>
        <name>hbase.table.namespace.mappings</name>
        <value>mytable1:/user/aaa/,mytable2:/user/bbb/,*:/tables_dir</value>
        </property>
```

Thread-pool Settings for Performance

The MapR Database C APIs internally have one thread pool per client. Threads work on the async tasks enqueued by a client application. There are two thread-pool parameters that you can modify in the `/opt/mapr/conf/dbclient.conf` file for better application performance.

fs.mapr.pool.threads

This parameter controls the number of connections that a client application makes with MapR Database for append, increment, read, and scan requests. For a higher rate of throughput to MapR Database, you can increase this value. The default value is 10.

fs.mapr.highpri.pool.threads

This parameter controls the number of threads that invoke application-provided callbacks. If an application's callbacks are not lightweight but instead perform complex calculations that require significant processing, increase this value to avoid delays in invoking callbacks. The default value is 2.

Building MapReduce Applications

This section provides information about building and running custom MapReduce application that access MapR Database binary tables.

The steps for building and running custom MapReduce applications that run against MapR Database are the same as the steps for building and running custom MapReduce applications that run against Apache HBase. The steps are documented in the Apache HBase Reference Guide at http://hbase.apache.org/book.html#mapreduce.

However, you must use an HBase JAR file from MapR's Maven repository at https://repository.mapr.com/nexus/content/groups/mapr-public/org/apache/hbase/hbase-server/. The name of the JAR file that you use must contain the version of HBase and the version of MapR that you are using.

For example, if you are using HBase 1.1 and MapR version 5.1, you would use the file `hbase-server-1.1.1-mapr-1602.jar`.

Performing Bulkloads with MapReduce

This section describes custom MapReduce applications used to perform bulkloads for MapR Database binary tables.

You can use the `HFileOutputFormat configureIncrementalLoad()` method for writing custom MapReduce applications to perform bulk loads. Although the name of the method implies that you can use

it only for incremental bulk loads, the method also works for full bulk loads, provided that the `-bulkload`, `BULKLOAD`, or `Bulkload` parameter for a table is set to true, as described in Bulk Loading and MapR Database Tables.

If you have a custom MapReduce applications that does not use `HFileOutputFormat.configureIncrementalLoad()`, simply use the path to the MapR Database table that you want to load. Using `HFileOutputFormat.configureIncrementalLoad()` provides at least two advantages:

This method performs a number of tasks that your application would otherwise need to do explicitly:

1. Inspects the table to configure a total order partitioner
2. Uploads the partitions file to the cluster and adds it to the DistributedCache
3. Sets the number of reduce tasks to match the current number of regions
4. Sets the output key/value class to match `HFileOutputFormat`'s requirements
5. Sets the reducer up to perform the appropriate sorting (either `KeyValueSortReducer` or `PutSortReducer`)

This method turns off Speculative Execution automatically. For details, see the note below.

Warning: Turning off Speculative Execution

Speculative Execution of MapReduce tasks is on by default. For custom applications that load MapR Database binary tables, it is recommended to turn Speculative Execution off. When it is on, the tasks that import data might run multiple times. Multiple tasks for an incremental bulkload could insert one or more versions of a record into a table. Multiple tasks for a full bulkload could cause loss of data if the source data continues to be updated during the load.

If your custom MapReduce application uses `HFileOutputFormat.configureIncrementalLoad()`, you do not have to turn off Speculative Execution manually. `HFileOutputFormat.configureIncrementalLoad()` turns it off automatically. Speculative Execution is automatically turned off for MapReduce utilities such as `CopyTable` and `ImportTsv`.

If you are writing a custom MapReduce application that does not use the `HFileOutputFormat configureIncrementalLoad()` method for bulk loading, you must turn off Speculative Execution manually.

Turn off Speculative Execution by setting the following MapReduce version 2 parameter to false: `mapreduce.map.speculative`

If the job is programmatically written, you can turn off Speculative Execution at the code level: `job.setSpeculativeExecution(false);`

Setting for OJAI Applications to Use MapR Client Features

Describes how to set the classpath for OJAI applications to use MapR client features.

When you launch an OJAI application that needs to access MapR client features, ensure that you prefix the classpath with `/opt/mapr/conf:/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop`. Alternatively, copy the `core-site.xml` file to the `/src/main/resources/` folder.

Developing Applications for JSON Tables

As part of its support for JSON tables, MapR Database implements the OJAI API. The OJAI API provides methods for creating, reading, updating, and deleting JSON documents in MapR Database JSON tables. It is available in Java, and starting in EEP 6.0, also available in Node.js, Python, C#, and Go. MapR

Database also provides a MapR Database JSON Client API for managing JSON tables and a MapR Database JSON REST API for performing basic operations using HTTP calls.

The following shows the general flow for developing an OJAI client application that accesses MapR Database JSON tables:

1. Make a connection to MapR Database using the OJAI Connection and Driver interfaces.
2. Request a MapR Database JSON table using the JSON DocumentStore.
3. Specify the table, document, or column family operation.
4. Perform the operation on the table.
5. Return the results.

For additional information about OJAI, refer to the following:

- OJAI wiki page
- OJAI github repository - The README file provides an introduction to OJAI

The MapR Database JSON Client API, implemented in Java, enables you to create, drop, and alter MapR Database JSON tables and column families.

You can also use HTTP calls to create, delete, and query MapR Database JSON tables Using the MapR Database JSON REST API on page 2696.

API Documentation

The following are links to the detailed API pages:

- Java OJAI Client API
- Node.js OJAI Client API
- Python OJAI Client API
- C# OJAI Client API
- Go OJAI Client API
- MapR Database JSON Client API

Note: Beginning with MapR version 6.0, the MapR Database `Table` interface in the MapR Database JSON Client API is deprecated and replaced by the `DocumentStore` interface in the OJAI API.

Sandbox Tutorial for JSON

This section provides a tutorial that will help you build a Java application using the MapR Database JSON Java API library.

This tutorial is an implementation of the Open JSON Application Interface (OJAI) API library. Basic information about using the MapR Database Shell (JSON Tables) on page 5286 utility for JSON tables is also included in this document.

About OJAI

The MapR Database JSON Java API library leverages MapR Database support for OJAI in many areas, including:

- Tables

- Sub-documents
- Efficient access to data
- Large document support
- Security

Key features of the API library include:

- APIs for manipulating JSON documents
- Extensions to the Hadoop ecosystem:
  - Efficient parsing of large files
  - MapReduce integration

JSON Support

MapR Database, in addition to its support of the key-value model (column family), now offers a Document Model. This means that applications can use a JSON document to represent data. JSON is built on two structures:

- A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array.
- An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.

A document looks like this:

```
{
    "_id" : "001",
    "first_name" : "John",
    "last_name" : "Doe",
    "age" : 45,
    "email" : "jd@mydoc.com",
    "interests" : ["sports", "movies"],
    "address" : {
        "street" : "1015 Main Street",
        "city" : "San Jose",
        "state" : "CA",
        "zip" : "95106"
    }
}
```

MapR Database stores the documents in tables. An interesting aspect of MapR Database JSON is that inside a table, documents can have a different structure.

Documents inside MapR Database must have a unique identifier stored in the `_id` field.

MapR Database does not store the documents as a whole in a single location. Instead, MapR Database creates fields for each attribute and nested documents/attributes. This allows MapR Database to access the information very efficiently. When you read, for example using projection, or when you edit a document, only the necessary fields will be modified. MapR Database can store very large documents, for example, multi-GB documents, if the application requires it.

Prerequisites

- Download MapR Sandbox (Version 5.1)
- Configure OJAI in your Maven project.

- Sample Projects
  - OJAI 101
  - REST API & AngularJS

Installation

Follow these steps to install and configure MapR Database.

Install MapR Database

The MapR Sandbox is available as a virtual machine (VM) so the installation is straightforward.

The VM is configured to use a host-only network. Check that you have a host-only network configured in your environment.

Add the IP address of the VM to your `/etc/hosts` file. On Windows:

```
Windows\System32\drivers\etc\hosts
```

In the `/etc/hosts` file, use the MapR Database machine name `maprdemo`.

The VM is a CentOS Linux. You can connect to it using SSH:

```
# password is mapr
ssh mapr@maprdemo
```

Set MAPR_HOME on Your Host

On your host Windows, Mac OSX, or Linux machine where you will write and run your Java applications, follow these steps:

1. Create the following directory structure:

   ```
   /opt/mapr/conf  (for Linux and OSX)
   C:\opt\mapr\conf (for Windows)
   ```

2. If you have a MapR client installed on your machine (existing `/opt/mapr` folder), use it; otherwise, create a new folder. Use an arbitrary name for the folder. For example:`/opt/mapr_51`
3. Create a file in this folder named `mapr-clusters.conf`.
4. Add the following line to this file:

   ```
   demo.mapr.com maprdemo:7222
   ```

   Note: `maprdemo` is the host name or IP address of the VM where the MapR Sandbox is running.

Java Library

In the sample application, you will see the entries required in your Maven `pom.xml` file:

```
<repositories>
    <repository>
        <id>mapr-releases</id>
        <url>https://repository.mapr.com/maven/</url>
        <snapshots><enabled>true</enabled></snapshots>
        <releases><enabled>true</enabled></releases>
    </repository>

</repositories>
```

```
<dependencies>
  <dependency>
    <groupId>com.mapr.db</groupId>
    <artifactId>maprdb</artifactId>
    <version>5.1.0-mapr</version>
  </dependency>
</dependencies>
```

Build Your Java Application

Follow the instructions in this section to build a Java application that uses OJAI and accesses MapR Database JSON tables.

1. Prepare the Directory Structure

This application uses the JSON tables that are mapped to the filesystem and uses the POSIX permission model (read/write for users and groups), and more. In this tutorial we do not use any advanced security features. To read more about security, see Performing File System Operations on MapR Database Tables on page 1041.

A particular directory structure is needed on the VM for this application. Change the permissions for the `apps` directory, and add a `blog` folder (used by the Java REST Sample application).

```
ssh mapr@maprdemo
cd /mapr/demo.mapr.com/
ls -la
chmod 777 apps
mkdir apps/blog
chmod 777 apps/blog
```

2. MapR Database and Java: Overview

The OJAI 101 repository contains the full application that shows the core operations of MapR Database and OJAI.

Before running any Java application, you have to set the library path to point to the MapR client library. For example:

```
-Dmapr.home.dir=/opt/mapr
```

If you used a different name for the folder in `opt`, use that name.

Code Snippets

Create a table:

```
Table table = MapRDB.createTable("/apps/user_profiles");
```

Create an JSON document (JSON):

```
Document doc = MapRDB.newDocument()
                          .set("firstName", "John")
                          .set("lastName", "Doe")
                          .set("age", 50);
```

This document is very simple, but you can use the API to create any valid JSON document, including nested documents and arrays. The JSON document looks like this:

```
{
  "firstName" : "John",
```


last-updated: Apr 11, 2022

```
  "lastName" : "Doe",
  "age" : 50
}
```

You can now insert the document into MapR Database using the `insert()` method. You just need to set the Document ID (or rowkey):

```
table.insert("jdoe", doc);
```

The Document ID is unique within a table. If you try to insert a document using an existing key, a `DocumentExistsException` will be raised. You can also use the `insertOrReplace` method; with this method, if the document with the ID exists, it will be replaced.

You can also retrieve the document with a simple get operation:

```
Document doc2  = table.findById("jdoe");
System.out.println(  doc2  );
System.out.println(
       doc2.getString("firstName") +":"+ doc2.getInt("age")
);
```

To update a document, you have to do the following operations:

```
DocumentMutation mutation = MapRDB.newMutation()
                                  .increment("age", 1)
                                  .set("interests",
Arrays.asList("sports", "movies"));
table.update("jdoe", mutation);
```

As you can see, not only can you update an existing attribute, such as the age increment, but you can also modify the structure of the document. For example, we added the `interests` field to the document.

We won't go into all the capabilities of the database, but let's take a look at a simple query. For example, return all the profiles with an age greater than 50:

```
QueryCondition condition = MapRDB.newCondition()
                                 .is("age", Condition.Op.GREATER, 50)
                                 .build();

DocumentStream rs = table.find(condition);
Iterator<Document> itrs =  rs.iterator();
while (itrs.hasNext()) {
    System.out.println( itrs.next() );
}
rs.close();
```

Finally let's delete a document:

```
table.delete("jdoe");
```

Let's also delete the table:

```
MapRDB.deleteTable("/apps/user_profiles");
```

You have now learned the basics of OJAI and you can see how to store, retrieve, and delete a document from a MapR Database table.

You can now start building your own application or look at the different sample applications available.

3. MapR Database and Java: Sample Application

Clone the following git repository: https://github.com/mapr-demos/maprdb-ojai-101

Look at the com.mapr.db.samples.basic.Ex01SimpleCRUD.run() method.

This method calls three other methods:

- `createDocuments()`, which shows different ways of creating documents
- `queryDocuments()`, which shows different ways of querying documents
- `updateDocuments()`, which shows different ways of updating documents

Execute the following Maven commands:

```
mvn clean package
mvn exec:java -Dexec.mainClass="com.mapr.db.samples.basic.Ex01SimpleCRUD"
```

You can use the MapR Database Shell described at the end of this document to query JSON documents, including the one created by running the sample application.

4. MapR Database and Java: REST API Sample

Clone the following git repository: https://github.com/mapr-demos/maprdb-ojai-rest-sample

This web application exposes a MapR Database document using a REST API (using JAX-RS and Swagger).

Execute the following Maven commands:

```
mvn clean package
mvn exec:java -Dexec.mainClass="com.mapr.db.samples.rest.Main"
```

Look at com.mapr.db.samples.rest.Main.

Start the application and go to: `http://localhost:8080`

Using the mapr dbshell Utility

This section assumes that you performed the following tasks described earlier in this document:

- Downloaded the MapR 5.1 Sandbox.
- Installed the VM.
- Connected to the MapR host using a terminal.

Assuming you are connected to your MapR host using a terminal, you can run the following command to launch the shell:

```
mapr dbshell
```

You can list all the commands of the shell by using `help`. Use the shell as shown in the following examples.

Create a table:

```
create /apps/my_users
```

Add a new document in the table:

```
insert /apps/my_users --id "001" --value '{"first_name":"John",
"last_name":"Doe", "age" : 34 }'
```

Get one document by ID:

```
findbyid /apps/my_users --id 001
```

Get all documents:

```
find /apps/my_users
```

Delete a document:

```
delete /apps/my_users --id "001"
```

Drop a table:

```
drop /apps/my_users
```

Managing JSON Tables
This section describes how to create, list, and delete JSON tables, alter JSON table attributes, set permissions, and manage column families. You can perform these operations using either the MapR Database JSON Client API library or MapR Database Shell commands.

MapR Database JSON Client API

The MapR Database JSON Client API is a Java library. There is not a Python implementation of the library, but you can create and drop MapR Database JSON tables in the Python OJAI client.

Admin API

Use the methods in this interface to perform these tasks:

- Create JSON tables
- Alter JSON tables
- Delete JSON tables
- List JSON tables in a folder
- See if a JSON table exists

For a full list of methods, see the MapR Admin interface

TableDescriptor API

Use the methods in this interface to perform these tasks:

- Create tables with non-default values for one or more of their parameters
- Alter tables

For a full list of interfaces and methods, see the MapR TableDescriptor interface.

MapR Database Shell

The mapr dbshell is a tool for the creation and lightweight manipulation of JSON tables and documents. To run dbshell, enter `mapr dbshell` on the command line after logging into a node in a MapR cluster. See MapR Database Shell (JSON Tables) on page 5286 for more information.

Creating JSON Tables

This topic describes how to create MapR Database JSON tables using either programmatic APIs or dbshell.

Note: Before creating a table, you typically create a directory and MapR volume. This is not required; however, it is a good practice. For example, assuming both the directory and volume names are `sample`, the command would be:

```
// Create directories with hadoop
hadoop fs -mkdir /sample

// Create a MapR volume using maprcli create volume
maprcli volume create -name sample -path /sample -type rw
```

Java

The following Java code examples show you how to create a table in the following ways:

- By using the default values for the table attributes,
- By setting specific values for the table attributes.

See the Admin and TableDescriptor APIs for more information.

The following example shows how to create a table by calling an `Admin` object's `createTable()` `method` and passing, as an argument, the path that you want to use for the new table:

```
public void createJSONTable(String tablePath) throws DBException {
    try (Admin admin = MapRDB.newAdmin()) {
        if (!admin.tableExists(tablePath)) {
            admin.createTable(tablePath);
         }
    }
}
```

Tables created with this version of the `createTable()` method use the default values for their attributes.

Alternatively, the following example how to create a table by passing a `TableDescriptor` object as an argument to the `createTable() method`:

```
/* Create a TableDescriptor for the table to create,
 * passing in the path of the table.
 */
TableDescriptor tableDescriptor = MapRDB.newTableDescriptor(tablePath);

/* Pass the TableDescriptor object and the path to the table
 * to the Admin.createTable() method.
 */
public void createJSONTable(String tablePath, TableDescriptor
tableDescriptor) throws DBException {
    try (Admin admin = MapRDB.newAdmin()) {
        if (!admin.tableExists(tablePath)) {
            admin.createTable(tableDescriptor);
        }
```

```
    }
}
```

This alternative allows you to set values for some of the table's attributes.

Node.js

To create a table in the Node.js OJAI client, call the `Connection.createStore()` method:

```
connection.createStore(table_path)
    .then((store) => {
        // Process result
        ...
    });
```

The method returns a `DocumentStore` object.

Python

To create a table in the Python OJAI client, call the `Connection.create_store()` method:

```
store = connection.create_store(store_path=table_path)
```

The method returns a `DocumentStore` object.

dbshell

The following **dbshell** command shows code syntax for creating a table:

```
# mapr dbshell
maprdb root:> create /<tablePath>/<tableName>
```

C#

To create a table in the C# OJAI client, call the `connection.CreateStore(string storePath)` method:

```
var store = connection.CreateStore(string storePath);
```

The method returns a `DocumentStore` object.

Go

To create a table in the Go OJAI client, call the `connection.CreateStore()` function:

```
store, error := connection.CreateStore("/store_path")
```

The function returns a new `DocumentStore` and an error.

Listing JSON Tables

This topic describes how to list the JSON tables by using either the MapR Database JSON Client API or MapR Database Shell.

Permission Required

The `readAce` permission on the volumes where the JSON tables are located. Setting Whole Volume ACEs on page 1459

Java

The table is listed by calling MapR Database JSON Client API `Admin` object's `listTables()` method and passing, as an argument, the path of the folder.

Use a conditional loop to iterate through the returned list and retrieve the names of the tables.

```
public void listTables(String parentFolder) throws DBException {
    try (Admin admin = MapRDB.newAdmin()) {
        for(Path tablePath : admin.listTables(parentFolder)) {
            System.out.println(tablePath);
        }
    }
}
```

Note: The parameter `parentFolder` provides a path to a folder that is in the MapR filesystem. See "**Table Paths**" in MapR Database JSON Tables for examples.

dbshell

```
# mapr dbshell
maprdb root:> list /demo
/demo/user
/demo/checkin
/demo/review
/demo/business
/demo/tip
5 table(s) found.
```

See dbshell list on page 5303 for further details.

Altering JSON Table Attributes

This topic describes how to change the values of table attributes by using the MapR Database JSON Client API.

In this example, the code turns off the `bulkload` flag on the table. Applications will typically need to turn off this flag after a bulk load of the table with the `import`, `importJSON`, or `copytable` utility.

Create a `TableDescriptor` object for an existing table by passing the path of the table to the `Admin` interface's `getTableDescriptor()` method.

Permissions Required

The `readAce` and `writeAce` permissions on the volumes where the JSON tables are located. For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

Example

Tables are altered a by using the `TableDescriptor` object and then passing that object to the `Admin` interface's `altertable()` method.

```
public void alterTable(String tablePath) throws DBException {
    try (Admin admin = MapRDB.newAdmin()) {
        TableDescriptor tableDesc = admin.getTableDescriptor(tablePath);
        // set bulk load to false
        tableDesc.setBulkLoad(false);
        admin.alterTable(tableDesc);
    }
}
```


last-updated: Apr 11, 2022

Deleting JSON Tables

This topic describes how to delete MapR Database JSON tables using either programmatic APIs or dbshell.

Java

To delete a table in the Java OJAI client, call an `Admin` object's `deleteTable()` method and pass, as an argument, the path of the table to delete:

```
public void deleteTable(String tablePath) throws DBException {
    try (Admin admin = MapRDB.newAdmin()) {
        if (admin.tableExists(tablePath)) {
            admin.deleteTable(tablePath);
        }
    }
}
```

Node.js

To delete a table in the Node.js OJAI client, call the `Connection.deleteStore()` method:

```
connection.deleteStore(table_path)
    then((deleteResponse) => {
        // Process deleteReponse
        ...
    });
```

Python

To delete a table in the Python OJAI client, call the `Connection.delete_store()` method:

```
rc = connection.delete_store(store_path=table_path)
```

dbshell

```
# mapr dbshell
maprdb root:> drop <table path>
```

See dbshell drop on page 5290 for additional details.

C#

To delete a table in the C# OJAI client, call the `connection.DeleteStore(string storePath)` method:

```
connection.DeleteStore(string storePath);
```

Go

To delete a table in the Go OJAI client, call the `connection.DeleteStore()` function:

```
err := connection.DeleteStore("/store_path")
```

Permissions Required

You must have both the `readAce` and `writeAce` permissions on the volumes where the JSON tables are located to delete it. For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

Permission Types for Fields and Column Families in JSON Tables

By using access-control expressions (ACEs), you can grant or deny access to fields and column families that are in JSON tables.

For an explanation of the syntax of ACEs, see ACE Syntax.

There are three types of permission:

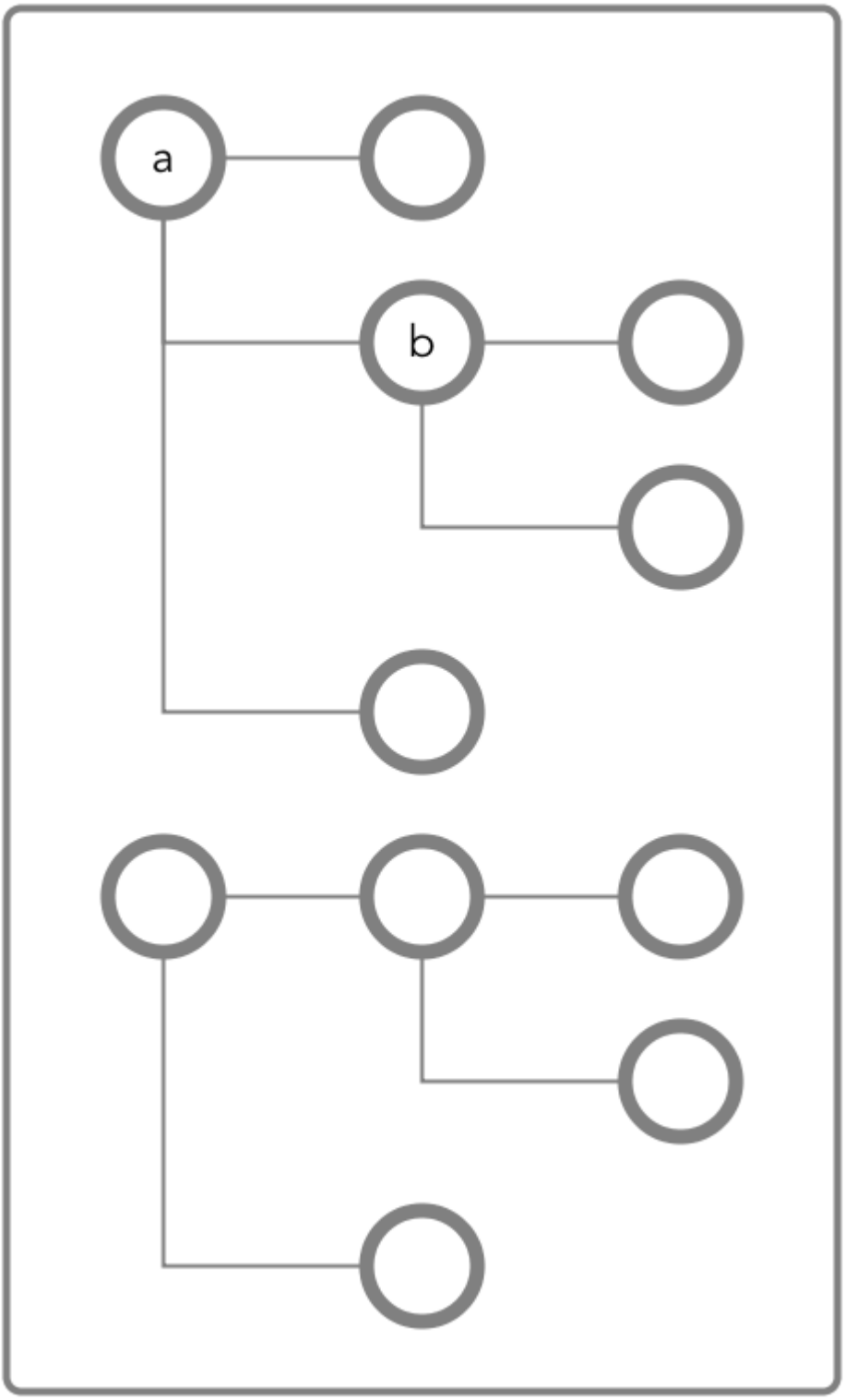
- Traverse (`traverseperm`)
- Read (`readperm`)
- Write (`writeperm`)

Traverse (`traverseperm`)

This permission allows the grantee to descend a hierarchy of fields to access the fields that the grantee has write or read permission on.

For example, suppose that a user has read and write access only on field b in this document.

last-updated: Apr 11, 2022

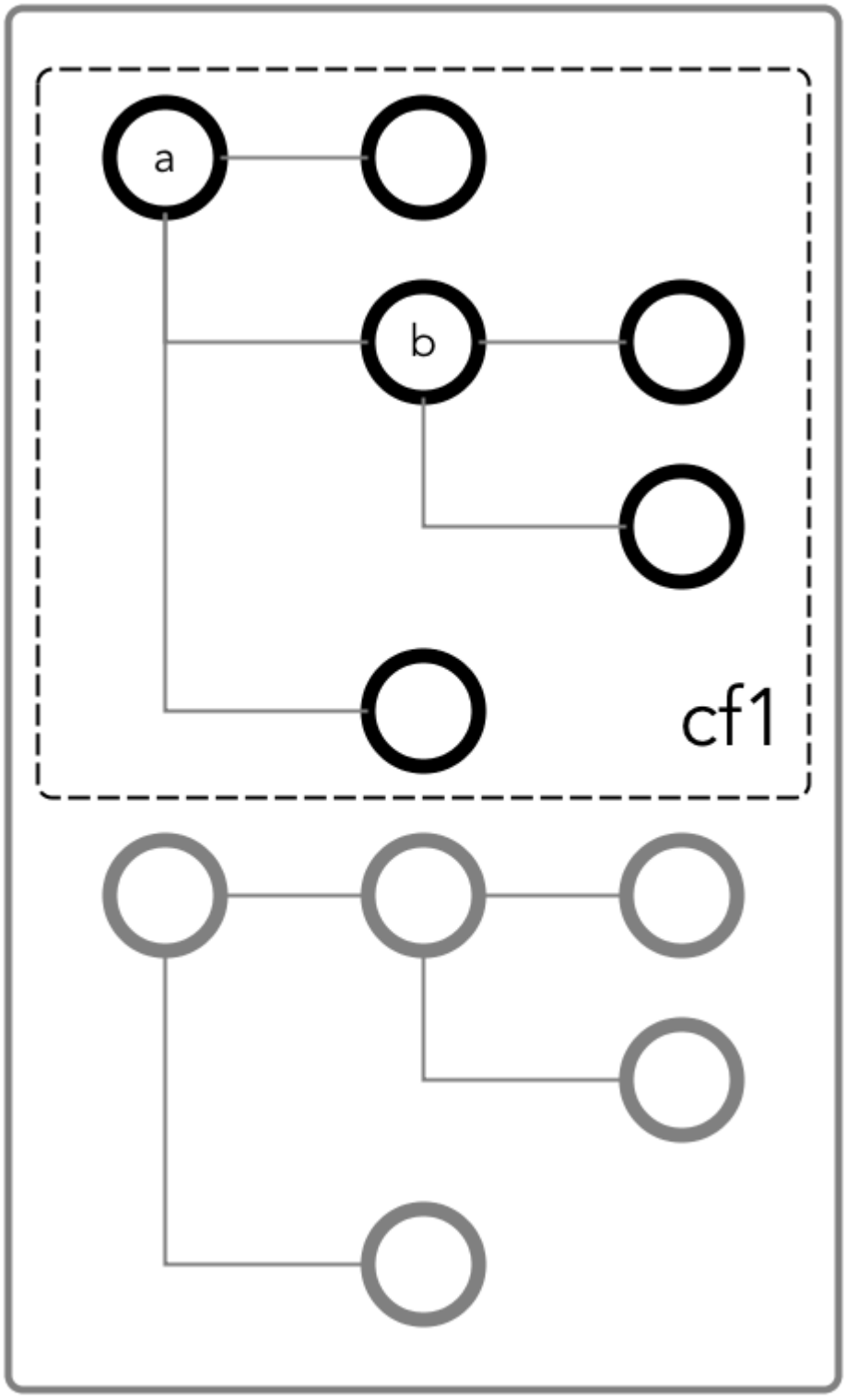


To access field b, the user would need to be able to traverse (pass through) field a. In this case, because the entire document is in the default column family, the user could be granted traverse permission on the default column family. Field a would inherit the traverse permission.

If traverse permission on the default column family were denied the user, it would not be possible for the user to access field b. Granting traverse permission on field a in this case would have no effect.

last-updated: Apr 11, 2022

In the next example, field a is a column family named `cf1`.



To be able to read and write at field b, the user could be granted the traverse permission on the column family.

last-updated: Apr 11, 2022

Read (`readperm`)

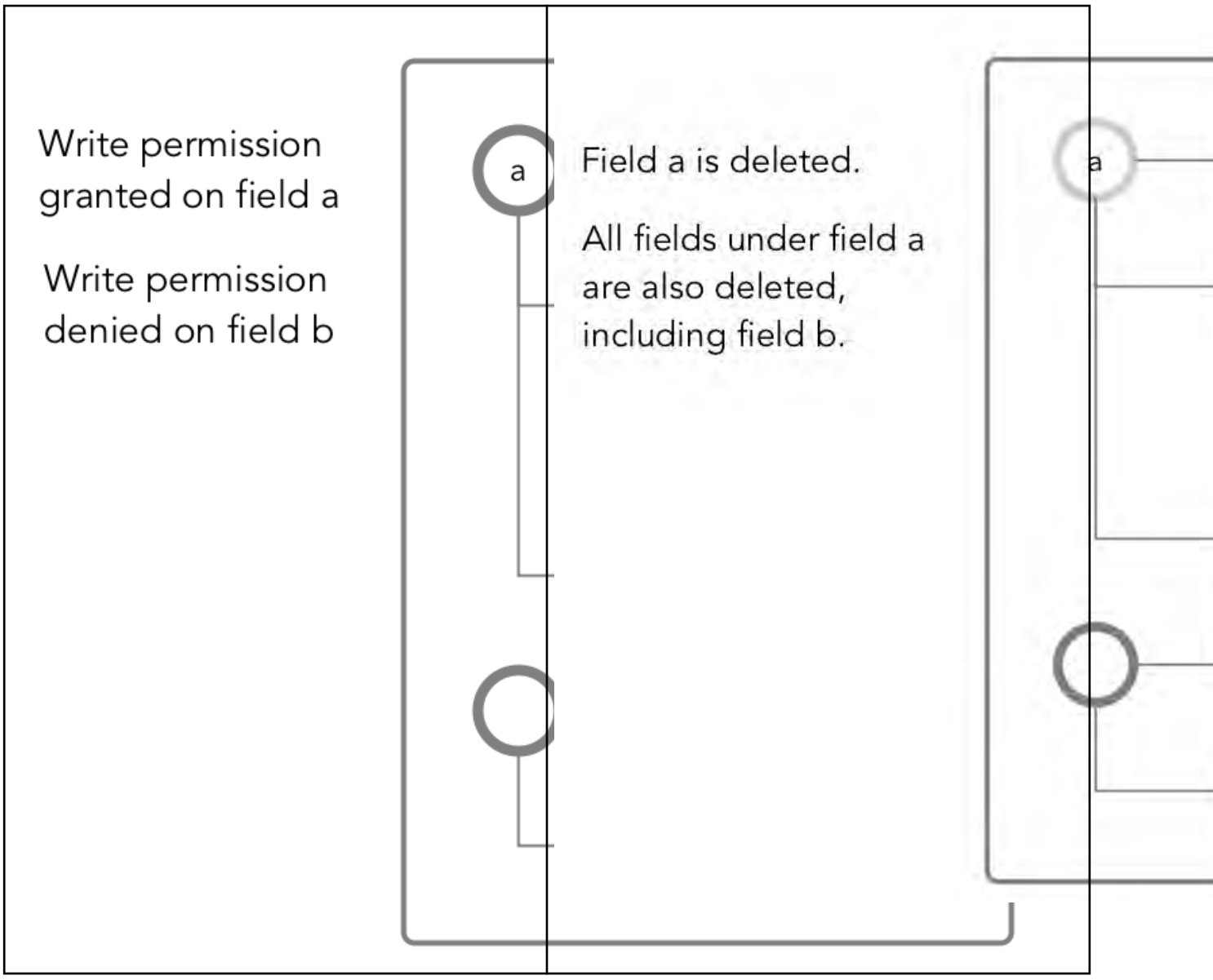
This permission allows the grantee to read from a field.

This permission extends to fields that are nested below the field that the permission was granted on. However, grantees can be explicitly denied the permission on any of the nested fields.

Write (`writeperm`)

This permission allows the grantee to delete a field, insert a value into a field, or overwrite a field's value.

As illustrated in the following two diagrams, deleting a field also deletes all fields that are nested within that field, even those fields on which the write permission is explicitly denied.



Obtaining `readperm` and `writeperm` on Fields
In this scenario, you want to perform an operation on a field, and the operation requires that you have readperm and writeperm permissions on that field. How you obtain these permissions depends on whether the field is in the default column family or a non-default column family.

When the field is in the default column family

In the document below, you want to perform an operation on field c, which is in the default column family. The operation requires you to have `readperm` and `writeperm` on field c.

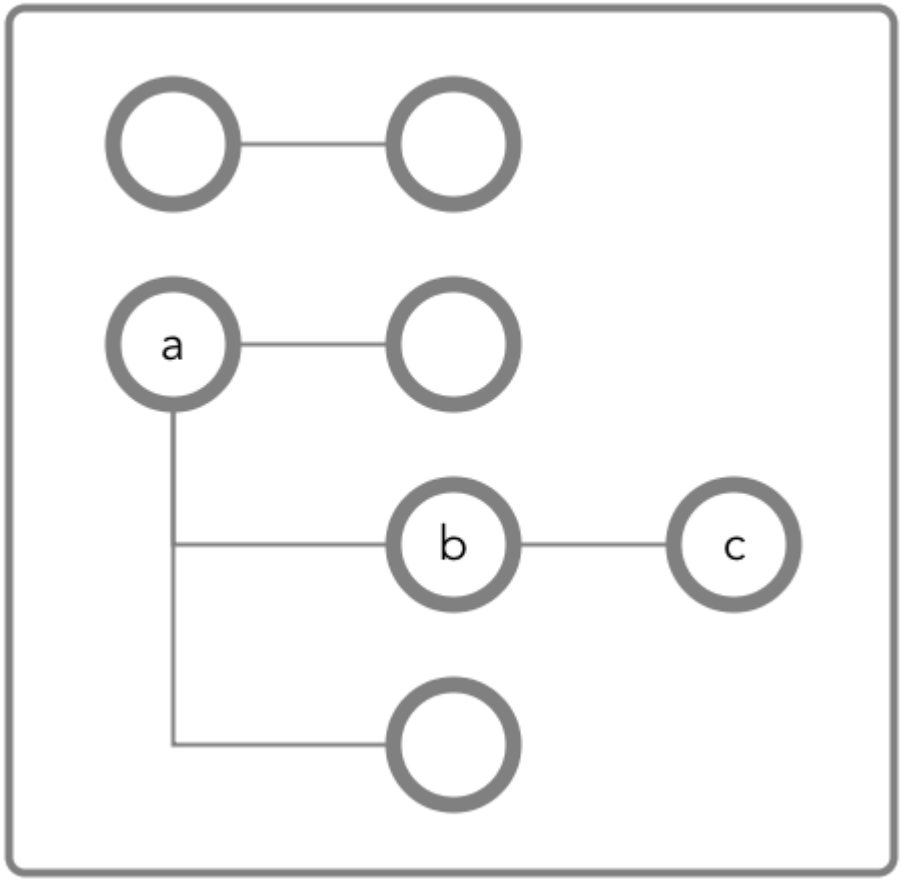


Figure 23: Schematic diagram of an JSON document in which all fields are in the default column family

Case 1: You have `readperm` and `writeperm` on the default column family

In this case, field c inherits these permissions, assuming that the permissions were not denied on field a or b.

If you do not have `readperm` and `writeperm` on field a or b, you need `traverseperm` on the field that denied you those permissions. You also need `readperm` and `writeperm` explicitly granted to you on field c. You could be granted these permissions with the `maprcli table cf colperm set` command, as in these examples:

```
maprcli table cf colperm set -path
<path to JSON table>
-cfname default -name
a.b -traverseperm u:<user ID> |
<existing ACE for this field>
maprcli table cf colperm set -path
<path to JSON table> -cfname default
```

```
-name a.b.c -readperm u:<user ID>
| <existing ACE for this
field> -writeperm
u:<user ID> | <existing ACE for this
field>
```

Case 2: You do not have `readperm` and `writeperm` on the default column family

In this case, you need the `traverseperm` permission on the default column family. Fields a and b inherit this permission. You also need `readperm` and `writeperm` on field c.

You could be granted these permissions with commands similar to these:

```
maprcli table cf edit -path
<path to JSON table> -cfname
default -traverseperm
u:<user ID> | <existing ACE for this
field>
maprcli table cf colperm set -path
<path to JSON table> -cfname
default -name a.b.c
-readperm u:<user ID> | <existing ACE
for this field> -writeperm u:<user
ID> |
<existing ACE for this field>
```

When the field is in a non-default column family

Note: Non-default column families are an advanced feature of MapR Database's native JSON support. For information about them, see Column Families in JSON table.

In the following document, you want to perform an operation on field c, which is in the column family `cf1` that is defined at field b with the path `a.b`.

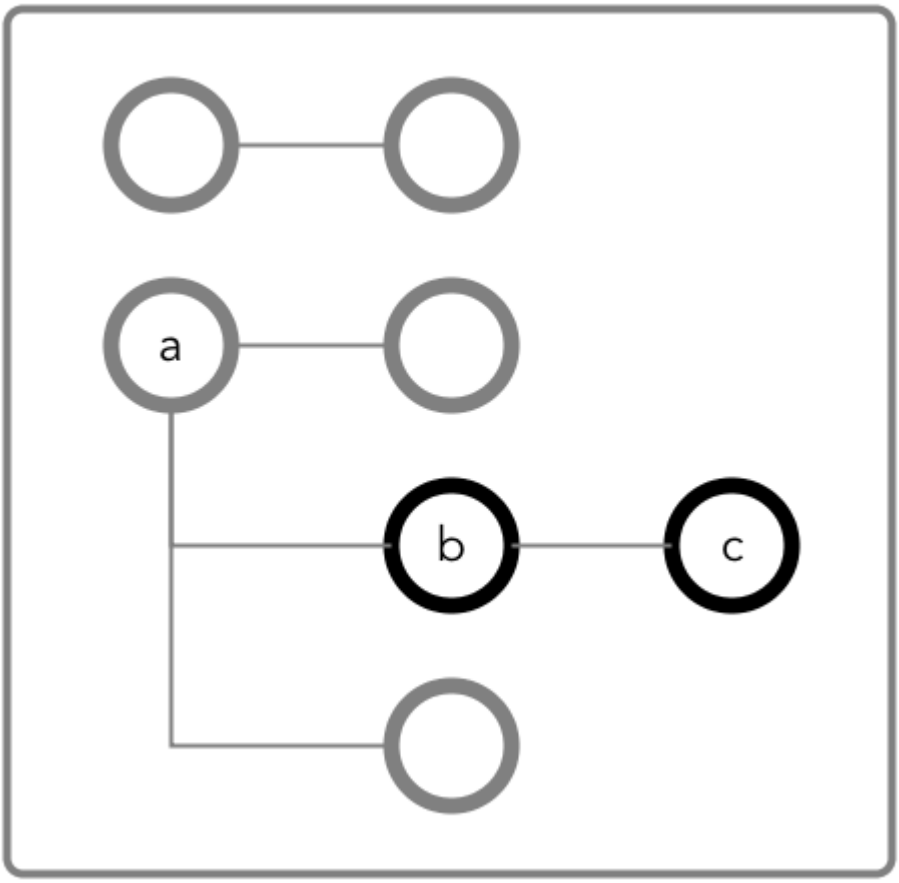


Figure 24: Schematic diagram of an JSON document in which fields b and c are in a column family that has the path `a.b`

Case 1: You do not have `readperm` and `writeperm` on field `b`	You need `traverseperm` on field b and both `readperm` and `writeperm` on field c. You can be granted these permissions with commands similar to these:
	`/opt/mapr/bin/maprcli table cf edit -path <path to JSON table> -cfname cf1 -traverseperm u:<user ID> | <existing ACE for this field>`
	`maprcli table cf colperm set -path <path to JSON table> -cfname cf1 -name a.b.c -readperm u:<user ID> | <existing ACE for this field> -writeperm u:<user ID> | <existing ACE for this field>`

Case 2: You do have `readperm` and `writeperm` on field `b`	You do not need any further permissions. Field c inherits your `readperm` and `writeperm` permissions from field b.

Obtaining `readperm` or `writeperm` on Fields

In this scenario, you want to perform an operation on a field, and the operation requires that you have readperm or writeperm permissions on that field. How you obtain either permission depends on whether the field is in the default column family or a non-default column family.

When the field is in the default column family

In the following document, you want to perform an operation on field `c,` which is in the default column family. The operation requires you to have `readperm` or `writeperm` on field `c`.

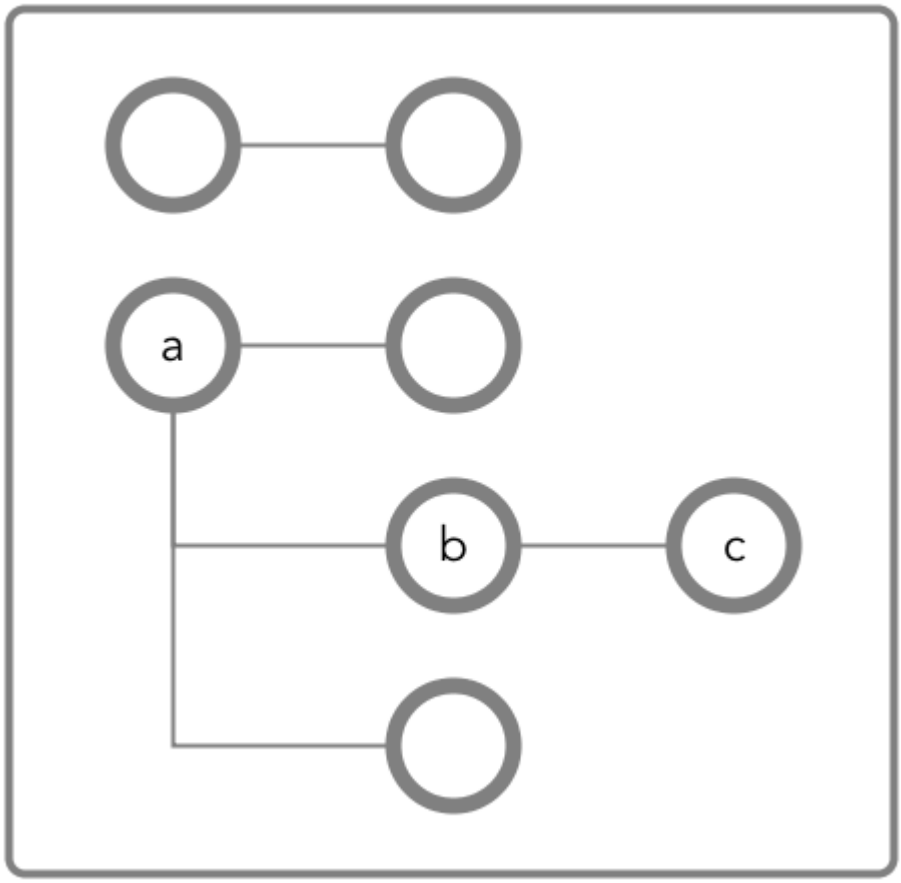


Figure 25: Schematic diagram of an JSON document in which all fields are in the default column family

Case 1: You have the same permission (`readperm` or `writeperm`) on the default column family

In this case, field c inherits the permission, assuming that the permission was not denied on field a or b.

If you do not have `readperm` or `writeperm` on field a or b, you need `traverseperm` on the field that denied you the permission that you need. You also need `readperm` or `writeperm` explicitly granted to you on field c.

Example commands to grant these permissions:

```
/opt/mapr/bin/maprcli table cf
colperm set -path <path to JSON
table> -cfname
```

```
default -name a.b -traverseperm
u:<user ID> | <existing ACE for this
field>
```

The next example command grants `readperm`:

```
/opt/mapr/bin/maprcli table cf
colperm set -path <path to JSON
table> -cfname
default -name a.b.c -readperm u:<user
ID> | <existing ACE for this field>
```

Case 2: You do not have the same permission (`readperm` or `writeperm`) on the default column family

In this case, you need the `traverseperm` permission on the default column family. You also need `readperm` or `writeperm` explicitly granted to you on field `c`.

Example commands to grant these permissions:

```
/opt/mapr/bin/maprcli table cf
edit -path <path to JSON
table> -cfname cf1
-traverseperm u:<user ID> | <existing
ACE for this field>
```

This next example command grants `readperm`:

```
/opt/mapr/bin/maprcli table cf
colperm set -path <path to JSON
table> -cfname cf1
-name a.b.c -readperm u:<user ID> |
<existing ACE for this field>
```

When the field is in a non-default column family

Note: Non-default column families are an advanced feature of MapR Database's native JSON support. For information about them, see Column Families in JSON Tables.

In the following document, you want to perform an operation on field `c`, which is in the column family that is defined at field `b` with the path `a.b`. The operation requires you to have `readperm` or `writeperm` on field `c`.

last-updated: Apr 11, 2022

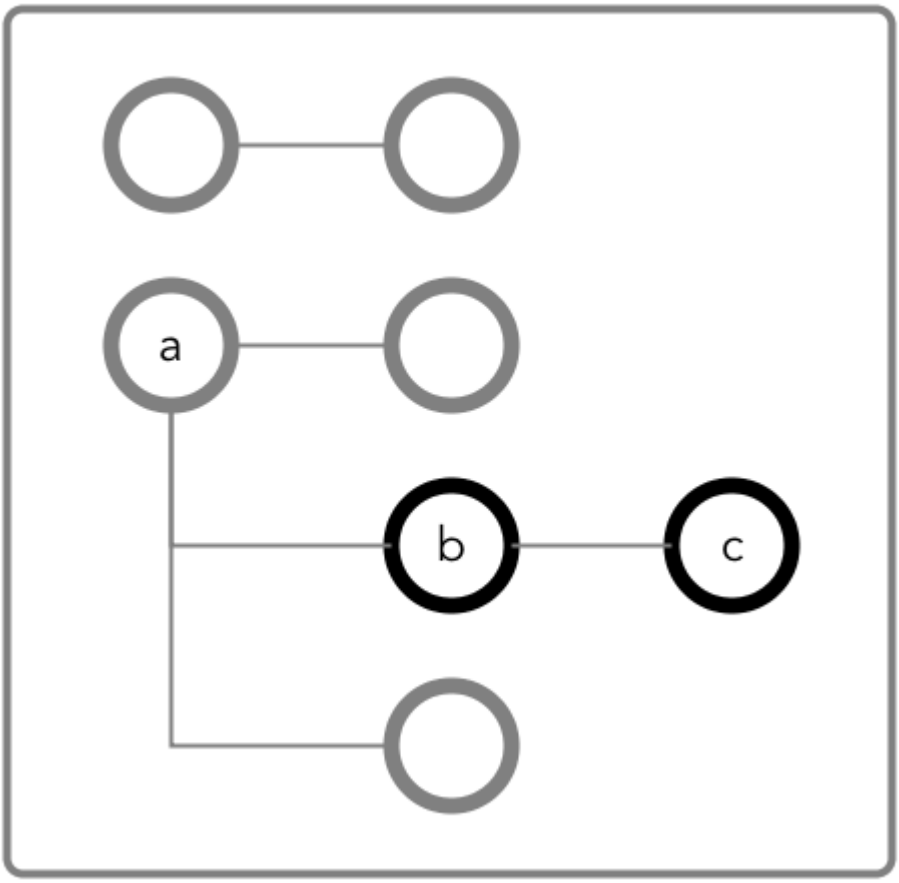


Figure 26: Schematic diagram of an JSON document in which fields b and c are in a column family that has the path `a.b`

Case 1: You do not have the permission you need (`readperm` or `writeperm`) on field `b`	You need `traverseperm` on field b, and you need `readperm` or `writeperm` granted to you explicitly on field c.
	Example commands to grant these permissions:
	`/opt/mapr/bin/maprcli table cf edit -path <path to JSON table> -cfname cf1 -traverseperm u:<user ID> | <existing ACE for this field>`
	`maprcli table cf colperm set -path <path to JSON table> -cfname cf1 -name a.b.c -readperm u:<user ID> | <existing ACE for this field>`

Case 2: You do have the permission you need (`readperm` or `writeperm`) on field `b`	You do not need any further permissions. Field c inherits your `readperm` and `writeperm` permissions from field b.

Setting Permissions on Arrays

When granting permissions on a field, if the field contains array data, you must grant the permission on the array field. This grants access not only to array data in the field, but also nested documents and scalar data. It is also possible to set permissions on subfields within nested documents that are stored in an array.

Note: This topic describes the behavior of permissions in MapR Database version 6.1 and later, regardless of the MapR version you used to grant the permissions. To understand how permissions on arrays behave in earlier releases, see Operational Changes (MapR 6.1.0) - Permissions on Arrays in MapR Database JSON.

Granting Permissions on Array Elements

Suppose you have the following documents where `person` is:

- An array of nested documents in document `id001`
- A single nested document in document `id002`
- A scalar value in document `id003`

```
{
    "_id" : "id001",
    "person" : [
        {"name" : {"last" : "Smith", "first" : "John"}},
        {"name" : {"last" : "Subramanium", "first" : "Ananya"}}
    ]
}
{
    "_id" : "id002",
    "person" : {"name" : {"last" : "Doe", "first" : "Jane"}}
}
{
    "_id" : "id003",
    "person" : "Unknown"
}
```

If you grant a user read permission on the array `person[]`, that user can read every field in every nested document within the array in document `id001`. The permission also enables the user to read the `person` field in documents `id002` and `id003`.

If you receive an error when trying to grant permission on `person[]` because you previously granted permission on `person`, then you (or an administrator with the appropriate permissions) must first remove the existing permission on `person`. If you expect the schema of the `person` field to evolve to include non-array and array data, then you should grant the permission on `person[]` rather than `person`, to avoid having to remove the conflicting `person` permission.

You cannot grant permissions on individual elements in an array; for example: `person[1]`. Granting permission on an array enables access to the entire array.

Granting Permissions on Nested Document Fields in an Array

If you want to restrict read access to only specific fields in `person`, whether the field is an array of nested documents or a single nested document, perform the following steps:

1. Deny the user read permission on the array `person[]`.
2. Grant the user traverse permission on the array `person[]`.
3. Grant the user read permission on the specific fields.

For example, to grant the user read permission on only the first names in the nested documents, for the third step, grant read permission on `person[].name.first`. The permission enables the user to read the field in all nested documents in documents `id001` and `id002`.

last-updated: Apr 11, 2022

If permissions already exist on `person.name.first`, then all attempts to define permissions on `person[].name.first` fails. You (or an administrator with the appropriate permissions) must first remove the existing permission on `person.name.first`. Similar to the scenario described in the previous section, if you expect the schema of the `person` field to evolve to include individual nested documents as well as arrays of nested documents, then you should grant the permission on `person[].name.first` to avoid having to remove the conflicting permission.

If you already have permissions on `person[].name.first`, then attempting to define permissions on `person.name.first` fails. There is no need to add this permission.

Granting Permissions on JSON Tables
This page summarizes the default access-control expressions for the supported ways of setting read, traverse, and write permissions.

The default permissions for column families are determined when tables are created. The default permissions for fields are inherited from the column family where the fields are located.

Action	Method	Permissions	Default Access-Control Expressions
Set default permissions on new column families when creating a JSON table.	Java API	`-defaultreadperm -defaulttraversepe rm -defaultwriteperm`	`u:<ID of the process>`
	`maprcli table create`		`u:<user ID of table creator>`
	mapr dbshell		
	Control System		
Set default permissions on new column families when editing a JSON table.	`maprcli table edit`		Current ACEs
	Control System		
Set permissions on a column family when creating the column family.	`maprcli table cf create`	`-readperm -traverseperm -writeperm -indexperm`	ACEs for `-defaultreadperm`, `-defaulttraverseperm`, and `-defaultwriteperm`
	Control System		
Set permissions on a column family when editing the column family.	`maprcli table cf edit`		Current ACEs
	Control System		
Set permissions on individual fields.	`maprcli table cf colperm set`		Inherited from column family or parent field
	Control System		

Managing Column Families
JSON tables store data in column families, which are collections of fields that are stored together on disk.

Each table has a default column family, which is default storage for all fields in the documents of a table. You can create additional column families to store data for a collection of fields in a separate location on disk. Queries that operate only on data that is stored in a column family are more efficient and better performing than queries on the same data when that data is stored with other data in a table. You can also cache values from a column family in memory.

Applications do not need to be aware of the existence of column families. They perform CRUD operations by using the paths of fields in a document. For example, to update any of the fields below `a.c`, an application does not need to be aware that the field is in the column family at the path `a.c`. The application simply moves through the document along the path to the field.

For more information, see Column Families in JSON Tables on page 527.

Creating Column Families
You can create column families with the MapR Database JSON Java API library by using the `Admin.createTable(TableDescriptor tableDescriptor)` method.

Add a column family to the `TableDescriptor` object before passing that object to the `createTable()` method.

Restriction

If any existing column family in a JSON table, including the default column family, uses a time-to-live that is greater than 0, you cannot create any additional column families in that table. See Setting TTL for Data.

Permissions Required

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `createrenamefamilyperm` on the table

Example

Here is an example of using the API to create two column families -- the default column family and a custom column family -- during the creation of a table:

```
/* Create a TableDescriptor for the table to create,
   passing in the path of the table. */
TableDescriptor tableDescriptor = MapRDB.newTableDescriptor(tablePath);

/* Create a FamilyDescriptor for the default column family.
   When you create a table with the API, you must also create
   the default column family.
   After creating the FamilyDescriptor, add it to
   the TableDescriptor. */
FamilyDescriptor defaultfamilyDesc = MapRDB.newDefaultFamilyDescriptor();
tableDescriptor.addFamily(defaultfamilyDesc);

/* Create a FamilyDescriptor for the custom column family
   to create. The setJsonFieldPath() method specifies the field
   at which to create the column family.
   After creating the FamilyDescriptor, add it to
   the TableDescriptor. */
FamilyDescriptor familyDescriptor = MapRDB.newFamilyDescriptor()
  .setName("CF1")
  .setJsonFieldPath("a.b");
tableDescriptor.addFamily(familyDescriptor);

// Pass the TableDescriptor to the Admin.createTable() method.
public void createJSONTable(String tablePath, TableDescriptor
tableDescriptor) throws DBException {
  try (Admin admin = MapRDB.newAdmin()) {
    if (!admin.tableExists(tablePath)) {
      admin.createTable(tableDescriptor);
    }
  }
}
```

Alternative Method

You can also create column families in JSON tables by running the command `table cf create`.

Altering Column Families
You can alter column families, including the default column family for a table, by using the `Admin.alterFamily()` method in the MapR Database JSON Java API library.

Permissions Required

- `readAce` and `writeAce` on the volume
- `lookupdir` on directories in the path
- `createrenamefamilyperm` on the table

Example

Here is an example of using the API to change the name of a column family:

```
public void alterColumnFamily(String tablePath, String familyName,
  String newFamilyName) throws DBException {
  try (Admin admin = MapRDB.newAdmin()) {

    /* Get a TableDescriptor object for the table. This object
       gives access to the column families that are in the table. */
    TableDescriptor tableDesc = admin.getTableDescriptor(tablePath);

    /* Get a FamilyDescriptor object for the column family to
       change the name of. /
    FamilyDescriptor familyDesc = tableDesc.getFamily(familyName);

    // Rename the column family.
    familyDesc.setName(newFamilyName);

    /* Call alterFamily(), passing in the path of the table,
       the original name of the column family, and the
       FamilyDescriptor in which the new name was set. */
    admin.alterFamily(tablePath, familyName, familyDesc);
  }
}
```

Alternative Method

You can also edit column families in JSON tables by running the command `table cf edit`.

Deleting Column Families
You can delete a column family (except for the default column family) in a JSON table with the `Admin.deleteFamily()` Java method.

Important: Starting in the 6.0 release, you cannot delete a column family from a JSON table.

Permissions Required

The `readAce` and `writeAce` permissions on the volumes where the JSON tables are located. For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

Behavior

The data that is in the specified column family is deleted. If the column family is followed by one or more column families in a hierarchy, the other column families in the hierarchy are unaffected and still accessible. For example, if column family `CF1` at path `a.c` is followed by column family `CF2` at path `a.c.f`, `CF2` remains accessible and only the data in `CF1` is deleted.

Before deleting the column family `CF1` at `a.c`	After deleting the column family `CF1` at `a.c`

<pre>{
 "a" : {
  "b" : "value_b",
  "c" : {
  "d" : "value_d",
  "e" : "value_e",
  "f" : {
  "g" :
"value_g",
  "h" : "value_h"
  }
  }
  }
}</pre>
	<pre>{
	 "a" : {
	  "b" : "value_b",
	  "c" : {
	  "d" : "",
	  "e" : "",
	  "f" : {
	  "g" :
	"value_g",
	  "h" : "value_h"
	  }
	  }
	  }
	}</pre>

Example of using the `Admin.deleteFamily()` method

```
public void deleteColumnFamily(String tablePath, String familyName) throws
DBException {
    try (Admin admin = MapRDB.newAdmin()) {
      if (admin.tableExists(tablePath)) {
        admin.deleteFamily(tablePath, familyName);
      }
    }
}
```

Parameter	Description
`tablePath`	The path of the table in the MapR filesystem. See the "Table Paths" section in MapR Database JSON Tables on page 524.
`familyName`	The name of the column family to delete. You cannot delete the default column family. If familyName is equal to "default", the API returns an exception.

Setting TTL for Data

You can delete stale JSON documents in JSON tables automatically by setting a time-to-live (TTL) value on the column family.

TTL is set only on the default column family in a JSON table. The duration that you set applies to each entire JSON document in the JSON table.

Note: Only the default column family can exist in order to set TTL; no other column families can exist in the JSON table. You also cannot set the TTL for a JSON table if it has secondary indexes.

Data can become stale. If the data in an JSON document has not been updated within a certain period of time, you might want to delete the document. In the case of a large amount of JSON documents, applications should not have to track the time between updates and then delete the expired documents.

Because the time-to-live that is set on a column family affects an entire JSON table, only the default column family is allowed to have a non-default time-to-live value. In addition, to prevent multiple column families from having non-default time-to-live values, additional column families can not be created in a table if the default column family has a non-default value. This is because if more than one column family had a non-default TTL value, fragments of documents would expire at different times, leading to inconsistent views of data.

last-updated: Apr 11, 2022

Permission Required

The `writeAce` permission on the volumes where the JSON tables are located. For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

Example: Setting the default column family to a non-default time-to-live value
If you set the time-to-live parameter for the default column family to 864,000 seconds, JSON documents in that table are considered to be stale if the document's data has not been updated within 10 days and are automatically deleted.

The following code example creates a JSON table, the default column family and sets the TTL to a non-default value of 10 days (864,000 seconds).

```
/* Create a TableDescriptor for the table to create,
   passing in the path of the table. */
TableDescriptor tableDescriptor = MapRDB.newTableDescriptor(tablePath);

/* Create a FamilyDescriptor for the default column family.
   When you create a table with the API, you must also create
   the default column family.
   Set the TTL to 10 days.
   After creating the FamilyDescriptor, add it to
   the TableDescriptor. */
FamilyDescriptor defaultfamilyDesc = MapRDB.newDefaultFamilyDescriptor()
   .setTTL(864000);
tableDescriptor.addFamily(defaultfamilyDesc);


// Pass the TableDescriptor to the Admin.createTable() method.
public void createJSONTable(String tablePath, TableDescriptor
tableDescriptor) throws DBException {
  try (Admin admin = MapRDB.newAdmin()) {
    if (!admin.tableExists(tablePath)) {
      admin.createTable(tableDescriptor);
    }
  }
}
```

Managing JSON Documents
To perform CRUD operations (create, read, update, and delete) on JSON documents in MapR Database JSON tables using the OJAI API, you use `Document`, `DocumentStore`, and `DocumentMutation` objects.

You can also perform these operations using MapR Database Shell (JSON Tables) on page 5286.

`Document`

To create a JSON document, you must create a `Document` object. See the following for information specific to each language:

Java	To create a JSON document in Java OJAI, use the Document interface. See Creating JSON Documents in Java OJAI on page 2545 to learn about the different ways to create `Document` objects in Java.
Node.js	To create a `Document` object in Node.js OJAI, simply create a JSON object.

	See Sample OJAI Code for Creating JSON Documents on page 2550 for an example of how to do this.
Python	The preferred approach is to create a `Document` object in Python is to create a Python dictionary. You can also use the Document interface. See Creating JSON Documents in Python OJAI on page 2549 to learn about these two ways to create JSON documents in Python.
C#	To create a `Document` object in C# OJAI, create a C# object. See Sample OJAI Code for Creating JSON Documents on page 2550 for an example of how to do this. For C# OJAI examples, see this Github page.
Go	To create a `Document` object in Go OJAI, create a Go structure. See Sample OJAI Code for Creating JSON Documents on page 2550 for an example of how to do this.

`DocumentStore`

After you create a `Document` object, you can pass it to the `DocumentStore` interface. The interface has methods to perform the following tasks:

- Delete documents from tables
- Insert documents into tables
- Replace documents in tables

See the following for API links to the `DocumentStore` interface in each language:

Java	DocumentStore **Note:** By default, OJAI implements non-buffered writes. If you want buffered writes instead, use the `ojai.mapr.documentstore.buffer-writes` option and with the `Document` object. This option is available only in the Java OJAI API. See Enabling Buffered Writes in Java OJAI on page 2670 for more information.
Node.js	DocumentStore
Python	DocumentStore
C#	DocumentStore For C# OJAI examples, see this Github page.
Go	DocumentStore

last-updated: Apr 11, 2022

`DocumentMutation`

To make changes to JSON documents, create a `DocumentMutation` object. A `DocumentMutation` enables you to perform OJAI mutations, which includes replacing, updating, combining, and deleting fields in a JSON document. For a list of available mutations, see Using OJAI Mutation Syntax on page 2561.

Java	To create a `DocumentMutation` object, call the methods in the `DocumentMutation` class corresponding to the mutation operations you want to perform. See DocumentMutation for a list of available methods. Pass the `DocumentMutation` object to either the DocumentStore.checkAndUpdate or DocumentStore.update method to apply the changes to the document. The first method accepts a QueryCondition parameter that must evaluate to true for the mutation to be applied. Both methods have an `_id` parameter corresponding to the document to be updated.
Node.js	To create a `DocumentMutation` object, create a JSON object Using OJAI Mutation Syntax on page 2561. Pass the `DocumentMutation` object to either the DocumentStore.checkAndUpdate or DocumentStore.update method to apply the changes to the document. The `DocumentStore.checkAndUpdate()` method accepts an OJAI query condition parameter that must evaluate to true for the mutation to be applied. Both methods have an `_id` parameter corresponding to the document to be updated.
Python	To create a `DocumentMutation` object, create a Python dictionary object Using OJAI Mutation Syntax on page 2561. Pass the `DocumentMutation` object to either the DocumentStore.check_and_update or DocumentStore.update method to apply the changes to the document. The `DocumentStore.check_and_update()` method accepts an OJAI query condition parameter that must evaluate to true for the mutation to be applied. Both methods have an `_id` parameter corresponding to the document to be updated.
C#	To create a `DocumentMutation` object, create a C# object Using OJAI Mutation Syntax on page 2561. Pass the `DocumentMutation` object to either the `DocumentStore.CheckAndUpdate` or `DocumentStore.Update` method to apply the changes to the document. The `DocumentStore.CheckAndUpdate` method accepts an OJAI query condition parameter that must evaluate to true for the mutation to be applied. Both methods have an `_id` parameter corresponding to the document to be updated. For C# OJAI examples, see this Github page.
Go	To create a `DocumentMutation` object, create a Go structure Using OJAI Mutation Syntax on page 2561.

Pass the `DocumentMutation` structure to either the `DocumentStore.CheckAndUpdate` or `DocumentStore.Update` method to apply the changes to the document. The `DocumentStore.CheckAndUpdate` method accepts an OJAI query condition parameter that must evaluate to true for the mutation to be applied. Both methods have an `_id` parameter corresponding to the document to be updated.

By default, the default maximum size of a JSON document is 32 MB. A `DocumentMutation` does not enforce this limit. MapR Database enforces the limit when you pass your `DocumentMutation` object to the `DocumentStore` method. See JSON Document Size on page 510 for information about how to increase this limit.

See Examples: Updating JSON Documents on page 2569 for examples that use mutations.

Creating JSON Documents in OJAI

The way you create a JSON document in your OJAI application depends on the language you use.

Creating JSON Documents in Java OJAI

There are several ways to create JSON documents in your Java OJAI application. They all require you to call the `Connection.newDocument` method to create a `Document` object.

Related information

Connection

Document

Create a Document Using a Document Object in Java OJAI

You can create a new JSON document in your Java OJAI client by first calling the `Connection.newDocument()` method to create a `Document` object, and then calling methods on the object to specify document fields and values.

The following shows the detailed sequence of steps:

1. Create a new JSON document by calling the `newDocument()` method in the `Connection` class.
2. Specify the ID of the document with the `setId()` method.
3. Specify field names and their values with the `set()` or `setArray()` method.
4. Return the results in a `Document` object.

For example, suppose you want to create the following JSON document:

```
{
    "_id" : "movie0000001",
    "title" : "OJAI -- The Documentary",
    "studio" : "MapR Technologies, Inc.",
    "release_date" : "2015-09-29",
    "trailers" : {
        "teaser" : "https://10.10.21.90/trailers/teaser",
        "theatrical" : "https://10.10.21.90/trailers/theatrical"
    },
    "characters" : [
        "Heroic Developer",
        "Evil Release Manager",
        "Mad Development Manager"
    ],
    "box_office_gross" : 1000000000L
}
```


last-updated: Apr 11, 2022

The following method creates the document:

```
public Document buildDocument() {
    return connection.newDocument()
        .setId("movie0000001")
        .set("title", "OJAI -- The Documentary")
        .set("studio", "MapR Technologies, Inc.")
        .set("release_date", Values.parseDate("2015-09-29"))
        .set("trailers.teaser", "https://10.10.21.90/trailers/teaser")
        .set("trailers.theatrical", "https://10.10.21.90/trailers/
theatrical")
        .setArray("characters",
            ImmutableList.of(
                "Heroic Developer", "Evil Release Manager", "Mad
Development Manager"))
        .set("box_office_gross", 1000000000L);
}
```

Create a Document from a JSON String in Java OJAI

You can create a new JSON document in your Java OJAI client by passing a JSON string to the `Connection.newDocument()` method.

To create the following JSON document:

```
{
    "_id":"id001",
    "a":1,
    "b":"aString",
    "array":[
        1,
        2,
        "arrStr",
        {
            "c":"arrMapStr"
        }
    ]
}
```

Call `Connection.newDocument()`, passing in a JSON string with escaped quotes:

```
Document pojoDoc = connection.newDocument(
    "{\"_id\":\"id001\",\"a\":1,\"b\":\"aString\",\"array\":[1,2,\"arrStr\",
{\"c\":\"arrMapStr\"}]}");
```

Create a Document from a JavaBean

You can create a new JSON document in your Java OJAI client by passing a JavaBean to the `Connection.newDocument(Object bean)` method. Through an example, the content shows you a sample JavaBean class, how to create a bean for that class, how to create a JSON document from the bean, and how to convert a JSON document back to a bean.

Sample JavaBean Class

Suppose that you are using a JavaBean class named `ExampleJson`:

```
package com.example;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import javax.annotation.Generated;
```

```
import com.fasterxml.jackson.annotation.JsonAnyGetter;
import com.fasterxml.jackson.annotation.JsonAnySetter;
import com.fasterxml.jackson.annotation.JsonIgnore;
import com.fasterxml.jackson.annotation.JsonInclude;
import com.fasterxml.jackson.annotation.JsonProperty;
import com.fasterxml.jackson.annotation.JsonPropertyOrder;

@JsonInclude(JsonInclude.Include.NON_NULL)
@Generated("org.jsonschema2pojo")
@JsonPropertyOrder({
    "a",
    "b",
    "array"
})

public class ExampleJson {

    @JsonProperty("a")
    private Double a;
    @JsonProperty("b")
    private String b;
    @JsonProperty("array")
    private List<Object> array = new ArrayList<Double>();
    @JsonIgnore
    private Map<String, Object> additionalProperties = new HashMap<String,
Object>();

    /**
      *
      * @return
      * The a
      */
    @JsonProperty("a")
    public Double getA() {
        return a;
    }

    /**
      *
      * @param a
      * The a
      */
    @JsonProperty("a")
    public void setA(Double a) {
        this.a = a;
    }

    /**
      *
      * @return
      * The b
      */
    @JsonProperty("b")
    public String getB() {
        return b;
    }

    /**
      *
      * @param b
      * The b
      */
    @JsonProperty("b")
    public void setB(String b) {
```

```
        this.b = b;
    }

    /**
      *
      * @return
      * The array
      */
    @JsonProperty("array")
    public List<Object> getArray() {
        return array;
    }

    /**
      *
      * @param array
      * The array
      */
    @JsonProperty("array")
    public void setArray(List<Object> array) {
        this.array = array;
    }

    @JsonAnyGetter
    public Map<String, Object> getAdditionalProperties() {
        return this.additionalProperties;
    }

    @JsonAnySetter
    public void setAdditionalProperty(String name, Object value) {
        this.additionalProperties.put(name, value);
    }
}
```

Create a Bean

You can create a bean for the `ExampleJson` class with the following code:

```
ExampleJson bean = new ExampleJson();

bean.setA(1);
bean.setB("aString");

List arrList = new ArrayList();
arrList.add(1);
arrList.add(2);
arrList.add("arrStr");

Map arrMap = new HashMap();
arrMap.put("c","arrMapStr");
arrList.add(arrMap);
bean.setArray(arrList);
```

Create a New Document from a Bean

After creating the `ExampleJson` bean, you can create a JSON document using the bean with the following call:

```
Document pojoDoc = connection.newDocument(bean);
```

The document will have the following structure:

```
{
    "a":1,
    "b":"aString",
    "array":[
        1,
        2,
        "arrStr",
        {
            "c":"arrMapStr"
        }
    ]
}
```

Create a JavaBean from a JSON Document

You can also create a JavaBean from a JSON document. For example, suppose you modify the document that you created earlier:

```
pojoDoc.set("d","10");
```

The following converts the modified document back into an `ExampleJson` bean:

```
ExampleJson bean = pojoDoc.toJavaBean(ExampleJson.class);
```

Creating JSON Documents in Python OJAI
There are two ways to create JSON documents in your Python OJAI application, one of which is the preferred approach.

The preferred way to create a Python dictionary object and then pass it to the `Connection.new_document()` method:

```
json_dict = {
    "_id" : "movie0000001",
    "title" : "OJAI -- The Documentary",
    "studio" : "MapR Technologies, Inc.",
    "release_date" : "2015-09-29",
    "trailers" : {
        "teaser" : "https://10.10.21.90/trailers/teaser",
        "theatrical" : "https://10.10.21.90/trailers/theatrical"
    },
    "characters" : [
        "Heroic Developer",
        "Evil Release Manager",
        "Mad Development Manager"
    ],
    "box_office_gross" : 1000000000L
}
new_document = connection.new_document(dictionary=json_dict)
```

Alternatively, you can call `Document` interface methods to set fields and values:

```
doc = connection.new_document()
        .set_id("movie0000001")
        .set('title', 'OJAI - The Documentary')
        .set('studio', 'MapR Technologies, Inc.')
        .set('release_date', ODate.parse(date_str='2015-09-29'))
        .set('trailers.teaser', 'https://10.10.21.90/trailers/teaser')
        .set('trailers.theatrical', 'https://10.10.21.90/trailers/
theatrical')
```

```
        .set('characters', ['Heroic Developer', 'Evil Release Manager',
'Mad Development Manager'])
        .set('box_office_gross', 1000000000)
```

See the following for more details about the APIs:

- Connection
- Document

Sample OJAI Code for Creating JSON Documents
The sample code in this section shows you how to create a JSON document.

Java

The code is available at OJAI_001_GetConnectionCreateDocument.java.

```
/**
 * Copyright (c) 2017 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.mapr.ojai.examples;

import org.ojai.Document;
import org.ojai.json.JsonOptions;
import org.ojai.store.Connection;
import org.ojai.store.DriverManager;

import com.mapr.ojai.examples.data.Dataset;
import com.mapr.ojai.examples.data.User;

public class OJAI_001_GetConnectionCreateDocument {

  public static void main(String[] args) {

    System.out.println("==== Start Application ===");


    // Create an OJAI connection to MapR cluster
    final Connection connection = DriverManager.getConnection("ojai:mapr:");

    for (final User someUser : Dataset.users) {
      // Create an OJAI Document form the Java bean (there are other ways
too)
      final Document userDocument = connection.newDocument(someUser);

      // Print the OJAI Document
      System.out.println(
          userDocument.asJsonString(           // serialize the OJAI
Document to JSON string
              new JsonOptions().pretty()       // in pretty format
      ));
```

```
    }

    // close the OJAI connection and release any resources held by the
connection
    connection.close();

    System.out.println("==== End Application ===");
  }

}
```

Node.js

The code is available at OJAI_001_GetConnectionCreateDocument.js.

```
/*
 * Copyright (c) 2018 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

const { ConnectionManager } = require('node-maprdb');

const connectionString = 'localhost:5678?' +
  'auth=basic;' +
  'user=mapr;' +
  'password=mapr;' +
  'ssl=true;' +
  'sslCA=/opt/mapr/conf/ssl_truststore.pem;' +
  'sslTargetNameOverride=node1.mapr.com';

// Create a connection to data access server
ConnectionManager.getConnection(connectionString)
  .then((connection) => {
    // create new document as a JavaScript object
    const newDocument = {
      "_id": "id001",
      "name": "Joe",
      "age": 50,
      "address": {
        "street": "555 Moon Way",
        "city": "Gotham"
      }
    };

    // Print the OJAI Document
    console.log(JSON.stringify(newDocument));

    // close the OJAI connection and release any resources held by the
connection
    connection.close();
  });
```

Python

The code is available at 001_get_connection_create_document.py.

```
from mapr.ojai.storage.ConnectionFactory import ConnectionFactory

# Create a connection to data access server
connection_str = "localhost:5678?auth=basic;user=mapr;password=mapr;" \
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/ssl_truststore.pem;" \
          "sslTargetNameOverride=node1.mapr.com"
connection = ConnectionFactory.get_connection(connection_str=connection_str)

# Json string or json dictionary
json_dict = {"_id": "id001",
             "name": "Joe",
             "age": 50,
             "address": {
                 "street": "555 Moon Way",
                 "city": "Gotham"}
             }

# Create new document from json_document
new_document = connection.new_document(dictionary=json_dict)

# Print the OJAI Document
print(new_document.as_json_str())

# close the OJAI connection
connection.close()
```

C#

The code is available at 001_GetConnectionCreateDocument.cs.

```
using System;
using MapRDB.Driver;

public class GetConnectionCreateDocument
{
    public void GetConnectionCreateDocument()
    {
        // Create a connection to data access server
        var connectionStr = $"localhost:5678?auth=basic;" +
            $"user=mapr;" +
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
            $"sslTargetNameOverride=node1.mapr.com";
        var connection = ConnectionFactory.CreateConnection(connectionStr);

        // Json string
        var jsonStr =
            @"{" +
                @""""_id"":""id001""," +
                @""""name"":""Joe""," +
                @""""age"":{""$numberInt"":""50""}," +
                @""""address"":" +
                    @"{" +
                        @""""street"":""555 Moon Way""," +
                        @""""city"":""Gotham""" +
                    @"}" +
            @"}";
```

```
        // Create a document from jsonStr
        var documentJson = connection.NewDocument(jsonStr);

        // Print the OJAI Document
        Console.WriteLine(documentJson.ToJsonString());

        // Create new document with the same fields using constructor
        var documentConstructed = connection.NewDocument()
            .SetID("id001")
            .Set("name", "Joe")
            .Set("age", 50)
            .Set("address.street", "555 Moon Way")
            .Set("address.city", "Gotham");

        // Print the OJAI Document
        Console.WriteLine(documentConstructed.ToJsonString());

        // Close the OJAI connection
        connection.Close();
    }
}
```

Go

The code is available at 001_get_connection_create_document.go.

```
package main

import (
    "fmt"
    client "github.com/mapr/private-maprdb-go-client"
)

func main() {
    // Create connection string
    connectionString := "192.168.33.11:5678?" +
        "auth=basic;" +
        "user=mapr;" +
        "password=mapr;" +
        "ssl=true;" +
        "sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
        "sslTargetNameOverride=node1.cluster.com"

    // Create a connection to data access server
    connection, err := client.MakeConnection(connectionString)
    if err != nil {
        panic(err)
    }

    // Json string or map from which the Document will be created
    newMap := map[string]interface{}{
        "_id":  "id001",
        "name": "Joe",
        "age":  50,
        "address": map[string]interface{}{
            "street": "555 Moon Way",
            "city":   "Gotham",
        },
    }

    // Create new document from json_document
    newDocument := connection.CreateDocumentFromMap(newMap)
```

```
    // Print the new OJAI Document
    fmt.Println(newDocument.AsJsonString())

    // Close connection
    connection.Close()
}
```

Examples: Inserting JSON Documents
This section contains sample code that inserts a JSON document into a MapR Database JSON table. It also shows the MapR Database Shell syntax for inserting documents.

Java

The following code is available at OJAI_002_GetStoreAndInsertDocuments.java.

After you create the JSON document, call the DocumentStore.insertOrReplace method to insert the document into MapR Database.

```
/**
 * Copyright (c) 2017 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.mapr.ojai.examples;

import org.ojai.Document;
import org.ojai.store.Connection;
import org.ojai.store.DocumentStore;
import org.ojai.store.DriverManager;

import com.mapr.ojai.examples.data.Dataset;
import com.mapr.ojai.examples.data.User;

public class OJAI_002_GetStoreAndInsertDocuments {

  public static void main(String[] args) {

    System.out.println("==== Start Application ===");

    // Create an OJAI connection to MapR cluster
    final Connection connection = DriverManager.getConnection("ojai:mapr:");

    // Get an instance of OJAI
    final DocumentStore store = connection.getStore("/demo_table");

    for (final User someUser : Dataset.users) {
      // Create an OJAI Document form the Java bean (there are other ways
too)
      final Document userDocument = connection.newDocument(someUser);

      System.out.println("\t inserting "+ userDocument.getId());
```

```
      // insert the OJAI Document into the DocumentStore
      store.insertOrReplace(userDocument);
    }

    // Close this instance of OJAI DocumentStore
    store.close();

    // close the OJAI connection and release any resources held by the
connection
    connection.close();

    System.out.println("==== End Application ===");
  }

}
```

Node.js

The following code is available at OJAI_002_GetStoreAndInsertDocuments.js.

The following code creates a list of JSON objects and then calls the DocumentStore.insertOrReplace method to insert the document into MapR Database.

```
/*
 * Copyright (c) 2018 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

const { ConnectionManager } = require('node-maprdb');

const connectionString = 'localhost:5678?' +
  'auth=basic;' +
  'user=mapr;' +
  'password=mapr;' +
  'ssl=true;' +
  'sslCA=/opt/mapr/conf/ssl_truststore.pem;' +
  'sslTargetNameOverride=node1.mapr.com';

let connection;

// Create a connection to data access server
ConnectionManager.getConnection(connectionString)
  .then((conn) => {
    connection = conn;
    // Get a store
    return connection.getStore('/demo_table');
  })
  .then((store) => {
    const documentList = [{'_id': 'user0000',
      'age': 35,
      'firstName': 'John',
      'lastName': 'Doe',
```

```
      'address': {
        'street': '350 Hoger Way',
        'city': 'San Jose',
        'state': 'CA',
        'zipCode': 95134
      },
      'phoneNumbers': [
        {'areaCode': 555, 'number': 5555555},
        {'areaCode': '555', 'number': '555-5556'}]
    },
      {'_id': 'user0001',
        'age': 26,
        'firstName': 'Jane',
        'lastName': 'Dupont',
        'address': {
          'street': '320 Blossom Hill Road',
          'city': 'San Jose',
          'state': 'CA',
          'zipCode': 95196
        },
        'phoneNumbers': [
          {'areaCode': 555, 'number': 5553827},
          {'areaCode': '555', 'number': '555-6289'}]
      },
      {'_id': 'user0002',
        'age': 45,
        'firstName': 'Simon',
        'lastName': 'Davis',
        'address': {
          'street': '38 De Mattei Court',
          'city': 'San Jose',
          'state': 'CA',
          'zipCode': 95142
        },
        'phoneNumbers': [
          {'areaCode': 555, 'number': 5425639},
          {'areaCode': '555', 'number': '542-5656'}]
      }
    ];
    const promiseList = documentList.map((doc) => {
      // Print the OJAI Document
      console.log(JSON.stringify(doc));
      // Insert the OJAI Document into the DocumentStore
      return store.insertOrReplace(doc);
    });
    return Promise.all(promiseList);
  })
  .then(() => {
    // close the OJAI connection
    connection.close();
  });
```

Python

The following code is available at 002_get_store_and_insert_documents.py.

The following code creates a list of JSON dictionary objects, creates Document objects, and calls the DocumentStore.insert_or_replace method to insert the documents into MapR Database.

```
from mapr.ojai.storage.ConnectionFactory import ConnectionFactory

# Create a connection to data access server
connection_str = "localhost:5678?auth=basic;user=mapr;password=mapr;" \
```

```
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/ssl_truststore.pem;" \
          "sslTargetNameOverride=node1.mapr.com"
connection = ConnectionFactory.get_connection(connection_str=connection_str)

# Get a store and assign it as a DocumentStore object
if connection.is_store_exists('/demo_table'):
    store = connection.get_store('/demo_table')
else:
    store = connection.create_store('/demo_table')

document_list = [{'_id': 'user0000',
                  'age': 35,
                  'firstName': 'John',
                  'lastName': 'Doe',
                  'address': {
                      'street': '350 Hoger Way',
                      'city': 'San Jose',
                      'state': 'CA',
                      'zipCode': 95134
                  },
                  'phoneNumbers': [
                      {'areaCode': 555, 'number': 5555555},
                      {'areaCode': '555', 'number': '555-5556'}]
                  },
                 {'_id': 'user0001',
                  'age': 26,
                  'firstName': 'Jane',
                  'lastName': 'Dupont',
                  'address': {
                      'street': '320 Blossom Hill Road',
                      'city': 'San Jose',
                      'state': 'CA',
                      'zipCode': 95196
                  },
                  'phoneNumbers': [
                      {'areaCode': 555, 'number': 5553827},
                      {'areaCode': '555', 'number': '555-6289'}]
                  },
                 {'_id': 'user0002',
                  'age': 45,
                  'firstName': 'Simon',
                  'lastName': 'Davis',
                  'address': {
                      'street': '38 De Mattei Court',
                      'city': 'San Jose',
                      'state': 'CA',
                      'zipCode': 95142
                  },
                  'phoneNumbers': [
                      {'areaCode': 555, 'number': 5425639},
                      {'areaCode': '555', 'number': '542-5656'}]
                  }
                 ]

for doc_dict in document_list:
    # Create new document from json_document
    new_document = connection.new_document(dictionary=doc_dict)
    # Print the OJAI Document
    print(new_document.as_json_str())

    # Insert the OJAI Document into the DocumentStore
    store.insert_or_replace(new_document)
```

```
# close the OJAI connection
connection.close()
```

dbshell

The following shows the syntax to insert a document with MapR Database Shell. See dbshell insert on page 5302 for more information and examples.

```
# mapr dbshell
maprdb root:>

// Syntax for inserting a document using the document ID
maprdb root:> insert <table path> --value '{"_id": "<row-key", < table
field >}'

// Syntax for inserting a document using document value
maprdb root:> insert <table path> --id <row-key> --value '{"_id":
"<row-key", < table field >}'
```

C#

The following code is available at 002_GetStoreAndInsertDocuments.cs.

The following code creates a list of JSON strings, creates Documents from the list, and calls the `DocumentStore.InsertOrReplace` method to insert the documents into the MapR Database.

```
using System;
using MapRDB.Driver;
using System.Collections.Generic;

public class GetStoreAndInsertDocuments
{
    public void GetStoreAndInsertDocuments()
    {
        // Create a connection to data access server
        var connectionStr = $"localhost:5678?auth=basic;" +
            $"user=mapr;" +
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
            $"sslTargetNameOverride=node1.mapr.com";
        var connection = ConnectionFactory.CreateConnection(connectionStr);

        // Get a store and assign it as a DocumentStore object
        if (!connection.StoreExist("/demo_table"))
            connection.CreateStore("/demo_table");
        var store = connection.GetStore("/demo_table");

        var documentList = new List<string>
            {
                @"{""_id"":""user0000"","" +
                @""age"":{""$numberInt"":""35""},"" +
                @""firstName"":""John"","" +
                @""lastName"":""Doe"","" +
                @""address"":{"" +
                    @""street"":""350 Hoger Way"","" +
                    @""city"":""San Jose"","" +
                    @""state"":""CA"","" +
                    @""zipCode"":{""$numberLong"":""95134""}"" +
                    @"},"" +
                @""phoneNumbers"":["" +
                    @"{""areaCode"":{""$numberInt"":""555""},""number"":
```

```
{""$numberLong"":""5555555""}}," +
                    @"{""areaCode"":""555"",""number"":""555-5556""}]" +
                @"}",
                @"{""_id"":""user0001""," +
                @"""age"":{""$numberInt"":""26""}," +
                @"""firstName"":""Jane""," +
                @"""lastName"":""Dupont""," +
                @"""address"":{" +
                    @"""street"":""320 Blossom Hill Road""," +
                    @"""city"":""San Jose""," +
                    @"""state"":""CA""," +
                    @"""zipCode"":{""$numberLong"":""95196""}" +
                    @"}," +
                @"""phoneNumbers"":[" +
                    @"{""areaCode"":{""$numberInt"":""555""},""number"":
{""$numberLong"":""5553827""}}," +
                    @"{""areaCode"":""555"",""number"":""555-6289""}]" +
                @"}",
                @"{""_id"":""user0002""," +
                @"""age"":{""$numberInt"":""45""}," +
                @"""firstName"":""Simon""," +
                @"""lastName"":""Davis""," +
                @"""address"":{" +
                    @"""street"":""38 De Mattei Court""," +
                    @"""city"":""San Jose""," +
                    @"""state"":""CA""," +
                    @"""zipCode"":{""$numberLong"":""95142""}" +
                    @"}," +
                @"""phoneNumbers"":[" +
                    @"{""areaCode"":{""$numberInt"":""555""},""number"":
{""$numberLong"":""5425639""}}," +
                    @"{""areaCode"":""555"",""number"":""542-5656""}]" +
                @"}"
            };

        foreach (var doc in documentList)
        {
            // Create new document from json string
            var document = connection.NewDocument(doc);

            // Print the OJAI Document
            Console.WriteLine(document.ToJsonString());

            // Insert the OJAI Document into the DocumentStore
            store.InsertOrReplace(document);
        }

        // Close the OJAI connection
        connection.Close();
    }
}
```

Go

The following code is available at 002_get_store_and_insert_documents.go.

The following code creates a list of JSON dictionary objects, creates Document objects, and calls the DocumentStore.InsertOrReplaceDocument function to insert the documents into MapR Database.

```
package main

import (
    "fmt"
```


last-updated: Apr 11, 2022

```
    client "github.com/mapr/private-maprdb-go-client"
)

func main() {
    // Create connection string
    connectionString := "192.168.33.11:5678?" +
        "auth=basic;" +
        "user=mapr;" +
        "password=mapr;" +
        "ssl=true;" +
        "sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
        "sslTargetNameOverride=node1.cluster.com"

    storeName := "/demo_table"

    // Create a connection to DAG
    connection, err := client.MakeConnection(connectionString)
    if err != nil {
        panic(err)
    }

    // Get a store and assign it as a DocumentStore struct
    isExists, err := connection.IsStoreExists(storeName)
    if err != nil {
        panic(err)
    }
    var store *client.DocumentStore
    if isExists {
        store, err = connection.GetStore(storeName)
        if err != nil {
            panic(err)
        }
    } else {
        store, err = connection.CreateStore(storeName)
        if err != nil {
            panic(err)
        }
    }

    // Slice of maps from which the Document will be created
    documentArray := []map[string]interface{}{
        {
            "_id":       "user0000",
            "age":       35,
            "firstName": "John",
            "lastName":  "Doe",
            "address": map[string]interface{}{
                "street":  "350 Hoger Way",
                "city":    "San Jose",
                "state":   "CA",
                "zipCode": 95134,
            },
            "phoneNumbers": []interface{}{
                map[string]interface{}{"areaCode": 555, "number": 5555555},
                map[string]interface{}{"areaCode": "555", "number":
"555-5556"},
            },
        },
        {
            "_id":       "user0001",
            "age":       26,
            "firstName": "Jane",
            "lastName":  "Dupont",
            "address": map[string]interface{}{
```

```
                "street":  "320 Blossom Hill Road",
                "city":    "San Jose",
                "state":   "CA",
                "zipCode": 95196,
            },
            "phoneNumbers": []interface{}{
                map[string]interface{}{"areaCode": 555, "number": 5553827},
                map[string]interface{}{"areaCode": "555", "number":
"555-6289"},
            },
        },
        {
            "_id":       "user0002",
            "age":       45,
            "firstName": "Simon",
            "lastName":  "Davis",
            "address": map[string]interface{}{
                "street":  "38 De Mattei Court",
                "city":    "San Jose",
                "state":   "CA",
                "zipCode": 95142,
            },
            "phoneNumbers": []interface{}{
                map[string]interface{}{"areaCode": 555, "number": 5425639},
                map[string]interface{}{"areaCode": "555", "number":
"542-5656"},
            },
        },
    }

    for _, docMap := range documentArray {
        // Create new document from json_document
        newDocument := connection.CreateDocumentFromMap(docMap)
        // Print the new OJAI Document
        fmt.Println(newDocument.AsJsonString())
        //Insert the OJAI Document into the DocumentStore
        store.InsertOrReplaceDocument(newDocument)
    }

    // Close connection
    connection.Close()
}
```

Using OJAI Mutation Syntax

To perform updates using OJAI, you specify the document you want to update using its `_id` field, create *mutations* for that document, and then update it in your document store. OJAI defines a syntax for specifying mutations. Mutations allow you to append, decrement, delete, increment, combine, replace, and update fields in a document. This topic describes the syntax for the supported mutation operations and provides examples.

The following table lists the mutations OJAI supports. Each entry in the table contains a brief description of the mutation and a link to a section in this topic that describes the mutation in more detail.

Mutation Operation	Description
Append	Appends values to binary, string, and array fields
Decrement	Decrements field values
Delete	Deletes fields
Increment	Increments field values
Merge	Combines a nested document with an existing document

last-updated: Apr 11, 2022

Mutation Operation	Description
Put	Replaces field values or adds new fields
Set	Updates field values or adds new fields

The examples in this topic use the following sample JSON document:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : false }, { "decimal" : 123.456 } ],
    "c" : {
      "d" : 10,
      "e" : "Hello"
    }
  },
  "m" : "MapR wins"
}
```

OJAI Append Mutations

Syntax

```
{"$append":{"fieldpath":value}}
```

```
{"$append":[{"fieldpath1":value1},
{"fieldpath2":value2},...]}
```

Description

The `$append` mutation is a read-modify-write operation. Use it to append specified values to existing binary, string, or array type fields. If there is type mismatch in any intermediate field specified in a *fieldpath* for the document, the mutation fails with an error. For example, an append mutation on field path `a.b.c` fails if the field `a` is a scalar.

To append multiple field paths, use an array notation to list the field paths.

Example

The following mutation appends an element to the array `a.b` and appends the string `" MapR"` to the end of the string already in the field path `a.c.e`:

```
{"$append":[{"a.b":{"appd":1}},
{"a.c.e":" MapR"}]}
```

The mutation results in the following document, with the field updates highlighted in bold:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : false },
{ "decimal" : 123.456 }, { "appd" :
1 } ],
    "c" : {
      "d" : 10,
      "e" : "Hello MapR"
    }
  },
```

```
  "m" : "MapR wins"
}
```

OJAI Decrement Mutations

Syntax

```
{"$decrement":"fieldpath"}
```

```
{"$decrement":
{"fieldpath":decrementValue}}
```

```
{"$decrement":
[{"fieldpath1":decrementValue1},
{"fieldpath2":decrementValue2},...]}
```

Description

The `$decrement` mutation decrements the value in the *fieldpath*. To decrement multiple field paths, use an array notation to list the field paths.

If the *fieldpath* does not exist, the mutation adds a new field to the document with the value *decrementValue*.

The *decrementValue* is optional and defaults to -1.

The mutation fails if there is a type mismatch in the field.

Example

The following updates the value 10 in `a.c.d` to 5 by using the decrement mutation:

```
{"$decrement":{"a.c.d":5}}
```

The mutation results in the following document, with the field update highlighted in bold:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : false },
{ "decimal" : 123.456 } ],
    "c" : {
      "d" : 5,
      "e" : "Hello"
    }
  },
  "m" : "MapR wins"
}
```

OJAI Delete Mutations

Syntax

```
{"$delete":"fieldpath"}
```

```
{"$delete":
["fieldpath1","fieldpath2",...]}
```


last-updated: Apr 11, 2022

Description

The `$delete` mutation removes either a single field or a list of fields from a document. If the field does not exist, the delete ignores that field.

Example

The following mutation removes two fields from the document:

```
{"$delete":["a.b[1]","a.c.e"]}
```

The mutation results in the following document:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : false } ],
    "c" : {
      "d" : 10
    }
  },
  "m" : "MapR wins"
}
```

OJAI Increment Mutations

Syntax

```
{"$increment":"fieldpath"}
```

```
{"$increment":
{"fieldpath":incrementValue}}
```

```
{"$increment":
[{"fieldpath1":incrementValue1},
{"fieldpath2":decrementValue2},...]}
```

Description

The `$increment` mutation increments the value in the *fieldpath*. To increment multiple field paths, use an array notation to list the field paths.

If the *fieldpath* does not exist, the mutation adds a new field to the document with the value *incrementValue*.

The *incrementValue* is optional and defaults to 1.

The mutation fails if there is a type mismatch in the field.

Example

The following updates the value 10 in `a.c.d` to 15 by using the increment mutation:

```
{"$increment":{"a.c.d":5}}
```

The mutation results in the following document, with the field update highlighted in bold:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : false },
{ "decimal" : 123.456 } ],
    "c" : {
```


last-updated: Apr 11, 2022

```
      "d" : 15,
      "e" : "Hello"
    }
  },
  "m" : "MapR wins"
}
```

OJAI Merge Mutations

Syntax

```
{"$merge":
{"fieldpath":nestedDocument}}
```

Description

The `$merge` mutation combines a *nestedDocument* with an existing document at a specified *fieldpath*. If the original document already contains subfields specified in the *nestedDocument*, then the mutation replaces the values for those subfields. Otherwise, it adds new subfields to the document.

Note: The `$merge` mutation does not support the array notation that other mutation operations provide.

To specify more than one merge operation in a single mutation, use the syntax described at either Specifying Multiple Mutation Operations on page 2567 or OJAI Mutations Without Explicit Mutation Operation Names on page 2569. When using these syntax variations, avoid specifying overlapping field paths. MapR Database treats these as conflicting mutations and discards conflicts.

Examples

The following mutation replaces the pre-existing field path `a.c.d` with the value 11. It adds a new subfield y to the nested document `a.c`.

```
{"$merge":{"a.c":{"d":11,"y":"yo"}}}
```

The mutation results in the following document, with the field updates highlighted in bold:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : false },
{ "decimal" : 123.456 } ],
    "c" : {
      "d" : 11,
      "e" : "Hello",
      "y" : "yo"
    }
  },
  "m" : "MapR wins"
}
```

The following mutation replaces the value in the field path `a.b` and adds a new subfield `a.d`:

```
{"$merge":{"a":{"b":1,"d":"MapR"}}}
```


last-updated: Apr 11, 2022

The mutation results in the following document, with the field updates highlighted in bold:

```
{
  "_id" : "id1",
  "a" : {
    "b" : 1,
    "c" : {
      "d" : 10,
      "e" : "Hello"
    },
    "d" : "MapR"
  },
  "m" : "MapR wins"
}
```

OJAI Put Mutations

Syntax

```
{"$put":{"fieldpath":value}}
```

```
{"$put":[{"fieldpath1":value1},
{"fieldpath2":value2},...]}
```

Description

The `$put` mutation is a replace operation. It is not a read-modify-write operation; it does no validation on the data. If the specified *fieldpath* exists, the mutation replaces the *fieldpath*'s value with the new *value,* regardless of the type of the original value. For example, you can update a field `a.b` from an array to a nested document. If the *fieldpath* does not exist, the mutation creates a new field with the given *value*. Because the operation does no data validation, it is significantly faster than the set operation.

To replace multiple field paths, use an array notation to list the field paths.

Example

The following example replaces the pre-existing fields `a.b` and `a.c.d` with values whose types differ from the original types. It also adds a new field `a.x`.

```
{"$put":[{"a.b":{"boolean":true}},
{"a.c.d":"eureka"},{"a.x":1}]}
```

The mutation results in the following document, with the field updates highlighted in bold:

```
{
  "_id" : "id1",
  "a" : {
    "b" : { "boolean" : true },
    "c" : {
      "d" : "eureka",
      "e" : "Hello"
    },
    "x" : 1
  },
  "m" : "MapR wins"
}
```

The mutation behaves as follows for the pre-existing fields:

- For `a.b`, the mutation replaces the original array of nested documents with a single nested document.
- For `a.c.d`, the mutation changes the field from an integer to a string.

OJAI Set Mutations

Syntax

```
{"$set":{"fieldpath":value}}
```

```
{"$set":[{"fieldpath1":value1},
{"fieldpath2":value2},...]}
```

Description

The `$set` mutation updates one or more fields in a document. It is a read-modify-write operation. It validates the type of the existing value before applying the mutation. If the specified *fieldpath* does not exist in a document, the mutation creates a new field. If the *fieldpath* exists but is not of the same type as the type of new *value*, then the entire mutation fails.

To update multiple field paths, use an array notation to list the field paths.

Example

The following example updates the pre-existing fields `a.b.[0]` and `a.c.d`. It also adds a new field `a.x`.

```
{"$set":[{"a.b[0].boolean":true},
{"a.c.d":11},{"a.x":1}]}
```

The mutation results in the following document, with the field updates highlighted in bold:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : true },
{ "decimal" : 123.456 } ],
    "c" : {
      "d" : 11,
      "e" : "Hello"
    },
    "x" : 1
  },
  "m" : "MapR wins"
}
```

Specifying Multiple Mutation Operations

You can specify more than one operation in a single mutation by specifying each operation separated by a comma.

The following is a mutation with six operations:

```
{
   "$set":{"x":[1,2,3]},
```

```
   "$put":{"a.c.e":{"$binary":"AAAADg=="}},
   "$increment":"a.b[1].decimal",
   "$delete":"a.b[0]",
   "$merge":{"newDoc":{"k":"MapR DBShell rocks!!"}},
   "$append":{"m":"!!!"}
}
```

It results in the following document, with the field updates highlighted in bold:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "decimal" : 124.456 } ],
    "c" : {
      "d" : 10,
      "e" : { "$binary" : "AAAADg==" }
    }
  },
  "m" : "MapR wins!!!",
  "newDoc" : { "k" : "MapR DBShell rocks!!" },
  "x" : [ 1, 2, 3 ]
}
```

The mutation applies the updates in the following manner:

- The `$set` mutation adds a new array field `x` with the value `[1, 2, 3]`.
- The `$put` mutation replaces the string `"Hello"` with the nested document `{"$binary":"AAAADg=="}`.
- The `$increment` mutation increments the value `123.456` in the second element of the array `a.b`.
- The `$delete` mutation deletes the field path `a.b[0]`, resulting in a single element array `a.b`.
- The `$merge` mutation adds a new field `newDoc` with the nested document `{"k":"MapR DBShell rocks!!"}` as its value.
- The `$append` mutation appends the string `"!!!"` to the end of the string `"MapR wins"`.

Conflicting Mutations

When you specify a mutation with field paths that are overlapping, MapR Database detects the conflict, discards the previous conflicting operation, and proceeds with the next operation.

For example, suppose you have the following document:

```
{"_id":"id1", "a":{"b":{"c":5}}}
```

The following mutation has two operations with overlapping fields `a.b`:

```
{"$delete":"a.b","$set":{"a.b.d":10}}
```

You may have intended for the mutation to first delete `a.b` and then to replace it with `a.b.d` as follows:

```
{"_id":"id1", "a":{"b":{"d":"10"}}}
```

But the *actual* result is the following:

```
{"_id":"id1", "a":{"b":{"c":5,"d":"10"}}}
```

In this case, the set operation on **`a.b.d`** causes the delete operation on **`a.b`** to be discarded.

Note: In the earlier example in this section, the `$increment` and `$delete` operations are not conflicting because one operates on `a.b[1]`, while the other operates on `a.b[0]`. On the other hand, the following are conflicting operations:

```
{"$increment":"a.b[1].decimal","$delete":"a.b"}
```

OJAI Mutations Without Explicit Mutation Operation Names

You can specify a mutation without using an explicit mutation name. These mutations run as merge operations.

For example, the following mutation merges the fields `k` and `a.c.d` to the document by adding a new field `k` and updating `a.c.d`:

```
{"k":"eureka","a":{"c":{"d":1234}}}
```

The mutation results in the following document, with updates highlighted in bold:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : false }, { "decimal" : 123.456 } ],
    "c" : {
      "d" : 1234,
      "e" : "Hello"
    }
  },
  "k" : "eureka",
  "m" : "MapR wins"
}
```

Examples: Updating JSON Documents

This section contains sample code that updates a JSON document in a MapR Database JSON table using an OJAI mutation. It also shows the MapR Database Shell syntax for updating documents.

See Using OJAI Mutation Syntax on page 2561 for more details about OJAI mutations.

Java

The following code is available at OJAI_012_UpdateDocument.java. It does the following:

- Finds a document using the DocumentStore.findById method
- Creates a DocumentMutation that updates a field
- Updates the document by calling the DocumentStore.update method

```
/**
 * Copyright (c) 2017 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
```

```
 * limitations under the License.
 */
package com.mapr.ojai.examples;

import org.ojai.store.Connection;
import org.ojai.store.DocumentMutation;
import org.ojai.store.DocumentStore;
import org.ojai.store.DriverManager;

public class OJAI_012_UpdateDocument {

  public static void main(String[] args) {

    System.out.println("==== Start Application ===");


    // Create an OJAI connection to MapR cluster
    final Connection connection = DriverManager.getConnection("ojai:mapr:");

    // Get an instance of OJAI DocumentStore
    final DocumentStore store = connection.getStore("/demo_table");

    String docId = "user0002";

    // Print the document before update
    System.out.println( "\t"+
store.findById(docId).getMap("address").toString() );



    // Create a DocumentMutation to update the zipCode field
    DocumentMutation mutation = connection.newMutation()
        .set("address.zipCode", 95196L);


    System.out.println("\tUpdating document "+ docId);

    // Update the Document with '_id' = "user0002"
    store.update(docId, mutation);

    // Print the document after update
    System.out.println( "\t"+
store.findById(docId).getMap("address").toString() );


    // Close this instance of OJAI DocumentStore
    store.close();

    // close the OJAI connection and release any resources held by the
connection
    connection.close();

    System.out.println("==== End Application ===");
  }

}
```

Node.js - Update

The following code is available at OJAI_011_UpdateDocument.js. It does the following:

- Finds a document using the DocumentStore.findById method

- Creates an OJAI mutation that updates a field
- Updates the document by calling the DocumentStore.update method

```
/*
 * Copyright (c) 2018 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

const { ConnectionManager } = require('node-maprdb');

const connectionString = 'localhost:5678?' +
  'auth=basic;' +
  'user=mapr;' +
  'password=mapr;' +
  'ssl=true;' +
  'sslCA=/opt/mapr/conf/ssl_truststore.pem;' +
  'sslTargetNameOverride=node1.mapr.com';

let connection;
let store;
const docId = 'user0002';

// Create a connection to data access server
ConnectionManager.getConnection(connectionString)
  .then((conn) => {
    connection = conn;
    // Get a store
    return connection.getStore('/demo_table');
  })
  .then((newStore) => {
    // Get a store and assign it as a DocumentStore object
    store = newStore;
    // Find the document before update
    return store.findById(docId);
  })
  .then((docBeforeUpdate) => {
    // Print the document before update
    console.log(`Document with id ${docId} before update`)
    console.log(docBeforeUpdate);

    const mutation = {'$put': {'address.zipCode': 95196}};
    return store.update(docId, mutation);
  })
  .then(() => {
    // Find the document after update
    return store.findById(docId);
  })
  .then((docAfterUpdate) => {
    // Print the document after update
    console.log(`Document with id ${docId} before update`)
```

```
    console.log(docAfterUpdate);
  });
```

Node.js - Check and Update

The following code is available at OJAI_012_CheckAndUpdateDocument.js. It does the following:

- Finds a document using the DocumentStore.findById method
- Creates an OJAI mutation that updates a field
- Creates an OJAI condition to apply in the check and update
- Performs the check and update on the document by calling the DocumentStore.checkAndUpdate method

```
/*
 * Copyright (c) 2018 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

const { ConnectionManager } = require('node-maprdb');

const connectionString = 'localhost:5678?' +
  'auth=basic;' +
  'user=mapr;' +
  'password=mapr;' +
  'ssl=true;' +
  'sslCA=/opt/mapr/conf/ssl_truststore.pem;' +
  'sslTargetNameOverride=node1.mapr.com';

let connection;
let store;
const docId = 'user0002';

// Create a connection to data access server
ConnectionManager.getConnection(connectionString)
  .then((conn) => {
    connection = conn;
    // Get a store
    return connection.getStore('/demo_table');
  })
  .then((newStore) => {
    // Get a store and assign it as a DocumentStore object
    store = newStore;
    // Find the document before update
    return store.findById(docId);
  })
  .then((docBeforeUpdate) => {
    // Print the document before update
    console.log(`Document with id ${docId} before update`)
```

```
    console.log(docBeforeUpdate);

    const mutation = {'$put': {'address.zipCode': 95196}};
    const condition = {'$eq': {'address.street': '320 Blossom Hill Road'}}
    return store.checkAndUpdate(docId, mutation, condition);
  })
  .then((updateResult) => {
    console.log(updateResult);
    // Find the document after update
    return store.findById(docId);
  })
  .then((docAfterUpdate) => {
    // Print the document after update
    console.log(`Document with id ${docId} before update`)
    console.log(docAfterUpdate);
  });
```

Python - Update

The following code is available at 012_update_document.py. It does the following:

- Finds a document using the DocumentStore.find_by_id method
- Creates an OJAI mutation that updates a field
- Updates the document by calling the DocumentStore.update method

```
from mapr.ojai.storage.ConnectionFactory import ConnectionFactory

# Create a connection to data access server
connection_str = "localhost:5678?auth=basic;user=mapr;password=mapr;" \
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/ssl_truststore.pem;" \
          "sslTargetNameOverride=node1.mapr.com"
connection = ConnectionFactory.get_connection(connection_str=connection_str)

# Get a store and assign it as a DocumentStore object
store = connection.get_store('/demo_table')

doc_id = 'user0002'

# Print the document before update
document_before_update = store.find_by_id(doc_id)
print("Document with id {0} before update".format(doc_id))
print(document_before_update)

# Create mutation to update the zipCode field
mutation = {'$set': {'address.zipCode': 95196}}

# Execute update
store.update(_id=doc_id, mutation=mutation)

document_after_update = store.find_by_id(doc_id)
print('Document with id {0} after update'.format(doc_id))
print(document_after_update)
```

Python - Check and Update

The following code is available at 013_check_and_update_document.py. It does the following:

- Finds a document using the DocumentStore.find_by_id method

- Creates an OJAI mutation that updates a field
- Creates an OJAI condition to apply in the check and update
- Performs the check and update on the document by calling the DocumentStore.check_and_update method

```
from mapr.ojai.storage.ConnectionFactory import ConnectionFactory

# Create a connection to data access server
connection_str = "localhost:5678?auth=basic;user=mapr;password=mapr;" \
                 "ssl=true;" \
                 "sslCA=/opt/mapr/conf/ssl_truststore.pem;" \
                 "sslTargetNameOverride=node1.mapr.com"
connection = ConnectionFactory.get_connection(connection_str=connection_str)

# Get a store and assign it as a DocumentStore object
store = connection.get_store('/demo_table')

doc_id = 'user0001'

# Print the document before update
document_before_update = store.find_by_id(doc_id)
print("Document with id {0} before update".format(doc_id))
print(document_before_update)

# Create mutation to update the zipCode field
mutation = {'$put': {'address.zipCode': 99999}}

# Create condition
condition = {'$eq': {'address.street': '320 Blossom Hill Road'}}

# Execute check_and_update.
# Returns True if condition True and document was updated.
update_result = store.check_and_update(_id=doc_id,
                                       mutation=mutation,
                                       query_condition=condition)

print(update_result)

document_after_update = store.find_by_id(doc_id)
print('Document with id {0} after update'.format(doc_id))
print(document_after_update)
```

dbshell

The following dbshell command is equivalent to the code examples. See dbshell update on page 5305 for more information and examples.

```
# mapr dbshell
maprdb root:> update /demo_table --id user002 --m {"$set":
{"address.zipCode":95196}}
```

C# - Update

The following code is available at 012_UpdateDocument.cs. It does the following:

- Finds a document using the `DocumentStore.FindById` method to print the document before update.
- Creates an OJAI mutation that updates a field.

- Updates the document by calling the `DocumentStore.Update` method.

```
using System;
using MapRDB.Driver;

public class UpdateDocument
{
    public void UpdateDocument()
    {
        // Create a connection to data access server
        var connectionStr = $"localhost:5678?auth=basic;" +
            $"user=mapr;" +
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
            $"sslTargetNameOverride=node1.mapr.com";
        var connection = ConnectionFactory.CreateConnection(connectionStr);

        // Get a store and assign it as a DocumentStore object
        var store = connection.GetStore("/demo_table");

        var docId = "user0002";

        // Print the document before update
        var documentBeforeUpdate = store.FindById(docId);
        Console.WriteLine($"Document with id {docId} before update:");
        Console.WriteLine(documentBeforeUpdate);

        // Create mutation to update the zipCode field
        var mutation =
connection.NewDocumentMutation().Set("address.zipCode", (long)95196);

        // Execute update
        store.Update(docId, mutation);

        // Print the document after update
        var documentAfterUpdate = store.FindById(docId);
        Console.WriteLine($"Document with id {docId} after update:");
        Console.WriteLine(documentAfterUpdate);

        // Close the OJAI connection
        connection.Close();
    }
}
```

C# - Check and Update

The following code is available at 013_CheckAndUpdateDocument.cs. It does the following:

- Finds a document using the `DocumentStore.FindById` method to print the document before update.
- Creates an OJAI mutation that updates a field.
- Creates an OJAI condition to apply in the check and update.
- Performs the check and update on the document by calling the `DocumentStore.CheckAndUpdate` method.

```
using System;
using MapRDB.Driver;
using MapRDB.Driver.Ojai;
```

```
public class CheckAndUpdateDocument
{
    public void CheckAndUpdateDocument()
    {
        // Create a connection to data access server
        var connectionStr = $"localhost:5678?auth=basic;" +
            $"user=mapr;" +
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
            $"sslTargetNameOverride=node1.mapr.com";
        var connection = ConnectionFactory.CreateConnection(connectionStr);

        // Get a store and assign it as a DocumentStore object
        var store = connection.GetStore("/demo_table");

        var docId = "user0001";

        // Print the document before update
        var documentBeforeUpdate = store.FindById(docId);
        Console.WriteLine($"Document with id {docId} before update:");
        Console.WriteLine(documentBeforeUpdate);

        // Create mutation to update the zipCode field
        var mutation =
connection.NewDocumentMutation().SetOrReplace("address.zipCode", 99999);

        // Create condition
        var condition = connection
            .NewQueryCondition()
                .Is("address.street", QueryOp.EQUAL, "320 Blossom Hill
Road")
                .Close()
            .Build();

        // Execute CheckAndUpdate.
        // Returns True if condition True and document was updated
        var updateResult = store.CheckAndUpdate(docId, condition, mutation);

        Console.WriteLine(updateResult);

        // Print the document after update
        var documentAfterUpdate = store.FindById(docId);
        Console.WriteLine($"Document with id {docId} after update:");
        Console.WriteLine(documentAfterUpdate);

        // Close the OJAI connection
        connection.Close();
    }
}
```

Go - Update

The following code is available at 012_update_document.go. It does the following:

- Finds a document using the `DocumentStore.FindByIdString` function to print the document before update
- Creates an OJAI mutation that updates a field

last-updated: Apr 11, 2022

- Updates the document by calling the `DocumentStore.Update` function

```
package main

import (
    "fmt"
    client "github.com/mapr/private-maprdb-go-client"
)

func main() {
    // Create connection string
    connectionString := "192.168.33.11:5678?" +
        "auth=basic;" +
        "user=mapr;" +
        "password=mapr;" +
        "ssl=true;" +
        "sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
        "sslTargetNameOverride=node1.cluster.com"

    storeName := "/demo_table"
    documentId := "user0002"

    // Create a connection to DAG
    connection, err := client.MakeConnection(connectionString)
    if err != nil {
        panic(err)
    }

    // Get a store and assign it as a DocumentStore struct
    store, err := connection.GetStore(storeName)
    if err != nil {
        panic(err)
    }

    // Print the document before update
    documentBeforeUpdate, err := store.FindByIdString(documentId)
    if err != nil {
        panic(err)
    }
    fmt.Printf("Document with id %v before update.\n %v", documentId,
documentBeforeUpdate.AsJsonString())

    // Create mutation to update the zipCode field
    mutation := map[string]interface{}{"$set": map[string]interface{}
{"address.zipCode": 95196}}

    // Execute update
    err = store.Update(client.BosiFromString(documentId),
client.MosmFromMap(mutation))
    if err != nil {
        panic(err)
    }

    // Print the document after update
    documentAfterUpdate, err := store.FindByIdString(documentId)
    if err != nil {
        panic(err)
    }
    fmt.Printf("Document with id %v after update.\n %v", documentId,
documentAfterUpdate.AsJsonString())

    // Close connection
```

```
    connection.Close()
}
```

Go - Check and Update

The following code is available at 013_check_and_update_document.go. It does the following:

- Finds a document using the DocumentStore.FindByIdString function to print the document before update
- Creates an OJAI mutation that updates a field
- Creates an OJAI condition to apply in the check and update
- Performs the check and update on the document by calling the `DocumentStore.CheckAndUpdate` function

```
package main

import (
    "fmt"
    client "github.com/mapr/private-maprdb-go-client"
)

func main() {
    // Create connection string
    connectionString := "192.168.33.11:5678?" +
        "auth=basic;" +
        "user=mapr;" +
        "password=mapr;" +
        "ssl=true;" +
        "sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
        "sslTargetNameOverride=node1.cluster.com"

    storeName := "/demo_table"
    documentId := "user0001"

    // Create a connection to DAG
    connection, err := client.MakeConnection(connectionString)
    if err != nil {
        panic(err)
    }

    // Get a store and assign it as a DocumentStore struct
    store, err := connection.GetStore(storeName)
    if err != nil {
        panic(err)
    }

    // Print the document before update
    documentBeforeUpdate, err := store.FindByIdString(documentId)
    if err != nil {
        panic(err)
    }
    fmt.Printf("Document with id %v before update.\n %v\n", documentId,
documentBeforeUpdate.AsJsonString())

    // Create mutation to update the zipCode field
    mutation := map[string]interface{}{"$put": map[string]interface{}
{"address.zipCode": 99999}}

    // Create condition
```

```
    condition := map[string]interface{}{"$eq": map[string]interface{}
{"address.street": "320 Blossom Hill Road"}}

    // Execute update
    // Returns True if condition True and document was updated.
    res, err := store.CheckAndUpdate(
        client.BosiFromString(documentId),
        client.MoscFromMap(condition),
        client.MosmFromMap(mutation))
    if err != nil {
        panic(err)
    }

    // Print the document after update
    documentAfterUpdate, err := store.FindByIdString(documentId)
    if err != nil {
        panic(err)
    }
    fmt.Printf("Update result: %v.\nDocument with id %v after update.\n
%v\n",
        res,
        documentId,
        documentAfterUpdate.AsJsonString())

    // Close connection
    connection.Close()
}
```

Querying JSON Documents

This section describes how to query JSON documents in MapR Database JSON tables using the OJAI API library and MapR Database Shell. It includes sample programs using the OJAI API library and shows how to run the same queries in MapR Database Shell.

Querying in OJAI Applications

To query MapR Database JSON tables in your OJAI applications, you use the OJAI `Query` interface. The typical flow of your application involves creating a connection, obtaining a handle to the MapR Database JSON table you want to query, constructing the query, performing the query, and then processing the results.

Note: The Node.js, Python, C#, and Go OJAI clients are supported starting in EEP 6.0.

Description

The `DocumentStore` interface includes a `Query` interface. The `Query` interface allows you to build a query programmatically.

Java

To construct an OJAI query, call the following methods in the `Query` interface:

- Query.select
- Query.where
- Query.orderBy
- Query.offset
- Query.limit

To run the query, pass the `Query` object to the DocumentStore.find method.

last-updated: Apr 11, 2022

Node.js

To construct an OJAI query, create a Node.js JSON object using OJAI Query Syntax on page 2603.

To run the query, pass the `Query` object to the DocumentStore.find method.

Python

To construct an OJAI query, create a Python dictionary object using OJAI Query Syntax on page 2603.

To run the query, pass the `Query` object to the DocumentStore.find method.

Note: The following `Query` methods are available in the Python OJAI API, but creating a Python dictionary is the preferred approach:

- Query.select
- Query.where
- Query.order_by
- Query.offset
- Query.limit

C#

To construct an OJAI query, create a C# object.

To run the query, pass the `Query` object to the `DocumentStore.Find` method.

Note: The following `Query` methods are available in the C# OJAI API:

- Query.Select
- Query.Where
- Query.OrderBy
- Query.Offset
- Query.Limit

Go

To construct an OJAI query, create a Go object.

To run the query, pass the `Query` object to the `DocumentStore.FindQuery` function.

Note: The following `Query` functions are available in the Go OJAI API:

- `Query.Select`
- `Query.WhereCondition`
- `Query.OrderBy`
- `Query.Offset`
- `Query.Limit`

last-updated: Apr 11, 2022

Basic Application Flow

The following steps describe the basics in developing client applications that query MapR Database JSON tables using the OJAI API.

Java

1. Create a Connection instance to your MapR cluster using the DriverManager class:

```
Connection connection =
DriverManager.getConnection("ojai:m
apr:");
```

Note: Do not omit the ending colon in the connection string.

2. Obtain a DocumentStore handle to a MapR Database JSON table using the connection object:

```
DocumentStore store =
connection.getStore(tablePath);
```

3. Create a Query object using the connection object:

```
Query query =
connection.newQuery();
```

4. Perform the query operation on the table:

```
QueryResult result =
store.find(query);
```


last-updated: Apr 11, 2022

5. Process the results.

   The following code snippet iterates through the QueryResult and prints each document as a JSON string:

```
for (final Document userDocument :
result) {
    // Print the OJAI Document

System.out.println(userDocument.asJ
sonString());
}
```

   To process individual fields within a document, use the DocumentReader interface. The following code snippet iterates through the fields in a document and prints the fields that are strings:

```
Iterable it =
result.documentReaders();
for (DocumentReader reader : it) {
    EventType et = null;
    while ((et = reader.next()) !=
null) {
        if (et ==
EventType.STRING) {

System.out.println("Value of field
" + reader.getFieldName() + ": " +
reader.getString());
        }
    }
}
```

6. Close the result stream, the connection to the document store, and the connection to MapR:

```
result.close();
store.close;
connection.close();
```

Node.js

1. Create a connection:

```
ConnectionManager.getConnection('lo
calhost:5678?;user=mapr;password=ma
pr;ssl=false')
    .then((connection) => {
        // Process connection
        ...
    });
```

2. Obtain a handle to a MapR Database JSON table using the connection object:

```
connection.getStore(tablePath)
    .then((store) => {
        // Process store
        ...
    });
```

3. Create a query object:

```
const query = {};
```

4. Perform the query operation on the table:

```
const stream = store.find(query)
```

5. Process the results:

```
stream.on('data', (document) =>
console.log(document));
```

6. Close the connection to MapR:

```
stream.on('end', () => {
  console.log('end');
  connection.close();
});
```

Python

1. Create a Connection instance to your MapR cluster using the `ConnectionFactory` class:

```
connection_str =
'localhost:5678?;user=mapr;password
=mapr;ssl=false'
connection =
ConnectionFactory.get_connection(co
nnection_str=connection_str)
```

2. Obtain a DocumentStore handle to a MapR Database JSON table using the connection object:

```
store =
connection.get_store(table_path)
```

3. Create a Query object using the connection object:

```
query =
connection.new_query().build()
```

4. Perform the query operation on the table:

```
query_result = store.find(query)
```

5. Process the results.

   The following code snippet iterates through the QueryResult and prints each document as a Python dictionary:

```
for doc in query_result:
    print(doc)
```

6. Close the connection to MapR:

```
connection.close()
```

C#

1. Create a `Connection` instance to your MapR cluster using the `ConnectionFactory` class:

```
var connectionStr =
$"localhost:5678?auth=basic;" +
            $"user=mapr;" +
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/
ssl_truststore.pem;" +

$"sslTargetNameOverride=node1.mapr.
com";
var connection =
ConnectionFactory.CreateConnection(
connectionStr);
```

2. Obtain a `DocumentStore` handle to a MapR Database JSON table using the connection object:

```
var store =
connection.GetStore(storePath);
```

3. Create a `Query` object using the connection object:

```
var query =
connection.NewQuery().Build();
```

4. Perform the query operation on the table:

```
var queryResult =
store.Find(query);
```


last-updated: Apr 11, 2022

5. Process the results.

   The following code snippet iterates through the `QueryResult` and prints each document as a JSON:

```
var documentStream = await
queryResult.GetDocumentAsyncStream(
).GetAllDocuments();
foreach (var document in
documentStream)
{

Console.WriteLine(document.ToJsonSt
ring());
}
```

6. Close the connection to MapR:

```
connection.Close();
```

Go

1. Create a `Connection` instance to your MapR cluster:

```
connectionString :=
"localhost:5678?
auth=basic;user=mapr;password=mapr;
ssl=false"
connection, error :=
client.MakeConnection(connectionStr
ing)
```

2. Obtain a `DocumentStore` handle to a MapR Database JSON table using the connection object:

```
store,
error := connection.CreateStore("/
store_path")
```

3. Create a Query object:

```
query, err := client.MakeQuery()
query.Build()
```

4. Perform the query operation on the table:

```
queryResult, err :=
store.FindQuery(query,
&client.FindOptions{})
```


last-updated: Apr 11, 2022

5. Process the results.

   The following code snippet iterates through the QueryResult and prints each document as a JSON:

```
    for _, doc := range
queryResult.DocumentList() {
        fmt.Println(doc)
    }
```

6. Close the connection to MapR:

```
connection.Close()
```

See Examples: Querying JSON Documents on page 2624 for complete code examples.

Related concepts

OJAI Distributed Query Service on page 505
OJAI queries either directly access MapR Database JSON or leverage the OJAI Distributed Query Service. The OJAI Distributed Query Service provides distributed query support for MapR Database JSON, powered by Apache Drill. The MapR client automatically determines whether OJAI queries benefit from using the OJAI Distributed Query Service, when the service is available. This section describes the architecture, including the code paths and components involved. It also discusses queries that originate from Drill SQL, which leverage the full functionality of MapR Drill.

Related information

Java OJAI Client API
Node.js OJAI Client API
Python OJAI Client API
C# OJAI Client API
Go OJAI Client API
OJAI github repository
OJAI wiki page

Comparisons and Sorts in OJAI Queries

When running OJAI queries with comparisons and sorts, you need to be aware of how different data types behave. You also need to understand how sorting works in MapR-DB queries. Depending on the component that runs the sort, you may encounter unexpected behavior.

OJAI supports comparisons using the `QueryCondition` interface. For information about how to use this interface, see Query Conditions in OJAI Applications on page 2590.

When using the OJAI `Query` `where` and `orderby`, and comparing and sorting across different data types, there are subtleties you should take into consideration. See Using Comparable JSON Document Data Types in Comparisons and Sorts on page 513 and Using Non-comparable JSON Document Data Types in Comparisons and Sorts on page 514 for more information.

If you do not have a secondary index defined that can generate your query's specified `orderby`, then your query requires an explicit sort. If you have installed the OJAI Distributed Query Service on page 505, the service performs the sort. If you have not, the MapR client performs the sort, but restricts the amount of data it can sort. The default sort limit is 5000 documents. For example, if your query returns 10,000 documents, and you specify a query result `limit` of 5000 documents, the MapR client can perform the sort.

Important: The MapR client returns an error if your query result size exceeds the client's sort limit.

You can avoid errors due to the client sort limitation by adhering to the following guidelines:

- If you know the largest possible query result size when your queries specify an `order by`, you can increase the sort limit of your client to that maximum size by setting the `ojai.mapr.query.max-client-sort-limit` parameter.

  The following code snippets increase the limit to 6000:

Java

```
query.setOption("ojai.mapr.query.ma
x-client-sort-limit", 6000);
```

Note: This option is not applicable to the Java OJAI Thin Client.

Node.js

```
const query
= {"$select":"col","$options":
{"ojai":{"mapr":{"query":
{"max-client-sort:6000}}}}
const stream = store.find(query)
```

Python

```
query = {"$select":"col","$options":
{"ojai":{"mapr":{"query":
{"max-client-sort:6000}}}}
query_result = store.find(query)
```

C#

```
            var query =
connection.NewQuery()
                .Select("col")
                .SetMaxClientSortLimi
t(6000)
                .Build();

            var queryResult =
store.Find(query);
```

Go

```
query := map[string]interface{}{
    "$select":"col",
    "$options":map[string]interface{}
{
        "ojai":map[string]interface{}
{

"mapr":map[string]interface{}{

"query":map[string]interface{}{

"max-client-sort":6000}}}}
queryResult, err :=
store.FindQueryMap(query,
&client.FindOptions{})
```

You can also set this option across all your OJAI clients by modifying a MapR Data Access Gateway property. See Administering the MapR Data Access Gateway - Application Properties for details.

- If you do not know the largest possible query result size, specify a `limit` in your queries. If your query result size exceeds that `limit`, the client sorts the entire result set but returns only a subset of the rows up to the specified limit. This avoids the error, but may result in unintended behavior if your application is not expecting a truncated result. You should take corrective action if necessary.

  See Querying with Order By on page 2655 for an example of how to set a query `limit`.

Permissions and OJAI Queries
You need to understand permission requirements because they affect filter conditions in your OJAI queries.

MapR Database enforces permissions when your application processes the query result. In the basic application flow shown in the previous section, this corresponds to step 5. In an application, if user1 performs the query while user2 processes the result, then the result corresponds to user2's permissions.

You should create a separate OJAI connection for each unique user. Sharing a connection across users can result in non-optimal queries or invalid permission errors.

The following permissions are required to query documents:

- The `readAce` permission on the volumes where the JSON tables that contain the documents are located. See Setting Whole Volume ACEs on page 1459.
- The `readperm` permission on the JSON table's column families containing fields being queried. See Enabling Table and Stream Authorizations with ACEs on page 1461.

If the user does not have the `readperm` permission on a field, MapR Database treats the field as non-existent for that user. When a query selects a non-existent field, MapR Database ignores the field. If a query filters on a non-existent field, the query behaves as follows:

Filter Condition on Non-existent Field	Behavior
Filter for specific values in the field	No documents qualify the filter because a non-existent field does not match any value.
Filter for non-matches in the field	All documents qualify the filter because a non-match on a non-existent field is a no-op.

The exception is the rowkey field. Access control on the rowkey is not available. Users can always select and filter on rowkey.

For information about setting permissions, see Permission Types for Fields and Column Families in JSON Tables on page 1462.

OJAI Query Options
OJAI supports query options that enable you to modify the behavior of your queries. This includes an option to force secondary index usage and options to influence the behavior of sorts.

Available Query Options

The following table lists available query options. Some options may or may not apply, depending on whether your query uses the OJAI Distributed Query Service. The detailed descriptions make a note of this.

Option Name	Description	Details
`ojai.mapr.query.hint-using-index`	Forces the MapR client to use a particular index, regardless of cost considerations	Forcing Secondary Index Usage in OJAI on page 2589
`ojai.mapr.query.force-noncovering-sort`	Enables sort behavior to avoid partial sorts due to secondary index lags	Avoiding Partial Sorts with Secondary Indexes in OJAI on page 2589
`ojai.mapr.query.max-client-sort-limit`	Sets the MapR client sort limit	Comparisons and Sorts in OJAI Queries on page 2586

last-updated: Apr 11, 2022

Option Name	Description	Details
`ojai.mapr.query.force-drill`	When set to `true`, forces the MapR client to use the OJAI Distributed Query Service	Forcing Usage of the OJAI Distributed Query Service on page 2589
`ojai.mapr.drill.<OJAI Distributed Query Service Property Name>`	Sets options for the OJAI Distributed Query Service	Setting OJAI Distributed Query Service Properties on page 2590

Setting Query Options

To set these options in your OJAI application, see the following topics:

Java	Setting Query Options in Java OJAI on page 2670
Node.js	Setting Query Options in Node.js Using OJAI Query Syntax on page 2678
Python	Setting Query Options in Python Using OJAI Query Syntax on page 2688
C#	Setting Query Options in C# OJAI on page 2692
Go	Setting Query Options in Go OJAI on page 2696

Forcing Secondary Index Usage in OJAI

To force the MapR client to use an index, specify the name of the index with the `ojai.mapr.query.hint-using-index` option.

Regardless of cost considerations, the MapR client attempts to use the specified index. To use the index, the index must benefit filter conditions, the order by, or projections in the query as described at Queries that Benefit from Secondary Indexes on page 570. Otherwise, the MapR client ignores the option.

To force the MapR client to *not* use any indexes, specify the table name without the full path as the second parameter in the calls shown earlier. For example, if the full path of your table is `/mapr/sanfrancisco/volume1/customer`, pass the name `customer` as the second parameter.

Note: Setting this option in your OJAI application has no effect if you are using the OJAI Distributed Query Service.

Avoiding Partial Sorts with Secondary Indexes in OJAI

Partial sorts can occur due to secondary index lags. To avoid these lags, set the `ojai.mapr.query.force-noncovering-sort` option to TRUE.

This option forces the OJAI Distributed Query Service to explicitly sort the data. Do not set this option if you do not expect to encounter index lags. Otherwise, you lose the ordering advantage that secondary indexes provide.

For more information about why partial sorts occur, see Partial Sorts with Non-Covering Indexes on page 579.

Forcing Usage of the OJAI Distributed Query Service

When set to `true`, the MapR client uses the OJAI Distributed Query Service execution path, rather than selecting an execution path that it determines to be most optimal. See OJAI Distributed Query Service on page 505 for more information about the different query execution paths.

last-updated: Apr 11, 2022

Setting OJAI Distributed Query Service Properties

OJAI queries may leverage the OJAI Distributed Query Service. To modify OJAI Distributed Query Service property settings in your OJAI application, prefix the OJAI Distributed Query Service property name with `ojai.mapr.drill`.

For example, the option `ojai.mapr.drill.planner.enable_index_planning` disables using secondary indexes when queries use the Query Service.

See Index Planning and Execution Configuration Options on page 3352 for the list of available OJAI Distributed Query Service properties.

Related concepts

OJAI Distributed Query Service on page 505
OJAI queries either directly access MapR Database JSON or leverage the OJAI Distributed Query Service. The OJAI Distributed Query Service provides distributed query support for MapR Database JSON, powered by Apache Drill. The MapR client automatically determines whether OJAI queries benefit from using the OJAI Distributed Query Service, when the service is available. This section describes the architecture, including the code paths and components involved. It also discusses queries that originate from Drill SQL, which leverage the full functionality of MapR Drill.

Query Conditions in OJAI Applications

You can create a query condition in an OJAI application in either of two ways. One way is to create an OJAI `QueryCondition` object and call methods in the class to construct your query condition. Another way is to create an OJAI query condition in a JSON format.

Creating an OJAI QueryCondition Object

The Java and Python OJAI clients support a `QueryCondition` interface. After you create a `QueryCondition` object, call methods in the class to construct your query condition.

Creating a `QueryCondition` Object

Java

Java OJAI provides a `QueryCondition.is()` method for specifying query conditions. The method takes three arguments:

- The field path to apply the condition to
- The condition operator, represented as a `QueryCondition.Op`
- The value to compare the field path against

The field path is either a field in a JSON document, a subfield within a nested document, or an array element. Starting in MapR 6.1, you can also specify a container field path. See OJAI Query Conditions Using Container Field Paths on page 2615 for details.

Depending on the type of the field path, you specify the comparison value as follows:

Scalar Data

You can specify the value using either a Java typed value (for example, `int`, `float`, or `String`) or a Java OJAI object. The API supports the following OJAI types:

- `ODate`
- `OInterval`

last-updated: Apr 11, 2022

	• `OTime` • `OTimestamp`
Nested Documents	You can specify only equality and non-equality conditions on nested documents. You specify the nested document using a Java `Map` object. In the case of equality, all of the fields in the nested document must match. The order of the fields is not relevant.
Arrays	You can specify only equality and non-equality conditions on arrays. You specify an array using a Java `List` object. In the case of equality, the order of the elements and the element values must match.

In addition to `QueryCondition.is()`, `QueryCondition` also supports the following methods:

`QueryCondition` Method	**Description**
`equals()` `notequals()`	Match for equality or non-equality on nested documents and arrays
`in()`	Search for individual elements in an array
`like()`	Search for string values using SQL LIKE expressions
`matches()`	Search for string values using regular expressions. You can use regular expressions that compose the Perl-Compatible Regular Expressions (PCRE) library as well as a subset of the regular expressions that are supported in `java.util.regex.pattern`. See HBase Java Regular Expressions Support on page 2506 for a list of supported regular expressions.
`and()`	Begins a new AND condition block
`or()`	Begins a new OR condition block

last-updated: Apr 11, 2022

QueryCondition Method	Description
`elementAnd()`	Begins a new `elementAnd` block. See OJAI Query Condition Operators on page 2606 for a detailed description of this operator.
`close()`	Closes a compound condition block
`build()`	Builds the condition

Note: The material described in this section is a subset of the `QueryCondition` API. It introduces you to the basics of the API. For the complete API, see the QueryCondition interface.

Python

Python OJAI provides a `QueryCondition.is_()` method for specifying query conditions. The method takes three arguments:

- The field path to apply the condition to
- The condition operator, represented as a `QueryConditionOp`
- The value to compare the field path against

The field path is either a field in a JSON document, a sub-field within a nested document, or an array element. Starting in MapR 6.1, you can also specify a container field path. See OJAI Query Conditions Using Container Field Paths on page 2615 for details.

Depending on the type of the field path, you specify the comparison value as follows:

Scalar Data

You can specify the value using either a Python scalar value (for example, `int`, `float`, or `str`) or a Python OJAI object. The API supports the following OJAI types:

- `ODate`
- `OInterval`
- `OTime`
- `OTimestamp`

Nested Documents

You can specify only equality and non-equality conditions on nested documents. You specify the nested document using a Python `dictionary` object. In the case of equality, all of the fields in the nested

	document must match. The order of the fields is not relevant.
Arrays	You can specify only equality and non-equality conditions on arrays. You specify an array using a Python `list` object. In the case of equality, the order of the elements and the element values must match.

In addition to `QueryCondition.is_()`, `QueryCondition` also supports the following methods:

`QueryCondition` Method	**Description**
`equals_()` `not_equals_()`	Match for equality or non-equality on nested documents and arrays
`in_()`	Search for individual elements in an array
`like_()`	Search for string values using SQL LIKE expressions
`matches_()`	Search for string values using regular expressions. You can use regular expressions that comprise the Perl-Compatible Regular Expressions (PCRE) library as well as a subset of the regular expressions that are supported in `java.util.regex.pattern`. See HBase Java Regular Expressions Support on page 2506 for a list of supported regular expressions.
`and_()`	Begins a new `AND` condition block
`or_()`	Begins a new `OR` condition block
`element_and()`	Begins a new `elementAnd` block. See OJAI Query Condition Operators on page 2606 for a detailed description of this operator.
`close()`	Closes a compound condition block
`build()`	Builds the condition

Note:

- The material described in this section is a subset of the `QueryCondition` API. It introduces you to the basics of the API. For the complete API, see the QueryCondition interface.
- The preferred approach for creating query conditions in Python is to create the condition in a JSON format. See Creating an OJAI Query Condition Using a JSON String on page 2601

C#

C# OJAI provides a `QueryCondition.ls()` method for specifying query conditions. The method takes three arguments:

- The field path to apply the condition to
- The condition operator, represented as a `QueryOp`
- The value to compare the field path against

The field path is either a field in a JSON document, a sub-field within a nested document, or an array element. Starting in MapR 6.1, you can also specify a container field path. For details, see OJAI Query Conditions Using Container Field Paths on page 2615.

Depending on the type of the field path, you specify the comparison value as follows:

Scalar Data	You can specify the value using either a C# scalar value (for example, `int`, `float`, or `string`) or a C# OJAI object. The API supports the following OJAI types: • `OjaiDate` • `OjaiInterval` • `OjaiTime` • `OjaiTimestamp`
Nested Documents	You can specify only equality and non-equality conditions on nested documents. You specify the nested document using a C# object. In the case of equality, all of the fields in the nested document must match. The order of the fields is not relevant.
Arrays	You can specify only equality and non-equality conditions on arrays. You specify an array using

a C# `list` of values of the specified type. In the case of equality, the order of the elements and the element values must match.

In addition to `QueryCondition.ls()`, `QueryCondition` also supports the following methods:

`QueryCondition` Method	**Description**
`Condition()`	Search for values using a specific condition.
`Equals()` `NotEquals()`	Match for equality or non-equality on nested documents and arrays.
`Exists()` `NotExists()`	Search for a field if the given field path exists, or verify that a field path does not exist.
`In()` `NotIn()`	Search for individual elements in an array or verify their absence.
`Like()` `NotLike()`	Search for string values using SQL LIKE expressions or verify they do not match the specified SQL LIKE expression.
`Matches()` `NotMatches()`	Search for string values using regular expressions. You can use regular expressions that comprise the Perl-Compatible Regular Expressions (PCRE) library, as well as a subset of the regular expressions that are supported in `java.util.regex.pattern`. For a list of supported regular expressions, see HBase Java Regular Expressions Support on page 2506.
`SizeOf()`	Search for a value of the specified size. The value must be one of the following types: • `string` • `binary` • `iDictionary` • `iList`
`TypeOf()` `NotTypeOf()`	Search for value of the specified Type or verify its absence.

last-updated: Apr 11, 2022

QueryCondition Method	Description
`And()`	Begins a new `AND` condition block.
`Or()`	Begins a new `OR` condition block.
`ElementAnd()`	Begins a new `ElementAnd` block. For a detailed description of this operator, see OJAI Query Condition Operators on page 2606.
`Close()`	Closes a compound condition block.
`Build()`	Builds the condition.

Note:

- The material described in this section is a subset of the QueryCondition API. It introduces you to the basics of the API. For the complete API, see the QueryCondition interface.
- The preferred approach for creating query conditions in C# is to create the condition in a JSON format. See 009_FindQueryWithSelectAndCondition.cs or Example: Creating a QueryCondition Object on page 2598.

Go

Go OJAI provides a `QueryCondition.ls()` function for specifying query conditions. The function takes three arguments:

- The field path to apply the condition to
- The condition operator, represented as a `QueryOp`
- The value to compare the field path against

The field path is either a field in a JSON document, a sub-field within a nested document, or an array element. Starting in MapR 6.1, you can also specify a container field path. For details, see OJAI Query Conditions Using Container Field Paths on page 2615.

Depending on the type of the field path, you specify the comparison value as follows:

Scalar Data

You can specify the value using either a Go scalar value (for example, `int`, `float64`, or `string`) or a Go OJAI object. The API supports the following OJAI types:

- `OjaiDate`
- `OjaiTime`

last-updated: Apr 11, 2022

	• `OjaiTimestamp`
Nested Documents	You can specify only equality and non-equality conditions on nested documents. You specify the nested document using a Go object. In the case of equality, all of the fields in the nested document must match. The order of the fields is not relevant.
Arrays	You can specify only equality and non-equality conditions on arrays. You specify an array using a Go `list` of values of the specified type. In the case of equality, the order of the elements and the element values must match.

In addition to `QueryCondition.ls()`, `QueryCondition` also supports the following functions:

`QueryCondition` Function	**Description**
`AddCondition()`	Search for values using a specific condition.
`Equals()` `NotEquals()`	Match for equality or non-equality on nested documents and arrays.
`Exists()` `NotExists()`	Search for a field if the given field path exists, or verify that it does not exist.
`In()` `NotIn()`	Search for individual elements in an array or verify their absence.
`Like()` `NotLike()`	Search for string values using SQL LIKE expressions or verify they do not match the specified SQL LIKE expression.
`Matches()` `NotMatches()`	Search for string values using regular expressions. You can use regular expressions that comprise the Perl-Compatible Regular Expressions (PCRE) library, as well as a subset of the regular expressions that are supported in `java.util.regex.pattern`. For a list of supported regular expressions, see HBase Java Regular Expressions Support on page 2506.

last-updated: Apr 11, 2022

QueryCondition Function	Description
`TypeOf()` `NotTypeOf()`	Search for value of the specified Type or verify its absence.
`And()`	Begins a new `AND` condition block.
`Or()`	Begins a new `OR` condition block.
`ElementAnd()`	Begins a new `ElementAnd` block. For a detailed description of this operator, see OJAI Query Condition Operators on page 2606.
`Close()`	Closes a compound condition block.
`Build()`	Builds the condition.

Note:

- The material described in this section is a subset of the `Query` API. It introduces you to the basics of the API. For the complete API, see the Query interface.
- The preferred approach for creating query conditions in Go is to create the condition in a JSON format. See 009_find_query_with_select_and_condition.go or Example: Creating a QueryCondition Object on page 2598.

Example: Creating a `QueryCondition` Object

The following example shows how to define a `QueryCondition` object for this query condition:

```
(a.b.[0].boolean == false && (a.c.d != 5 || a.b[1].decimal > 1 ||
a.b[1].decimal < 10))
```

Java

```
QueryCondition qc =
connection.newCondition()
    .and()
        .is("a.b[0].boolean",
Op.EQUAL, false)
        .or()
            .is("a.c.d",
Op.NOT_EQUAL, 5)
            .is("a.b[1].decimal",
Op.GREATER, 1)
            .is("a.b[1].decimal",
Op.LESS, 10)
            .close()
    .close()
    .build();
```

Pass the `QueryCondition` object to the Query.where method. For a complete Java code example, see the *Java - OJAI QueryCondition Object* example at Querying with Conditions on page 2642.

Python

```
qc = connection.new_condition()
    .and_()
        .is_('a.b[0].boolean',
QueryConditionOp.EQUAL, False)
        .or_()
            .is_('a.c.d',
QueryConditionOp.NOT_EQUAL, 5)
            .is_('a.b[1].decimal',
QueryConditionOp.GREATER, 1)
            .is_('a.b[1].decimal',
QueryConditionOp.LESS, 10)
            .close()
    .close()
    .build()
```

Pass the `QueryCondition` object to the Query.where method. For a complete Python code example, see the *Python - OJAI QueryCondition Object* example at Querying with Conditions on page 2642.

C#

```
var condition =
connection.NewQueryCondition()
    .And()
        .Is("a.b[0].boolean",
QueryOp.EQUAL, false)
        .Or()
            .Is("a.c.d",
QueryOp.NOT_EQUAL, 5)
            .Is("a.b[1].decimal",
QueryOp.GREATER, 1)
            .Is("a.b[1].decimal",
QueryOp.LESS, 10)
            .Close()
    .Close()
    .Build();
```

Pass the `Condition` object to the `Query.Where` method. For a complete C# code example, see the *C# - OJAI QueryCondition Object* example at Querying with Conditions on page 2642.

Go

```
    condition, err :=
client.MakeCondition(
        client.And(),

client.Is("a.b[0].boolean",
client.EQUAL, false),
            client.Or(),
                client.Is("a.c.d",
client.NOT_EQUAL, 5),

client.Is("a.b[1].decimal",
client.GREATER, 1),

client.Is("a.b[1].decimal",
```

```
client.LESS, 10),
            client.Close(),
        client.Close())
    condition.Build()
```

Pass the `Condition` object to the `Query.WhereCondition` function. For a complete Go code example, see the *Go - OJAI QueryCondition Object* example at Querying with Conditions on page 2642.

Examples: Using the `QueryCondition.elementAnd` Method

The following example shows how to write the `elementAnd` condition described at Using elementAnd with Nested Documents on page 2619, using a `QueryCondition` object:

Java

```
QueryCondition qc =
connection.newCondition()
    .elementAnd("grades[]")
        .is("course",
QueryConditionOp.EQUALS, "history")
        .is("score",
QueryConditionOp.EQUALS, 12)
        .close()
    .build();
```

Python

```
qc = connection.new_condition()
    .element_and("grades[]")
        .is_("course",
QueryConditionOp.EQUALS, "history")
        .is_("score",
QueryConditionOp.EQUALS, 12)
        .close()
    .build()
```

C#

```
var condition =
connection.NewQueryCondition()
    .ElementAnd("grades[]")
        .Is("course", QueryOp.EQUALS,
"history")
        .Is("score", QueryOp.EQUALS,
12)
        .Close()
    .Build();
```

Go

```
    condition, err :=
client.MakeCondition(
        client.ElementAnd("grades[]"),
            client.Is("course",
client.EQUAL, "history"),
            client.Is("score",
client.EQUAL, 12),
        client.Close())
    condition.Build()
```

The following code corresponds to the example described at Using elementAnd with Scalar Values on page 2620 using a `QueryCondition` object:

Java

```
QueryCondition qc =
connection.newCondition()
    .elementAnd("values[]")
        .is("$",
QueryConditionOp.GREATER, 7)
        .is("$",
QueryConditionOp.LESS, 14)
        .close()
    .build();
```

Python

```
qc = connection.new_condition()
    .element_and("values[]")
        .is_("$",
QueryConditionOp.GREATER, 7)
        .is_("$",
QueryConditionOp.LESS, 14)
        .close()
    .build()
```

C#

```
var condition =
connection.NewQueryCondition()
    .EementAnd("values[]")
        .Is("$", QueryOp.GREATER, 7)
        .Is("$", QueryOp.LESS, 14)
        .Close()
    .Build();
```

Go

```
    condition, err :=
client.MakeCondition(
        client.ElementAnd("values[]"),
            client.Is("$",
client.EQUAL, 7),
            client.Is("$",
client.EQUAL, 14),
        client.Close())
    condition.Build()
```

Creating an OJAI Query Condition Using a JSON String
You can create a query condition using OJAI syntax to specify the condition in JSON format. This is the preferred approach for the Node.js, Python, C#, and Go OJAI clients.

The following example shows you how to create the following query condition using the syntax:

```
(a.b.[0].boolean == false && (a.c.d != 5 || a.b[1].decimal > 1 ||
a.b[1].decimal < 10))
```

Java

This is a Java string for the condition:

```
String jc = new String(
    '{ \
        "$and":[ \
            {"$eq":
{"a.b[0].boolean":false}}, \
```


last-updated: Apr 11, 2022

```
            {"$or":[ \
                {"$ne":{"a.c.d":5}}, \
                {"$gt":
{"a.b[1].decimal":1}}, \
                {"$lt":
{"a.b[1].decimal":10}} \
            ]} \
        ]} \
    )';
```

Pass the string to the Query.where method. See the *Java - OJAI Query Condition in JSON Format* example at Querying with Conditions on page 2642 for a complete Java code example.

Node.js

This is a Node.js JSON object for the condition:

```
query =
    {"$where":
        {"$and":[
            {"$eq":
{"a.b[0].boolean":false}},
            {"$or":[
                {"$ne":{"a.c.d":5}},
                {"$gt":
{"a.b[1].decimal":1}},
                {"$lt":
{"a.b[1].decimal":10}}
            ]}
        ]}
    };
```

See the *Node.js - OJAI Query Condition in JSON Format* example at Querying with Conditions on page 2642 for a complete Node.js code example.

Python

This is a Python dictionary for the condition:

```
query =
    {"$where":
        {"$and":[
            {"$eq":
{"a.b[0].boolean":false}},
            {"$or":[
                {"$ne":{"a.c.d":5}},
                {"$gt":
{"a.b[1].decimal":1}},
                {"$lt":
{"a.b[1].decimal":10}}
            ]}
        ]}
    }
```

See the *Python - OJAI Query Condition in JSON Format* example at Querying with Conditions on page 2642 for a complete Python code example.

C#

This is a JSON string for the condition:

```
var query =
    @"{""$where"":" +
        @"{""$and"":[" +
```

```
            @"{""$eq"":
{""a.b[0].boolean"":false}}," +
            @"{""$or"":["
+
                @"{""$ne"":{""a.c.d"":
{""$numberInt"":""5""}}}," +

@"{""$gt"":{""a.b[1].decimal"":
{""$decimal"":""1""}}}," +

@"{""$lt"":{""a.b[1].decimal"":
{""$decimal"":""10""}}}" +
            @"]}" +
        @"]" +
   @"}";
```

Go

This is a JSON string for the condition:

```
query := "{\"$where\": +
              "{\"$and\":[" +
                    "{\"$eq\":
{\"a.b[0].boolean\":false}}," +
                    "{\"$or\":[" +
                         "{\"$ne\":
{\"a.c.d\":5}}," +
                         "{\"$gt\":
{\"a.b[1].decimal\":1}}," +
                         "{\"$lt\":
{\"a.b[1].decimal\":10}}" +
                    "]}" +
              "]}" +
         "}"
```

To learn about the complete OJAI syntax for query conditions, see OJAI Query Condition Syntax on page 2606.

Note: The OJAI clients are supported starting in EEP 6.0.0.

OJAI Query Syntax

OJAI defines a syntax for specifying queries on JSON documents. You can use this syntax in Node.js and Python OJAI client applications and MapR Database shell.

See Query with --query on page 5292 to learn about how to use this syntax in MapR Database shell.

An OJAI query can include the following components:

- Projection
- Condition
- Order by
- Limit
- Offset
- Options

You can specify some or all these components in a query, separating each component with a comma.

OJAI Query Projection

Syntax

```
"$select":"fieldpath"
```

```
"$select":
["fieldpath1","fieldpath2",...]
```

Description

The projection is the list of field paths to select in your query. You can specify a single field path or multiple. When specifying multiple, use an array notation to list the field paths.

See JSON Document Field Paths on page 515 for more information about the syntax of different JSON document field paths.

Examples

Single field path:

```
"$select":"a.c.d"
```

Multiple field paths:

```
"$select":["a.c.d", "a.c.e", "m[0]"]
```

OJAI Query Condition

Syntax

```
"$where":OJAIQueryCondition
```

Description

The condition filters your query result. See OJAI Query Condition Syntax on page 2606 for more information about the syntax of an *OJAIQueryCondition*.

Example

If you have the following condition:

```
(a.b.[0].boolean == false && (a.c.d !
= 5 || a.b[1].decimal > 1 ||
a.b[1].decimal < 10))
```

This is the OJAI JSON syntax for the condition:

```
"$where":{
    "$and":[
        {"$eq":
{"a.b[0].boolean":false}},
        {"$or":[
            {"$ne":{"a.c.d":5}},
            {"$gt":
{"a.b[1].decimal":1}},
            {"$lt":
{"a.b[1].decimal":10}}
            ]
        }
    ]
}
```


last-updated: Apr 11, 2022

OJAI Query Order By

Syntax

```
"$orderby":"fieldpath"
```

```
"$orderby":{"fieldpath":"order"}
```

```
"$orderby":[fieldpath1,fieldpath2,...]
```

```
"$orderby":[{"fieldpath1":"order"},
{"fieldpath2":"order"},...]
```

Description

The order by specifies the field paths on which to sort your query result. You can specify a single field path or multiple. When specifying multiple, use an array notation to list the field paths. For each field path, you can optionally specify an *order* of either `asc` or `desc`. Both *order* keywords are case insensitive. The default is `asc`. When specifying an *order*, enclose the *fieldpath* and *order* with curly braces.

Examples

Order on a single field path in the default `asc` order:

```
"$orderby":"a.c.e"
```

Order on a single field path in the `desc` order:

```
"$orderby":{"a.c.e":"desc"}
```

Order on two field paths, where the second specifies a `desc` order:

```
"$orderby":["a.c.d",{"a.c.e":"desc"}]
```

OJAI Query Limit

Syntax

```
"$limit":positive-integer
```

Description

The number of documents to return from the query.

Example

Return only ten documents:

```
"$limit:10
```

OJAI Query Offset

Syntax

```
"$offset":positive-integer
```

Description

The number of documents to skip before returning results to the client. The offset value has a direct effect on query time; as the offset value increases, query time also increases.

last-updated: Apr 11, 2022

Example

Process the query and skip the first five documents in the result set before returning the results to the client.

```
"$offset":5
```

OJAI Query Options

Syntax

```
"$options":{optionName:optionValue}
```

```
"$options":
[{optionName1:optionValue1},
{optionName2:optionValue2},...]
```

Description

Settings that influence a query's execution path. See OJAI Query Options on page 2588 for a list of available options.

When specifying the *optionName*, you must separate the components of the option name, replacing the dots with curly braces and colons and enclosing each component in quotes.

Example

Force the query to use the OJAI Distributed Query Service by setting the `ojai.mapr.query.force-drill` option:

```
"$options":{"ojai":{"mapr":{"query":
{"force-drill":true}}}}
```

OJAI Query Condition Syntax

OJAI defines a syntax for specifying query conditions that allows you to express query conditions in a JSON format. This topic describes the supported operators and provides examples of these query conditions.

When writing an OJAI application, you can also apply query conditions by calling OJAI API methods, corresponding to specific operators. This section does not discuss this alternative. For details on that alternative, see Query Conditions in OJAI Applications on page 2590.

Note: Using the JSON format is the preferred approach for Python and Node.js OJAI clients.

OJAI Query Condition Operators

OJAI supports comparison, existence, between, match, like, type of, size of, in, and logical operators.

Click the name in the following box to navigate to the section that provides details on each operator.

last-updated: Apr 11, 2022

Comparison Operators

Operators

Operator	Syntax
Equals	`{"$eq": {"fieldpath":value}}`
Greater Than	`{"$gt": {"fieldpath":value}}`
Greater Than or Equals	`{"$ge": {"fieldpath":value}}`
Less Than	`{"$lt": {"fieldpath":value}}`
Less Than or Equals	`{"$le": {"fieldpath":value}}`
Not Equals	`{"$ne": {"fieldpath":value}}`

Description

Compares the data in *`fieldpath`* against *`value`* for the specified operator.

Float and double data are approximate representations of decimal values. They may not return true in equality comparisons against their equivalent decimal values.

You can specify only equality and non-equality conditions on nested documents and arrays.

In the case of equality on nested documents, all of the fields in the nested document must match. The order of the fields is not relevant.

In the case of equality on arrays, both the order of the elements and the element values must match.

last-updated: Apr 11, 2022

Existence Operators

Exists

Syntax

```
{"$exists":"fieldpath"}
```

Description

Checks for existence of *fieldpath*.

Not Exists

Syntax

```
{"$notexists":"fieldpath"}
```

Description

Checks for non-existence of *fieldpath*.

See Existence Conditions with Container Field Paths on page 2617 for details about how these operators behave when you use them with container field paths.

Between

Syntax

```
{"$between":{"fieldpath":
[startValue,endValue]}}
```

Description

Checks if the value in *fieldpath* is in the range specified by *startValue* and *endValue*, where the values are inclusive.

Matches Operators

Operators

Operator	Syntax
Matches	`{"$matches": {"fieldpath":matchValue}}`
Not Matches	`{"$notmatches": {"fieldpath":matchValue}}`

Description

Performs a regular expression match on *fieldPath* using *matchValue*.

You can use regular expressions that compose the Perl-Compatible Regular Expressions (PCRE) library as well as a subset of the regular expressions that are supported in `java.util.regex.pattern`. See HBase Java Regular Expressions Support for a list of supported regular expressions.

Like Operators

Operators

Operator	Syntax
Like	`{"$like": {"fieldpath":likeValue}}`
Not Like	`{"$notlike": {"fieldpath":likeValue}}`

Description

Performs a SQL LIKE comparison on `fieldPath` where *`likeValue`* is a string with wildcard characters '%' and '_'.

Special-Purpose Characters for the Like Operators

The OJAI API allows you to use four special-purpose characters or patterns with `$like` operator expressions:

Special-Purpose Character	Description	Example
%	Matches any string of zero or more characters.	`"abc%"` matches `"abc"`, `"abcd"`, `"abcde232136"`, etc. `"%abc"` matches `"abc"`, `"pqrabc"`, etc.
_	Matches a single character.	`"_am"` matches `"ram"`, `"sam"`, `"Sam"`, `"cam"`.
[]	Matches a single character in the specified set or range.	`"[r-t]am"` matches `"ram"`, `"sam"`, and `"tam"` but not `"Sam"` or `"cam"`.
[^]	Matches a single character **not** in the specified set or range.	`"[^r-t]am"` matches `"Sam"`, `"pam"`, `"jam"`, or `"cam"` but not `"ram"`, `"sam"` and `"tam"`.

When any of these special characters is used as a literal, the character can be enclosed within `[]`:

Literal	Corresponding Like Expression
`"[a]"`	`"[[]a]"`

last-updated: Apr 11, 2022

Literal	Corresponding Like Expression
`"a%"`	`"a[%]"`
`"a_c"`	`"a[_]c"`

Note that `"^"` and `"]"` need not be escaped.

Type of Operators

Type Of

Syntax

```
{"$typeof":{"fieldpath":"typeValue"}}
```

Description

Checks whether *`fieldpath`* is of type *`typeValue`*.

Not Type Of

Syntax

```
{"$nottypeof":{"fieldpath":"typeValue"}}
```

Description

Checks whether *`fieldpath`* is not of type *`typeValue`*.

`typeValue` can be any of `map`, `array`, `binary`, `date`, `time`, `timestamp`, `interval`, `double`, `float`, `long`, `int`, `short`, `byte`, `string`, `boolean`, or `null`.

Size Of

Syntax

```
{"$sizeof":{"fieldpath":{"comparisonOp":intValue}}}
```

Description

Compares the size of the data in *`fieldpath`* against *`intValue`*, using *`comparisonOp`* . The size varies depending on the type of *`fieldPath`*:

String	Length of string
Array	Number of elements in the array
Nested document	Number of subfields in the nested document

last-updated: Apr 11, 2022

comparisonOp can be any of `$eq`, `$lt`, `$le`, `$gt`, `$ge`, or `$ne`.

In Operators

In

Syntax

```
{"$in":
{"fieldpath":inOp
Values}}
```

Description

Checks whether the data in *fieldpath* is in the list specified by *inOpValues*.

Not In

Syntax

```
{"$notin":
{"fieldpath":inOp
Values}}
```

Description

Checks whether the data in *fieldpath* is not in the list specified by *inOpValues*

Logical Operators

And

Syntax

```
{"$and":
[OJAIQueryConditi
ons]}
```

Description

Applies logical AND on a list of conditions. *OJAIQueryConditions* is a comma-separated list of OJAI query conditions.

Or

Syntax

```
{"$or":
[OJAIQueryConditi
ons]}
```

Description

Applies logical OR on a list of conditions. *OJAIQueryConditions* is a comma-separated list of OJAI query conditions.

Element And

Syntax

```
{

"$elementAnd":{
```

```
"containerFieldPath":

[OJAIQueryConditions]
    }
}
```

Note: Supported starting in MapR 6.1.

Description

Applies multiple conditions as part of a group. All conditions must be true for a common array element.

OJAIQueryConditions is the comma-separated list of the OJAI query conditions.

containerFieldPath exhibits the following behaviors:

- *containerFieldPath* specifies the container path prefix of the common container element.
- If *containerFieldPath* refers to a container of nested documents, then you must use field paths relative to the common prefix in your *OJAIQueryConditions* .
- If the *containerFieldPath* refers to a container of scalar values, then you use the $ symbol to refer to individual elements in your *OJAIQueryConditions*.

Note: There is no `elementOr` operator because it is semantically equivalent to an OR operator.

Related concepts

OJAI Query Condition Examples on page 2613

last-updated: Apr 11, 2022

This section contains examples that show you how to use different OJAI query condition operators in combination with different field references and data types.

OJAI Query Conditions Using Container Field Paths on page 2615
Starting in MapR 6.1, MapR Database supports the notion of a *container field path*. A container field path enables you to perform comparisons on a field path that is either a single value or an arbitrary array element. You can use container field paths with arrays and nested documents, including nested documents with multiple levels of nesting and multidimensional arrays.

OJAI Query Conditions Using elementAnd on page 2619
The `elementAnd` operator allows you to specify multiple conditions on the same array element using a container field path. This is in contrast to the `and` operator where conditions can refer to any array element. You can use `elementAnd` with both nested documents and scalar values. You can also use it in combination with other operators, including `between`, `and`, and `or`.

OJAI Query Condition Examples
This section contains examples that show you how to use different OJAI query condition operators in combination with different field references and data types.

The examples in this section use the following JSON documents:

```
{ "_id" : "001", "name" : "Ipod 001", "tags" : [ "electronics", "ipod",
"apple" ] }
{ "_id" : "002", "name" : "Ipod 002", "tags" : "ipod" }
{ "_id" : "003", "name" : "Ipod 003", "tags" : 10 }
{ "_id" : "004", "name" : "Ipod 004", "tags" : [ 10, "ipod", { "t" :
"ipod" } ] }
{ "_id" : "005", "name" : "Ipod 005", "tags" : { "t" : "ipod" } }
{ "_id" : "006", "name" : "Ipod 006", "tags" : [ { "t" : "ipod" }, { "t" :
"apple" } ] }
{ "_id" : "007", "name" : "Ipod 007", "tags" : [ { "t" : "ipod", "v" :
10 }, { "t" : "apple", "v" : 9 } ] }
{ "_id" : "008", "name" : "Ipod 008", "tags" : { "t" : "ipod", "v" : 10 } }
```

Example	Documents Returned
`{"$exists":"tags.v"}` Matches documents where `tags` is a nested document that has a `v` subfield.	008
`{"$eq":{"tags":10}}` Matches documents where `tags` equals the scalar value 10.	003
`{"$eq":{"tags.t":"ipod"}}` Matches documents where `tags` is a nested document with a subfield `t` equal to "ipod".	005, 008
`{"$eq":{"tags":{"t":"ipod"}}}` Matches documents where `tags` is a nested document with a single subfield `t` equal to "ipod".	005
`{"$eq":{"tags":{"v":10,"t":"ipod"}}}` Matches documents where `tags` is a nested document with two subfields, `v` and `t`. `v` is equal to 10, and `t` is equal to "ipod". The order of subfields in the condition does not matter.	008

last-updated: Apr 11, 2022

Example	Documents Returned
`{"$eq":{"tags":["electronics","ipod","apple"]}}` Matches documents where `tags` is an array with the three elements listed.	001
`{"$eq":{"tags":["ipod","electronics","apple"]}}` This example does not match any document, whereas the previous does, because the order of the elements in this example does not match the order in document 001.	None
`{"$between":{"tags":[5,15]}}` Matches documents where `tags` is a scalar value between 5 and 15.	003
`{"$like":{"tags[1]":"ip%"}}` Matches documents where the first array element in `tags` qualifies the wildcard string `"ip%"`	001, 004
`{"$typeof":{"tags":"map"}}` Matches documents where `tags` is a nested document.	005, 008
`{"$typeof":{"tags":"array"}}` Matches documents where `tags` is an array.	001, 004, 006, 007
`{"$sizeof":{"tags":{"$ge":3}}}` Matches documents where the size of the data in `tags` is greater than or equal to three. • 001 matches because the array has three elements • 002 matches because the string is of length four • 004 matches because the nested document has three subfields	001, 002, 004
`{"$in":{"tags":["ipod", 10, {"t":"ipod"}]}}` Matches documents where `tags` equals any of the values listed. Note that the values can be of different types.	002, 003, 005
`{ "$and":[ {"$lt":{"tags[1].v":10}}, {"$matches":{"tags[1].t":"ap{2}"}} ] }` Matches documents where the first array element in `tags` has a nested document with a subfield v less than one, and the same nested document also matches the regular expression `"ap{2}"`.	007

Example	Documents Returned

<pre>{
    "$or":[
        {"$exists":"tags.v"},
        {"$typeof":{"tags":"string"}}
    ]
}</pre> Matches documents where either `tags` is a nested document with a subfield `v`, or `tags` is a scalar string.
	002, 008

Note: You can improve the performance of queries with conditions by using secondary indexes. See Queries that Benefit from Secondary Indexes on page 570 for more details.

Related concepts

OJAI Query Conditions Using Container Field Paths on page 2615
Starting in MapR 6.1, MapR Database supports the notion of a *container field path*. A container field path enables you to perform comparisons on a field path that is either a single value or an arbitrary array element. You can use container field paths with arrays and nested documents, including nested documents with multiple levels of nesting and multidimensional arrays.

OJAI Query Conditions Using elementAnd on page 2619
The `elementAnd` operator allows you to specify multiple conditions on the same array element using a container field path. This is in contrast to the `and` operator where conditions can refer to any array element. You can use `elementAnd` with both nested documents and scalar values. You can also use it in combination with other operators, including `between`, `and`, and `or`.

Related reference

OJAI Query Condition Operators on page 2606
OJAI supports comparison, existence, between, match, like, type of, size of, in, and logical operators.

OJAI Query Conditions Using Container Field Paths

Starting in MapR 6.1, MapR Database supports the notion of a *container field path*. A container field path enables you to perform comparisons on a field path that is either a single value or an arbitrary array element. You can use container field paths with arrays and nested documents, including nested documents with multiple levels of nesting and multidimensional arrays.

Conditions with Container Field Paths on Arrays

If you have a field that has a single value rather than an array of values, when using the container notation, MapR Database treats the single value as an array with one element. This enables you to use a container field path to access a field that has both array elements and scalar values. The array elements and scalar values can be of any type.

Suppose you have the following set of documents:

```
{ "_id" : "001", "name" : "Ipod 001", "tags" : [ "electronics", "ipod",
"apple" ] }
{ "_id" : "002", "name" : "Ipod 002", "tags" : "ipod" }
{ "_id" : "003", "name" : "Ipod 003", "tags" : 10 }
{ "_id" : "004", "name" : "Ipod 004", "tags" : [ 10, "ipod", { "t" :
"ipod" } ] }
{ "_id" : "005", "name" : "Ipod 005", "tags" : { "t" : "ipod" } }
{ "_id" : "006", "name" : "Ipod 006", "tags" : [ { "t" : "ipod" }, { "t" :
"apple" } ] }
{ "_id" : "007", "name" : "Ipod 007", "tags" : [ { "t" : "ipod", "v" :
```


last-updated: Apr 11, 2022

```
10 }, { "t" : "apple", "v" : 9 } ] }
{ "_id" : "008", "name" : "Ipod 008", "tags" : { "t" : "ipod", "v" : 10 } }
```

To find all documents that contain the tag named `"ipod"`, you can use the following OJAI query condition, where you reference `tags` using a container field path:

```
{"$eq":{"tags[]":"ipod"}}
```

The expression matches the following documents, with the matching condition highlighted in bold:

```
{ "_id" : "001", "name" : "Ipod 001", "tags" : [ "electronics", "ipod",
"apple" ] }
{ "_id" : "002", "name" : "Ipod 002", "tags" : "ipod" }
{ "_id" : "004", "name" : "Ipod 004", "tags" : [ 10, "ipod", { "t" :
"ipod" } ] }
```

Note that the matching documents have the following characteristics:

- In 001 and 004, `tags` are array fields.
- In 002, `tags` is a scalar value.
- In 001 and 004, the `tags` arrays have elements in addition to `"ipod"`.

You can also use the `AND` operator to match multiple container field path conditions.

For example, the following condition finds all documents that have both `"ipod"` and `"apple"` tags:

```
{
    "$and":[
        {"$eq":{"tags[]":"ipod"}},
        {"$eq":{"tags[]":"apple"}}
    ]
}
```

The expression matches the following document, with the matching conditions highlighted in bold:

```
{ "_id" : "001", "name" : "Ipod 001", "tags" : [ "electronics", "ipod",
"apple" ]
```

Conditions with Container Field Paths on Nested Documents

You can also use the container field path in combination with a nested document subfield reference.

For example, using the same set of documents shown earlier, the following OJAI query condition finds all documents in which `"ipod"` is specified in the subfield named `t` within the `tags` nested document:

```
{"$eq":{"tags[].t":"ipod"}}
```

This expression returns the following documents, with the matching condition highlighted in bold:

```
{ "_id" : "004", "name" : "Ipod 004", "tags" : [ 10, "ipod", { "t" : "ipod"
} ] }
{ "_id" : "005", "name" : "Ipod 005", "tags" : { "t" : "ipod" } }
{ "_id" : "006", "name" : "Ipod 006", "tags" : [ { "t" : "ipod" }, { "t" :
"apple" } ] }
{ "_id" : "007", "name" : "Ipod 007", "tags" : [ { "t" : "ipod", "v" :
10 }, { "t" : "apple", "v" : 9 } ] }
{ "_id" : "008", "name" : "Ipod 008", "tags" : { "t" : "ipod", "v" : 10 } }
```

Note that the matching documents have the following characteristics:

- In 005 and 008, `tags` is a single nested document.
- In 006 and 007, `tags` is an array of nested documents.
- In 004, the `tags` array has both scalar data and a nested document.
- In 004 and 006, the `tags` array have other array elements that do not match the nested document subfield `t`.

Existence Conditions with Container Field Paths

Existence Operators on page 2608 check for the existence and non-existence of a specified field path. When you use `$exists` with a container field path, the specified field path can be any element in an array.

Using the same set of documents shown earlier, the following OJAI query condition finds all documents where the `tags` array has a nested document with a subfield `t`:

```
{"$exists":"tags[].t"}
```

The expression matches the following documents with the matching condition highlighted in bold:

```
{"_id":"004","name":"Ipod 004","tags":[10,"ipod",{"t":"ipod"}]}
{"_id":"005","name":"Ipod 005","tags":{"t":"ipod"}}
{"_id":"006","name":"Ipod 006","tags":[{"t":"ipod"},{"t":"apple"}]}
{"_id":"007","name":"Ipod 007","tags":[{"t":"ipod","v":10},
{"t":"apple","v":9}]}
```

When you use `$notexists` with a container field path, it matches *any* element in the array that does not meet the existence condition:

```
{"$notexists":"tags[].t"}
```

The expression returns the following documents with the matching condition highlighted in bold:

```
{"_id":"001","name":"Ipod 001","tags":["electronics","ipod","apple"]}
{"_id":"002","name":"Ipod 002","tags":"ipod"}
{"_id":"003","name":"Ipod 003","tags":10}
{"_id":"004","name":"Ipod 004","tags":[10,"ipod",{"t":"ipod"}]}
```

Even document `004` has a `tags[].t` element, the other elements in that document's `tags` array do not; therefore, the document qualifies the condition.

Conditions with Container Field Paths Across Multiple Levels of Nested Documents

The following are examples of query conditions that match the sample document shown at Container Field Paths Across Multiple Levels of Nested Documents:

```
{"$eq":{"projects[].customer.contacts[].emails[].value":"jdoe@gmail.com"}}
```

```
{"$eq":{"projects[].customer.contacts[].role":"CEO"}}
```

Conditions with Container Field Paths on Multidimensional Arrays

The following examples reference documents that store the high and low temperatures for each day in a week. They use a two-dimensional array to store this data. The first element of each nested array element is the high temperature for a day, and the second element is the low. Typically, the two-dimensional array

last-updated: Apr 11, 2022

has seven array pairs, one for each day of the week. But in cases where data is unavailable, the document has only the days available.

For example, document 002 has a single dimensional array because it has data for only one day that week.

```
{
   "_id" : "001",
   "temps" : [[61,49],[74,51],[75,51],[74,52],[78,54],[75,53],[75,54]],
   "weekOf" : "4/29/2018"
}
{
   "_id" : "002",
   "temps" : [81,60],
   "weekOf" : "5/12/2018"
}
{
   "_id" : "003",
   "temps" : [[80,55],[78,54],[79,54],[77,53],[79,54],[77,54],[78,54]],
   "weekOf" : "5/13/2018"
}
```

As described at Container Field Paths with Multidimensional Arrays, you can specify a container field path in a dimension only if it does not precede a dimension that specifies an explicit element. For example, the following condition is not allowed because the first dimension specifies a container field path and precedes element 1 in the second dimension:

```
// Invalid condition
{"$ge":{"temps[][1]":60}}
```

The following table shows examples of conditions on multidimensional arrays that MapR Database supports:

Example	Documents Returned
`{"$ge":{"temps[][]":60}}` Matches documents that have any temperature greater than 60. • Documents 001 and 003 match because all days have high temperatures above 60. • Document 002 matches because day 1 has a low temperature of 60. Although `temps` in this document is a one-dimensional array, the container notation treats it as a two-dimensional array.	001, 002, 003
`{"$ge":{"temps[1][]":75}}` Matches documents that have any temperature greater than 75 on the second day of the week	003
`{"$eq":{"temps[]":[78,54]}}` Matches documents that have a high and low temperature of 78 and 54 on the same day. • Day 5 from document 001 matches this condition. • Days 2 and 7 from document 003 match this condition.	001, 003

Related concepts

OJAI Query Conditions Using elementAnd on page 2619

last-updated: Apr 11, 2022

The `elementAnd` operator allows you to specify multiple conditions on the same array element using a container field path. This is in contrast to the `and` operator where conditions can refer to any array element. You can use `elementAnd` with both nested documents and scalar values. You can also use it in combination with other operators, including `between`, `and`, and `or`.

Related information

Container Field Paths on page 518
Indexes on Container Field Paths in Equality Conditions on page 573
Indexes on Container Field Paths in Range Conditions on page 576

OJAI Query Conditions Using `elementAnd`

The `elementAnd` operator allows you to specify multiple conditions on the same array element using a container field path. This is in contrast to the `and` operator where conditions can refer to any array element. You can use `elementAnd` with both nested documents and scalar values. You can also use it in combination with other operators, including `between`, `and`, and `or`.

Using `elementAnd` with Nested Documents

Assume that you have the following set of documents that reflect student scores on courses. Each document has an array of `grades`. `Grades` is a nested document that reflects how the students scored on each course they took.

```
{"_id":"001","grades":[{"course":"math","score":15.5},
{"score":12,"course":"history"},{"course":"english","score":8}]}
{"_id":"002","grades":[{"course":"math","score":4},
{"course":"history","score":12,"cmts":"..."},
{"course":"english","score":18}]}
{"_id":"003","grades":[{"course":"math","score":11},
{"course":"history","score":15},{"course":"english","score":12},
{"course":"sports","score":4}]}
{"_id":"004","grades":[{"course":"math","score":15.5},
{"course":"history","score":12,"details":{"info":"..."}}]}
{"_id":"005","grades":[{"course":"math","score":15.5},
{"course":"history","score":10},{"course":"physics","score":11}]}
```

If you want to find the students who scored 12 in history, you use the following `elementAnd` condition:

```
{
    "$elementAnd":{
        "grades[]":[
            {"$eq":{"course":"history"}},
            {"$eq":{"score":12}}
        ]
    }
}
```

The condition matches the following documents, with the matching conditions highlighted in bold:

```
{"_id":"001","grades":[{"course":"math","score":15.5},
{"course":"history","score":12},{"course":"english","score":8}]}
{"_id":"002","grades":[{"course":"math","score":4},
{"course":"history","score":12,"cmts":"..."},
{"course":"english","score":18}]}
{"_id":"004","grades":[{"course":"math","score":15.5},
{"course":"history","score":12,"details":{"info":"..."}}]}
```

The example illustrates the following behavior:

- The positions of the subfields in the nested document are not significant.

- In document 002, there are other subfields in the nested document that do not match the specified conditions.

In contrast, the following example expresses a different condition, using `and` instead of `elementAnd`:

```
{
    "$and":[
        {"$eq":{"grades[].course":"history"}},
        {"$eq":{"grades[].score":12}}
    ]
}
```

This condition returns documents corresponding to students who have taken history and scored 12 on *any* course. The following are the matching documents, with the matching conditions highlighted in bold:

```
{"_id":"001","grades":[{"course":"math","score":15.5},
{"course":"history","score":12},{"course":"english","score":8}]}
{"_id":"002","grades":[{"course":"math","score":4},
{"course":"history","score":12,"cmts":"..."},
{"course":"english","score":18}]}
{"_id":"003","grades":[{"course":"math","score":11},
{"course":"history","score":15},{"course":"english","score":12},
{"course":"sports","score":4}]}
{"_id":"004","grades":[{"course":"math","score":15.5},
{"course":"history","score":12,"details":{"info":"..."}}]}
```

The example illustrates the following behavior:

- Besides returning the same documents as the previous `elementAnd` example, this condition also returns document 003.
- Document 003 matches because that student took history and scored 12 on english, rather than history.
- Document 005 does not match because although the student took history, the student did not score 12 on any courses.

Using `elementAnd` with Scalar Values

If you apply `elementAnd` to a container of scalar values, you use the `$` symbol to denote an unspecified container element.

Suppose you have the following documents:

```
{"_id" : "001", "name" : "a", "values" : [1, 2, 3, 6, 15] }
{"_id" : "002", "name" : "b", "values" : [3, 6, 9, 10, 15] }
{"_id" : "003", "name" : "c", "values" : [14] }
{"_id" : "004", "name" : "c", "values" : 11 }
```

To find all documents where `values[]` contains a number between 7 and 11 (inclusive), you can use the following condition:

```
{
    "$elementAnd":{
        "values[]":[
            {"$ge":{"$":7}},
            {"$le":{"$":11}}
        ]
    }
}
```

The condition returns the following documents, with the matching numbers highlighted in bold:

```
{"_id":"002","name":"b","values":[3,6,9,10,15]}
{"_id":"004","name":"c","values":11}
```

The example illustrates the following behavior:

- In document 002, multiple elements in the array match the condition.
- In document 004, `values` is a scalar value.

Suppose you apply the following condition that uses `and` instead of `elementAnd`:

```
{
    "$and":[
        {"$ge":{"values[]":7}},
        {"$le":{"values[]":11}}
    ]
}
```

All documents except 003 match this `and` condition because in the matching documents, `values[]` contains *some* number greater than or equal to 7 and *some* number less than or equal to 11. The difference is that the same number does not need to match both conditions, which is the case for document 001.

Using `between` with `elementAnd`

You cannot use a container field path in a `between` condition. To use the `between` operator to match against an arbitrary array element, you must include the `between` condition in an `elementAnd` condition.

The following table shows the proper way to specify a `between` condition that is equivalent to the `elementAnd` example from the previous section:

Correct Condition	Incorrect Condition
`{ "$elementAnd":{ "values[]":[ {"$between":{"$":[7,11]}} ] } }`	`{"$between":{"values[]":[7,11]}}`

This example uses `between` to match against an arbitrary scalar array element. You can also use `between` to match against a subfield in a nested document, in which the nested document is an arbitrary array element.

For example, using the sample documents shown earlier, the following table shows the correct way to apply the `between` operator on the subfield `score` in the nested documents that are elements in the `grades` array:

Correct Condition	Incorrect Condition

<pre>{
    "$elementAnd":{
        "grades[]":[
            {"$between":{"score":
[15.5,20]}}
        ]
    }
}</pre>
	<pre>{"$between":{"grades[].score":
	[15.5,20]}}</pre>

The condition returns the following documents, with the matching conditions highlighted in bold:

<pre>{"_id":"001","grades":[{"course":"math","score":15.5},
{"course":"history","score":12},{"course":"english","score":8}]}
{"_id":"002","grades":[{"course":"math","score":4},
{"cmts":"...","course":"history","score":12},
{"course":"english","score":18}]}
{"_id":"004","grades":[{"course":"math","score":15.5},
{"course":"history","details":{"info":"..."},"score":12}]}
{"_id":"005","grades":[{"course":"math","score":15.5},
{"course":"history","score":10},{"course":"physics","score":11}]}</pre>

Using Other Operators in `elementAnd` Conditions

You can also use operators like `or` in `elementAnd`'s query condition list.

For example, the following condition finds all students who scored 12 in either history or english:

<pre>{
    "$elementAnd":{
        "grades[]":[
            {"$or":[
                {"$eq":{"course":"history"}},
                {"$eq":{"course":"english"}}
            ]},
            {"$eq":{"score":12}}
        ]
    }
}</pre>

The condition returns the following documents, with the matching conditions highlighted in bold:

<pre>{"_id":"001","grades":[{"course":"math","score":15.5},
{"course":"history","score":12},{"course":"english","score":8}]}
{"_id":"002","grades":[{"course":"math","score":4},
{"cmts":"...","course":"history","score":12},
{"course":"english","score":18}]}
{"_id":"003","grades":[{"course":"math","score":11},
{"course":"history","score":15},{"course":"english","score":12},
{"course":"sports","score":4}]}
{"_id":"004","grades":[{"course":"math","score":15.5},
{"course":"history","score":12,"details":{"info":"..."}}]}</pre>

Combining `elementAnd` with Other Operators

You can combine `elementAnd` with other operators like `and`.

last-updated: Apr 11, 2022

For example, using the sample documents shown earlier, suppose you want to find all students who scored 12 in history as well scored 15.5 in math. The following condition expresses this criteria:

```
{
   "$and":[
      {
         "$elementAnd":{
            "grades[]":[
               {"$eq":{"course":"history"}},
               {"$eq":{"score":12}}
            ]
         }
      },
      {
         "$elementAnd":{
            "grades[]":[
               {"$eq":{"course":"math"}},
               {"$eq":{"score":15.5}}
            ]
         }
      }
   ]
}
```

The condition returns the following documents, with the matching conditions highlighted in bold:

```
{"_id":"001","grades":[{"course":"math","score":15.5},
{"course":"history","score":12},{"course":"english","score":8}]}
{"_id":"004","grades":[{"course":"math","score":15.5},
{"course":"history","score":12,"details":{"info":"..."}}]}
```

Related reference

OJAI Query Condition Operators on page 2606
OJAI supports comparison, existence, between, match, like, type of, size of, in, and logical operators.

Related information

Composite Indexes and Container Field Paths on page 554

Querying with MapR Database Shell

This section describes how to query JSON documents using either the `find` or `findbyid` command in MapR Database Shell (dbshell). It introduces the functionality the `find` command supports and describes the two ways to specify your queries. It also provides links to reference pages and examples.

The `findbyid` command allows you to retrieve a single document with a specified id from a MapR Database JSON table.

The `find` command allows you to specify projections and filter conditions (using JSON strings) to retrieve specific documents. It also allows you to specify the following options:

- Range of document IDs to retrieve
- Offset from which to start retrieval
- Order by to sort fields in the document
- Limit on the number of documents to retrieve

To invoke MapR Database shell, run the following command on a MapR cluster node:

```
% mapr dbshell
```


last-updated: Apr 11, 2022

For a complete list and description of options available, see dbshell find or findbyid on page 5290.

Alternatives for Writing Dbshell Query Commands

You can construct your dbshell queries in one of two ways:

- Use individual options in the `find` command
- Use the `--query` option in the `find` command and specify keywords as arguments to `--query`

The following example illustrates the differences between the two alternatives.

Suppose you want to query the table `/apps/tab` with the following criteria:

- Select fields `f1` and `f2`
- Limit the result to ten documents
- Skip the first two documents
- Filter documents where the field `f3` equals 15
- Sorts on field `f1`

Click on each of the following tabs to see the syntax for each alternative:

Use Individual Options in `find`

```
find /apps/tab --fields f1,f2 --limit
10 --offset 2 --where {"$eq":
{"f3":15}} --orderby f1
```

Use the `--query` Option in `find`

```
find /apps/tab --query {"$select":
["f1","f2"],"$limit":10,"$offset":2,"$
where":{"$eq":
{"f3":15}},"$orderby":"f1"}
```

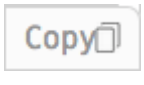
For more examples on how to use the two query alternatives, see the following links:

Use Individual Options in `find`	Use the `--query` Option in `find`
Query Examples with Other Options on page 5297	• Query with --query on page 5292 • Query with --orderby on page 5296

Note: With both options, you need to specify the query condition using OJAI Query Condition Syntax on page 2606.

Examples: Querying JSON Documents

This section provides query examples using the OJAI API. The examples include querying by document ID, retrieving all documents in a store, selecting individual fields, specifying query conditions, and ordering your query result. For reference, the examples also include the equivalent MapR Database Shell (dbshell) commands.

If you hover over the right hand side of all code examples, you can use the 

icon to copy and paste the code.

You can also download the code examples from github at https://github.com/mapr-demos/ojai-examples.git.

last-updated: Apr 11, 2022

Querying By ID

The examples in this section show you how to query for a single document ID.

Java

This example retrieves a single document identified by the ID `user001`.

Note: To query for a range of document IDs, you must specify an OJAI QueryCondition. See Querying with Conditions on page 2642 for examples of the syntax.

```
/**
 * Copyright (c) 2017 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.mapr.ojai.examples;

import org.ojai.Document;
import org.ojai.store.Connection;
import org.ojai.store.DocumentStore;
import org.ojai.store.DriverManager;

public class OJAI_003_FindById {

  public static void main(String[] args) {

    System.out.println("==== Start Application ===");


    // Create an OJAI connection to MapR cluster
    final Connection connection = DriverManager.getConnection("ojai:mapr:");

    // Get an instance of OJAI DocumentStore
    final DocumentStore store = connection.getStore("/demo_table");

    // fetch the OJAI Document by its '_id' field
    final Document userDocument = store.findById("user0001");

    // Print the OJAI Document
    System.out.println(userDocument.asJsonString());

    // Close this instance of OJAI DocumentStore
    store.close();

    // close the OJAI connection and release any resources held by the
connection
    connection.close();

    System.out.println("==== End Application ===");
  }

}
```


last-updated: Apr 11, 2022

Node.js

This example retrieves a single document identified by the ID `user0001`.

Note: To query for a range of document IDs, you must specify an OJAI query condition. See Querying with Conditions on page 2642 for examples of the syntax.

```
/*
 * Copyright (c) 2018 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

const { ConnectionManager } = require('node-maprdb');

const connectionString = 'localhost:5678?' +
  'auth=basic;' +
  'user=mapr;' +
  'password=mapr;' +
  'ssl=true;' +
  'sslCA=/opt/mapr/conf/ssl_truststore.pem;' +
  'sslTargetNameOverride=node1.mapr.com';

let connection;

ConnectionManager.getConnection(connectionString)
  .then((conn) => {
    connection = conn;
    // Get a store
    return connection.getStore('/demo_table');
  })
  .then((store) => {
    // fetch the OJAI Document by its '_id' field
    return store.findById('user0001');
  })
  .then((doc) => {
    // Print the OJAI Document
    console.log(doc);
    connection.close();
  });
```

Python

This example retrieves a single document identified by the ID `user0001`.

Note: To query for a range of document IDs, you must specify an OJAI QueryCondition. See Querying with Conditions on page 2642 for examples of the syntax.

```
from mapr.ojai.storage.ConnectionFactory import ConnectionFactory

# Create a connection to data access server
connection_str = "localhost:5678?auth=basic;user=mapr;password=mapr;" \
```

```
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/ssl_truststore.pem;" \
          "sslTargetNameOverride=node1.mapr.com"
connection = ConnectionFactory.get_connection(connection_str=connection_str)

# Get a store and assign it as a DocumentStore object
store = connection.get_store('/demo_table')

# fetch the OJAI Document by its '_id' field
doc = store.find_by_id("user0001")

# Print the OJAI Document
print(doc)

# close the OJAI connection
connection.close()
```

dbshell

The following is the equivalent of the code examples using dbshell. See dbshell find or findbyid on page 5290 for more details about the syntax dbshell provides.

```
# mapr dbshell
maprdb root:> findbyid /demo_table --id user0001
```

C#

This example retrieves a single document identified by the ID `user0001`.

Note: To query for a range of document IDs, you must specify an OJAI `QueryCondition`. See Querying with Conditions on page 2642 for examples of the syntax.

```
using System;
using MapRDB.Driver;

public class FindById
{
    public void FindById()
    {
        // Create a connection to data access server
        var connectionStr = $"localhost:5678?auth=basic;" +
            $"user=mapr;" +
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
            $"sslTargetNameOverride=node1.mapr.com";
        var connection = ConnectionFactory.CreateConnection(connectionStr);

        // Get a store and assign it as a DocumentStore object
        var store = connection.GetStore("/demo_table");

        // Fetch the OJAI Document by its '_id' field
        var document = store.FindById("user0001");

        // Print the OJAI Document
        Console.WriteLine(document);

        // Close the OJAI connection
        connection.Close();
```


last-updated: Apr 11, 2022

```
    }
}
```

Go

This example retrieves a single document identified by the ID `user0001`.

Note: To query for a range of document IDs, you must specify an OJAI Condition. See Querying with Conditions on page 2642 for examples of the syntax.

```
package main

import (
    "fmt"
    client "github.com/mapr/private-maprdb-go-client"
)

func main() {
    // Create connection string
    connectionString := "192.168.33.11:5678?" +
        "auth=basic;" +
        "user=mapr;" +
        "password=mapr;" +
        "ssl=true;" +
        "sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
        "sslTargetNameOverride=node1.cluster.com"

    storeName := "/demo_table"

    // Create a connection to DAG
    connection, err := client.MakeConnection(connectionString)
    if err != nil {
        panic(err)
    }

    // Get a store and assign it as a DocumentStore struct
    store, err := connection.GetStore(storeName)
    if err != nil {
        panic(err)
    }

    // Fetch the OJAI Document by its '_id' field
    doc, err := store.FindByIdString("id0001")
    if err != nil {
        panic(err)
    }

    // Print the OJAI Document
    fmt.Println(doc.AsJsonString())

    // Close connection
    connection.Close()
}
```

Querying and Returning All Documents

The examples in this section show you two ways of retrieving all documents from a document store.

Java - Example 1

The following example queries a document store and returns all documents by using the DocumentStore.find method.

```
/**
 * Copyright (c) 2017 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.mapr.ojai.examples;

import org.ojai.Document;
import org.ojai.DocumentStream;
import org.ojai.store.Connection;
import org.ojai.store.DocumentStore;
import org.ojai.store.DriverManager;

public class OJAI_004_FindAll {

  public static void main(String[] args) {

    System.out.println("==== Start Application ===");


    // Create an OJAI connection to MapR cluster
    final Connection connection = DriverManager.getConnection("ojai:mapr:");

    // Get an instance of OJAI DocumentStore
    final DocumentStore store = connection.getStore("/demo_table");

    // fetch all OJAI Documents from this store
    final DocumentStream stream = store.find();

    for (final Document userDocument : stream) {
      // Print the OJAI Document
      System.out.println(userDocument.asJsonString());
    }

    // Close this instance of OJAI DocumentStore
    store.close();

    // close the OJAI connection and release any resources held by the
connection
    connection.close();

    System.out.println("==== End Application ===");
  }

}
```

Java - Example 2

The following example queries a document store and returns all documents. It creates a Query object and passes that to the DocumentStore.findQuery method.

```
/**
 * Copyright (c) 2017 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License"); you may
not use this file except in compliance with
 * the License. You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on
 * an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
express or implied. See the License for the
 * specific language governing permissions and limitations under the
License.
 */
package com.mapr.ojai.examples;

import org.ojai.Document;
import org.ojai.DocumentStream;
import org.ojai.store.Connection;
import org.ojai.store.DocumentStore;
import org.ojai.store.DriverManager;
import org.ojai.store.Query;

public class OJAI_005_FindAllQuery {

  public static void main(final String[] args) {

    System.out.println("==== Start Application ===");

    // Create an OJAI connection to MapR cluster
    final Connection connection = DriverManager.getConnection("ojai:mapr:");

    // Get an instance of OJAI DocumentStore
    final DocumentStore store = connection.getStore("/demo_table");

    // Build an OJAI query
    final Query query = connection.newQuery().build();

    // fetch all OJAI Documents from this store
    final DocumentStream stream = store.find(query);

    for (final Document userDocument : stream) {
      // Print the OJAI Document
      System.out.println(userDocument.asJsonString());
    }

    // Close this instance of OJAI DocumentStore
    store.close();

    // close the OJAI connection and release any resources held by the
connection
    connection.close();

    System.out.println("==== End Application ===");
  }
```

```
}
```

Node.js - Example 1

The following example queries a document store and returns all documents by using the DocumentStore.find method.

```
/*
 * Copyright (c) 2018 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

const { ConnectionManager } = require('node-maprdb');

const connectionString = 'localhost:5678?' +
  'auth=basic;' +
  'user=mapr;' +
  'password=mapr;' +
  'ssl=true;' +
  'sslCA=/opt/mapr/conf/ssl_truststore.pem;' +
  'sslTargetNameOverride=node1.mapr.com';

let connection;

// Create a connection to data access server
ConnectionManager.getConnection(connectionString)
  .then((conn) => {
    connection = conn;
    // Get a store
    return connection.getStore('/demo_table');
  })
  .then((store) => {
    // fetch all OJAI Documents from table
    return store.find({});
  })
  .then((queryResult) => {
    queryResult.on('data', (document) => console.log(document));
    queryResult.on('end', () => {
      // close the OJAI connection
      connection.close();
    });
  });
```

Node.js - Example 2

The following example queries a document store and returns all documents. It creates an empty query and passes that to the DocumentStore.find method.

```
/*
 * Copyright (c) 2018 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

const { ConnectionManager } = require('node-maprdb');

const connectionString = 'localhost:5678?' +
  'auth=basic;' +
  'user=mapr;' +
  'password=mapr;' +
  'ssl=true;' +
  'sslCA=/opt/mapr/conf/ssl_truststore.pem;' +
  'sslTargetNameOverride=node1.mapr.com';

let connection;

// Create a connection to data access server
ConnectionManager.getConnection(connectionString)
  .then((conn) => {
    connection = conn;
    // Get a store
    return connection.getStore('/demo_table');
  })
  .then((store) => {
    // options for find request
    const options = {
      'ojai.mapr.query.include-query-plan': true,
      'ojai.mapr.query.timeout-milliseconds': 10000
    }
    // fetch all OJAI Documents from table
    return store.find({}, options)
  })
  .then((queryResult) => {
    // get query plan
    console.log(queryResult.queryPlan);

    queryResult.on('data', (document) => {
      // Print OJAI Documents from document stream
      console.log(document);
    });
    queryResult.on('end', () => {
      // close the OJAI connection
      connection.close();
    });
  });
```

Python - Example 1

The following example queries a document store and returns all documents by using the DocumentStore.find method.

```
from mapr.ojai.storage.ConnectionFactory import ConnectionFactory

# Create a connection to data access server
connection_str = "localhost:5678?auth=basic;user=mapr;password=mapr;" \
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/ssl_truststore.pem;" \
          "sslTargetNameOverride=node1.mapr.com"
connection = ConnectionFactory.get_connection(connection_str=connection_str)

# Get a store and assign it as a DocumentStore object
store = connection.get_store('/demo_table')

# fetch all OJAI Documents from table
query_result = store.find()

# Print OJAI Documents from document stream
for doc in query_result:
    print(doc)

# close the OJAI connection
connection.close()
```

Python - Example 2

The following example queries a document store and returns all documents. It creates a Query object and passes that to the DocumentStore.find method.

```
from mapr.ojai.storage.ConnectionFactory import ConnectionFactory

# Create a connection to data access server
connection_str = "localhost:5678?auth=basic;user=mapr;password=mapr;" \
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/ssl_truststore.pem;" \
          "sslTargetNameOverride=node1.mapr.com"
connection = ConnectionFactory.get_connection(connection_str=connection_str)

# Get a store and assign it as a DocumentStore object
store = connection.get_store('/demo_table')

# Build an OJAI query
query = connection.new_query().build()

# options for find request
options = {
    'ojai.mapr.query.include-query-plan': True,
    'ojai.mapr.query.result-as-document': True,
    'ojai.mapr.query.timeout-milliseconds': 10000
}

# fetch all OJAI Documents from table
query_result = store.find(query, options=options)

# get query plan
print(query_result.get_query_plan())

doc_stream = query_result
# Print OJAI Documents from document stream
```


last-updated: Apr 11, 2022

```
for doc in doc_stream:
    print(doc.as_dictionary())

# close the OJAI connection
connection.close()
```

dbshell

The following is the equivalent of the code examples using dbshell. See dbshell find or findbyid on page 5290 for more details about the syntax dbshell provides.

```
# mapr dbshell
maprdb root:> find /demo_table
```

C# - Example 1

The following example queries a document store and returns all documents by using the `GetAllDocuments` method.

```
using System;
using MapRDB.Driver;
using MapRDB.Driver.Ojai;

public class FindAllDocuments
{
    public async void FindAllDocuments()
    {
        // Create a connection to data access server
        var connectionStr = $"localhost:5678?auth=basic;" +
            $"user=mapr;" +
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
            $"sslTargetNameOverride=node1.mapr.com";
        var connection = ConnectionFactory.CreateConnection(connectionStr);

        // Get a store and assign it as a DocumentStore object
        var store = connection.GetStore("/demo_table");

        // Fetch all OJAI Documents from table
        var queryResult = store.Find();
        var documentStream = await queryResult.GetAllDocuments();

        // Print OJAI Documents from document stream
        foreach (var document in documentStream)
        {
            Console.WriteLine(document.ToJsonString());
        }

        // Close the OJAI connection
        connection.Close();
    }
}
```


last-updated: Apr 11, 2022

C# - Example 2

The following example queries a document store and returns all documents. It creates a `Query` object and passes that to the `DocumentStore.Find` method.

```
using System;
using MapRDB.Driver;
using MapRDB.Driver.Ojai;

public class FindAllQuery
{
    public async void FindAllQuery()
    {
        // Create a connection to data access server
        var connectionStr = $"localhost:5678?auth=basic;" +
            $"user=mapr;" +
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
            $"sslTargetNameOverride=node1.mapr.com";
        var connection = ConnectionFactory.CreateConnection(connectionStr);

        // Get a store and assign it as a DocumentStore object
        var store = connection.GetStore("/demo_table");

        // Build an OJAI query
        var query = connection.NewQuery().Build();

        // Options for find request
        var options = new QueryOptions()
        {
            IncludeQueryPlan = true,
            Timeout = 1000
        };

        // Fetch all OJAI Documents from table
        var queryResult = store.Find(query, options);

        // Get query plan
        Console.WriteLine(queryResult.GetQueryPlan());

        var documentStream = await
queryResult.GetDocumentAsyncStream().GetAllDocuments();
        // Print OJAI Documents from document stream
        foreach (var document in documentStream)
        {
            Console.WriteLine(document.ToDictionary());
        }

        // Close the OJAI connection
        connection.Close();
    }
}
```

Go - Example 1

The following example queries a document store and returns all documents by using the DocumentStore.FindAll function.

```
package main

import (
```


last-updated: Apr 11, 2022

```
    "fmt"
    client "github.com/mapr/private-maprdb-go-client"
)

func main() {
    // Create connection string
    connectionString := "192.168.33.11:5678?" +
        "auth=basic;" +
        "user=mapr;" +
        "password=mapr;" +
        "ssl=true;" +
        "sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
        "sslTargetNameOverride=node1.cluster.com"

    storeName := "/demo_table"

    // Create a connection to DAG
    connection, err := client.MakeConnection(connectionString)
    if err != nil {
        panic(err)
    }

    // Get a store and assign it as a DocumentStore struct
    store, err := connection.GetStore(storeName)
    if err != nil {
        panic(err)
    }

    // Fetch all OJAI Documents from table
    findResult, err := store.FindAll(&client.FindOptions{IncludeQueryPlan:
false, ResultAsDocument: true})

    // Print OJAI Documents from document stream
    for _, doc := range findResult.DocumentList() {
        fmt.Println(doc)
    }

    // Close connection
    connection.Close()
}
```

Go - Example 2

The following example queries a document store and returns all documents. It creates a Query object and passes that to the DocumentStore.FindQuery function.

```
package main

import (
    "fmt"
    client "github.com/mapr/private-maprdb-go-client"
)

func main() {
    // Create connection string
    connectionString := "192.168.33.11:5678?" +
        "auth=basic;" +
        "user=mapr;" +
        "password=mapr;" +
        "ssl=true;" +
        "sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
        "sslTargetNameOverride=node1.cluster.com"
```

```
    storeName := "/demo_table"

    // Create a connection to DAG
    connection, err := client.MakeConnection(connectionString)
    if err != nil {
        panic(err)
    }

    // Get a store and assign it as a DocumentStore struct
    store, err := connection.GetStore(storeName)
    if err != nil {
        panic(err)
    }

    // Options for find request
    options := &client.FindOptions{IncludeQueryPlan: true,
ResultAsDocument: true}

    // Build an OJAI query
    query, err := client.MakeQuery()
    if err != nil {
        panic(err)
    }

    // Fetch all OJAI Documents from table
    findResult, err := store.FindQuery(query, options)

    // Get query plan
    fmt.Println(findResult.QueryPlan())

    // Print OJAI Documents from document stream
    for _, doc := range findResult.DocumentList() {
        fmt.Println(doc)
    }

    // Close connection
    connection.Close()
}
```

Paginating Your Result

An alternative to returning all documents from a store is to specify a limit in the query. Another alternative is to paginate the result using offset and limit. Querying with Order By on page 2655 contains an example that shows you how to use offset and limit. Although the example also uses order by, you can use offset and limit independent of order by.

Querying with Select

The examples in this section query a document store and retrieve specific fields from the documents.

Note: Selecting a specific field is also known as a *projection*. You can improve the performance of projection queries by using secondary indexes. See Using Indexes to Optimize Projections in Queries on page 580 for more details.

Java

The following example shows how to retrieve the `_id` and `address.zipCode` fields from all documents in a store using the Query.select method.

```
/**
 * Copyright (c) 2017 MapR, Inc.
```

```
 *
 * Licensed under the Apache License, Version 2.0 (the "License"); you may
not use this file except in compliance with
 * the License. You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on
 * an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
express or implied. See the License for the
 * specific language governing permissions and limitations under the
License.
 */
package com.mapr.ojai.examples;

import org.ojai.Document;
import org.ojai.DocumentStream;
import org.ojai.store.Connection;
import org.ojai.store.DocumentStore;
import org.ojai.store.DriverManager;
import org.ojai.store.Query;

public class OJAI_006_FindQueryWithSelect {

  public static void main(final String[] args) {

    System.out.println("==== Start Application ===");

    // Create an OJAI connection to MapR cluster
    final Connection connection = DriverManager.getConnection("ojai:mapr:");

    // Get an instance of OJAI DocumentStore
    final DocumentStore store = connection.getStore("/demo_table");

    // Build an OJAI query
    final Query query = connection.newQuery()
        .select("_id", "address.zipCode")
        .build();

    // fetch all OJAI Documents from this store
    final DocumentStream stream = store.find(query);

    for (final Document userDocument : stream) {
      // Print the OJAI Document
      System.out.println(userDocument.asJsonString());
    }

    // Close this instance of OJAI DocumentStore
    store.close();

    // close the OJAI connection and release any resources held by the
connection
    connection.close();

    System.out.println("==== End Application ===");
  }

}
```

Node.js

The following example shows how to retrieve the `_id` and `address.zipCode` fields from all documents in a store using an OJAI query.

```
/*
 * Copyright (c) 2018 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

const { ConnectionManager } = require('node-maprdb');

const connectionString = 'localhost:5678?' +
  'auth=basic;' +
  'user=mapr;' +
  'password=mapr;' +
  'ssl=true;' +
  'sslCA=/opt/mapr/conf/ssl_truststore.pem;' +
  'sslTargetNameOverride=node1.mapr.com';

let connection;

// Create a connection to data access server
ConnectionManager.getConnection(connectionString)
  .then((conn) => {
    connection = conn;
    // Get a store
    return connection.getStore('/demo_table');
  })
  .then((store) => {
    // Create an OJAI query
    const query = {"$select": ["_id", "address.zipCode"]};
    // fetch OJAI Documents by query
    return store.find(query);
  })
  .then((queryResult) => {
    queryResult.on('data', (document) => {
      // Print OJAI Documents from document stream
      console.log(document);
    });
    queryResult.on('end', () => {
      // close the OJAI connection
      connection.close();
    });
  });
```

Python

The following example shows how to retrieve the `_id` and `address.zipCode` fields from all documents in a store using an OJAI query.

```
from mapr.ojai.storage.ConnectionFactory import ConnectionFactory

# Create a connection to data access server
connection_str = "localhost:5678?auth=basic;user=mapr;password=mapr;" \
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/ssl_truststore.pem;" \
          "sslTargetNameOverride=node1.mapr.com"
connection = ConnectionFactory.get_connection(connection_str=connection_str)

# Get a store and assign it as a DocumentStore object
store = connection.get_store('/demo_table')

# Create an OJAI query
query = {"$select": ["_id", "address.zipCode"]}

# options for find request
options = {
    'ojai.mapr.query.result-as-document': True
    }

# fetch OJAI Documents by query
query_result = store.find(query, options=options)

# Print OJAI Documents from document stream
for doc in query_result:
    print(doc.as_dictionary())

# close the OJAI connection
connection.close()
```

dbshell

The following two dbshell commands are equivalent to the code examples. See dbshell find or findbyid on page 5290 for more details about the syntax dbshell provides.

```
# mapr dbshell
maprdb root:> find /demo_table --query {"$select":["_id","address.zipcode"]}

maprdb root:> find /demo_table --fields _id,address.zipcode
```

C#

The following example shows how to retrieve the `_id` and `address.zipCode` fields from all documents in a store using an OJAI query.

```
using System;
using MapRDB.Driver;
using MapRDB.Driver.Ojai;

public class FindQueryWithSelect
{
    public async void FindQueryWithSelect()
    {
        // Create a connection to data access server
        var connectionStr = $"localhost:5678?auth=basic;" +
            $"user=mapr;" +
```


last-updated: Apr 11, 2022

```
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
            $"sslTargetNameOverride=node1.mapr.com";
        var connection = ConnectionFactory.CreateConnection(connectionStr);

        // Get a store and assign it as a DocumentStore object
        var store = connection.GetStore("/demo_table");

        //Create an OJAI query
        var query = connection.NewQuery().Select("_id",
"address.zipCode").Build();

        // Options for find request
        var options = new QueryOptions(1000, true);

        // Fetch OJAI Documents by query
        var queryResult = store.Find(query, options);

        var documentStream = await
queryResult.GetDocumentAsyncStream().GetAllDocuments();
        // Print OJAI Documents from document stream
        foreach (var document in documentStream)
        {
            Console.WriteLine(document.ToDictionary());
        }

        // Close the OJAI connection
        connection.Close();
    }
}
```

Go

The following example shows how to retrieve the `_id` and `address.zipCode` fields from all documents in a store using an OJAI query.

```
package main

import (
    "fmt"
    client "github.com/mapr/private-maprdb-go-client"
)

func main() {
    // Create connection string
    connectionString := "192.168.33.11:5678?" +
        "auth=basic;" +
        "user=mapr;" +
        "password=mapr;" +
        "ssl=true;" +
        "sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
        "sslTargetNameOverride=node1.cluster.com"

    storeName := "/demo_table"

    // Create a connection to DAG
    connection, err := client.MakeConnection(connectionString)
    if err != nil {
        panic(err)
    }
```

```
    // Get a store and assign it as a DocumentStore struct
    store, err := connection.GetStore(storeName)
    if err != nil {
        panic(err)
    }

    // Options for find request
    options := &client.FindOptions{ResultAsDocument: true}

    // Create an OJAI query
    query := map[string]interface{}{"$select": []interface{}{"_id",
"address.zipCode"}}

    // Fetch all OJAI Documents from table
    findResult, err := store.FindQueryMap(query, options)

    // Print OJAI Documents from document stream
    for _, doc := range findResult.DocumentList() {
        fmt.Println(doc)
    }

    // Close connection
    connection.Close()
}
```

Querying with Conditions

The examples in this section query a document store and return documents that have specific conditions.

For more information about how to specify query conditions in OJAI, see Query Conditions in OJAI Applications on page 2590.

Note: You can improve the performance of queries with conditions by using secondary indexes. See Queries that Benefit from Secondary Indexes on page 570 for more details.

Java - OJAI `QueryCondition` Object

The following example shows how to return all documents from a store where `address.zipCode` equals 95196, using the Query.where method. It uses an OJAI QueryCondition to specify the condition.

```
/**
 * Copyright (c) 2017 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License"); you may
not use this file except in compliance with
 * the License. You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on
 * an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
express or implied. See the License for the
 * specific language governing permissions and limitations under the
License.
 */
package com.mapr.ojai.examples;

import org.ojai.Document;
import org.ojai.DocumentStream;
import org.ojai.store.Connection;
import org.ojai.store.DocumentStore;
import org.ojai.store.DriverManager;
```

```
import org.ojai.store.Query;
import org.ojai.store.QueryCondition.Op;

public class OJAI_007_FindQueryWithCondition {

  public static void main(final String[] args) {

    System.out.println("==== Start Application ===");

    // Create an OJAI connection to MapR cluster
    final Connection connection = DriverManager.getConnection("ojai:mapr:");

    // Get an instance of OJAI DocumentStore
    final DocumentStore store = connection.getStore("/demo_table");

    // Build an OJAI query with QueryCondition
    final Query query = connection.newQuery()
        .where(
            connection.newCondition()                    //
                .is("address.zipCode", Op.EQUAL, 95196)   // Build an OJAI
QueryCondition
                .build())                                  //
        .build();

    // fetch all OJAI Documents from this store
    final DocumentStream stream = store.find(query);

    for (final Document userDocument : stream) {
      // Print the OJAI Document
      System.out.println(userDocument.asJsonString());
    }

    // Close this instance of OJAI DocumentStore
    store.close();

    // close the OJAI connection and release any resources held by the
connection
    connection.close();

    System.out.println("==== End Application ===");
  }

}
```

Java - OJAI Query Condition in JSON Format

The following example shows how to return all documents from a store where `address.zipCode` equals 95196, using the Query.where method. It specifies the query condition using a JSON string.

```
/**
 * Copyright (c) 2017 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License"); you may
not use this file except in compliance with
 * the License. You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on
 * an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
express or implied. See the License for the
```

```
 * specific language governing permissions and limitations under the
License.
 */
package com.mapr.ojai.examples;

import org.ojai.Document;
import org.ojai.DocumentStream;
import org.ojai.store.Connection;
import org.ojai.store.DocumentStore;
import org.ojai.store.DriverManager;
import org.ojai.store.Query;

public class OJAI_008_FindQueryWithConditionJson {

  public static void main(final String[] args) {

    System.out.println("==== Start Application ===");

    // Create an OJAI connection to MapR cluster
    final Connection connection = DriverManager.getConnection("ojai:mapr:");

    // Get an instance of OJAI DocumentStore
    final DocumentStore store = connection.getStore("/demo_table");

    // Build an OJAI query with the condition specified as a JSON string
    final Query query = connection.newQuery()
        .where("{\"$eq\": {\"address.zipCode\": 95196}}")
        .build();

    // fetch all OJAI Documents from this store
    final DocumentStream stream = store.find(query);

    for (final Document userDocument : stream) {
      // Print the OJAI Document
      System.out.println(userDocument.asJsonString());
    }

    // Close this instance of OJAI DocumentStore
    store.close();

    // close the OJAI connection and release any resources held by the
connection
    connection.close();

    System.out.println("==== End Application ===");
  }

}
```

Node.js - OJAI Query Condition in JSON Format

The following example uses an OJAI query condition specified in JSON format to return all documents from a store where `address.zipCode` equals 95196.

```
/*
 * Copyright (c) 2018 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
```

```
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

const { ConnectionManager } = require('node-maprdb');

const connectionString = 'localhost:5678?' +
  'auth=basic;' +
  'user=mapr;' +
  'password=mapr;' +
  'ssl=true;' +
  'sslCA=/opt/mapr/conf/ssl_truststore.pem;' +
  'sslTargetNameOverride=node1.mapr.com';

let connection;

// Create a connection to data access server
ConnectionManager.getConnection(connectionString)
  .then((conn) => {
    connection = conn;
    // Get a store
    return connection.getStore('/demo_table');
  })
  .then((store) => {
    // Create an OJAI query
    const query = {"$where": {"$eq": { 'address.zipCode': 95196 }}};
    // fetch OJAI Documents by query
    return store.find(query);
  })
  .then((queryResult) => {
    queryResult.on('data', (document) => {
      // Print OJAI Documents from document stream
      console.log(document);
    });
    queryResult.on('end', () => {
      // close the OJAI connection
      connection.close();
    });
  });
```

Python - OJAI `QueryCondition` Object

The following example shows how to return all documents from a store where `address.zipCode` equals 95196, using the Query.where method. It uses an OJAI QueryCondition to specify the condition.

```
from mapr.ojai.ojai_query.QueryOp import QueryOp
from mapr.ojai.storage.ConnectionFactory import ConnectionFactory

# Create a connection to data access server
connection_str = "localhost:5678?auth=basic;user=mapr;password=mapr;" \
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/ssl_truststore.pem;" \
          "sslTargetNameOverride=node1.mapr.com"
connection = ConnectionFactory.get_connection(connection_str=connection_str)

# Get a store and assign it as a DocumentStore object
store = connection.get_store('/demo_table')
```

```
# Create an OJAI query
query = connection.new_query()\
    .where(connection.new_condition()
           .is_('address.zipCode', QueryOp.EQUAL, 95196)
           .close()
           .build())\
    .build()

# fetch the OJAI Documents by query
query_result = store.find(query)

# Print OJAI Documents from document stream
for doc in query_result:
    print(doc)

# close the OJAI connection
connection.close()
```

Python - OJAI Query Condition in JSON Format

The following example uses an OJAI query condition specified in JSON format to return all documents from a store where `address.zipCode` equals 95196.

```
from mapr.ojai.storage.ConnectionFactory import ConnectionFactory

# Create a connection to data access server
connection_str = "localhost:5678?auth=basic;user=mapr;password=mapr;" \
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/ssl_truststore.pem;" \
          "sslTargetNameOverride=node1.mapr.com"
connection = ConnectionFactory.get_connection(connection_str=connection_str)

# Get a store and assign it as a DocumentStore object
store = connection.get_store('/demo_table')

# Create an OJAI query
query = {"$where": {"$eq": {"address.zipCode": 95196}}}

# options for find request
options = {
    'ojai.mapr.query.result-as-document': True
    }

# fetch OJAI Documents by query
query_result = store.find(query, options=options)

# Print OJAI Documents from document stream
for doc in query_result:
    print(doc.as_dictionary())

# close the OJAI connection
connection.close()
```

dbshell

The following two dbshell commands are equivalent to the code examples. See dbshell find or findbyid on page 5290 for more details about the syntax dbshell provides.

```
# mapr dbshell
maprdb root:> find /demo_table --q {"$where":{"$eq":
{"address.zipCode":95196}}}
```

```
maprdb root:> find /demo_table --where {"$eq":
{"address.zipCode":95196}}
```

C# - OJAI `QueryCondition` Object

The following example shows how to return all documents from a store where `address.zipCode` equals 95196, using the `Query.Where` method. It uses an OJAI QueryCondition to specify the condition.

```
using System;
using MapRDB.Driver;
using MapRDB.Driver.Ojai;

public class FindQueryWithCondition
{
    public async void FindQueryWithCondition()
    {
        // Create a connection to data access server
        var connectionStr = $"localhost:5678?auth=basic;" +
            $"user=mapr;" +
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
            $"sslTargetNameOverride=node1.mapr.com";
        var connection = ConnectionFactory.CreateConnection(connectionStr);

        // Get a store and assign it as a DocumentStore object
        var store = connection.GetStore("/demo_table");

        //Create an OJAI query
        var query = connection
            .NewQuery()
                .Where(connection
                        .NewQueryCondition()
                            .Is("address.zipCode", QueryOp.EQUAL, 95196)
                            .Close()
                        .Build())
                 .Build();

        // Fetch the OJAI Documents by query
        var queryResult = store.Find(query);

        var documentStream = await
queryResult.GetDocumentAsyncStream().GetAllDocuments();
        // Print OJAI Documents from document stream
        foreach (var document in documentStream)
        {
            Console.WriteLine(document);
        }

        // Close the OJAI connection
        connection.Close();
    }
}
```

C# - OJAI Query Condition in JSON Format

The following example uses an OJAI query condition specified in JSON format to return all documents from a store where `address.zipCode` equals 95196.

```
using System;
using MapRDB.Driver;
using MapRDB.Driver.Ojai;

public class FindQueryWithConditionJson
{
    public async void FindQueryWithConditionJson()
    {
        // Create a connection to data access server
        var connectionStr = $"localhost:5678?auth=basic;" +
            $"user=mapr;" +
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
            $"sslTargetNameOverride=node1.mapr.com";
        var connection = ConnectionFactory.CreateConnection(connectionStr);

        // Get a store and assign it as a DocumentStore object
        var store = connection.GetStore("/demo_table");

        // Create an OJAI query
        var query =
            @"{" +
                @"""$where"":" +
                    @"{" +
                        @"""$eq"":{""address.zipCode"":
{""$numberLong"":""95196""}}" +
                    @"}" +
            @"}";

        // Fetch OJAI Documents by query
        var queryResult = store.FindQuery(query);

        var documentStream = await queryResult.GetAllDocuments();
        // Print OJAI Documents from document stream
        foreach (var document in documentStream)
        {
            Console.WriteLine(document.ToDictionary());
        }

        // Close the OJAI connection
        connection.Close();
    }
}
```

Go - OJAI `QueryCondition` Object

The following example shows how to return all documents from a store where `address.zipCode` equals 95196, using the `Query.WhereCondition` function. It uses an OJAI Condition to specify the condition.

```
package main

import (
    "fmt"
    client "github.com/mapr/private-maprdb-go-client"
)
```

```
func main() {
    // Create connection string
    connectionString := "192.168.33.11:5678?" +
        "auth=basic;" +
        "user=mapr;" +
        "password=mapr;" +
        "ssl=true;" +
        "sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
        "sslTargetNameOverride=node1.cluster.com"

    storeName := "/demo_table"

    // Create a connection to DAG
    connection, err := client.MakeConnection(connectionString)
    if err != nil {
        panic(err)
    }

    // Get a store and assign it as a DocumentStore struct
    store, err := connection.GetStore(storeName)
    if err != nil {
        panic(err)
    }

    // Options for find request
    options := &client.FindOptions{ResultAsDocument: true}

    // Create a condition
    condition, err := client.MakeCondition(client.Is("address.zipCode",
client.EQUAL, 95196), client.Close())
    if err != nil {
        panic(err)
    }
    condition.Build()

    // Create an OJAI query
    query, err := client.MakeQuery(client.WhereCondition(condition))
    if err != nil {
        panic(err)
    }
    query.Build()

    // Fetch all OJAI Documents from table
    findResult, err := store.FindQuery(query, options)

    // Print OJAI Documents from document stream
    for _, doc := range findResult.DocumentList() {
        fmt.Println(doc)
    }

    // Close connection
    connection.Close()
}
```

Go - OJAI Query Condition in JSON Format

The following example uses an OJAI query condition specified in JSON format to return all documents from a store where `address.zipCode` equals 95196.

```
package main

import (
```

```
    "fmt"
    client "github.com/mapr/private-maprdb-go-client"
)

func main() {
    // Create connection string
    connectionString := "192.168.33.11:5678?" +
        "auth=basic;" +
        "user=mapr;" +
        "password=mapr;" +
        "ssl=true;" +
        "sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
        "sslTargetNameOverride=node1.cluster.com"

    storeName := "/demo_table"

    // Create a connection to DAG
    connection, err := client.MakeConnection(connectionString)
    if err != nil {
        panic(err)
    }

    // Get a store and assign it as a DocumentStore struct
    store, err := connection.GetStore(storeName)
    if err != nil {
        panic(err)
    }

    // Options for find request
    options := &client.FindOptions{ResultAsDocument: true}

    // Create an OJAI query
    query := map[string]interface{}{
        "$where": map[string]interface{}{
            "$eq": map[string]interface{}{
                "address.zipCode": 95196}}}

    // Fetch all OJAI Documents from table
    findResult, err := store.FindQueryMap(query, options)

    // Print OJAI Documents from document stream
    for _, doc := range findResult.DocumentList() {
        fmt.Println(doc)
    }

    // Close connection
    connection.Close()
}
```

Querying with Select and Conditions
The examples in this section query a document store and return specific fields from documents that have specific conditions.

For more information about how to specify query conditions in OJAI, see Query Conditions in OJAI Applications on page 2590.

Note: You can improve the performance of queries with conditions by using secondary indexes. See Queries that Benefit from Secondary Indexes on page 570 for more details.

Java

The following example shows how to return the `name`, `address.zipCode`, `age`, and `phoneNumber` fields from documents that have `address.zipCode` equal to 95196. It uses the Query.select and Query.where methods, specifying the query condition as a JSON string.

```
/**
 * Copyright (c) 2017 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License"); you may
not use this file except in compliance with
 * the License. You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on
 * an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
express or implied. See the License for the
 * specific language governing permissions and limitations under the
License.
 */
package com.mapr.ojai.examples;

import org.ojai.Document;
import org.ojai.DocumentStream;
import org.ojai.store.Connection;
import org.ojai.store.DocumentStore;
import org.ojai.store.DriverManager;
import org.ojai.store.Query;

public class OJAI_009_FindQueryWithSelectAndCondition {

  public static void main(final String[] args) {

    System.out.println("==== Start Application ===");

    // Create an OJAI connection to MapR cluster
    final Connection connection = DriverManager.getConnection("ojai:mapr:");

    // Get an instance of OJAI DocumentStore
    final DocumentStore store = connection.getStore("/demo_table");

    // Build an OJAI query with the condition specified as a JSON string
    final Query query = connection.newQuery()
        .select("name",
"address.zipCode").select("age").select("phoneNumbers[0]")
        .where("{\"$eq\": {\"address.zipCode\": 95196}}")
        .build();

    // fetch all OJAI Documents from this store
    final DocumentStream stream = store.find(query);

    for (final Document userDocument : stream) {
      // Print the OJAI Document
      System.out.println(userDocument.asJsonString());
    }

    // Close this instance of OJAI DocumentStore
    store.close();

    // close the OJAI connection and release any resources held by the
connection
```

```
    connection.close();

    System.out.println("==== End Application ===");
  }

}
```

Node.js

The following example shows how to return the `name`, `address.zipCode`, `age`, and `phoneNumber` fields from documents that have `address.zipCode` equal to 95196. It uses an OJAI query and condition.

```
/*
 * Copyright (c) 2018 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

const { ConnectionManager } = require('node-maprdb');

const connectionString = 'localhost:5678?' +
  'auth=basic;' +
  'user=mapr;' +
  'password=mapr;' +
  'ssl=true;' +
  'sslCA=/opt/mapr/conf/ssl_truststore.pem;' +
  'sslTargetNameOverride=node1.mapr.com';

let connection;

// Create a connection to data access server
ConnectionManager.getConnection(connectionString)
  .then((conn) => {
    connection = conn;
    // Get a store
    return connection.getStore('/demo_table');
  })
  .then((store) => {
    // Create an OJAI query
    const query = {"$select": ["name",
        "adress.zipCode",
        "age",
        "phoneNumbers[0]"],
      "$where": {"$eq": {"address.zipCode": 95196}}};
    // fetch OJAI Documents by query
    return store.find(query);
  })
  .then((queryResult) => {
    queryResult.on('data', (document) => {
      // Print OJAI Documents from document stream
      console.log(document);
    });
```

```
    queryResult.on('end', () => {
      // close the OJAI connection
      connection.close();
    });
  });
```

Python

The following example shows how to return the `name`, `address.zipCode`, `age`, and `phoneNumber` fields from documents that have `address.zipCode` equal to 95196. It uses an OJAI query and condition.

```
from mapr.ojai.storage.ConnectionFactory import ConnectionFactory

# Create a connection to data access server
connection_str = "localhost:5678?auth=basic;user=mapr;password=mapr;" \
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/ssl_truststore.pem;" \
          "sslTargetNameOverride=node1.mapr.com"
connection = ConnectionFactory.get_connection(connection_str=connection_str)

# Get a store and assign it as a DocumentStore object
store = connection.get_store('/demo_table')

# Create an OJAI query
query = {"$select": ["name",
                     "adress.zipCode",
                     "age",
                     "phoneNumbers[0]"],
         "$where": {"$eq": {"address.zipCode": 95196}}}

# options for find request
options = {
    'ojai.mapr.query.result-as-document': True
    }

# fetch OJAI Documents by query
query_result = store.find(query,
                          options=options)

# Print OJAI Documents from document stream
for doc in query_result:
    print(doc.as_dictionary())

# close the OJAI connection
connection.close()
```

dbshell

The following two dbshell commands are equivalent to the code examples. See dbshell find or findbyid on page 5290 for more details about the syntax dbshell provides.

```
find /demo_table --query {
   "$select":["name","address.zipCode","age","phoneNumber[0]"],
   "$where":{"$eq":{"address.zipCode":95196}}
}

find /demo_table
   --fields name,address.zipCode,age,phoneNumber[0]
   --where {"$eq":{"address.zipCode":95196}}
```

Note: The commands are shown split across multiple lines for readability. When using dbshell, you must enter them in a single line.

C#

The following example shows how to return the `name`, `address.zipCode`, `age`, and `phoneNumber` fields from documents that have `address.zipCode` equal to 95196. It uses an OJAI query and condition.

```
using System;
using MapRDB.Driver;
using MapRDB.Driver.Ojai;

public class FindQueryWithSelectAndCondition
{
    public async void FindQueryWithSelectAndCondition()
    {
        // Create a connection to data access server
        var connectionStr = $"localhost:5678?auth=basic;" +
            $"user=mapr;" +
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
            $"sslTargetNameOverride=node1.mapr.com";
        var connection = ConnectionFactory.CreateConnection(connectionStr);

        // Get a store and assign it as a DocumentStore object
        var store = connection.GetStore("/demo_table");

        // Create an OJAI condition
        var condition = connection
            .NewQueryCondition()
                .Is("address.zipCode", QueryOp.EQUAL, 95196)
                .Close()
            .Build();

        // Create an OJAI query
        var query = connection
            .NewQuery()
                .Select("name", "adress.zipCode", "age", "phoneNumbers[0]")
                .Where(condition)
            .Build();

        // Fetch OJAI Documents by query
        var queryResult = store.Find(query);

        var documentStream = await
queryResult.GetDocumentAsyncStream().GetAllDocuments();
        // Print OJAI Documents from document stream
        foreach (var document in documentStream)
        {
            Console.WriteLine(document.ToJsonString());
        }

        // Close the OJAI connection
        connection.Close();
    }
}
```

Go

The following example shows how to return the `name`, `address.zipCode`, `age`, and `phoneNumber` fields from documents that have `address.zipCode` equal to 95196. It uses an OJAI query and condition.

```
package main

import (
    "fmt"
    client "github.com/mapr/private-maprdb-go-client"
)

func main() {
    // Create connection string
    connectionString := "192.168.33.11:5678?" +
        "auth=basic;" +
        "user=mapr;" +
        "password=mapr;" +
        "ssl=true;" +
        "sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
        "sslTargetNameOverride=node1.cluster.com"

    storeName := "/demo_table"

    // Create a connection to DAG
    connection, err := client.MakeConnection(connectionString)
    if err != nil {
        panic(err)
    }

    // Get a store and assign it as a DocumentStore struct
    store, err := connection.GetStore(storeName)
    if err != nil {
        panic(err)
    }

    // Options for find request
    options := &client.FindOptions{ResultAsDocument: true}

    // Create an OJAI query
    query := map[string]interface{}{"$select": []interface{}{"firstName",
"address.zipCode", "age", "phoneNumbers[0]"},
        "$where": map[string]interface{}{
            "$eq": map[string]interface{}{"address.zipCode": 95196}}}

    // Fetch all OJAI Documents from table
    findResult, err := store.FindQueryMap(query, options)

    // Print OJAI Documents from document stream
    for _, doc := range findResult.DocumentList() {
        fmt.Println(doc)
    }

    // Close connection
    connection.Close()
}
```

Querying with Order By

The examples in this section query a document store and return specific fields from the documents, sorted in a specific order. One of the examples also uses offset and limit.

last-updated: Apr 11, 2022

Note: You can improve the performance of order by queries by using secondary indexes. See Using Indexes to Optimize ORDER BY Queries on page 577 for more information.

Java - Order By

The following example shows how to return the `_id`, `firstName`, `lastName`, and `address.zipCode` fields from documents in a store, sorting the documents by `_id`. It uses the Query.select and Query.orderBy methods.

Note: The example sorts in the default ascending order. To sort in descending order, modify the `orderby` method call as follows:

```
orderBy("_id", SortOrder.DESC)
```

```
/**
 * Copyright (c) 2017 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License"); you may
not use this file except in compliance with
 * the License. You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on
 * an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
express or implied. See the License for the
 * specific language governing permissions and limitations under the
License.
 */
package com.mapr.ojai.examples;

import org.ojai.Document;
import org.ojai.DocumentStream;
import org.ojai.store.Connection;
import org.ojai.store.DocumentStore;
import org.ojai.store.DriverManager;
import org.ojai.store.Query;

public class OJAI_010_FindQueryWithOrderBy {

  public static void main(final String[] args) {

    System.out.println("==== Start Application ===");

    // Create an OJAI connection to MapR cluster
    final Connection connection = DriverManager.getConnection("ojai:mapr:");

    // Get an instance of OJAI DocumentStore
    final DocumentStore store = connection.getStore("/demo_table");

    // Build an OJAI query with an order by
    final Query query = connection.newQuery()
        .select("_id", "firstName", "lastName", "address.zipCode")
        .orderBy("_id")
        .build();

    // fetch all OJAI Documents from this store
    final DocumentStream stream = store.find(query);
```

```
    for (final Document userDocument : stream) {
      // Print the OJAI Document
      System.out.println(userDocument.asJsonString());
    }

    // Close this instance of OJAI DocumentStore
    store.close();

    // close the OJAI connection and release any resources held by the
connection
    connection.close();

    System.out.println("==== End Application ===");
  }

}
```

Java - Order By with Offset and Limit

The following example shows how to return the `_id`, `firstName`, `lastName`, and `address.zipCode` fields from documents in a store, sorting the documents by `_id`. It uses the Query.select and Query.orderBy methods. In addition, the returned documents are offset and limited by using the Query.offset and Query.limit methods.

Note: The example sorts in the default ascending order. To sort in descending order, modify the `orderBy` method call as follows:

```
orderBy("_id", SortOrder.DESC)
```

```
/**
 * Copyright (c) 2017 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License"); you may
not use this file except in compliance with
 * the License. You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on
 * an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
express or implied. See the License for the
 * specific language governing permissions and limitations under the
License.
 */
package com.mapr.ojai.examples;

import org.ojai.Document;
import org.ojai.DocumentStream;
import org.ojai.store.Connection;
import org.ojai.store.DocumentStore;
import org.ojai.store.DriverManager;
import org.ojai.store.Query;

public class OJAI_011_FindQueryWithOrderByLimitOffset {

  public static void main(final String[] args) {

    System.out.println("==== Start Application ===");
```

```
    // Create an OJAI connection to MapR cluster
    final Connection connection = DriverManager.getConnection("ojai:mapr:");

    // Get an instance of OJAI DocumentStore
    final DocumentStore store = connection.getStore("/demo_table");

    // Build an OJAI query with an order by, offset, and limit
    final Query query = connection.newQuery()
        .select("_id", "firstName", "lastName", "address.zipCode")
        .orderBy("_id")
        .offset(2)
        .limit(1)
        .build();

    // fetch all OJAI Documents from this store
    final DocumentStream stream = store.find(query);

    for (final Document userDocument : stream) {
      // Print the OJAI Document
      System.out.println(userDocument.asJsonString());
    }

    // Close this instance of OJAI DocumentStore
    store.close();

    // close the OJAI connection and release any resources held by the
connection
    connection.close();

    System.out.println("==== End Application ===");
  }

}
```

Node.js - Order By

The following example uses an OJAI query to return the `_id` and `name` fields from documents in a store and to sort the documents by `_id`.

Note: The example sorts in the default ascending order. To sort in descending order, modify the `orderby` specification as follows:

```
order_by('_id', desc)
```

```
/*
 * Copyright (c) 2018 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

const { ConnectionManager } = require('node-maprdb');
```

```
const connectionString = 'localhost:5678?' +
  'auth=basic;' +
  'user=mapr;' +
  'password=mapr;' +
  'ssl=true;' +
  'sslCA=/opt/mapr/conf/ssl_truststore.pem;' +
  'sslTargetNameOverride=node1.mapr.com';

let connection;

// Create a connection to data access server
ConnectionManager.getConnection(connectionString)
  .then((conn) => {
    connection = conn;
    // Get a store
    return connection.getStore('/demo_table');
  })
  .then((store) => {
    // Create an OJAI query
    const query = {"$select": ["_id", "name"], "$orderby": {"_id": "asc"}};
    // fetch OJAI Documents by query
    return store.find(query);
  })
  .then((queryResult) => {
    queryResult.on('data', (document) => {
      // Print OJAI Documents from document stream
      console.log(document);
    });
    queryResult.on('end', () => {
      // close the OJAI connection
      connection.close();
    });
  });
```

Node.js - Order By with Offset and Limit

The following example uses an OJAI query to return the `_id`, `firstName`, `lastName`, and `address.zipCode` fields from documents in a store, sort the documents by `_id`, offset the result by two documents, and limit the result to a single document.

Note: The example sorts in the default ascending order. To sort in descending order, modify the `orderby` specification as follows:

```
"$orderby": {"_id": "desc"}
```

```
/*
 * Copyright (c) 2018 MapR, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
```


last-updated: Apr 11, 2022

```
const { ConnectionManager } = require('node-maprdb');

const connectionString = 'localhost:5678?' +
  'auth=basic;' +
  'user=mapr;' +
  'password=mapr;' +
  'ssl=true;' +
  'sslCA=/opt/mapr/conf/ssl_truststore.pem;' +
  'sslTargetNameOverride=node1.mapr.com';

let connection;

// Create a connection to data access server
ConnectionManager.getConnection(connectionString)
  .then((conn) => {
    connection = conn;
    // Get a store
    return connection.getStore('/demo_table');
  })
  .then((store) => {
    // Create an OJAI query
    const query = {"$offset": 2,
      "$select": ["_id",
        "firstName",
        "lastName",
        "address.zipCode"],
      "$limit": 1,
      "$orderby": {"_id": "asc"}};
    // fetch OJAI Documents by query
    return store.find(query);
  })
  .then((queryResult) => {
    queryResult.on('data', (document) => {
      // Print OJAI Documents from document stream
      console.log(document);
    });
    queryResult.on('end', () => {
      // close the OJAI connection
      connection.close();
    });
  });
```

Python - Order By

The following example uses an OJAI query to return the `_id` and `name` fields from documents in a store and to sort the documents by `_id`.

Note: The example sorts in the default ascending order. To sort in descending order, modify the `orderby` specification as follows:

```
"$orderby": {"_id": "desc"}
```

```
from mapr.ojai.storage.ConnectionFactory import ConnectionFactory

# Create a connection to data access server
connection_str = "localhost:5678?auth=basic;user=mapr;password=mapr;" \
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/ssl_truststore.pem;" \
          "sslTargetNameOverride=node1.mapr.com"
connection = ConnectionFactory.get_connection(connection_str=connection_str)
```

```
# Get a store and assign it as a DocumentStore object
store = connection.get_store('/demo_table')

# Create an OJAI query
query = {"$select": ["_id", "name"], "$orderby": {"_id": "asc"}}

# fetch OJAI Documents by query
query_result = store.find(query)

# Print OJAI Documents from document stream
for doc in query_result:
    print(doc)

# close the OJAI connection
connection.close()
```

Python - Order By with Offset and Limit

The following example uses an OJAI query to return the `_id`, `firstName`, `lastName`, and `address.zipCode` fields from documents in a store, sort the documents by `_id`, offset the result by two documents, and limit the result to a single document.

Note: The example sorts in the default ascending order. To sort in descending order, modify the `orderby` method call as follows:

```
order_by('_id', desc)
```

```
from mapr.ojai.storage.ConnectionFactory import ConnectionFactory

# Create a connection to data access server
connection_str = "localhost:5678?auth=basic;user=mapr;password=mapr;" \
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/ssl_truststore.pem;" \
          "sslTargetNameOverride=node1.mapr.com"
connection = ConnectionFactory.get_connection(connection_str=connection_str)

# Get a store and assign it as a DocumentStore object
store = connection.get_store('/demo_table')

# Create an OJAI query
query = {"$offset": 2,
         "$select": ["_id",
                     "firstName",
                     "lastName",
                     "address.zipCode"],
         "$limit": 1,
         "$orderby": {"_id": "asc"}}

# options for find request
options = {
    'ojai.mapr.query.result-as-document': True
    }

# fetch OJAI Documents by query
query_result = store.find(query, options=options)

# Print OJAI Documents from document stream
for doc in query_result:
    print(doc.as_dictionary())
```


last-updated: Apr 11, 2022

```
# close the OJAI connection
connection.close()
```

dbshell

The following dbshell commands are equivalent to the code examples. See dbshell find or findbyid on page 5290 for more details about the syntax dbshell provides.

```
find /demo_table --query {
  "$select":["_id","firstName","lastName","address.zipCode"],
  "$orderby":"_id"
}

find /demo_table
   --fields _id,firstName,lastName,address.zipCode
   --orderby _id

find /demo_table --query {
  "$select":["_id","firstName","lastName","address.zipCode"],
  "$orderby":"_id",
  "$offset":2,
  "$limit":1
}

find /demo_table
   --fields _id,firstName,lastName,address.zipCode
   --orderby _id
   --offset 2
   --limit 1
```

Note: The commands are shown split across multiple lines for readability. When using dbshell, you must enter them in a single line.

C# - Order By

The following example uses an OJAI query to return the `_id` and `name` fields from documents in a store and to sort the documents by `_id`.

Note: The example sorts in the default ascending order. To sort in descending order, modify the `OrderBy` specification as follows:

```
.OrderBy("_id", SortOrder.DESC)
```

```
using System;
using MapRDB.Driver;
using MapRDB.Driver.Ojai;

public class FindQueryWithOrderBy
{
    public async void FindQueryWithOrderBy()
    {
        // Create a connection to data access server
        var connectionStr = $"localhost:5678?auth=basic;" +
            $"user=mapr;" +
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
            $"sslTargetNameOverride=node1.mapr.com";
        var connection = ConnectionFactory.CreateConnection(connectionStr);
```

```
        // Get a store and assign it as a DocumentStore object
        var store = connection.GetStore("/demo_table");

        // Create an OJAI query
        var query = connection
            .NewQuery()
                .Select("_id", "name")
                .OrderBy("_id", SortOrder.ASC)
            .Build();

        // Fetch OJAI Documents by query
        var queryResult = store.Find(query);

        var documentStream = await
queryResult.GetDocumentAsyncStream().GetAllDocuments();
        // Print OJAI Documents from document stream
        foreach (var document in documentStream)
        {
            Console.WriteLine(document);
        }

        // Close the OJAI connection
        connection.Close();
    }
}
```

C# - Order By with Offset and Limit

The following example uses an OJAI query to return the `_id`, `firstName`, `lastName`, and `address.zipCode` fields from documents in a store, sort the documents by `_id`, offset the result by two documents, and limit the result to a single document.

Note: The example sorts in the default ascending order. To sort in descending order, modify the `OrderBy` method call as follows:

```
.OrderBy("_id", SortOrder.DESC)
```

```
using System;
using MapRDB.Driver;
using MapRDB.Driver.Ojai;

public class FindQueryWithOrderByLimitOffset
{
    public async void FindQueryWithOrderByLimitOffset()
    {
        // Create a connection to data access server
        var connectionStr = $"localhost:5678?auth=basic;" +
            $"user=mapr;" +
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
            $"sslTargetNameOverride=node1.mapr.com";
        var connection = ConnectionFactory.CreateConnection(connectionStr);

        // Get a store and assign it as a DocumentStore object
        var store = connection.GetStore("/demo_table");

        // Create an OJAI query
        var query = connection
            .NewQuery()
```

```
                .Select("_id", "firstName", "lastName", "address.zipCode")
                .Offset(2)
                .Limit(1)
                .OrderBy("_id", SortOrder.ASC)
            .Build();

        // Fetch OJAI Documents by query
        var queryResult = store.Find(query);

        var documentStream = await
queryResult.GetDocumentAsyncStream().GetAllDocuments();
        // Print OJAI Documents from document stream
        foreach (var document in documentStream)
        {
            Console.WriteLine(document.ToJsonString());
        }

        // Close the OJAI connection
        connection.Close();
    }
}
```

Go - Order By

The following example uses an OJAI query to return the `_id` and `firstName` fields from documents in a store and to sort the documents by `_id`.

The example sorts in the default ascending order. To sort in descending order, modify the `orderby` specification as follows:

```
"$orderby": map[string]interface{}{"_id": "desc"}}
```

```
package main

import (
    "fmt"
    client "github.com/mapr/private-maprdb-go-client"
)

func main() {
    // Create connection string
    connectionString := "192.168.33.11:5678?" +
        "auth=basic;" +
        "user=mapr;" +
        "password=mapr;" +
        "ssl=true;" +
        "sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
        "sslTargetNameOverride=node1.cluster.com"

    storeName := "/demo_table"

    // Create a connection to DAG
    connection, err := client.MakeConnection(connectionString)
    if err != nil {
        panic(err)
    }

    // Get a store and assign it as a DocumentStore struct
    store, err := connection.GetStore(storeName)
    if err != nil {
        panic(err)
    }
```

```
    // Options for find request
    options := &client.FindOptions{ResultAsDocument: true}

    // Create an OJAI query
    query := map[string]interface{}{"$select": []interface{}{"_id",
"firstName"},
        "$orderby": map[string]interface{}{"_id": "asc"}}

    // Fetch all OJAI Documents from table
    findResult, err := store.FindQueryMap(query, options)

    // Print OJAI Documents from document stream
    for _, doc := range findResult.DocumentList() {
        fmt.Println(doc)
    }

    // Close connection
    connection.Close()
}
```

Go - Order By with Offset and Limit

The following example uses an OJAI query to return the `_id`, `firstName`, `lastName`, and `address.zipCode` fields from documents in a store, sort the documents by `_id`, offset the result by two documents, and limit the result to a single document.

The example sorts in the default ascending order. To sort in descending order, modify the `orderby` function call as follows:

```
"$orderby": map[string]interface{}{"_id": "desc"}}
```

```
package main

import (
    "fmt"
    client "github.com/mapr/private-maprdb-go-client"
)

func main() {
    // Create connection string
    connectionString := "192.168.33.11:5678?" +
        "auth=basic;" +
        "user=mapr;" +
        "password=mapr;" +
        "ssl=true;" +
        "sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
        "sslTargetNameOverride=node1.cluster.com"

    storeName := "/demo_table"

    // Create a connection to DAG
    connection, err := client.MakeConnection(connectionString)
    if err != nil {
        panic(err)
    }

    // Get a store and assign it as a DocumentStore struct
    store, err := connection.GetStore(storeName)
    if err != nil {
        panic(err)
    }
```

```
    // Options for find request
    options := &client.FindOptions{ResultAsDocument: true}

    // Create an OJAI query
    query := map[string]interface{}{"$select": []interface{}{"_id",
"firstName", "lastName", "address.zipCode"},
        "$offset":  2,
        "$limit":   1,
        "$orderby": map[string]interface{}{"_id": "asc"}}

    // Fetch all OJAI Documents from table
    findResult, err := store.FindQueryMap(query, options)

    // Print OJAI Documents from document stream
    for _, doc := range findResult.DocumentList() {
        fmt.Println(doc)
    }

    // Close connection
    connection.Close()
}
```

Using the Java OJAI Client
This topic describes MapR Database functionality that is applicable to only the Java OJAI client. This includes instructions on how to compile your Java OJAI application, enable buffered writes, use the read your own writes feature, and enable available query options.

Additional Resources

Examples: https://github.com/mapr-demos/ojai-examples/tree/master/java/src/main/java/com/mapr/ojai/examples

Compiling and Running Java OJAI Applications
For applications that use the Java OJAI API, use Maven to compile and determine the application's dependencies. Then, when you run the application, specify those dependencies in the application's classpath.

Compile and Determine Dependencies

Use Maven to compile and determine the application dependencies.

1. Add MapR's Maven repository to your `pom.xml` file, if it is not already added:

```
<repositories>
    <repository>
        <id>mapr-releases</id>
        <url>https://repository.mapr.com/nexus/content/repositories/
releases</url>
        <snapshots><enabled>true</enabled></snapshots>
        <releases><enabled>true</enabled></releases>
    </repository>
</repositories>
```

2. Add a dependency to the MapR OJAI driver project:

```
<dependencies>
    <dependency>
        <groupId>com.mapr.ojai</groupId>
        <artifactId>mapr-ojai-driver</artifactId>
        <version>6.0.0-mapr</version>
    </dependency>
</dependencies>
```

Note: Replace the `<version>` property with the MapR Database version that you are using.

3. Use Maven to compile the application and resolve dependencies.

Run the Application

When you develop a Java application, you can use a dependency management tool such as Maven to compile your application. However, it is recommended that do the following instead:

1. Compile the Java application without including dependencies
2. Specify the required classpath when you submit the application to the cluster

If you choose to bundle the JAR file, and there is a mismatch between the bundled JAR file and the version that your MapR cluster expects, this can result in failures. The failures differ depending on the version of MapR you are using. For more information, see Using the File System JAR to Connect to the Cluster on page 2367.

When the cluster is secure, the node must also have a MapR ticket configured for the user that runs the application.

You can use the following command to launch MapR Database JSON applications:

```
java -cp <classpath>:. -Djava.library.path=/opt/mapr/lib <main class JAR>
<command line arguments>
```

Enable OJAI Tracing

To help debug your Java OJAI application, you can enable OJAI tracing. MapR uses the `log4j` API to log tracing messages. To enable writing these messages to standard output, follow these steps:

1. Set the following property in your `/opt/mapr/conf/log4j.properties` file:

```
log4j.logger.com.mapr.ojai.store.impl=TRACE, stdout
```

2. Add the following to your `java` launch command:
   - In your java classpath, add the library that includes custom MapR `log4j` classes:

     ```
     -cp /opt/mapr/lib/central-logging-6.1.1-mapr.jar:<other classpaths>:.
     ```

   - Define the location of the `log4j.properties` file:

     ```
     -Dlog4j.configuration=file:/opt/mapr/conf/log4j.properties
     ```


last-updated: Apr 11, 2022

Reading Your Own Writes in Java OJAI

The Java OJAI `DocumentStore` and `Query` APIs provide the ability to track writes to JSON tables. Use these APIs to ensure your application reads recent writes on JSON tables with secondary indexes.

Description

You should use this feature if it is important for your query results to reflect synchronized data between a JSON table and its secondary indexes. Because MapR Database updates secondary indexes asynchronously, it is possible for a JSON table and its secondary indexes to become out-of-sync while the index update is in progress.

For example, consider the following scenario:

- Your application updates a JSON table.
- The JSON table includes an `address` field that is a nested document with a `zipCode` subfield.
- You have a secondary index on `zipCode`.
- Later in your application, you query the JSON table filtering on `zipCode`.

You want your query result to reflect the updates from earlier in your application. To achieve this, use the `DocumentStore` and `Query` APIs that enable you to retrieve up-to-date information from the index. The APIs synchronize write operations on the JSON table with read operations on a secondary index.

See Asynchronous Secondary Index Updates on page 587 for more information about index updates.

Note: The Python and Node.js OJAI APIs do not support this feature.

API Details

The OJAI `DocumentStore` and `Query` interfaces provide the following methods to support this functionality.

`DocumentStore.beginTrackingWrites`	Begins tracking the write operations performed through this instance of `DocumentStore`. The method takes an optional `previousWritesContext` parameter. If you specify this parameter, the tracking uses that context as the base state.
`DocumentStore.endTrackingWrites`	Flushes any buffered writes operations for this `DocumentStore` and returns a `writesContext`. Use this context to ensure that writes are visible to later queries. You can use the context across `DocumentStore` objects in the same, as well as different, client processes, when the stores refer to the same JSON table. For example, you can pass the `writesContext` returned by one `DocumentStore` to a second `DocumentStore`, to begin write tracking on the second store.
`DocumentStore.clearTrackedWrites`	Stops the write tracking and clears any state on this `DocumentStore` instance.
`Query.waitForTrackedWrites`	Sets the `writesContext` parameter for this query. A `writesContext` allows this query to "see" all the writes that happened inside the `writesContext` of a `DocumentStore`.

For the complete API, see Java OJAI Client API.

last-updated: Apr 11, 2022

Read Your Own Writes Example

A complete code example is available on github at OJAI_013_ReadYourOwnWrite.java. The following are code snippets from that example. Each step contains links to corresponding lines of code in the github example:

1. Call beginWriteTracking to set the starting point for the commit context on the JSON table `/demo_table`:

```
// Create an OJAI connection to MapR cluster
final Connection connectionNode1 =
DriverManager.getConnection("ojai:mapr:");

// Get an instance of OJAI DocumentStore
final DocumentStore storeNode1 = connectionNode1.getStore("/demo_table");

// initiate tracking of commit-context
storeNode1.beginTrackingWrites();
```

2. Update the `zipCode` of an existing user and insert a new user in `/demo_table`:

```
// issue a set of mutations/insert/delete/etc
storeNode1.update("user0000",
connectionNode1.newMutation().set("address.zipCode", 95110L));
storeNode1.insertOrReplace(connectionNode1.newDocument(
    "{\"_id\": \"user0004\", \"firstName\": \"Joel\",
\"lastName\": \"Smith\", \"age\": 56, \"address\": {\"zipCode\":
{\"$numberLong\":95110}}}"));
```

3. Call endWriteTracking to flush the write operations after step 1, including updates to the secondary index:

```
final String commitContext = storeNode1.endTrackingWrites();
```

   The call also returns a `commitContext`.

4. Issue a query that calls waitForTrackedWrites with the `commitContext` from step 3:

```
/*
 * Next section of the code can run on the same or on a different node,
 * the `commitContext` obtained earlier needs to be propagated to that
node.
 */

// Create an OJAI connection to MapR cluster
final Connection connectionNode2 =
DriverManager.getConnection("ojai:mapr:");

// Get an instance of OJAI DocumentStore
final DocumentStore storeNode2 = connectionNode2.getStore("/demo_table");

// Build an OJAI query and set its commit context with timeout of 2
seconds
final Query query = connectionNode2.newQuery()
    .select("_id", "firstName", "lastName", "address.zipCode")
    .where("{\"$gt\": {\"address.zipCode\": 95110}}")
    .waitForTrackedWrites(commitContext)
    .build();
```

   The query filters on the indexed subfield `address.zipCode`. The `commitContext` ensures that the query result includes the changes made in step 2.

Setting Query Options in Java OJAI
This topic describes how to set query options in your Java OJAI application.

See OJAI Query Options on page 2588 for a list of available query options.

To set an option, pass the option name as the first parameter to the Query.setOption method:

```
query.setOption("ojai.mapr.query.hint-using-index", indexName);
```

Enabling Buffered Writes in Java OJAI
By default, MapR Database JSON does not buffer writes. You can improve performance by enabling buffered writes in your Java OJAI application.

Description

The buffered writes option can be set in the Connection.getStore method. You pass the option setting through a `Document` object in the second parameter to the method. The `Document` object sets `ojai.mapr.documentstore.buffer-writes` to either `true` or `false`. The default value is `false`, which means that writes are not buffered.

Example Code Snippet

The following code sample enables buffered writes:

```
final DocumentStore store =
    connection.getStore(
        "/demo_table",

connection.newDocument().set("ojai.mapr.documentstore.buffer-writes",
true));
```

Using the Java OJAI Thin Client
Starting with EEP 6.3.0, you can use the Java OJAI Thin Client to write MapR Database JSON applications. The Java OJAI Thin Client provides a lightweight library that supports the OJAI API. You

can connect to MapR Database JSON, and add, update, and query documents in a MapR Database JSON table.

Java OJAI Thin Client Benefits

The client provides you with the following benefits:

- Easy installation and use
- Access to MapR Database JSON through the OJAI interface
- An OJAI interface that is tailored to Java developers
- Support for L3/L4 (transport level) and L7 (application level) proxy load balancing

Comparing the Java OJAI Client and the Java OJAI Thin Client

Note these considerations when deciding whether to use the Java OJAI Client or the Java OJAI Thin Client:

- Both the Java OJAI client and Java OJAI Thin Client use the same API (Java OJAI Client API).
- The Java OJAI client is more scalable, more performant, and more fault tolerant, but also more complicated to deploy. See Using the Java OJAI Client on page 2666.
- The Java OJAI Thin Client requires you to specify the service (MapR Data Access Gateway) to which you will connect.
- The Java OJAI Thin Client is a pure Java client, while the Java OJAI client requires a JNI library.

Installing the MapR Data Access Gateway

To use the Java OJAI Thin Client, you must install the MapR Data Access Gateway on your MapR cluster. The gateway serves as a proxy for translating requests between the Java OJAI Thin Client and the MapR cluster. To administer the gateway and configure load balancing, see Administering the MapR Data Access Gateway on page 1492.

Java OJAI Thin Client Security

The client supports username/password authentication. The initial connection (and token renewal) use these credentials. Subsequent communication uses JWT.

When connecting to a secure MapR cluster, the client uses:

- X.509 certificates to authenticate with the MapR Data Access Gateway
- TLS v1.2 to encrypt communication between the client and the Data Access Gateway

Java OJAI Thin Client Connection String

The string you use to connect your OJAI client to a MapR cluster must have the following format:

```
"ojai:mapr:thin:@<hostname>[:<port>][?<option_name>=<option_value>;...]"
```

<hostname>	Name of the MapR Data Access Gateway host
<port>	Port number (see Ports Used by MapR Software on page 2290) that gRPC clients use to connect to the MapR Data Access Gateway Default: 5678

last-updated: Apr 11, 2022

auth=<scheme_name>	The authentication scheme for the current connection; currently, only `basic`
user=<username>	The user name for `basic` authentication
password=<password>	The password for `basic` authentication
ssl=true|false	Whether to establish a secure connection using SSL/TLS An error is returned if there is a mismatch between your client and cluster security settings. The default for this option is `true`, which is the required setting if connecting to a secure MapR cluster. If connecting to a nonsecure MapR cluster, set it to `false`. If set to `false`, the other SSL parameters are ignored.
sslCA=<path to PEM file containing CA certificate>	Path to a local file containing Certificate Authority (CA) signed certificates in PEM format. Must be set if the `ssl` option is `true`.

Here is an example of a connection string:

```
"ojai:mapr:thin:@localhost:5678?
auth=basic;user=fred;password=george;sslCA=/opt/app/conf/rootca.pem"
```

Advanced Parameters

Advanced parameters such as `maxmsgsize` are optional for the Java OJAI Thin Client connection string:

maxmsgsize	If you use thin-client version `1.0.2-mapr` and later, `maxmsgsize` sets the maximum message size that the gRPC client accepts. The default is set to 32 MB, as this is the default maximum document size for MapR Database JSON tables. The value specified in the connection string should be less than or equal to the value set in the Data Access Gateway configuration on the server side (see Administering the MapR Data Access Gateway on page 1492).

Maven Coordinates

The Maven coordinates are:

```
<dependency>
    <groupId>com.mapr.ojai</groupId>
    <artifactId>mapr-ojai-driver-thin</artifactId>
    <version>1.0.2-mapr</version>
</dependency>
```

Additional Resources

The Java OJAI Client examples at the following location also apply to the Java OJAI Thin Client. Only the connection string and the Maven artifact name will be different for the thin client: https://github.com/mapr-demos/ojai-examples/tree/master/java/src/main/java/com/mapr/ojai/examples

Using the Node.js OJAI Client

Starting with EEP 6.0, you can use the Node.js OJAI client to write MapR Database JSON applications. The client provides you with a lightweight library that supports the OJAI API. You can connect to MapR Database JSON from middleware components, and add, update, and query documents in a MapR Database JSON table.

The client provides you with the following benefits:

- Easy installation and use
- Access to MapR Database JSON through the OJAI interface in Node.js
- An OJAI interface that is tailored to Node.js developers
- Support for Callback and Promise/Async Node.js asynchronous programming models
- Support for L3/L4 (transport level) and L7 (application level) proxy load balancing
- Use of JavaScript to manipulate MapR Database JSON documents

To use the Node.js OJAI client, you must install the MapR Data Access Gateway on your MapR cluster. The gateway serves as a proxy for translating requests between the Node.js client and the MapR cluster. The gateway also performs data processing to keep the client lightweight. See Administering the MapR Data Access Gateway on page 1492 for information about how to administer the gateway and configure load balancing.

Important: MapR does not support running the Node.js OJAI client in a web browser.

Additional Resources

Examples: https://github.com/mapr-demos/ojai-examples/tree/master/nodejs

Source Code: https://github.com/mapr/maprdb-node-client

Getting Started with the Node.js OJAI Client

This section describes the software required to run the Node.js OJAI client, client/server security, and how to specify your connection string. It also provides links to documentation that shows you how to write Node.js OJAI applications.

The Node.js OJAI client is available starting in the EEP 6.0 release.

Software Requirements

You must have the following software installed to run the client:

Client Software	Installation Notes
Node.js	Supported versions: • 6.x • 8.x • 9.x • 10.x
Node.js OJAI client	Install the client by using the following command: `npm install node-maprdb`

You also must have access to the following software:

last-updated: Apr 11, 2022

- MapR cluster 6.1 or later
- MapR Data Access Gateway 2.0 or later

To run a Node.js OJAI application, you simply need to install and configure the MapR Data Access Gateway:

- Installing the Data Access Gateway Service on page 1492
- Modifying Configuration Settings for the Data Access Gateway Service on page 1493

Node.js OJAI Client Security

The client supports username/password authentication. The initial connection (and token renewal) use these credentials. Subsequent communication uses JWT.

When connecting to a secure MapR cluster, the client uses:

- X.509 certificates to authenticate with the MapR Data Access Gateway
- TLS v1.2 to encrypt communication between the client and the Data Access Gateway

Node.js OJAI Client Connection String

The string you use to connect your OJAI client to the cluster must have the following format:

```
"[ojai:mapr:thin:v1@]<hostname>[:<port>][?<option_name>=<option_value>;...]"
```

The prefix `ojai:mapr:thin:v1@` is optional.

<hostname>	Name of the MapR Data Access Gateway host
<port>	Port number (see Ports Used by MapR Software on page 2290) that gRPC clients use to connect to the MapR Data Access Gateway Default: 5678
auth=<scheme_name>	The authentication scheme for the current connection; currently, only `basic`
user=<username>	The user name for `basic` authentication
password=<password>	The password for `basic` authentication
ssl=true|false	Whether to establish a secure connection using SSL/TLS An error is returned if there is a mismatch between your client and Data Access Gateway security settings. The default for this option is `true`, which is the required setting if connecting to a secure Data Access Gateway. If connecting to a nonsecure Data Access Gateway, set it to `false`. If set to `false`, the other SSL parameters are ignored. Note that the `grpc.service.ssl.enabled` property controls the SSL setting for the Data Access Gateway. For more information, see Administering the MapR Data Access Gateway on page 1492.
sslCA=<path to PEM file containing CA certificate>	Path to a local file containing Certificate Authority (CA) signed certificates in PEM format.

	Must be set if the `ssl` option is `true`.
`sslTargetNameOverride=<CA certificate common name>`	Fully qualified domain name specified in the CA certificate, which is different from the `<hostname>` in the connection string. For example, imagine that you are using the following: • Public network host name is `ec2-203-0-113-25.compute-1.amazonaws.com`. • Internal DNS is `node1.mydomain.com`. • CA signed certificate is issued to `node1.mydomain.com`. Using these names, you must specify the following connection string: `"ec2-203-0-113-25.compute-1.amazonaws.com:5678?ssl=true;sslCA=/opt/app/conf/rootca.pem;`**`sslTargetNameOverride=node1.mydomain.com`**`"`

Other examples of connection strings are the following:

```
"ojai:mapr:thin:v1@localhost:5678?
auth=basic;user=fred;password=george;sslCA=/opt/app/conf/rootca.pem"
"localhost:5678?ssl=false;auth=basic;user=fred;password=george"
```

Node.js OJAI Connection Retry Options

If your OJAI client cannot connect to your MapR cluster, it waits 10 ms. After 10 ms, it makes a second connection attempt. If that fails, it continues the attempts up to a configurable number of retries. The following parameters control the number of retries and the wait time between attempts:

Connection Option Parameter	Description	Default Value
`ojai.mapr.rpc.wait-multiplier`	Multiplier that determines the wait time for subsequent attempts after the initial 10 ms wait. The previous wait time is multiplied by this parameter.	1000
`ojai.mapr.rpc.wait-max-attempt`	Maximum wait time between attempts regardless of the multiplier parameter	18000 ms
`ojai.mapr.rpc.max-retries`	Maximum number of retry attempts	7

The following examples demonstrate how these parameters work, including the default case:

Attempt #	Wait Time (in ms) for each Retry Attempt	

	Default Parameters:
	`{`
	`'ojai.mapr.rpc.wait-multiplier': 1000,`
	`'ojai.mapr.rpc.wait-max-attempt': 18000,`
	`'ojai.mapr.rpc.max-retries': 7`
	`}`
		`{`
		`'ojai.mapr.rpc.wait-multiplier': 2,`
		`'ojai.mapr.rpc.wait-max-attempt': 90,`
		`'ojai.mapr.rpc.max-retries': 5`
		`}`

1	10	10
2	10*1000 = 10000	10*2 = 20
3	18000 10000*1000 = 10,000,000, which exceeds 18000	20*2 = 40
4	18000	40*2 = 80
5	18000	90 80*2 = 160, which exceeds 90
6	18000	Error
7	18000	N/A
8	Error	N/A

To set these retry options, you must pass them in the `ConnectionManager.getConnection` call:

```
const connectionString = 'localhost:5678?' +
  'auth=basic;' +
  'user=mapr;' +
  'password=mapr;' +
  'ssl=true;' +
  'sslCA=/opt/mapr/conf/ssl_truststore.pem;' +
  'sslTargetNameOverride=node1.mapr.com';
const options = {
  'ojai.mapr.rpc.wait-multiplier': 5,
  'ojai.mapr.rpc.wait-max-attempt': 50,
  'ojai.mapr.rpc.max-retries': 3
}

let connection;

ConnectionManager.getConnection(connectionString,options)
  .then((conn) => {
    connection = conn;
    // Get a store
    return connection.getStore('/demo_table');
  })
```

Writing Node.js OJAI Applications

For information about writing a Node.js OJAI application, see the Node.js sections in the following topics:

Querying in OJAI Applications on page 2579	Provides an introduction to the basic flow of an OJAI application that queries a MapR Database JSON table
Examples: Querying JSON Documents on page 2624	Contains code samples of OJAI applications that query MapR Database JSON tables
Managing JSON Documents on page 2542	Describes how to perform CRUD (create, query, update, and delete) operations on JSON documents in MapR Database JSON tables

Node.js OJAI Client Classes and Methods

This topic lists and describes the classes supported by the Node.js OJAI client and provides a link to document pages that describe the methods in each class.

Class Name	Description
`ConnectionManager`	Manages connections to MapR Database JSON.
`Connection`	Provides a logical connection to an OJAI data source: for example, MapR Database JSON.
`DocumentStore`	Encapsulates a store, typically persistent, of OJAI documents: for example, MapR Database JSON tables.
`QueryResult`	Encapsulates the stream of result sets for an OJAI query.
`OTime`	Encapsulates the OJAI `TIME` type.
`OTimestamp`	Encapsulates the OJAI `TIMESTAMP` type.
`ODate`	Encapsulates the OJAI `DATE` type.

See Node.js OJAI Client API for details about each class, including the methods available in each class.

Setting Query Options in Node.js OJAI

There are two categories of options you can set in your Node.js OJAI application. This topic describes both and shows you how to set each.

Setting Query Options Using a Node.js OJAI Method Call

Option Name	Description
`ojai.mapr.query.include-query-plan`	Enables or disables availability of the query plan for retrieval Value: `true|false` Default: `false`
`ojai.mapr.query.timeout-milliseconds`	Query timeout in milliseconds Maximum allowed value is 2147483647. Default: None; no timeout

To set any of these query options, you must pass the option as the second parameter in the `DocumentStore.find` method.

The following code snippet sets the query timeout to 3000 milliseconds:

```
const docStream = store.find(
    query,
    { 'ojai.mapr.query.timeout-milliseconds': 3000 }
);
```


last-updated: Apr 11, 2022

Setting Query Options in Node.js Using OJAI Query Syntax

OJAI Query Options on page 2588 describes query options that are available in all OJAI clients. To use these options in Node.js OJAI, you must construct your query in JSON format and use the `$options` keyword. See OJAI Query Syntax for details about the syntax, including an example.

Using the Python OJAI Client

Starting with EEP 6.0, you can use the Python OJAI client to write MapR Database JSON applications. The client provides you with a lightweight library that supports the OJAI API. You can connect to MapR Database JSON, and add, update, and query documents in a MapR Database JSON table.

The client provides you with the following benefits:

- Easy installation and use
- Access to MapR Database JSON through the OJAI interface in Python
- An OJAI interface that is tailored to Python developers
- Use of Python types to manipulate MapR Database JSON documents
- Support for Python multiprocessing and multithreading modules
- Support for L3/L4 (transport level) and L7 (application level) proxy load balancing

To use the Python OJAI client, you must install the MapR Data Access Gateway on your MapR cluster. The gateway serves as a proxy for translating requests between the Python client and the MapR cluster. The gateway also performs data processing to keep the client lightweight. See Administering the MapR Data Access Gateway on page 1492 for information about how to administer the gateway and configure load balancing.

Additional Resources

Examples: https://github.com/mapr-demos/ojai-examples/tree/master/python

Source Code: https://github.com/mapr/maprdb-python-client

Getting Started with the Python OJAI Client

This section describes the software required to run the Python OJAI client, client/server security, and how to specify your connection string. It also provides links to documentation that shows you how to write Python OJAI applications.

The Python OJAI client is available starting with the EEP 6.0 release.

Software Requirements

You must have the following software installed to run the client:

Client Software	Installation Notes
Python	Use Python 2.7 or later
pip	See https://pip.pypa.io/en/stable/installing/ for instructions specific to your environment.
Python OJAI client	Install the client by using the following command: `pip install maprdb-python-client`

You also must have access to the following software:

- MapR cluster 6.1 or later

- MapR Data Access Gateway 2.0 or later

To run a Python OJAI application, you simply need to install and configure the MapR Data Access Gateway:

- Installing the Data Access Gateway Service on page 1492
- Modifying Configuration Settings for the Data Access Gateway Service on page 1493

Python OJAI Client Security

The client supports username/password authentication. The initial connection (and token renewal) use these credentials. Subsequent communication uses JWT.

When connecting to a secure MapR cluster, the client uses:

- X.509 certificates to authenticate with the MapR Data Access Gateway
- TLS v1.2 to encrypt communication between the client and the Data Access Gateway

Python OJAI Client Connection String

The string you use to connect your OJAI client to the cluster must have the following format:

```
"[ojai:mapr:thin:v1@]<hostname>[:<port>][?<option_name>=<option_value>;...]"
```

The prefix `ojai:mapr:thin:v1@` is optional.

<hostname>	Name of the MapR Data Access Gateway host
<port>	Port number (see Ports Used by MapR Software on page 2290) that gRPC clients use to connect to the MapR Data Access Gateway Default: 5678
auth=<scheme_name>	The authentication scheme for the current connection; currently, only `basic`
user=<username>	The user name for `basic` authentication
password=<password>	The password for `basic` authentication
ssl=true|false	Whether to establish a secure connection using SSL/TLS An error is returned if there is a mismatch between your client and Data Access Gateway security settings. The default for this option is `true`, which is the required setting if connecting to a secure Data Access Gateway. If connecting to a nonsecure Data Access Gateway, set it to `false`. If set to `false`, the other SSL parameters are ignored. Note that the `grpc.service.ssl.enabled` property controls the SSL setting for the Data Access Gateway. For more information, see Administering the MapR Data Access Gateway on page 1492.
sslCA=<path to PEM file containing CA certificate>	Path to a local file containing Certificate Authority (CA) signed certificates in PEM format. Must be set if the `ssl` option is `true`.

last-updated: Apr 11, 2022

sslTargetNameOverride=<CA certificate common name>	Fully qualified domain name specified in the CA certificate, which is different from the <hostname> in the connection string. For example, imagine that you are using the following: • Public network host name is ec2-203-0-113-25.compute-1.amazonaws.com. • Internal DNS is node1.mydomain.com. • CA signed certificate is issued to node1.mydomain.com. Using these names, you must specify the following connection string: `"ec2-203-0-113-25.compute-1.amazonaws.com:5678?ssl=true;sslCA=/opt/app/conf/rootca.pem;`**`sslTargetNameOverride=node1.mydomain.com`**`"`

Other examples of connection strings are the following:

```
"ojai:mapr:thin:v1@localhost:5678?
auth=basic;user=fred;password=george;sslCA=/opt/app/conf/rootca.pem"
"localhost:5678?ssl=false;auth=basic;user=fred;password=george"
```

Python OJAI Connection Retry Options

If your OJAI client cannot connect to your MapR cluster, it waits 10 ms. After 10 ms, it makes a second connection attempt. If that fails, it continues the attempts up to a configurable number of retries. The following parameters control the number of retries and the wait time between attempts:

Connection Option Parameter	Description	Default Value
ojai.mapr.rpc.wait-multiplier	Multiplier that determines the wait time for subsequent attempts after the initial 10 ms wait. The previous wait time is multiplied by this parameter.	1000
ojai.mapr.rpc.wait-max-attempt	Maximum wait time between attempts regardless of the multiplier parameter	18000 ms
ojai.mapr.rpc.max-retries	Maximum number of retry attempts	7

The following examples demonstrate how these parameters work, including the default case:

Attempt #	Wait Time (in ms) for each Retry Attempt	

	Default Parameters:
	`{`
	`'ojai.mapr.rpc.wait-multiplier': 1000,`
	`'ojai.mapr.rpc.wait-max-attempt': 18000,`
	`'ojai.mapr.rpc.max-retries': 7`
	`}`
		`{`
		`'ojai.mapr.rpc.wait-multiplier': 2,`
		`'ojai.mapr.rpc.wait-max-attempt': 90,`
		`'ojai.mapr.rpc.max-retries': 5`
		`}`

1	10	10
2	10*1000 = 10000	10*2 = 20
3	18000 10000*1000 = 10,000,000, which exceeds 18000	20*2 = 40
4	18000	40*2 = 80
5	18000	90 80*2 = 160, which exceeds 90
6	18000	Error
7	18000	N/A
8	Error	N/A

To set these retry options, you must pass them in the `ConnectionFactory.get_connection` call:

```
connection_str = 'localhost:5678?auth=basic;user=mapr;password=mapr;' \
    'ssl=true;' \
    'sslCA=/opt/mapr/conf/ssl_truststore.pem;' \
    'sslTargetNameOverride=node1.mapr.com'
options = {
    'ojai.mapr.rpc.wait-multiplier': 5,
    'ojai.mapr.rpc.wait-max-attempt': 50,
    'ojai.mapr.rpc.max-retries': 3
}
connection =
ConnectionFactory.get_connection(connection_str=connection_str,options=optio
ns)
```

Writing a Python OJAI Application

For information about writing a Python OJAI application, see the Python sections in the following topics:

Querying in OJAI Applications on page 2579	Provides an introduction to the basic flow of an OJAI application that queries a MapR Database JSON table
Examples: Querying JSON Documents on page 2624	Contains code samples of OJAI applications that query MapR Database JSON tables
Managing JSON Documents on page 2542	Describes how to perform CRUD (create, query, update, and delete) operations on JSON documents in MapR Database JSON tables

last-updated: Apr 11, 2022

Python OJAI Client Classes and Methods
This topic lists and describes the classes supported by the Python OJAI client and provides a link to document pages that describe the methods in each class.

Class Name	Description
`Connection`	Provides a logical connection to an OJAI data source: for example, MapR Database JSON.
`ConnectionDriver`	Provides a connection handler, which enables you to get and check connections
`ConnectionManager`	Manages connections to MapR Database JSON.
`Document`	Provides the primary, DOM-based interface for inspecting OJAI documents.
`DocumentMutation`	Encapsulates a mutation to an existing OJAI document in a store.
`DocumentStore`	Encapsulates a store, typically persistent, of OJAI documents: for example, MapR Database JSON tables.
`DocumentStream`	Encapsulates the result set of an OJAI query.
`QueryResult`	Encapsulates the result set of an OJAI query.
`Query`	Encapsulates an OJAI query.
`QueryCondition`	Encapsulates a query condition; similar to a SQL `where` clause.
`Value`	Encapsulates the value of a field, scalar, or complex type in an OJAI document.
`OTime`	Encapsulates the OJAI `TIME` type.
`OTimestamp`	Encapsulates the OJAI `TIMESTAMP` type.
`ODate`	Encapsulates the OJAI `DATE` type.
`OInterval`	Encapsulates the OJAI `INTERVAL` type.

See Python OJAI Client API for details about each class, including the methods available in each class.

Multiprocessing and Multithreading in Python OJAI Applications
Python supports multiprocessing and multithreading modules that enable you to spawn either multiple processes or multiple threads in a Python program. This section contains examples that show you how to use these modules in your Python OJAI application.

Multiprocessing in Python OJAI Applications

The following code example spawns multiple processes using the Python `multiprocessing` module. When you use the module, you must create a separate OJAI connection for each process.

The code example is available at 014_multiprocessing_example.py.

```
"""Following example works with Python Client"""

import multiprocessing
from mapr.ojai.storage.ConnectionFactory import ConnectionFactory

"""Create a connection, get store, insert_or_replace/update document in
 store via multiprocessing"""

# Create a connection string using path:user@password
connection_str = "localhost:5678?auth=basic;user=mapr;password=mapr;" \
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/ssl_truststore.pem;" \
          "sslTargetNameOverride=node1.mapr.com"
```

```
# Create method which will be used for multiprocessing
def sample():
    # Create connection from connection_url
    # Cannot share connection for processes,
    # so need to create connection for each process.
    connection =
ConnectionFactory().get_connection(connection_str=connection_str)

    # Get a store and assign it as a DocumentStore object
    store = connection.get_or_create_store('/tmp/store_name')

    # Insert 15 documents, represented as Python dictionaries,
    # into DocumentStore
    for i in range(15):
        store.insert_or_replace(doc={'_id': str(i), 'name': 'Greg'})

    # Create DocumentMutation object using the OJAIConnection object
    mutation = connection.new_mutation()

    # Set mutation value
    mutation.set_or_replace(field_path='name', value='T')

    # Update 15 Document in store
    for i in range(15):
        store.update(_id=str(i), mutation=mutation)


# Create simple method for run process from Pool
def run(unused_var):
    pass


# Create data for multiprocessing
proces_count = 7
map_iterable = [1]  # simple iterator

# Create Pool object using the function and process_count value
p = multiprocessing.Pool(proces_count, initializer=sample)

# Run processes from the Pool
p.map(run, map_iterable)
```

Multithreading in Python OJAI Applications

You can use either the Python `thread` or `threading` module to spawn multiple threads in your Python application. When you use these modules, you can share an OJAI connection across threads.

Thread Module

The following code example uses the `thread` module. It is available at 015_thread_example.py.

```
"""Following example works with
Python Client"""
import thread
import time
from
mapr.ojai.storage.ConnectionFactory
import ConnectionFactory

"""Create a connection, get store,
insert_or_replace/update document in
 store via thread using same
connection"""
```

```
# Create method which will be used
for threads
def run_thread(name, conn):
    # Print that thread started with
threadName
    print('\n Start thread ', name)

    # Get a store and assign it as a
DocumentStore object
    store =
conn.get_or_create_store('/tmp/
store_name')

    # Insert 15 documents,
represented as Python dictionaries,
    # into DocumentStore
    for index in range(15):

store.insert_or_replace(doc={'_id':
str(index), 'name': 'Greg'})

    # Create DocumentMutation object
using the OJAIConnection object
    mutation = conn.new_mutation()

    # Set mutation value

mutation.set_or_replace(field_path='na
me', value='T')

    # Update 15 Document in store
    for index in range(15):
        store.update(_id=str(index),
mutation=mutation)

    # Print that thread done with
threadName
    print('\n Done thread ', name)


# Create a connection string using
path:user@password
connection_str = "localhost:5678?
auth=basic;user=mapr;password=mapr;" \
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/
ssl_truststore.pem;" \

"sslTargetNameOverride=node1.mapr.com"

# Create connection from
connection_url
# Can share connection for processes,
# so need to only one connection
instance for all threads
connection =
ConnectionFactory.get_connection(conne
ction_str)

# Create 10 threads using the same
```

```
connection instance
for i in range(10):
    thread_name =
'Thread-{}'.format(str(i))

thread.start_new_thread(run_thread,
(thread_name, connection,))

# This thread implementation doesn't
return thread object
# so thread status cannot be checked
# Wait 10 seconds
time.sleep(10)

# Close connection
connection.close()
```

Threading Module

The following code example uses the `threading` module. It is available at 016_threading_example.py.

```
"""Following example works with
Python Client"""
import threading
import time

from
mapr.ojai.storage.ConnectionFactory
import ConnectionFactory

"""Create a connection, get store,
insert_or_replace/update document in
 store via thread using same
connection"""

# Create a connection string using
path:user@password
connection_str = "localhost:5678?
auth=basic;user=mapr;password=mapr;" \
          "ssl=true;" \
          "sslCA=/opt/mapr/conf/
ssl_truststore.pem;" \

"sslTargetNameOverride=node1.mapr.com"

# Create connection from
connection_url
# Can share connection for processes,
# so need to only one connection
instance for all threads
connection =
ConnectionFactory.get_connection(conne
ction_str)


# Create child for sample threading
implementation
class MyThread(threading.Thread):
    # Implement __init__() method,
which takes thread name and
    # connection object
    def __init__(self, name,
```

```
connection):

threading.Thread.__init__(self)
        self.name = name
        self.connection = connection

    # Implement run() method
    def run(self):
        # Print that thread started
with threadName
        print('\n Start thread ',
self.name)

        # Get a store and assign it
as a DocumentStore object
        store =
connection.get_or_create_store('/tmp/
store_name')

        # Insert 15 documents,
represented as Python dictionaries,
        # into DocumentStore
        for index in range(15):

store.insert_or_replace(doc={'_id':
str(index), 'name': 'Greg'})

        # Create DocumentMutation
object using the OJAIConnection object
        mutation =
connection.new_mutation()

        # Set mutation value

mutation.set_or_replace(field_path='na
me', value='T')

        # Update 15 Document in store
        for index in range(15):

store.update(_id=str(index),
mutation=mutation)

        # Print that thread done with
threadName
        print('\n Done thread ',
self.name)


# This thread implementation return
thread object
# so thread status can be checked via
native methods
# Simple thread waiter for thread
list:
def waiter(threads):
    for my_thread in threads:
        # Check that current thread
is alive
        if my_thread.is_alive():
            time.sleep(1)
            # Wait until current
```

```
thread finished
            waiter(threads)
        # Move to the next thread if
this is not alive
        elif not my_thread.is_alive():
            pass


# Create list instance for storing
created threads objects
thread_list = []

# Create and run 10 threads
for i in range(10):
    # Create thread instance using
MyThread and OJAIConnection object
    thread =
MyThread(name='Thread-{0}'.format(str(
i)),

connection=connection)

    # Start current thread
    thread.start()

    # Append thread object into
thread_list
    thread_list.append(thread)

# Wait until all threads will finished
waiter(thread_list)

# Close connection
connection.close()
```

Setting Query Options in Python OJAI

There are two categories of options you can set in your Python OJAI application. This topic describes both and shows you how to set each.

Setting Query Options Using a Python OJAI Method Call

Option Name	Description
`ojai.mapr.query.include-query-plan`	Enables or disables availability of the query plan for retrieval Value: `True`|`False` Default: `False`
`ojai.mapr.query.result-as-document`	Enables or disables returning the query result as an OJAI `Document` class object versus a Python dictionary Value: `True`|`False` Default: `False`; returns query result as a Python dictionary
`ojai.mapr.query.timeout-milliseconds`	Query timeout in milliseconds Maximum allowed value is 2147483647. Default: None; no timeout

last-updated: Apr 11, 2022

To set any of these query options, you must pass the option as the second parameter in the DocumentStore.find method.

The following code snippet sets the option to return the query result as a Document object:

```
options = {'ojai.mapr.query.result-as-document': True}
query_result = store.find(query, options=options)
```

Setting Query Options in Python Using OJAI Query Syntax

OJAI Query Options on page 2588 describes query options that are available in all OJAI clients. To use these options in Python OJAI, you must construct your query in JSON format and use the `$options` keyword. See OJAI Query Syntax for details about the syntax, including an example.

Using the C# OJAI Client

Starting with EEP 6.1.0, you can use the C# OJAI client to write MapR Database JSON applications. The client provides you with a lightweight library that supports the OJAI API. You can connect to MapR Database JSON, and add, update, and query documents in a MapR Database JSON table.

The client provides you with the following benefits:

- Easy installation and use
- Access to MapR Database JSON through the OJAI interface in C#
- An OJAI interface that is tailored to C# developers
- Use of C# types to manipulate MapR Database JSON documents
- Support for C# asynchronous programming and threading mechanism
- Support for L3/L4 (transport level) and L7 (application level) proxy load balancing

To use the C# OJAI client, you must install the MapR Data Access Gateway on your MapR cluster. The gateway serves as a proxy for translating requests between the C# client and the MapR cluster. The gateway also performs data processing to keep the client lightweight. See Administering the MapR Data Access Gateway on page 1492 for information about how to administer the gateway and configure load balancing.

Additional Resources

Examples: https://github.com/mapr-demos/ojai-examples/tree/master/csharp

Getting Started with the C# OJAI Client

This section describes the software required to run the C# OJAI client, client/server security, and how to specify your connection string. It also provides links to documentation that shows you how to write C# OJAI applications.

The C# OJAI client is available starting in the EEP 6.0.0 release.

Software Requirements

You must have the following software installed to run the client:

Client Software	Installation Notes
Visual Studio 2017	https://visualstudio.microsoft.com/vs/

Client Software	Installation Notes

C# OJAI client	• Install the client by using the NuGet Command Line Interface (CLI).
	• Install the `MapRDB.Driver` using the NuGet Package Manager Console:
	`Install-Package MapRDB.Driver -Version 1.0.0`
	• Use the dotnet command line interface:
	`dotnet add package MapRDB.Driver`
	• Use the NuGet Package Manager UI. For instructions, follow this link.

You also must have access to the following software:

- MapR cluster 6.1 or later
- MapR Data Access Gateway 2.0 or later

To run a C# OJAI application, you simply need to install and configure the MapR Data Access Gateway:

- Installing the Data Access Gateway Service on page 1492
- Modifying Configuration Settings for the Data Access Gateway Service on page 1493

C# OJAI Client Security

The client supports username/password authentication. The initial connection (and token renewal) use these credentials. Subsequent communication uses JWT.

When connecting to a secure MapR cluster, the client uses:

- X.509 certificates to authenticate with the MapR Data Access Gateway
- TLS v1.2 to encrypt communication between the client and the Data Access Gateway

C# OJAI Client Connection String

The string you use to connect your OJAI client to the cluster must have the following format:

```
"[ojai:mapr:thin:v1@]<hostname>[:<port>][?<option_name>=<option_value>;...]"
```

The prefix `ojai:mapr:thin:v1@` is optional.

`<hostname>`	Name of the MapR Data Access Gateway host
`<port>`	Port number (see Ports Used by MapR Software on page 2290) that gRPC clients use to connect to the MapR Data Access Gateway Default: 5678
`auth=<scheme_name>`	The authentication scheme for the current connection; currently, only `basic`
`user=<username>`	The user name for `basic` authentication
`password=<password>`	The password for `basic` authentication

last-updated: Apr 11, 2022

ssl=true|false	Whether to establish a secure connection using SSL/TLS An error is returned if there is a mismatch between your client and Data Access Gateway security settings. The default for this option is `true`, which is the required setting if connecting to a secure Data Access Gateway. If connecting to a nonsecure Data Access Gateway, set it to `false`. If set to `false`, the other SSL parameters are ignored. Note that the `grpc.service.ssl.enabled` property controls the SSL setting for the Data Access Gateway. For more information, see Administering the MapR Data Access Gateway on page 1492.
sslCA=<path to PEM file containing CA certificate>	Path to a local file containing Certificate Authority (CA) signed certificates in PEM format. Must be set if the `ssl` option is `true`.
sslTargetNameOverride=<CA certificate common name>	Fully qualified domain name specified in the CA certificate, which is different from the `<hostname>` in the connection string. For example, imagine that you are using the following: • Public network host name is `ec2-203-0-113-25.compute-1.amazonaws.com`. • Internal DNS is `node1.mydomain.com`. • CA signed certificate is issued to `node1.mydomain.com`. Using these names, you must specify the following connection string: `"ec2-203-0-113-25.compute-1.amazonaws.com:5678?ssl=true;sslCA=/opt/app/conf/rootca.pem;`**`sslTargetNameOverride=node1.mydomain.com`**`"`

Other examples of connection strings are the following:

```
"ojai:mapr:thin:v1@localhost:5678?
auth=basic;user=fred;password=george;sslCA=/opt/app/conf/rootca.pem"
"localhost:5678?ssl=false;auth=basic;user=fred;password=george"
```

C# OJAI Connection Retry Options

If your OJAI client cannot connect to your MapR cluster, it waits 10 ms. After 10 ms, it makes a second connection attempt. If that fails, it continues the attempts up to a configurable number of retries. The following parameters control the number of retries and the wait time between attempts:

Connection Option Parameter	Description	Default Value
`retryExponentialMultiplier`	Multiplier that determines the wait time for subsequent attempts after the initial 10 ms wait. The previous wait time is multiplied by this parameter.	1000

last-updated: Apr 11, 2022

Connection Option Parameter	Description	Default Value
`retryCount`	Maximum number of retry attempts	7

To set these retry options, you must pass them in the `ConnectionFactory.CreateConnection` call:

```
var connectionStr = $"localhost:5678?auth=basic;" +
            $"user=mapr;" +
            $"password=mapr;" +
            $"ssl=true;" +
            $"sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
            $"sslTargetNameOverride=node1.mapr.com";
var connection = ConnectionFactory.CreateConnection(connectionStr, 3, 5);
```

Writing a C# OJAI Application

For information about writing a C# OJAI application, see the C# sections in the following topics:

Querying in OJAI Applications on page 2579	Provides an introduction to the basic flow of an OJAI application that queries a MapR Database JSON table
Examples: Querying JSON Documents on page 2624	Contains code samples of OJAI applications that query MapR Database JSON tables
Managing JSON Documents on page 2542	Describes how to perform CRUD (create, query, update, and delete) operations on JSON documents in MapR Database JSON tables

C# OJAI Client Classes and Methods

This topic lists and describes the classes supported by the C# OJAI client and provides a link to document pages that describe the methods in each class.

Class Name	Description
`Value`	Encapsulates the value of a field, scalar, or complex in an OJAI document.
`OjaiDocument`	Provides the primary, DOM-based interface for inspecting OJAI documents
`OjaiDocumentStream`/ `QueryResult`	Encapsulates the result set of an OJAI query.
`OjaiDocumentMutation`	Encapsulates a mutation to an existing OJAI document in a store.
`Query`	Encapsulates an OJAI query.
`QueryCondition`	Encapsulates a query condition; similar to a SQL `where` clause.
`ConnectionFactory`	Provides a logical connection to an OJAI data source: for example, MapR Database JSON.
`MapRDBConnection`	Provides a connection handler, which enables you to get and check connections.
`OjaiDocumentStore`	Encapsulates a store, typically persistent, of OJAI documents: for example, MapR Database JSON tables.
`OTime`	Encapsulates the OJAI `TIME` type.
`OTimestamp`	Encapsulates the OJAI `TIMESTAMP` type.
`ODate`	Encapsulates the OJAI `DATE` type.
`OInterval`	Encapsulates the OJAI `INTERVAL` type.

See the C# OJAI Client API for details about each class, including the methods available in each class.

Setting Query Options in C# OJAI

There are two categories of options you can set in your C# OJAI application. This topic describes both and shows you how to set each.

Setting Query Options Using a C# OJAI Method Call

Option Name	Description
`IncludeQueryPlan`	Enables or disables availability of the query plan for retrieval Value: `true|false` Default: `false`
`Timeout`	Query timeout in milliseconds Maximum allowed value is 2147483647. Default: None; no timeout

To set any of these query options, you must pass the option as the second parameter in the `DocumentStore.Find` method.

The following code snippet sets the query timeout to 3000 milliseconds:

```
var queryResult = store.Find(query, new QueryOptions() { Timeout = 3000 });

var queryResult = store.Find(query, new QueryOptions(3000));
```

Setting Query Options in C# Using OJAI Query Syntax

OJAI Query Options on page 2588 describes query options that are available in all OJAI clients. To use these options in C# OJAI, you must construct your query in JSON format and use the `$options` keyword. See OJAI Query Syntax for details about the syntax, including an example.

Using the Go OJAI Client

Starting with EEP 6.0.0, you can use the Go OJAI client to write MapR Database JSON applications. The client provides you with a lightweight library that supports the OJAI API. You can connect to MapR Database JSON, and add, update, and query documents in a MapR Database JSON table.

The client provides you with the following benefits:

- Easy installation and use
- Access to MapR Database JSON through the OJAI interface in Go
- An OJAI interface that is tailored to Go developers
- Use of Go types to manipulate MapR Database JSON documents
- Support for Go multithreading using Goroutines
- Support for L3/L4 (transport level) and L7 (application level) proxy load balancing

To use the Go OJAI client, you must install the MapR Data Access Gateway on your MapR cluster. The gateway serves as a proxy for translating requests between the Go client and the MapR cluster. The gateway also performs data processing to keep the client lightweight. To administer the gateway and configure load balancing, see Administering the MapR Data Access Gateway on page 1492.

Additional Resources

Blog: CRUD with the New Golang Client for MapR Database

Examples: https://github.com/mapr-demos/ojai-examples/tree/master/golang

Source Code: https://github.com/mapr/maprdb-go-client

Getting Started with the Go OJAI Client

This section describes the software required to run the Go OJAI client, client/server security, and how to specify your connection string. It also provides links to documentation that shows you how to write Go OJAI applications.

The Go OJAI client is available starting in the EEP 6.0.0 release.

Software Requirements

You must have the following software installed to run the client:

Client Software	Installation Notes
Golang 1.10 (or later)	
Go OJAI client	Install the client using the following command: `go get github.com/mapr/maprdb-go-client`

You also must have access to the following software:

- MapR cluster 6.1 or later
- MapR Data Access Gateway 2.0 or later

To run a Go OJAI application, you simply need to install and configure the MapR Data Access Gateway:

- Installing the Data Access Gateway Service on page 1492
- Modifying Configuration Settings for the Data Access Gateway Service on page 1493

For some sample code, see https://github.com/magpierre/mapr_go_client_mqtt. `main.go` shows a simple Go client that reads from an MQTT messaging protocol and writes to a MapR JSON database.

Go OJAI Client Security

The client supports username/password authentication. The initial connection (and token renewal) use these credentials. Subsequent communication uses JWT.

When connecting to a secure MapR cluster, the client uses:

- X.509 certificates to authenticate with the MapR Data Access Gateway
- TLS v1.2 to encrypt communication between the client and the Data Access Gateway

Go OJAI Client Connection String

The string you use to connect your OJAI client to the cluster must have the following format:

```
"[ojai:mapr:thin:v1@]<hostname>[:<port>][?<option_name>=<option_value>;...]"
```

The prefix `ojai:mapr:thin:v1@` is optional.

`<hostname>` Name of the MapR Data Access Gateway host

last-updated: Apr 11, 2022

<port>	Port number (see Ports Used by MapR Software on page 2290) that gRPC clients use to connect to the MapR Data Access Gateway Default: 5678
auth=<scheme_name>	The authentication scheme for the current connection; currently, only `basic`
user=<username>	The user name for `basic` authentication
password=<password>	The password for `basic` authentication
ssl=true|false	Whether to establish a secure connection using SSL/TLS An error is returned if there is a mismatch between your client and Data Access Gateway security settings. The default for this option is `true`, which is the required setting if connecting to a secure Data Access Gateway. If connecting to a nonsecure Data Access Gateway, set it to `false`. If set to `false`, the other SSL parameters are ignored. Note that the `grpc.service.ssl.enabled` property controls the SSL setting for the Data Access Gateway. For more information, see Administering the MapR Data Access Gateway on page 1492.
sslCA=<path to PEM file containing CA certificate>	Path to a local file containing Certificate Authority (CA) signed certificates in PEM format. Must be set if the `ssl` option is `true`.
sslTargetNameOverride=<CA certificate common name>	Fully qualified domain name specified in the CA certificate, which is different from the <`hostname`> in the connection string. For example, imagine that you are using the following: • Public network host name is `ec2-203-0-113-25.compute-1.amazonaws.com`. • Internal DNS is `node1.mydomain.com`. • CA signed certificate is issued to `node1.mydomain.com`. Using these names, you must specify the following connection string: `"ec2-203-0-113-25.compute-1.amazonaws.com:5678?ssl=true;sslCA=/opt/app/conf/rootca.pem;`**`sslTargetNameOverride=node1.mydomain.com`**`"`

Other examples of connection strings are the following:

```
"ojai:mapr:thin:v1@localhost:5678?
auth=basic;user=fred;password=george;sslCA=/opt/app/conf/rootca.pem"
"localhost:5678?ssl=false;auth=basic;user=fred;password=george"
```

Go OJAI Connection Retry Options

If your OJAI client cannot connect to your MapR cluster, it waits 10 ms. After 10 ms, it makes a second connection attempt. If that fails, it continues the attempts up to a configurable number of retries. The following parameters control the number of retries and the wait time between attempts:

Connection Option Parameter	Description	Default Value
`MaxAttempt`	Maximum number of retry attempts	9
`WaitBetweenSeconds`	Maximum wait time between attempts	12 s
`CallTimeoutSeconds`	Maximum call timeout	60 s

To set these retry options, you must pass them in the `client.MakeConnectionWithRetryOptions` call:

```
    connectionString := "localhost:5678?" +
        "auth=basic;" +
        "user=mapr;" +
        "password=mapr;" +
        "ssl=true;" +
        "sslCA=/opt/mapr/conf/ssl_truststore.pem;" +
        "sslTargetNameOverride=node1.cluster.com"
    options := &client.ConnectionOptions{MaxAttempt:3,
WaitBetweenSeconds:10, CallTimeoutSeconds:60}
    connection, _ :=
client.MakeConnectionWithRetryOptions(connectionString, options)
```

Writing a Go OJAI Application

For information about writing a Go OJAI application, see the Go sections in the following topics:

Querying in OJAI Applications on page 2579	Provides an introduction to the basic flow of an OJAI application that queries a MapR Database JSON table
Examples: Querying JSON Documents on page 2624	Contains code samples of OJAI applications that query MapR Database JSON tables
Managing JSON Documents on page 2542	Describes how to perform CRUD (create, query, update, and delete) operations on JSON documents in MapR Database JSON tables

Go OJAI Client Structures and Functions

This topic lists and describes the structures supported by the Go OJAI client and provides a link to document pages that describe the functions in each structure.

Structure Name	Description
`Document`	Provides the primary, DOM-based interface for inspecting OJAI documents.
`QueryResult`	Encapsulates the result set of an OJAI query.
`DocumentMutation`	Encapsulates a mutation to an existing OJAI document in a store.
`Query`	Encapsulates an OJAI query.
`Condition`	Encapsulates a query condition; similar to a SQL `where` clause.
`Connection`	Provides a logical connection to an OJAI data source: for example, MapR Database JSON.

last-updated: Apr 11, 2022

Structure Name	Description
`DocumentStore`	Encapsulates a store, typically persistent, of OJAI documents: for example, MapR Database JSON tables.
`OTime`	Encapsulates the OJAI `TIME` type.
`OTimestamp`	Encapsulates the OJAI `TIMESTAMP` type.
`ODate`	Encapsulates the OJAI `DATE` type.

For details about each structure, including the functions available in each structure, see the Go OJAI Client API.

Setting Query Options in Go OJAI
There are two categories of options you can set in your Go OJAI application. This topic describes both and shows you how to set each.

Setting Query Options Using a Go OJAI Function Call

Option Name	Description
`IncludeQueryPlan`	Enables or disables availability of the query plan for retrieval. Value: `true`|`false` Default: `false`
`ResultAsDocument`	Enables or disables returning the query result as an OJAI `Document` object list versus a Go map list. Value: `True`|`False` Default: `False`: returns query result as a Go map list
`Timeout`	Query timeout in milliseconds. Maximum allowed value is 2147483647 Default: None; no timeout

To set any of these query options, you must pass the option as the second parameter in the `DocumentStore.FindQueryWithContext` function.

The following code snippet sets the query timeout to 3000 milliseconds:

```
timeoutCtx, cancel := context.WithTimeout(context.Background(),
time.Duration(3*time.Second))
result, err := suite.store.FindQueryWithContext(query, findOptions,
timeoutCtx)
cancel()
```

Setting Query Options in Go Using OJAI Query Syntax

OJAI Query Options on page 2588 describes query options that are available in all OJAI clients. To use these options in Go OJAI, you must construct your query in JSON format and use the `$options` keyword. See OJAI Query Syntax for details about the syntax, including an example.

Using the MapR Database JSON REST API
Starting in the EEP 5.0 release, you can use a REST API to access MapR Database JSON tables. The REST API allows you to use HTTP calls to perform basic operations on MapR Database JSON tables.

The API supports the following operations:

last-updated: Apr 11, 2022

- Create and delete MapR Database JSON tables
- Insert, update, and delete documents from a table
- Retrieve documents while specifying filter conditions and projections

The REST API has the following characteristics:

- Operations are stateless
- Operations are synchronous
- Request responses are not buffered
- Web connections are secure when connecting to secure MapR clusters
- Supports the following methods of authentication:
  - Basic authentication
  - Token-based authentication using JSON Web Tokens (JWT)
- Supports user impersonation - All data access calls are run on behalf of the authenticated user
- Returns HTTP error codes and detailed error responses in the response message body

When connecting to a MapR cluster, you must use HTTPS in your requests.

With basic authentication, you pass a username and password in your Web client. With token based authentication, you generate a token and then pass the token in the header of subsequent API requests.

The MapR Data Access Gateway is the service that supports this web API. You should configure multiple instances of this service across your MapR cluster to distribute request processing. To achieve load balancing, you must install an external load balancer. Using token based authentication and an external load balancer, you can achieve high availability and failover. Because the REST API is stateless, you do not have to regenerate your authentication token when different service instances process your API request. This applies even in the event of failovers and service restart. You must regenerate your token when it expires.

The API does not support the following features:

- MapR Database JSON administrative commands, except the commands noted earlier
- Read Your Own Writes

To modify properties that the MapR Database JSON REST API uses, see Application Properties.

Related concepts

Understanding the MapR Data Access Gateway on page 750
The MapR Data Access Gateway is a service that acts as a proxy and gateway for translating requests between lightweight client applications and the MapR cluster.

Administering the MapR Data Access Gateway on page 1492
The MapR Data Access Gateway is a service that acts as a proxy and gateway for translating requests between lightweight client applications and the MapR cluster. This section describes considerations when upgrading the service, how to modify configuration settings, and how to administer and manage the service.

Getting Started with the MapR Database JSON REST API

A simple way to invoke the MapR Database JSON REST API is to use cURL commands. This section contains a sequence cURL commands that demonstrate the basic functionality of the API.

last-updated: Apr 11, 2022

Note: The MapR Database JSON REST API is available starting in the EEP 5.0 release.

The operations shown are the following:

- Create a MapR Database JSON table
- Insert documents into the table
- Retrieve documents from the table, including retrievals that contain field projections and conditions
- Update individual documents and fields within a document

To learn about the complete API, see the reference material at Understanding the MapR Database JSON REST API on page 2703.

The examples in this section assume that you installed the MapR Data Access Gateway on the host `10.10.100.42`. The examples use HTTPS with the default HTTPS port of `8243`. For information about installing the Data Access Gateway, see Installing the Data Access Gateway Service on page 1492.

Note: The examples URL encode the slashes in the table path (`%2F`) to differentiate them from the slashes in the command API.

Using Basic Authentication

The commands in this section use basic authentication. To use this form of authentication, you must pass the username and password in all commands, using the `-u` option.

1. Create a MapR Database JSON table in the path `/apps/employees`:

```
curl -X PUT \
  'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees' \
  -u root:mapr
```

2. Insert 3 documents into the table:

```
curl -X POST \
  'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees' \
  -u root:mapr \
  -H 'Content-Type: application/json' \
  -d '[{"_id":"user001","first_name":"John","last_name":"Doe"},
 {"_id":"user002","first_name":"Jane","last_name":"Doe"},
 {"_id":"user003","first_name":"Simon","last_name":"Davis"}]'
```

3. Retrieve all of the documents:

```
curl -X GET \
  'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees%2F' \
  -u root:mapr
```

The command returns the following:

```
{
    "DocumentStream": [
        {
            "_id": "user001",
            "first_name": "John",
            "last_name": "Doe"
        },
        {
            "_id": "user002",
            "first_name": "Jane",
            "last_name": "Doe"
        },
        {
            "_id": "user003",
            "first_name": "Simon",
            "last_name": "Davis"
        }
    ]
}
```

4. Limit the `GET` request to 2 documents starting at offset 1:

```
curl -X GET \
  'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees%2F?
offset=1&limit=2' \
  -u root:mapr
```

The command returns the following:

```
{
    "DocumentStream": [
        {
            "_id": "user002",
            "first_name": "Jane",
            "last_name": "Doe"
        },
        {
            "_id": "user003",
            "first_name": "Simon",
            "last_name": "Davis"
        }
    ]
}
```

5. Retrieve only the first names in the documents:

```
curl -X GET \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees?
fields=first_name' \
  -u root:mapr
```

The command returns the following:

```
{
    "DocumentStream": [
        {
            "first_name": "John"
        },
        {
            "first_name": "Jane"
        },
        {
            "first_name": "Simon"
        }
    ]
}
```

6. Retrieve all documents with a last name of `'Doe'`:

```
curl -g -X GET \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees?
condition={"$eq":{"last_name":"Doe"}}' \
  -u root:mapr
```

Note: You must pass `'-g'` in the cURL command due to the nested braces in the condition.

The command returns 2 documents:

```
{
    "DocumentStream": [
        {
            "_id": "user001",
            "first_name": "John",
            "last_name": "Doe"
        },
        {
            "_id": "user002",
            "first_name": "Jane",
            "last_name": "Doe"
        }
    ]
}
```

7. Retrieve the id and first name of documents with a last name of `'Doe'`:

```
curl -g -X GET \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees?
condition={"$eq":{"last_name":"Doe"}}&fields=_id,first_name' \
  -u root:mapr
```

Note: You must pass `'-g'` in the cURL command due to the nested braces in the condition.

The command returns the following:

```
{
    "DocumentStream": [
        {
            "_id": "user001",
            "first_name": "John"
        },
        {
            "_id": "user002",
            "first_name": "Jane"
        }
    ]
}
```

8. Run the same command, also retrieving the query plan:

```
curl -g -X GET \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees?
condition={"$eq":
{"last_name":"Doe"}}&fields=_id,first_name&getPlan=true' \
  -u root:mapr
```

The output includes the query plan:

```
{
    "DocumentStream": [
        {
            "_id": "user001",
            "first_name": "John"
        },
        {
            "_id": "user002",
            "first_name": "Jane"
        }
    ],
    "QueryPlan": [
        [
            {
                "streamName": "DBDocumentStream",
                "parameters": {
                    "queryConditionPath": true,
                    "projectionPath": [
                        "_id",
                        "first_name"
                    ],
                    "primaryTable": "/apps/employees"
                }
            }
        ]
    ]
}
```

9. Update the first name in one of the documents, specifying the `id` in the command:

```
curl -X POST \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees/document/
user001' \
  -H 'Content-Type: application/json' \
  -u root:mapr \
  -d '{"$set":{"first_name":"Jay"}}'
```

10. Retrieve the updated document, specifying the `id` in the command:

```
curl -X GET \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees/document/
user001' \
  -u root:mapr
```

    The document contains an updated first name:

```
{
    "_id": "user001",
    "first_name": "Jay",
    "last_name": "Doe"
}
```

11. Replace the same document, but this time with a user who has only a first name:

```
curl -X PUT \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees/document/
user001' \
  -H 'Content-Type: application/json' \
  -u root:mapr \
  -d '{"_id":"user001","first_name":"Jonathan"}'
```

12. Retrieve the new document by passing the `id` in the request:

```
curl -X GET \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees/document/
user001' \
  -u root:mapr
```

    The document contains only a first name:

```
{
    "_id": "user001",
    "first_name": "Jonathan"
}
```

Using Token-Based Authentication

To use token-based authentication, you first create a token, authenticating with a username and password. You then pass the generated token in all subsequent commands.

1. Generate a token using the following command:

```
curl -X POST \
  'https://10.10.100.42:8243/auth/v2/token' \
  -u root:mapr
```

   The command returns the following token:

```
{
    "token":
"eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJyb290IiwiYXVkIjoid2ViIiwiZXhwIjoxNTIwMjY
5MTQwLCJpYXQiOjE1MjAyNjczNDB9.NT8L2deiA6v55bfbU_opiG1XXGPP0IwfSex3jW5A1Zs
oI1ar09it7-XwNtRqfL_I29IHLyfmuHcT5eSIpwq6ng"
}
```

2. Pass the token in your cURL command, as shown in the following GET:

```
curl -X GET \
  https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees \
  -H 'Authorization:
Bearer
eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJyb290IiwiYXVkIjoid2ViIiwiZXhwIjoxNTIwMjY5
MTQwLCJpYXQiOjE1MjAyNjczNDB9.NT8L2deiA6v55bfbU_opiG1XXGPP0IwfSex3jW5A1Zso
I1ar09it7-XwNtRqfL_I29IHLyfmuHcT5eSIpwq6ng'
```

Understanding the MapR Database JSON REST API
The section describes the details of the MapR Database JSON REST API.

The following table summarizes the commands the API supports. Click on the command for details about the command parameters and examples.

Category	REST API Command	Description
Authentication	POST /auth/v2/token on page 2703	Creates an authentication token
Table Operations	PUT /api/v2/table/{path} on page 2704	Creates a MapR Database JSON table
	DELETE /api/v2/table/{path} on page 2704	Drops a MapR Database JSON table
Table Document Operations	POST /api/v2/table/{path} on page 2705	Adds one or more documents to a MapR Database JSON table
	PUT /api/v2/table/{path}/document/{id} on page 2706	Updates a document by id in a MapR Database JSON table
	POST /api/v2/table/{path}/document/{id} on page 2707	Updates a partial document by id in a MapR Database JSON table using mutations
	DELETE /api/v2/table/{path}/document/{id} on page 2707	Deletes a single document by id in a MapR Database JSON table
	GET /api/v2/table/{path} on page 2708	Retrieves one or more documents from a MapR Database JSON table
	GET /api/v2/table/{path}/document/{id} on page 2711	Retrieves a single document by id from a MapR Database JSON table

POST /auth/v2/token
Authenticates a user. If successful, creates an authentication token that you use in subsequent API requests. By default, the token expires in 30 minutes. After a token expires, you must rerun this command to generate a new token.

last-updated: Apr 11, 2022

Request Example

The following creates an authentication token for user `root`:

```
curl -X POST \
  'https://10.10.100.42:8243/auth/v2/token' \
  -u root:mapr
```

Response Example

```
200 OK

{
  "token":
"eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJtYXByIiwiYXVkIjoid2ViIiwiZXhwIjoxNTE2NzQ2MD
c4LCJpYXQiOjE1MTY3NDQyNzh9.6YXWX72UP9_U9DPmT8c-_DQRDwY_TL0DEdsBaBqoaLf8iK0qH
NctyBTbFO5ktUJMTubVOj6D7pFOEyEuV8lhjA"
}
```

For an example that shows how to use the token returned by this API call in a subsequent `GET` command, see Using Token-Based Authentication on page 2702.

PUT /api/v2/table/{path}
Creates a MapR Database JSON table

Parameters

Name	Description
path `string` *(path)*	**Required**: Path to the new MapR Database JSON table

Request Example

The following creates a table with the path `/apps/employees`:

```
curl -X PUT \
  'https://10.10.100.42:8243/api/v2/table/%2Fapps/employees' \
  -u root:mapr
```

Response Example

```
201 Created
```

DELETE /api/v2/table/{path}
Drops a MapR Database JSON table

Parameters

Name	Description
path `string` *(path)*	**Required**: Path to the MapR Database JSON table

Request Example

The following drops a table with the path `/apps/employees`:

```
curl -X DELETE \
  'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees' \
  -u root:mapr
```

Response Example

```
200 OK
```

POST /api/v2/table/{path}
Adds or replaces one or more documents in a MapR Database JSON table

Parameters

Name	Description
path `string` *(path)*	**Required**: Path to the MapR Database JSON table
fieldAsKey `string` *(query)*	The name of the field that serves as the key in the JSON document
mode `string` *(query)*	Defines the behavior of the operation. The following are the possible values: • `insertOrReplace` - Inserts new document if specified document ID does not exist; otherwise, replaces existing document specified by the ID. • `insert` - Inserts new document; if the specified document ID already exists, returns an error. • `replace` - Replaces document with specified ID; returns an error if document does not exist. Default: `insertOrReplace`
body *(body)*	The body of the documents to add or replace

Request Example

The following inserts 3 documents into `/apps/employees`:

```
curl -X POST \
  'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees' \
  -u root:mapr \
  -H 'Content-Type: application/json' \
  -d '[{"_id":"user001","first_name":"John","last_name":"Doe"},
 {"_id":"user002","first_name":"Jane","last_name":"Doe"},
 {"_id":"user003","first_name":"Simon","last_name":"Davis"}]'
```


last-updated: Apr 11, 2022

Response Example

```
200 OK
```

PUT /api/v2/table/{path}/document/{id}
Updates a single document by id in a MapR Database JSON table

Parameters

Name	Description
path `string` *(path)*	**Required**: Path to the MapR Database JSON table
id `string` *(path)*	**Required**: Id of the document to update. If the document with the specified id does not exist, the mode parameter determines the behavior.
condition `string` *(query)*	Query condition (in JSON format) used to perform OJAI `DocumentStore.checkAndReplace` evaluation. See OJAI Query Condition Syntax on page 2606 for a description of the syntax.
mode `string` *(query)*	Defines the behavior of the operation. The following are the possible values: • `insertOrReplace` - Inserts new document if specified document ID does not exist; otherwise, replaces existing document specified by the ID. • `insert` - Inserts new document; if the specified document ID already exists, returns an error. • `replace` - Replaces document with specified ID if it exists Default: `insertOrReplace`
body *(body)*	**Required**: The body of the new document

Request Example

The following replaces the document with id `user001` in `/apps/employees` with an employee who has only a first name:

```
curl -X PUT \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees/document/
user001' \
  -H 'Content-Type: application/json' \
  -u root:mapr \
  -d '{"_id":"user001","first_name":"Jonathan"}'
```

Response Example

```
200 OK
```

POST /api/v2/table/{path}/document/{id}
Updates a partial document by id in a MapR Database JSON table using mutations

Parameters

Name	Description
path `string` *(path)*	**Required**: Path to the MapR Database JSON table
id `string` *(path)*	**Required**: Id of the document to update. If the document does not exist, inserts a new document.
condition `string` *(query)*	Query condition (in JSON format) used to perform OJAI `DocumentStore.checkAndUpdate` evaluation. See OJAI Query Condition Syntax on page 2606 for a description of the syntax.
body *(body)*	**Required**: The mutation specifying updates to the document. See Using OJAI Mutation Syntax on page 2561 for a description of the syntax.

Request Example

The following updates the `first_name` field the document in `/apps/employees` with id `user001`, replacing the field with the value `Jay`:

```
curl -X POST \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees/document/
user001' \
  -H 'Content-Type: application/json' \
  -u root:mapr \
  -d '{"$set":{"first_name":"Jay"}}'
```

Response Example

```
200 OK
```

DELETE /api/v2/table/{path}/document/{id}
Deletes a single document by id in a MapR Database JSON table

Parameters

Name	Description
path `string` *(path)*	**Required**: Path to the MapR Database JSON table
id `string` *(path)*	**Required**: Id of the document to delete

last-updated: Apr 11, 2022

Name	Description
condition `string` *(query)*	Query condition (in JSON format) used to perform OJAI `DocumentStore.checkAndDelete` evaluation. See OJAI Query Condition Syntax on page 2606 for a description of the syntax.

Request Example

The following deletes the document with id `user003` in `/apps/employees`:

```
curl -X DELETE \
  'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees/document/
user003' \
  -u root:mapr
```

Response Example

```
200 OK
```

GET /api/v2/table/{path}
Retrieves one or more documents from a MapR Database JSON table

Parameters

Name	Description
path `string` *(path)*	**Required**: Path to the MapR Database JSON table
condition `string` *(query)*	Query condition (in JSON format) to evaluate on documents retrieved. See OJAI Query Condition Syntax on page 2606 for a description of the syntax.
fields `string` *(query)*	The fields from the document to retrieve. See JSON Document Field Paths on page 515 for details about how to specify field paths.
fromId `string` *(query)*	Starting id of the range of documents to retrieve (inclusive)
toId `string` *(query)*	Ending id of the range of documents to retrieve (exclusive)
getPlan `string` *(query)*	If set to `true`, returns the query plan used to retrieve the documents Value: True|False Default: False

Name	Description
limit `integer` *(query)*	The maximum number of documents to retrieve
offset `integer` *(query)*	The number of documents to skip past before returning results
orderBy `string` *(query)*	The fields on which to sort the result. Specify the fields in a comma separated list, in the format `<field name>:<sort order>` where `<sort order>` is either `asc` or `desc`. `<sort order>` is optional and defaults to `asc`.
query `string` *(query)*	Query string with predefined keywords that define the behavior of the query. See Query with --query on page 5292 for syntax details.
withTags `string` *(query)*	Enables or disables output with extended type tags Value: True|False Default: True

Request Examples

1. The following retrieves all documents from `/apps/employees`:

```
curl -X GET \
  'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees' \
  -u root:mapr
```

2. The following specifies an offset and limit in the GET request:

```
curl -X GET \
  'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees%2F?
offset=1&limit=2' \
  -u root:mapr
```

3. The following retrieves only the first names in the documents:

```
curl -X GET \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees?
fields=first_name' \
  -u root:mapr
```

4. The following retrieves all documents with a last name of `'Doe'`:

```
curl -g -X GET \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees?
condition={"$eq":{"last_name":"Doe"}}' \
  -u root:mapr
```

Note: You must pass `'-g'` in the cURL command due to the nested braces in the condition.

5. The following retrieves the id and first name of documents with a last name of `'Doe'`:

```
curl -g -X GET \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees?
condition={"$eq":{"last_name":"Doe"}}&fields=_id,first_name' \
  -u root:mapr
```

6. The following runs the same command and includes a request for the query plan:

```
curl -g -X GET \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees?
condition={"$eq":
{"last_name":"Doe"}}&fields=_id,first_name&getPlan=true' \
  -u root:mapr
```

Response Examples

```
200 OK

{
    "DocumentStream": [
        {
            "_id": "user001",
            "first_name": "John",
            "last_name": "Doe"
        },
        {
            "_id": "user002",
            "first_name": "Jane",
            "last_name": "Doe"
        },
        {
            "_id": "user003",
            "first_name": "Simon",
            "last_name": "Davis"
        }
    ]
}
```

If you have configured the MapR Data Access Gateway to limit the number of documents in retrieval requests, and your result set exceeds the limit, the API response includes a warning. In the following example, the limit is set to 2:

```
{
    "DocumentStream": [
        {
            "_id": "user001",
            "first_name": "John",
            "last_name": "Doe"
        },
        {
            "_id": "user002",
            "first_name": "Jane",
            "last_name": "Doe"
        }
    ],
    "WARNING": "result truncated due to limit set to 2."
}
```

The following shows an example of output that includes a query plan. It corresponds to the output from example #6 in the previous section:

```
{
    "DocumentStream": [
        {
            "_id": "user001",
            "first_name": "John"
        },
        {
            "_id": "user002",
            "first_name": "Jane"
        }
    ],
    "QueryPlan": [
        [
            {
                "streamName": "DBDocumentStream",
                "parameters": {
                    "queryConditionPath": true,
                    "projectionPath": [
                        "_id",
                        "first_name"
                    ],
                    "primaryTable": "/apps/employees"
                }
            }
        ]
    ]
}
```

GET /api/v2/table/{path}/document/{id}
Retrieves a single document by id from a MapR Database JSON table

Parameters

Name	Description
path `string` *(path)*	**Required**: Path to the MapR Database JSON table
id `string` *(path)*	**Required**: Id of the document to retrieve
condition `string` *(query)*	Query condition (in JSON format) to evaluate on document retrieved. See OJAI Query Condition Syntax on page 2606 for a description of the syntax.
fields `string` *(query)*	The fields from the document to retrieve. See JSON Document Field Paths on page 515 for details about how to specify field paths.
withTags `string` *(query)*	Enables or disables output with extended type tags Value: True|False Default: True

Request Example

The following retrieves the document with id `user003` from `/apps/employees`:

```
curl -X GET \
'https://10.10.100.42:8243/api/v2/table/%2Fapps%2Femployees/document/
user003' \
  -u root:mapr
```

Response Example

```
200 OK

{
    "_id": "user003",
    "first_name": "Simon",
    "last_name": "Davis"
}
```

MapR Database JSON MapReduce API

This API library extends the Apache Hadoop MapReduce framework, so that you can write your own MapReduce applications to write data from one JSON table to another.

Prerequisites to using this API Library

- Ensure that you have a firm grasp of MapReduce concepts and experience writing MapReduce applications.
- Before running a MapReduce application that uses this API, ensure that the destination JSON table or tables already exist and that any column families other than the default are already created on the destination tables.

Classes

The following table summarizes the information that is in the MapR Database JSON MapReduce API, which you can refer to for complete details of the classes.

Category	Class	Description
Utility	`MapRDBMapReduceUtil`	Simplifies the use of the API for most use cases.
Input formatters	`TableInputFormat`	Describes how to read documents from MapR Database JSON tables.
Record reader	`TableRecordReader`	Reads documents (records) from MapR Database JSON tables.
	`TableRecordReaderImpl`	Iterates over MapR Database JSON table data. Returns key-value pair as `ByteBufWritableComparable` and `Document` respectively.
Record writers	`BulkLoadRecordWriter`	Bulk loads documents into MapR Database JSON tables.
	`TableMutationRecordWriter`	Modifies documents that are in MapR Database JSON tables.
	`TableRecordWriter`	Writes documents to MapR Database JSON tables.

last-updated: Apr 11, 2022

Category	Class	Description
Output formatters	`BulkLoadOutputFormat`	Describes how to bulk load documents into MapR Database JSON tables.
	`TableOutputFormat`	Describes how to write documents to MapR Database JSON tables.
	`TableMutationOutputFormat`	Writes `DocumentMutation` from the MapReduce phase to JSON tables . The key is of type `Value` and the value is a `DocumentMutation`.
Serializers	`DocumentSerialization`	Defines the serializer and deserializer for passing data from `Document` objects between map and reduce phases.
	`DBDocumentSerialization`	Converts a JSON document from MapR Database format to binary SequenceFile format.
	`ValueSerialization`	Serializes a JSON key and passes it between MapReduce phases.
Partitioner	`TablePartitioner`	Specifies how to partition data from the source JSON table.
	`TotalOrderPartitioner<K,V>`	Globally sorts data according to row key and then partitions the sorted data. This class is useful when the destination table has been pre-split into two or more tablets.

Using MapRDBMapReduceUtil to Set Default Values in Configurations and Jobs

The centerpiece of this API is the `MapRDBMapReduceUtil` class, which you can use in the `createSubmittableJob()` method of your applications to perform these actions:

- Set default values in the configuration for a MapReduce application and set the input and output format classes.
- Set default types for output keys and values.
- Configure a `TotalOrderPartitioner` and return the number of reduce tasks to use for a job.

To set default values in the configuration for a MapReduce application and set the input and output format classes, use the following methods:

```
configureTableInputFormat(org.apache.hadoop.mapreduce.Job job, String
srcTable)
```

The `configureTableInputFormat` method performs the following actions:

- Set the serialization class for `Document` and `Value` objects. These interfaces are part of the OJAI (Open JSON Application Interface) API.
- Set the field `INPUT_TABLE` in `TableInputFormat` to the path and name of the source table, and pass this value to the configuration for the MapReduce application.

last-updated: Apr 11, 2022

- Set the input format class for the job to `TableInputFormat`.

```
configureTableOutputFormat(org.apache.hadoop.mapreduce.Job job, String
destTable)
```

The `configureTableOutputFormat` method performs the following actions:

- Set the field `OUTPUT_TABLE` in `TableOutputFormat` to the path and name of the destination table, and pass this value to the configuration for the MapReduce applications.
- Set the output format class for the job to `TableOutputFormat`.

If you want to set values for other fields in `TableInputFormat` or `TableOutputFormat`, or write your own logic for them, you can pass field values to configurations and specify these classes for jobs as you would in common MapReduce applications.

To set default types for output keys and values, use the following methods:

```
setMapOutputKeyValueClass(org.apache.hadoop.mapreduce.Job job)
setOutputKeyValueClass(org.apache.hadoop.mapreduce.Job job)
```

Note: You can also set types for output keys and values from the map phase, if those types will differ from the final output types.

To configure `TotalOrderPartitioner` and return the number of reduce tasks to use for a job, you can use a code line similar to the following in your application's method for creating a job:

```
int numReduceTasks =
MapRDBMapReduceUtil.setPartitioner(org.apache.hadoop.mapreduce.Jobjob,
String destPath);
```

The `setPartitioner()` method finds out whether a table has been pre-split into two or more tablets, counts the number of tablets, writes the number to a partitioner file, and sends that file to an instance of `TotalOrderPartitioner`. This line also returns the number of tablets to `numReduceTasks`. Your code can then use that variable to set the number of reducers, like the following:

```
job.setNumReduceTasks(numReduceTasks);
```

Note: The sample application gives an example of how to use `MapRDBMapReduceUtil`.

Mutating Rows in Destination Tables

Use the `TableMutationRecordWriter` class when you need to mutate rows.

For example, suppose that you are tracking the number of users who are performing various actions on your retail website. To do this, at intervals you run your MapReduce application and save the results in JSON documents in MapR Database. Suppose that you count the number of users who went through the order process but abandoned their orders. After every run of the application, you want to update an JSON document by adding the current count to the total count and by updating a field that tracks the date and time that the MapReduce application was last run.

You could do that by setting values in a DocumentMutation object (see the OJAI (Open JSON Application Interface) Javadoc. You would then serialize that and write it to the table with `TableMutationRecordWriter`.

Compiling and Running Applications

You can compile applications that use the MapR Database Java API by using the required JAR file from the MapR installation. Run applications with the mapr command.

To compile an application, use the following command:

```
javac -cp 'mapr classpath' <Application jars>
```

To launch an application, use the following command

```
mapr <Main class jar> <commandline arguments>
```

Note: If you want to add JAR files to the classpath that the mapr command uses, add them with the environment variable MAPR_CLASSPATH. For example:

```
export MAPR_CLASSPATH=/home/apps/awesome-1.0.jar
         mapr com.company.MyAwesomeApp
```

Important: Turn off speculative execution

Speculative execution of MapReduce tasks is on by default. For custom applications that load MapR Database tables, it is recommended to turn speculative execution off. When it is on, the tasks that import data might run multiple times. Multiple tasks for an incremental bulkload could insert one or more versions of a record into a table. Multiple tasks for a full bulkload could cause loss of data if the source data continues to be updated during the load.

If your custom MapReduce application uses `MapRDBMapReduceUtil.configureTableOutputFormat()`, you do not have to turn off speculative execution manually. This method turns it off automatically.

Turn off speculative execution by using either of these methods:

- Set the following MapReduce version 2 parameter to false: `mapreduce.map.speculative`
- Include the following line in the method in your application that sets parameters for jobs:

  ```
  job.setSpeculativeExecution(false);
  ```

MapR Database JSON MapReduce: Sample App

This sample application reads records (JSON documents) from a JSON table and inserts new documents into another JSON table.

After reading records from a JSON table, the application aggregates data within those records, creates new JSON documents that contain the aggregated records, and then inserts the new documents into another JSON table. Each record contains the name of an author and the name of a book that the author has written.

The JSON documents have this structure:

```
{
    "_id" : <string or binary>,
    "authorid":"<string>",
    "name":"<string>",
    "book":{
        "id":<int>,
        "title":"<string>"
    }
}
```


last-updated: Apr 11, 2022

The structure of each aggregate record will look like this:

```
{
    "_id" : <string or binary>,
    "authorid":"<string>",
    "book":{
        [
            "title":"<string>",
            "title":"<string>",
            ...
        ]
    }
}
```

Prerequisites

- Ensure that your user ID has the `-readAce` and `-writeAce` privileges on the volumes where you plan to create the source and destination tables.
- Create the source JSON table. You can create the source table and populate it with sample records by running sample_dataset.txt from the `mapr dbshell` utility.

  ```
  $ mapr dbshell < sample_dataset.txt
  ```

- Create the destination JSON table. A simple way to create this table is to use the `create` command in the MapR Database Shell (JSON Tables) on page 5286 utility.

Compiling and Running

To compile an application, use the following command:

```
javac -cp <classpath> <java source file(s)>
```

To launch an application, use the following command:

```
java -cp <classpath>:. -Djava.library.path=/opt/mapr/lib <main class>
<command line arguments>
```

To run the application, supply the paths and names of the source and destination tables as arguments:

```
CombineBookList <source_table> <destination_table>
```

Code Walkthrough

```
  private static Job createSubmittableJob(Configuration conf, String[]
otherArgs)
      throws IOException {

    srcTable = otherArgs[0];
    destTable = otherArgs[1];

    Job job = new Job(conf, NAME + "_" + destTable);
    job.setJarByClass(CombineBookList.class);
    MapRDBMapReduceUtil.configureTableInputFormat(job, srcTable);
    job.setMapperClass(CombineBookListMapper.class);
    MapRDBMapReduceUtil.setMapOutputKeyValueClass(job);
    MapRDBMapReduceUtil.configureTableOutputFormat(job, destTable);
    job.setReducerClass(CombineBookListReducer.class);
```

```
    MapRDBMapReduceUtil.setOutputKeyValueClass(job);
    job.setNumReduceTasks(1);
    return job;
  }
```

The `createSubmittableJob()` method uses methods that are in the `MapRDBMapReduceUtil` class to perform the following tasks:

Set the input format to the default table input format

You can call the `configureTableInputFormat()` method, passing in the job and also passing in the path and name of the source table:

```
MapRDBMapReduceUtil.configureTableInput
tFormat(job, srcTable);
```

The default behavior is to do the following:

- Set the serialization class for `Document` and `Value` objects. These interfaces are part of the OJAI (Open JSON Application Interface) API.
- Set the field `INPUT_TABLE` in `TableInputFormat` to the path and name of the source table, and pass this value to the configuration for the MapReduce application.
- Set the input format class for the job to `TableInputFormat`.

If you want to customize `TableInputFormat` , you can call it as you would normally set the input format for a job:

```
job.setInputFormatClass(TableInputForm
at.class);
```

Set the type for keys and values that are output from the mapper

You can call the `setMapOutputKeyValueClass()` method to use the default type for keys and values:

```
MapRDBMapReduceUtil.setMapOutputKeyVal
ueClass(job);
```

If you want to customize the output keys and values, you can call `Job.setMapOutputKeyClass()` and `Job.setMapOuputValueClass()` as you would normally for MapReduce applications.

Set the output format to the default table output format

You can call the `configureTableOutputFormat()` method, passing in the job and also passing in the path and name of the destination table, which must already exist at runtime:

```
MapRDBMapReduceUtil.configureTableOutp
utFormat(job, destTable);
```

The default behavior is to do the following:

- Set the field `OUTPUT_TABLE` in `TableOutputFormat` to the path and name of the destination table, and pass this value to the configuration for the MapReduce applications.

last-updated: Apr 11, 2022

	• Set the output format class for the job to `TableOutputFormat`. If you want to customize `TableOutputFormat`, you can call it as you would normally set the output format for a job: `job.setOutputFormatClass(TableOutputFormat.class);` You also have the option of using the `BulkLoadOutputFormat` class for bulk loading.
Set the type of the keys and values that are output from the reducer	You can call the `setOutputKeyValueClass()` method to use the default type for keys and values: `MapRDBMapReduceUtil.setOutputKeyValueClass(job);` If you want to customize the output keys and values, you can call `Job.setOutputKeyClass()` and `Job.setOuputValueClass()` as you would normally for MapReduce applications.

The `map()` method in the mapper class `CombineBookListMapper` takes the value of the `_id` field in a document as a key and the JSON document with that `_id` field value as a `Document`. The mapper does nothing with the `Value` object. For each `record`, the mapper writes the value of the `authorid` field and the full JSON document itself to the context.

```
  public static class CombineBookListMapper extends Mapper<Value, Document,
Value, Document> {
    @Override
    public void map(Value key, Document record, Context context) throws
IOException, InterruptedException {
      context.write(record.getValue("authorid"), record);
    }
  }
```

Both the `Value` and `Document` interfaces are part of the OJAI (Open JSON Application Interface) API. The javadoc for the OJAI API is here.

The `reduce()` method in the reducer class `CombineBookListReducer` takes the map output key, which is the value of the `authorid` field, and the map output value, which is an iterator of `Document` objects that each contain a full record. For each author ID, the reducer creates a document. For each document in the iterator, the reducer extracts the value of the `book` field and adds that value to the list `books` within a new JSON document.

```
  public static class CombineBookListReducer extends Reducer<Value,
Document, Value, Document> {

    @Override
    public void reduce(Value key, Iterable<Document> values,
        Context context) throws IOException, InterruptedException {
      Document d = MapRDB.newDocument();
      List<Document> books = new ArrayList<Document>();

      for (Document b : values) {
        books.add((Document)b.getValue("book"));
      }

      d.setId(key);
```

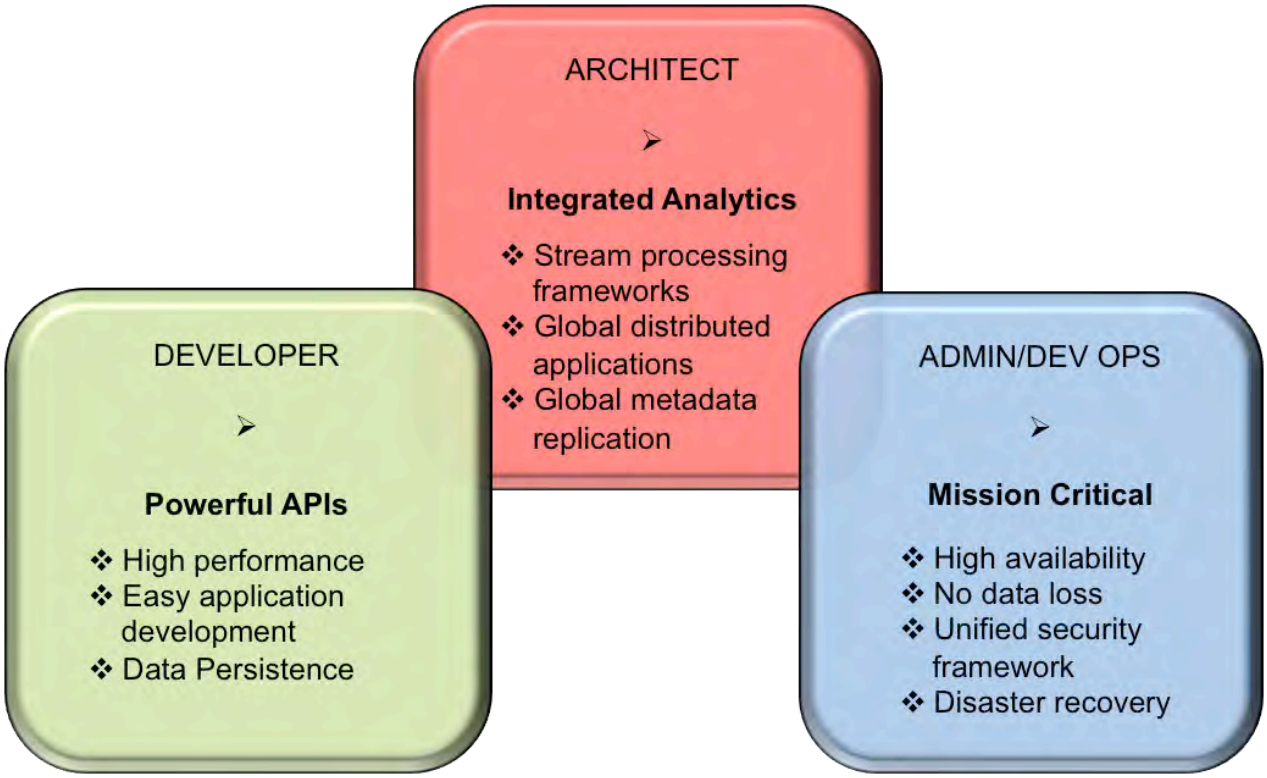
```
      d.set("books", books);
      context.write(key, d);
    }
  }
```

The MapRDB class is part of the MapR Database JSON API, not the MapR Database JSON MapReduce API.

MapR Event Store For Apache Kafka and Apps

MapR Event Store For Apache Kafka brings integrated publish and subscribe messaging to MapR Data Platform.

MapR Event Store For Apache Kafka is built into the MapR Data Platform. It requires no additional process to manage, leverages the same architecture as the rest of the platform, and requires minimal additional management.



1. The Getting Started with MapR Event Store For Apache Kafka section provides overall instructions for setting up, producing, and consuming streams.
2. The MapR Event Store For Apache Kafka section provides conceptual information.
3. The Administering Streams section provides information about creating and managing streams, topics, and stream replication.

Attention: As of core version 6.1, the MapR Event Store For Apache Kafka API enforces a maximum of 4096 partitions for a topic. If you create an application with the MapR Event Store For Apache Kafka 6.1 API, the maximum number of partitions is 4096. If you previously created an application with MapR Event Store For Apache Kafka 6.0.1 API (or older) and you have upgraded, the original number of partitions can be used. For example, if you were using more than 4096 partitions in core version 6.0.1 or earlier, you can continue with the same number of partitions after upgrading.

last-updated: Apr 11, 2022

Getting Started with MapR Event Store For Apache Kafka

If you have a basic understanding of MapR Event Store For Apache Kafka components and the typical flow of messages from producers to consumers, you can get started.

- Ensure that your Linux, Windows, or OS X system has Java SDK 7 or later installed.
- Install the latest version of MapR Data Platform on a cluster.
- Install the core client (mapr-client) package, if you want to run the producer and consumer from a machine outside the cluster. See Installing the MapR Client on page 389 for more information.

1. On a node in the MapR Data Platform cluster, follow these steps:

a) Create a stream.

- Run this command if you plan to run the producer and consumer with the same user ID that you are using to create the stream:

  ```
  maprcli stream create -path /<path to and name of the stream>
  ```

- Run this command if you plan to run the producer and consumer with user IDs that are different from the user ID that you are using to create the stream:

  ```
  maprcli stream create -path /<path to and name of the
  stream> -consumeperm u:<user ID> -produceperm u:<user ID>
  ```

  The two additional parameters grant security permissions. By default, these permissions are granted to the user ID that ran the `maprcli stream create` command.

`-consumeperm`	Grants permission to read messages from topics that are in the stream.
`-produceperm`	Grants permission to publish messages to topics that are in the stream.

b) Create a topic.

Run this command to create the topic:

```
maprcli stream topic create -path <path and name of the stream> -topic
<name of the topic>
```

2. On the system where the mapr-client is installed, compile and launch the Java consumer first and then launch the Java producer.

In both the consumer and producer, change this text to the path and name of your stream and to the name of the first of the topics:

```
/<path to and name of the stream>:<name of topic>
```

For the steps of compiling and launching, see Compiling and Running MapR Event Store For Apache Kafka Java Apps on page 2777.

Launch the consumer first, and then launch the producer. If you launch the producer first and then the consumer, the producer publishes 50 messages, but the consumer (as consumers do by default) starts reading from the head of the partition, which is after the 50 messages.

Note: As of MapR 6.0, the message offset in a partition starts from zero (0). If you are upgrading and do not enable the MapR Database/MapR Event Store For Apache Kafka feature, **mfs.feature.db.streams.v6.support**, the message offset in a partition starts from one (1).

last-updated: Apr 11, 2022

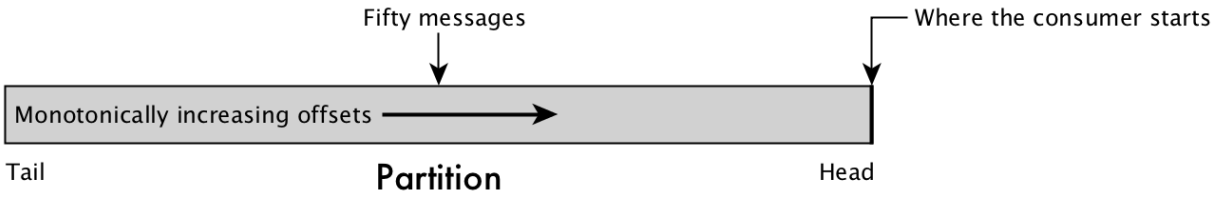


Figure 27: Result of starting the producer before starting the consumer for this step

If you launch the consumer first, the partition is empty and the consumer continuously polls for new messages.

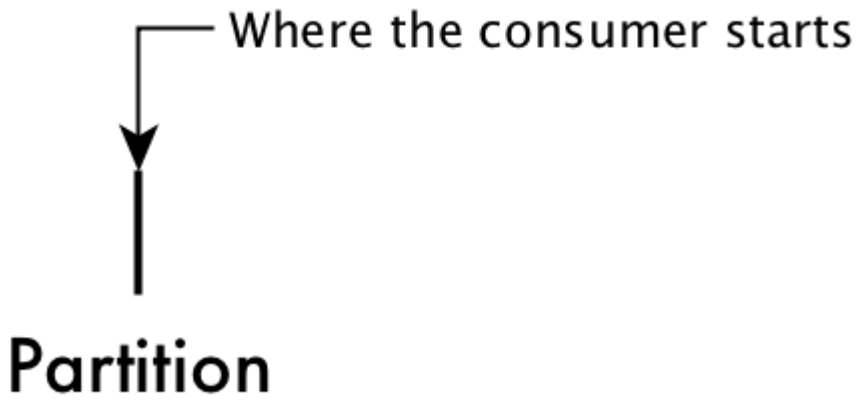


Figure 28: The position of a consumer on an empty partition

After you launch the producer, the fifty messages are published to the partition, and the consumer can move forward in the partition, reading the messages.

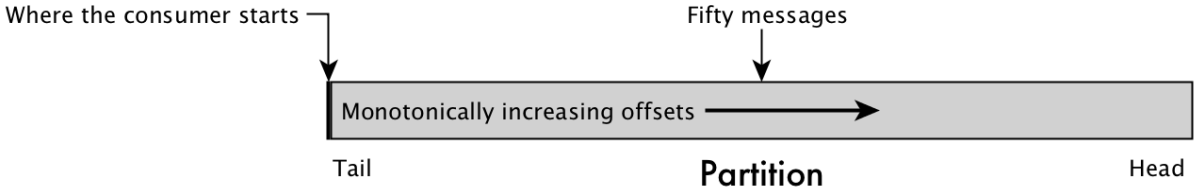


Figure 29: Result of starting the consumer first and then starting the producer for this step

Sample Java Consumer

You need to first add the following dependency to the POM file:

```
<dependency>
  <groupId>commons-logging</groupId>
  <artifactId>commons-logging</artifactId>
  <version>1.1.1</version>
</dependency>
```

```
/* This code is successfully tested for common-logging version 1.11 and
1.2. */
```

```
import org.apache.kafka.clients.consumer.ConsumerConfig;
import org.apache.kafka.clients.consumer.ConsumerRecords;
import org.apache.kafka.clients.consumer.KafkaConsumer;

import java.time.Duration;
import java.util.Collections;
import java.util.Properties;

public class SampleConsumer {
    // Set the stream and topic to read from
    public static String topic = "/<path to and name of the stream>:<name
of topic>";

    // Declare a new consumer.
    public static KafkaConsumer<Integer, String> consumer;

    public static void main(String[] args) {
        configureConsumer();

        // Subscribe to the topic.
        consumer.subscribe(Collections.singletonList(topic));

        // Set the timeout interval for requests for unread messages.
        Duration pollTimeout = Duration.ofMillis(1000);

        try {
            while (true) {
                ConsumerRecords<Integer, String> records =
consumer.poll(pollTimeout);
                records.forEach(record -> {
                    System.out.printf("%s %d %d %s %s \n", record.topic(),
                            record.partition(), record.offset(),
record.key(), record.value());

                });
            }
        } finally {
            consumer.close();
        }
    }

    /* Set the value for a configuration parameter.
       This configuration parameter specifies which
       class to use to deserialize the value of each message. */
    public static void configureConsumer() {
        Properties props = new Properties();
        props.put(ConsumerConfig.GROUP_ID_CONFIG, "consumer-group");
        props.put(ConsumerConfig.AUTO_OFFSET_RESET_CONFIG, "earliest");
        props.put(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG,

"org.apache.kafka.common.serialization.IntegerDeserializer");
        props.put(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG,
                "org.apache.kafka.common.serialization.StringDeserializer");
        consumer = new KafkaConsumer(props);
    }
}
```

Sample Java Producer

```
import org.apache.kafka.clients.producer.KafkaProducer;
import org.apache.kafka.clients.producer.ProducerConfig;
import org.apache.kafka.clients.producer.ProducerRecord;
```

```
import java.util.Properties;

public class SampleProducer {
    // Set the stream and topic to publish to.
    public static String topic = "/<path to and name of the stream>:<name
of topic>";
    // Set the number of messages to send.
    public static int numMessages = 50;

    // Declare a new producer.
    public static KafkaProducer<Integer, String> producer;

    public static void main(String[] args) {
        configureProducer();

        for(int i = 0; i < numMessages; i++) {
            // Set content of each message.
            String messageText = "Msg " + i;

           /* Add each message to a record. A ProducerRecord object
              identifies the topic or specific partition to publish
           a message to. */
            ProducerRecord<Integer, String> rec = new ProducerRecord(topic,
i, messageText);

            // Send the record to the producer client library.
            producer.send(rec);
            System.out.println("Sent message number " + i);
        }
        producer.close();
        System.out.println("All done.");
    }

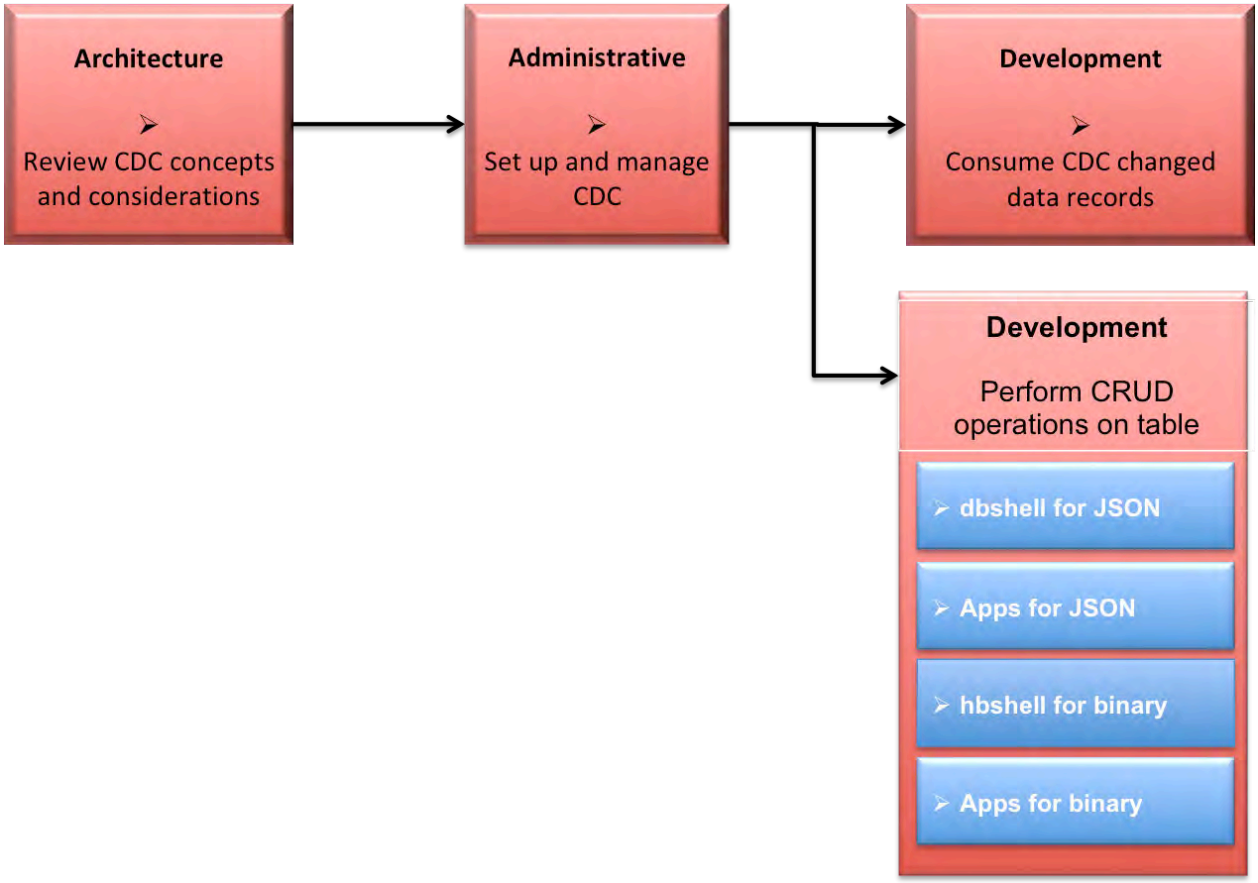
    /* Set the value for a configuration parameter.
       This configuration parameter specifies which class
       to use to serialize the value of each message. */
    public static void configureProducer() {
        Properties props = new Properties();
        props.put(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG,
                "org.apache.kafka.common.serialization.IntegerSerializer");
        props.put(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG,
                "org.apache.kafka.common.serialization.StringSerializer");
        producer = new KafkaProducer(props);
    }
}
```

For additional information, see https://github.com/mapr-demos/mapr-streams-sample-programs.

Consuming CDC Records

The OJAI changelog interfaces are used to consume changed data records (propagated by the Change Data Capture feature).

The general CDC flow of understanding architectural concepts, performing administrative tasks to set up and use CDC, performing CRUD operations on a database table, and developing applications for consuming CDC changed data records. This diagram provides hotspot links to help you navigate to the applicable documentation.



1. Learning about CDC
2. Administering Change Data Capture
3. Building a consumer app for CDC
4. Using dbshell to perform CRUD operations on MapR Database JSON tables
5. Developing client applications for MapR Database JSON tables.
6. Using hbshell to perform CRUD operations on MapR Database binary tables.
7. Developing client applications for MapR Database binary tables.

Javadoc

See the following Java documentation for detailed information about CDC APIs.

Java OJAI CDC API

Deserializer for consuming CDC records

The deserializer converts stream messages into individual change data records. When your application creates a CDC consumer, you must also register the ChangeData deserializer by setting the `value.deserializer` configuration parameter to `com.mapr.db.cdc.ChangeDataRecordDeserializer`.

last-updated: Apr 11, 2022

Note: When applications consume from a CDC change topic, the record key retrieved from `poll()` is not deserialized. The record key is not equal to the `_id` field of the document. If you want to retrieve the exact `_id` of the document, you must call the `ChangeDataRecord.getId()` method.

Interfaces for working with CDC records

The following OJAI interfaces and enumerations create consumers for CDC changed data.

ChangeNode	Contains the change to a single field in a document.
ChangeEvent	Identifies the change event associated with the current change node. The value of ChangeEvent can be one of the following: • NULL (no event) • NODE (a change with real value) • START_MAP (a node representing the beginning of a map) • END_MAP (a node representing the end of a map) • START_ARRAY (a node representing the beginning of an array) • END_ARRAY (a node representing the end of an array)
ChangeOp	Identifies the type of the operation performed on the current field. The values of ChangeOp can be one of the following: • NULL (no operation) • SET (replace the current field with the given value) • PUT (add an extra version of the value) • MERGE (combine the given value with the existing values in the table) • DELETE (delete all values older than or equal to the delete operation timestamp) • DELETE_EXACT (delete the version of the value with the given timestamp)
ChangeDataRecord	Contains all the changes made on a single document/row in the source table.
ChangeDataRecordType	Specifies the mode of change for the change data record. The following values are specified: • RECORD_INSERT • RECORD_UPDATE • RECORD_DELETE
ChangeDataReader	Is a parser that traverses over the individual change tree nodes on a change data record. It provides cursor-like semantics that can be moved, one tree node at a time, by invoking the `next` method. The APIs retrieve the properties of individual change nodes

last-updated: Apr 11, 2022

(for example: data type, field name, field value, and so on).

Open Data Format

The CDC Open Format feature allows you to create applications in languages other then Java that consume CDC (Change Data Capture) changed data records. For example, C/C++, Python, and C#/.NET) are supported.

This functionality is provided with an open format decoder/serializer in the MapR Event Store For Apache Kafka C library. The decoder translates the internal format to the open data format, decodes/deserializes the data, and returns the value of the changed data record as a human readable JSON string.

All languages that are binding through the MapR Event Store For Apache Kafka C library can retrieve the open data format and, with a simple JSON parser, consume changed data records.

Building Consumers for CDC

MapR Event Store For Apache Kafka consumers read and process CDC changed data records. The consumer is built with the OJAI API library.

Description

When building a consumer, the general steps are to:

- Set the consumer properties using Apache Kafka and MapR Data Platform configuration parameters.
- Subscribe to the stream topic.
- Consume the events and determine record type.
- Process the change data records.

The following examples refer to the MapR CDC Sample. See the Java OJAI CDC API for specific API information.

Set Configuration

This code snippet configures the consumer properties using the Apache Kafka configuration parameters. See MapR Event Store For Apache Kafka Configuration Parameters for Consumers. This could be externalized in a file or hard coded in the application code. The following code examples show both methods.

Note: CDC uses a optimized serialization format for all the events, so `value.deserializer` must be set to **com.mapr.db.cdc.ChangeDataRecordDeserializer**.

```
// Consumer configuration parameters specified in application

    Properties consumerProperties = new Properties();
    consumerProperties.setProperty("group.id",
"cdc.consumer.demo_table.fts_geo");
    consumerProperties.setProperty("enable.auto.commit", "true");
    consumerProperties.setProperty("auto.offset.reset", "latest");
    consumerProperties.setProperty("key.deserializer",
"org.apache.kafka.common.serialization.ByteArrayDeserializer");
    consumerProperties.setProperty("value.deserializer",
"com.mapr.db.cdc.ChangeDataRecordDeserializer");
```

```
// Consumer configuration parameters specified in an external file
```


last-updated: Apr 11, 2022

```
key.deserializer=org.apache.kafka.common.serialization.ByteArrayDeserializer
value.deserializer=com.mapr.db.cdc.ChangeDataRecordDeserializer
enable.auto.commit=true
auto.offset.reset=latest
group.id=cdc.consumer.demo_table.fts_geo
```

Subscribe to topic

This code snippet creates the consumer and subscribes to the MapR Event Store For Apache Kafka topic that contains the change data records. The consumer is created using a key (bytes[]) and a `ChangeDataRecord` object for the value.

```
// Consumer used to consume MapR-DB CDC events

KafkaConsumer<byte[], ChangeDataRecord> consumer = new
KafkaConsumer<byte[], ChangeDataRecord>(consumerProperties);
consumer.subscribe(Arrays.asList("/demo_changelog:demo_table"));
```

Consume the events and determine record type

This code snippet polls the topic to determine whether there are any changes and, if so, iterates through the change data records to retrieve the change data record IDs based on the change data record type. The ChangeDataRecordType interface is used to determine the type of record and the ChangeDataRecord interface is used to retrieve the record type and record ID.

```
    while (true) {
      ConsumerRecords<byte[], ChangeDataRecord> changeRecords =
consumer.poll(500);
      Iterator<ConsumerRecord<byte[], ChangeDataRecord>> iter =
changeRecords.iterator();

      while (iter.hasNext()) {
        ConsumerRecord<byte[], ChangeDataRecord> crec = iter.next();
        // The ChangeDataRecord contains all the changes made to a document
        ChangeDataRecord changeDataRecord = crec.value();
        String documentId = changeDataRecord.getId().getString();

        if (changeDataRecord.getType() ==
ChangeDataRecordType.RECORD_INSERT) {
          System.out.println("\n\t Document Inserted " + documentId);
          insertAndUpdateDocument(changeDataRecord, producer);
        } else if (changeDataRecord.getType() ==
ChangeDataRecordType.RECORD_UPDATE) {
          System.out.println("\n\t Document Updated " + documentId);
          insertAndUpdateDocument(changeDataRecord, producer);
        } else if (changeDataRecord.getType() ==
ChangeDataRecordType.RECORD_DELETE) {
          System.out.println("\n\t Document Deleted " + documentId);
          deleteDocument(changeDataRecord, producer);
        }


      }
    }

  }
```


last-updated: Apr 11, 2022

Process the records

This code snippet processes the change data records and based on the type of event (insert, update, delete), using the `ChangeDataRecordType` class and the `changeDataRecord.getType()` method, checks and retrieves the record type.

```
// Use the ChangeNode Iterator to capture all the individual changes

    Iterator<KeyValue<FieldPath, ChangeNode>> cdrItr =
changeDataRecord.iterator();

    while (cdrItr.hasNext()) {
      Map.Entry<FieldPath, ChangeNode> changeNodeEntry = cdrItr.next();
      String fieldPathAsString = changeNodeEntry.getKey().asPathString();
      ChangeNode changeNode = changeNodeEntry.getValue();
      ...
      ...
    }
```

To process and retrieve an inserted new document, you can check to see if the field path is NULL or empty. When *a new document is inserted*, all the changes are made in a single object represented as a Map. You then retrieve the map value by using the `changeNode.getMap()` or `changeNode.getString()` methods depending on the field value.

```
if (fieldPathAsString == null || fieldPathAsString.equals("")) { // Insert
        Map<String, Object> documentInserted = changeNode.getMap();

        if (documentInserted.containsKey("firstName")) {
          fieldToIndex.put("firstName", (String)
documentInserted.get("firstName"));
          sendIndexingMessage = true;
        }

        if (documentInserted.containsKey("lastName")) {
          fieldToIndex.put("lastName", (String)
documentInserted.get("lastName"));
          sendIndexingMessage = true;
        }

        if (documentInserted.containsKey("address")) {
          addressMessage.set("address",
jsonMapper.convertValue((Map)documentInserted.get("address"),
JsonNode.class) );
          sendAddressMessage = true;
        }
     }
```

To process and retrieve updated documents, you can check the field path and retrieve the value depending on the expected value type. When *a document is updated*, the iterator contains one ChangeNode by updated field. You can then access the field path and value directly. You then retrieve the map value by using the `changeNode.getMap()` or `changeNode.getString()` methods depending on the field value.

```
if (fieldPathAsString.equalsIgnoreCase("firstName")) {
          fieldToIndex.put("firstName", changeNode.getString());
          sendIndexingMessage = true;
        }
          else if (fieldPathAsString.equalsIgnoreCase("lastName")) {
          fieldToIndex.put("lastName", changeNode.getString());
          sendIndexingMessage = true;
        }
```

```
          else if (fieldPathAsString.equalsIgnoreCase("address")) {
          addressMessage.set("address",
jsonMapper.convertValue( changeNode.getMap(), JsonNode.class) );
          sendAddressMessage = true;
        }
```

To process delete operations, you can directly retrieve the document ID using the `changeDataRecord.getId()` method and process the document deletion with the `deleteDocument` method. The delete operation is a single change data record.

Consumer Application for CDC JSON Data
This example consumes changed data records from MapR Database JSON tables.

Example of Consuming JSON Changed Data Records

In this example, the following occurs:

- Initialize the consumer properties using Apache Kafka and MapR configuration parameters.
- Display the change data record properties.
- Iterate through the change nodes, determine the type of operation, and retrieve the operation value.
- Retrieve the properties of individual change node (for example: data type, field name, field value, and so on) by using various methods of the `ChangeDataReader` interface.
- Display the change data record values by using the `ChangeNode` interface.
- Subscribe to the stream topic, consume the events, and determine record type.

For changed data records from MapR Database JSON table data, the following are unique:

- There are multiple property values that can be retrieved through the `ChangeDataReader` interface. For example, getDouble or getFloat.
- There are multiple values for single fields in documents that can be retrieved through `ChangeNode` interface. See the code line: `Value value = changeNode.getValue();`

```
package example.cdps;

import com.mapr.db.MapRDB;
import org.apache.kafka.clients.consumer.ConsumerRecord;
import org.apache.kafka.clients.consumer.ConsumerRecords;
import org.apache.kafka.clients.consumer.KafkaConsumer;
import org.ojai.*;
import org.ojai.store.cdc.*;
import java.util.*;

public class CDPConsumer {

    /**
     * Initialize Basic Consumer Properties
     * @return
     */
    public Properties getBasicListnerProperties() {
        Properties props = new Properties();
        props.put("bootstrap.servers", "mfs220.qa.lab:9211");
        props.put("key.deserializer",
"org.apache.kafka.common.serialization.StringDeserializer");
        // Use MapR CDP Specific Deserializer to parse the change contents
        props.put("value.deserializer",
```

```
"com.mapr.db.cdc.ChangeDataRecordDeserializer");
        props.put("fetch.min.bytes", "10");
        props.put("fetch.wait.max.ms", "5000");
        props.put("auto.offset.reset", "earliest");
        props.put("enable.auto.commit", "false");
        return props;
    }

    /**
     * Display Utility
     * @param consumerRecordkey
     * @param id
     * @param changeDataRecordType
     * @param recordOpTime
     * @param recordServerOpTime
     * @param field
     * @param op
     * @param changeNodeOpTime
     * @param changeNodeServerOpTime
     * @param valueType
     * @param value
     */
    public void display(String consumerRecordkey,
                        Value id,
                        ChangeDataRecordType changeDataRecordType,
                        Long recordOpTime,
                        Long recordServerOpTime,
                        String field,
                        ChangeOp op,
                        Long changeNodeOpTime,
                        Long changeNodeServerOpTime,
                        Value.Type valueType,
                        Value value) {

        Document document = MapRDB.newDocument();
        document.set("consumerRecordkey",  consumerRecordkey);

        if(id != null)
            document.set("id", id);

        if(changeDataRecordType != null)
            document.set("changeDataRecordType",
changeDataRecordType.name());

        document.set("recordOpTime", recordOpTime);
        document.set("recordServerOpTime", recordServerOpTime);

        if(field != null)
            document.set("field", field);

        document.set("op", op.name());

        document.set("changeNodeOpTime", changeNodeOpTime);
        document.set("changeNodeServerOpTime", changeNodeServerOpTime);

        if(valueType != null)
            document.set("valueType", valueType.name());

        if(value != null)
            document.set("value", value);

        System.out.println("\t\n********* Propagated Change
************************\t\n");
        System.out.println("\t\n" + document.asJsonString() + "\t\n");
```

```
System.out.println("\t\n**************************************************
**\t\n");
    }

    /**
     * Parse change node contents via iterator
     * @param consumerRecordkey
     * @param changeDataRecord
     */
    public void iteratorDisplay(Value id,
                                ChangeDataRecordType changeDataRecordType,
                                Long recordOpTime,
                                Long recordServerOpTime,
                                String consumerRecordkey,
                                ChangeDataRecord changeDataRecord) {

        for (KeyValue<FieldPath, ChangeNode> fieldChangePair :
changeDataRecord) {

            // field if operation was done one a field
            String field = fieldChangePair.getKey().asJsonString();

            // Actual change node object, which holds change values
            ChangeNode changeNode = fieldChangePair.getValue();

            // Change Op, based on op done can be NULL, SET, MERGE, DELETE,
DELETE_EXACT
            ChangeOp op = changeNode.getOp();

            // change node op time
            Long changeNodeOpTime = changeNode.getOpTimestamp();
            Long changeNodeServerOpTime  = changeNode.getServerTimestamp();

            // the value type if it was non delete operation, such as
insert replace etc
            Value.Type valueType = changeNode.getType();

            // value of the operation such as insert value or replace
            Value value = changeNode.getValue();

            // display the change contents
            display(consumerRecordkey, id, changeDataRecordType,
recordOpTime, recordServerOpTime,
                    field, op, changeNodeOpTime, changeNodeServerOpTime,
valueType, value);
        }
    }

    /**
     * Get Parsed Value
     * @param changeDataReader
     * @param field
     * @param valueType
     * @return
     */
    public Value getValue(ChangeDataReader changeDataReader, String field,
Value.Type valueType) {
        Document valDoc = MapRDB.newDocument();

        if(field == null) {
            valDoc.setNull("null");
            field = "null";
        }
```


last-updated: Apr 11, 2022

```
        switch (valueType) {
            case NULL:
                valDoc.setNull(field);
                break;
            case BOOLEAN:
                valDoc.set(field, changeDataReader.getBoolean());
                break;
            case STRING:
                valDoc.set(field, changeDataReader.getString());
                break;
            case SHORT:
                valDoc.set(field, changeDataReader.getShort());
                break;
            case BYTE:
                valDoc.set(field, changeDataReader.getByte());
                break;
            case INT:
                valDoc.set(field, changeDataReader.getInt());
                break;
            case LONG:
                valDoc.set(field, changeDataReader.getLong());
                break;
            case FLOAT:
                valDoc.set(field, changeDataReader.getFloat());
                break;
            case DOUBLE:
                valDoc.set(field, changeDataReader.getDouble());
                break;
            case DECIMAL:
                valDoc.set(field, changeDataReader.getDecimal());
                break;
            case DATE:
                valDoc.set(field, changeDataReader.getDate());
                break;
            case TIME:
                valDoc.set(field, changeDataReader.getTime());
                break;
            case TIMESTAMP:
                valDoc.set(field, changeDataReader.getTimestamp());
                break;
            case INTERVAL:
                valDoc.set(field, changeDataReader.getInterval());
                break;
            case BINARY:
                valDoc.set(field, changeDataReader.getBinary());
                break;
            default:
                break;
        }
        return valDoc.getValue(field);
    }
    /**
     * Parse change node contents via reader
     * @param consumerRecordkey
     * @param changeDataRecord
     */
    public void readerDisplay(Value id,
                              ChangeDataRecordType changeDataRecordType,
                              Long recordOpTime,
                              Long recordServerOpTime,
                              String consumerRecordkey,
                              ChangeDataRecord changeDataRecord) {
        System.out.println("Reader");
```


last-updated: Apr 11, 2022

```
        ChangeEvent changeEvent;
        // get reader from the event
        ChangeDataReader changeDataReader = changeDataRecord.getReader();

        while ((changeEvent = changeDataReader.next()) != null) {
            // parse through change events
            switch (changeEvent) {
                case NODE:
                    System.out.println("node event get the value type");
                    Value.Type valueType = changeDataReader.getType();
                    String field = changeDataReader.getFieldName();
                    Long serverTimestamp =
changeDataReader.getServerTimestamp();
                    Long opTimestamp = changeDataReader.getOpTimestamp();
                    ChangeOp op = changeDataReader.getOp();
                    Value value = getValue(changeDataReader, field,
valueType);

                    display(consumerRecordkey, id, changeDataRecordType,
                        recordOpTime, recordServerOpTime, field, op,
opTimestamp,
                            serverTimestamp, valueType, value);
                    break;
            }
            break;
        }
    }

    /**
     * Consume from changelog topics
     * @param pollTimeout
     * @param topics
     */
    public void consume(long pollTimeout, String topics, boolean method) {
        System.out.println("consume...");
        // initialize consumer
        KafkaConsumer<String, ChangeDataRecord> consumer = new
KafkaConsumer<String, ChangeDataRecord>
                (getBasicListnerProperties());

        // subscribe to /stream:topic
        List<String> topicList = new ArrayList<String>();
        topicList.add(topics);
        consumer.subscribe(topicList);
        consumer.seekToBeginning();

        // Get consumer records
        ConsumerRecords<String, ChangeDataRecord> consumerRecords =
consumer.poll(pollTimeout);

        // iterate over consumer records
        for(ConsumerRecord<String, ChangeDataRecord> consumerRecord:
consumerRecords) {

            String consumerRecordkey = consumerRecord.key().trim();
            ChangeDataRecord changeDataRecord = consumerRecord.value();

            // record key for the change
            Value id = changeDataRecord.getId();

            // record level op can be either RECORD_INSERT, RECORD_UPDATE,
RECORD_DELETE
            ChangeDataRecordType changeDataRecordType =
changeDataRecord.getType();
```


last-updated: Apr 11, 2022

```
            // record level op-time & server op-time
            Long recordOpTime = changeDataRecord.getOpTimestamp();
            Long recordServerOpTime = changeDataRecord.getServerTimestamp();

            if(method) {
                // Method 1 - via iterator interface
                iteratorDisplay(id, changeDataRecordType,
                        recordOpTime, recordServerOpTime,
                        consumerRecordkey, changeDataRecord);
            } else {
                // Method 2 - via reader interface
                readerDisplay(id, changeDataRecordType,
                        recordOpTime, recordServerOpTime,
                        consumerRecordkey, changeDataRecord);
            }
        }
        consumer.close();
    }

    /**
     * Driver
     * @param args
     */
    public static void main(String[] args) {
        Long pollTimeout = Long.parseLong(args[0]);
        String topic = args[1];
        boolean method = Boolean.parseBoolean(args[2]);
        CDPConsumer cdpConsumer = new CDPConsumer();
        cdpConsumer.consume(pollTimeout, topic, method);
    }
}
```

Consumer Application for CDC Binary Data
This example consumes changed data records from MapR Database Binary tables.

Example of Consuming Binary Changed Data Records

In this example, the following occurs:

- Initialize the consumer properties using Apache Kafka and MapR Data Platform configuration parameters.
- Display the change data record properties.
- Iterate through the change nodes, determine the type of operation, and retrieve the operation value.
- Display the change data record values by using the `ChangeNode` interface.
- Subscribe to the stream topic, consume the events, and determine record type.

For changed data records from MapR Database Binary table data, the following are unique:

- An additional package must be imported: `java.nio.ByteBuffer`
- There is single value for single fields in documents that can be retrieved through `ChangeNode` interface. See the code line: `ByteBuffer value = changeNode.getBinary();`

```
package com.mapr.qa.cdc.tests.binary;

import org.apache.kafka.clients.consumer.ConsumerRecord;
import org.apache.kafka.clients.consumer.ConsumerRecords;
```

```
import org.apache.kafka.clients.consumer.KafkaConsumer;
import org.ojai.*;
import org.ojai.store.Connection;
import org.ojai.store.Driver;
import org.ojai.store.DriverManager;
import org.ojai.store.cdc.*;

import java.nio.ByteBuffer;
import java.util.*;

public class CDCBinaryExample {

    /**
     * Initialize Basic Consumer Properties
     *
     * @return
     */
    public Properties getBasicListnerProperties() {
        Properties props = new Properties();
        props.put("bootstrap.servers", "broker:9092");
        props.put("key.deserializer",
"org.apache.kafka.common.serialization.StringDeserializer");
        // Use MapR CDC Specific Deserializer to parse the change contents
        props.put("value.deserializer",
"com.mapr.db.cdc.ChangeDataRecordDeserializer");
        props.put("fetch.min.bytes", "10");
        props.put("fetch.wait.max.ms", "5000");
        props.put("auto.offset.reset", "earliest");
        return props;
    }

    /**
     * Display Utility
     *
     * @param consumerRecordkey
     * @param id
     * @param changeDataRecordType
     * @param recordOpTime
     * @param recordServerOpTime
     * @param field
     * @param op
     * @param changeNodeOpTime
     * @param changeNodeServerOpTime
     * @param valueType
     * @param value
     */
    public void display(String consumerRecordkey,
                        Value id,
                        ChangeDataRecordType changeDataRecordType,
                        Long recordOpTime,
                        Long recordServerOpTime,
                        String field,
                        ChangeOp op,
                        Long changeNodeOpTime,
                        Long changeNodeServerOpTime,
                        Value.Type valueType,
                        ByteBuffer value) {

        Connection mConnection = DriverManager.getConnection("ojai:mapr:");
        Driver mDriver = mConnection.getDriver();
        Document document = mDriver.newDocument();
        document.set("consumerRecordkey", consumerRecordkey);

        if (id != null)
```

```
            document.set("id", id);

        if (changeDataRecordType != null)
            document.set("changeDataRecordType",
changeDataRecordType.name());

        document.set("recordOpTime", recordOpTime);
        document.set("recordServerOpTime", recordServerOpTime);

        if (field != null)
            document.set("field", field);

        document.set("op", op.name());

        document.set("changeNodeOpTime", changeNodeOpTime);
        document.set("changeNodeServerOpTime", changeNodeServerOpTime);

        if (valueType != null)
            document.set("valueType", valueType.name());

        if (value != null)
            document.set("value", new String(value.array()));

        System.out.println("\t\n********* Propagated Change
************************\t\n");
        System.out.println("\t\n" + document.asJsonString() + "\t\n");

System.out.println("\t\n************************************************
**\t\n");
    }

    /**
     * Parse change node contents via iterator
     *
     * @param consumerRecordkey
     * @param changeDataRecord
     */
    public void iteratorDisplay(Value id,
                                ChangeDataRecordType changeDataRecordType,
                                Long recordOpTime,
                                Long recordServerOpTime,
                                String consumerRecordkey,
                                ChangeDataRecord changeDataRecord) {

        for (KeyValue<FieldPath, ChangeNode> fieldChangePair :
changeDataRecord) {

            // field if operation was done on a field
            String field = fieldChangePair.getKey().asJsonString();

            // Actual change node object, which holds change values
            ChangeNode changeNode = fieldChangePair.getValue();

            // Change Op, based on op done can be NULL, PUT, DELETE,
DELETE_EXACT
            ChangeOp op = changeNode.getOp();

            // change node op time
            Long changeNodeOpTime = changeNode.getOpTimestamp();
            Long changeNodeServerOpTime = changeNode.getServerTimestamp();

            // the value type BINARY, if it is non delete operation
            Value.Type valueType = changeNode.getType();
```

```
            // value of the operation
            ByteBuffer value = changeNode.getBinary();

            // display the change contents
            display(consumerRecordkey, id, changeDataRecordType,
recordOpTime, recordServerOpTime,
                    field, op, changeNodeOpTime, changeNodeServerOpTime,
valueType, value);
        }
    }

    /**
     * Parse change node contents via reader
     *
     * @param consumerRecordkey
     * @param changeDataRecord
     */
    public void readerDisplay(Value id,
                              ChangeDataRecordType changeDataRecordType,
                              Long recordOpTime,
                              Long recordServerOpTime,
                              String consumerRecordkey,
                              ChangeDataRecord changeDataRecord) {

        ChangeEvent changeEvent;
        // get reader from the event
        ChangeDataReader changeDataReader = changeDataRecord.getReader();

        while ((changeEvent = changeDataReader.next()) != null) {
            // parse through change events
            switch (changeEvent) {
                case NODE:
                    System.out.println("node event get the value type");
                    Value.Type valueType = changeDataReader.getType();
                    String field = changeDataReader.getFieldName();
                    Long serverTimestamp =
changeDataReader.getServerTimestamp();
                    Long opTimestamp = changeDataReader.getOpTimestamp();
                    ChangeOp op = changeDataReader.getOp();
                    ByteBuffer value = changeDataReader.getBinary();

                    display(consumerRecordkey, id, changeDataRecordType,
                            recordOpTime, recordServerOpTime, field, op,
opTimestamp,
                            serverTimestamp, valueType, value);
                    break;
            }
            break;
        }
    }

    /**
     * Consume from changelog topics
     *
     * @param pollTimeout
     * @param topics
     */
    public void consume(long pollTimeout, String topics, boolean method) {
        System.out.println("consume...");
        // initialize consumer
        KafkaConsumer<String, ChangeDataRecord> consumer = new
KafkaConsumer<String, ChangeDataRecord>
                (getBasicListnerProperties());
```


last-updated: Apr 11, 2022

```
        // subscribe to /stream:topic
        List<String> topicList = new ArrayList<String>();
        topicList.add(topics);
        consumer.subscribe(topicList);
        consumer.seekToBeginning();

        // Get consumer records
        ConsumerRecords<String, ChangeDataRecord> consumerRecords =
consumer.poll(pollTimeout);

        // iterate over consumer records
        for (ConsumerRecord<String, ChangeDataRecord> consumerRecord :
consumerRecords) {

            String consumerRecordkey = consumerRecord.key().trim();
            ChangeDataRecord changeDataRecord = consumerRecord.value();

            // record key for the change
            Value id = changeDataRecord.getId();

            // record level op can be either RECORD_INSERT, RECORD_UPDATE,
RECORD_DELETE
            ChangeDataRecordType changeDataRecordType =
changeDataRecord.getType();

            // record level op-time & server op-time
            Long recordOpTime = changeDataRecord.getOpTimestamp();
            Long recordServerOpTime = changeDataRecord.getServerTimestamp();

            if (method) {
                // Method 1 - via iterator interface
                iteratorDisplay(id, changeDataRecordType,
                        recordOpTime, recordServerOpTime,
                        consumerRecordkey, changeDataRecord);
            } else {
                // Method 2 - via reader interface
                readerDisplay(id, changeDataRecordType,
                        recordOpTime, recordServerOpTime,
                        consumerRecordkey, changeDataRecord);
            }
        }
        consumer.close();
    }

    /**
     * Driver
     *
     * @param args
     */
    public static void main(String[] args) {
        Long pollTimeout = Long.parseLong(args[0]);
        String topic = args[1];
        boolean method = Boolean.parseBoolean(args[2]);
        CDCBinaryExample cdcBinaryExample = new CDCBinaryExample();
        cdcBinaryExample.consume(pollTimeout, topic, method);
    }
}
```

Open Format
Describes the CDC open format.

Open Format Mapping

The following shows the mapping between the MapR Data Platform CDC data types and the JSON open format data types.

```
{
        "map": {
        "null": null,
        "boolean" : true,
        "string": "eureka",
        "byte" : {"$numberByte": 127},
        "short": {"$numberShort": 32767},
        "int": {"$numberInt": 2147483647},
        "long": {"$numberLong":9223372036854775807},
        "float" : {"$numberFloat":3.4028235E38},
        "double" : 1.7976931348623157e308,
        "decimal": {"$decimal":
"12345678901234567890189012345678901.23456789"},
        "date": {"$dateDay": "yyyy-mm-dd"},
        "time" : {"$time" : "HH:mm:ss[.sss]"},
        "timestamp" : {"$date" : "yyyy-MM-ddTHH:mm:ss.SSSXXX"},
        "interval" : {"$interval" : number_of_millisecods},
        "binary" : {"$binary" : "base64_encoded_binary_value"},
        "array" : [42, "open sesame", 3.14, {"$dateDay": "2015-01-21"}]
        }
}
```

JSON Record Format

When the consumer retrieves the changed data record (by the key-value pair), the record is returned as a string in JSON format (a readable open format). The information about the mutation is returned as an array where each array element is one (1) change.

Note: If you use the default print, the string returns float values of up to six (6) digits of precision and double values of up to fifteen (15) digits. If the data exceeds this default and you want the exact number returned, use the CDC API that returns a float or double value.

The following example changed data record shows two (2) mutations.

```
{
        "_id":"row1"
        "$opType":"$RECORD_UPDATE",
        "$opTime":1518654391801,

        "$mutations":[
        {"$fieldPath":"arrayB,
        "$fieldOp":"$SET",
        "$fieldValue":[{"$numberInt":100},false,"set a map"]
        }
        {"$fieldPath":"arrayC,
        "$fieldOp":"$SET",
        "$fieldValue":[{"$numberInt":200},false,"set a map"]
        }
        ]
}
```


last-updated: Apr 11, 2022

Example

The following sample code initialized consumer properties for open format and consumes the changelog data from the topic.

```
/*
 * Initialize Basic Consumer Properties for Open Format
 * @return
 */

private Properties getOpenFormatListenerProperties() {
   Properties props = new Properties();
   props.put(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG,
"org.apache.kafka.common.serialization.StringDeserializer");
   props.put(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG,
"org.apache.kafka.common.serialization.StringDeserializer");
   return props; }

/*
 * Consume from changelog topic
 *
 */
public void startConsume(String topic) {
   KafkaConsumer<String, String> consumer = new KafkaConsumer<String,
String> (getOpenFormatListnerProperties());
   List<String> topicList = new ArrayList<>();
   topicList.add(topic);
   consumer.subscribe(topicList);

   ConsumerRecords<String, String> consumerRecords =
consumer.poll(pollTimeout);
   Iterator<ConsumerRecord<String, String>> iterator =
consumerRecords.iterator();
   while (iterator.hasNext())

      { ConsumerRecord<String, String> record = iterator.next(); String
cdcResult = record.value(); }

}
```

Consuming Audit Logs

Audit Streaming (available from v6.0.1) provides a way to process the audit data in real-time.

When audit streaming is enabled, the MapR Data Platform generates audit logs that are sent as an audit stream, opening the possibility of real-time processing of the audit data. See Streaming Audit Logs on page 701 for more information.

Use the sample consumer application, or build your custom consumer application, to consume the audit logs that are available as a stream topic, when audit streaming is enabled.

The sample application uses filesystem APIs to get the file path and name from the FID, and the volume name from the volume ID.

Determine When to use Cached or Uncached Version of the Filesystem API

Caching the file path and file name, along with the volume name at the initial API call, reduces the load on CLDB for subsequent API calls.

However, there could be cases when the uncached version of the application is more suitable for use. Consider the following example:

For the initial API call, File1 is returned as the file name for FID 1. The result is cached.

The file is then renamed to File2. For subsequent API calls, to get the file name for FID 1, the result from the cache is used. The cache, unaware of the rename operation, returns the name as File1, which is incorrect, as the file is already renamed to File2. For such a case, use the uncached version.

Evaluate your use case, and then use the cached, or the uncached version, as appropriate.

Sample Cached Consumer Application for Audit Stream
The Consumer.java application demonstrates how to connect to the MapR Filesystem filesystem, and consume the messages in a stream topic.

Sample Application

Before running this application, ensure that you have access to a cluster running MapR File System. To build and run this application:

1. Set the classpath as shown below:

```
export CLASSPATH=`hadoop classpath`
```

2. Compile the Java file as shown below:

```
javac -cp .:`mapr classpath` Consumer.java
```

3. Run the final `Consumer.class` file. For example:

```
java -cp .:`mapr classpath` Consumer
```

This application requires the following imports:

- `org.apache.kafka.clients.consumer.ConsumerRecord`
- `org.apache.kafka.clients.consumer.ConsumerRecords`
- `org.apache.kafka.clients.consumer.KafkaConsumer`
- `org.apache.hadoop.conf.Configuration`
- `com.mapr.fs.MapRFileSystem`
- `com.google.common.io.Resources`
- `java.net.URI`
- `java.io.IOException`
- `java.io.InputStream`
- `java.util.Iterator`
- `java.util.Properties`
- `java.util.Random`
- `java.util.StringTokenizer`
- `java.util.regex.Pattern`

The application performs the actions described in the following sections.

Initializes the consumer properties

The configuration parameters for the consumer are stored in `consumer.props` file. This file should be present in the current directory or mapr classpath. For example, your `consumer.props` file could look similar to the following:

```
#bootstrap.servers=localhost:9092
group.id=test
enable.auto.commit=true
key.deserializer=org.apache.kafka.comm
on.serialization.StringDeserializer
value.deserializer=org.apache.kafka.co
mmon.serialization.StringDeserializer
# fast session timeout makes it more
fun to play with failover
## apps specific ?
session.timeout.ms=10000

# These buffer sizes are needed to
avoid consumer switching to
# a mode where it processes one
bufferful every 5 seconds with
multiple
# timeouts along the way.
fetch.min.bytes=50000
# receive.buffer.bytes=262144 //
fixed size buffer
max.partition.fetch.bytes=2097152

auto.offset.reset=earliest
```

The application initializes the consumer properties stored in the `consumer.props` file.

```
public static void main(String[]
args) throws
IOException,InterruptedException {
    KafkaConsumer<String, String>
consumer;
    try (InputStream props =
Resources.getResource("consumer.props"
).openStream()) {
        Properties properties = new
Properties();
        properties.load(props);
        if
(properties.getProperty("group.id")
== null) {

properties.setProperty("group.id",
"group-" + new
Random().nextInt(100000));
        }
        consumer = new
KafkaConsumer<>(properties);
    }
}
```

Subscribes to the topic to read from

The application initializes the filesystem object, with the last parameter as `true` so that the audit logs generated by the operations for converting fid to file path and volid to volume name are sent

to the `ExpandAudit.json.log` file used by the `expandaudit on page 2096` utility and not to the stream. It then selects the stream and subscribes to the topic to read at path `/var/mapr/auditstream/auditlogstream:my.cluster.com`.

```
Configuration conf = new
Configuration();
String uri = MAPRFS_URI;
uri = uri + "mapr/";
conf.set("fs.default.name", uri);
MapRFileSystem fs = new
MapRFileSystem();
fs.initialize(URI.create(uri), conf,
true);
Pattern pattern =
Pattern.compile("/var/mapr/
auditstream/
auditlogstream:my.cluster.com.auth.+")
;
consumer.subscribe(pattern,null);
```

Requests unread messages from the topic

The application requests to read unread messages in the subscribed topic. It then iterates through the returned records, extracts the value of each message, and prints the value to the standard output.

```
boolean stop = false;
int pollTimeout = 1000;
while (!stop) {
    ConsumerRecords<String,
String> consumerRecords =
consumer.poll(pollTimeout);
    Iterator<ConsumerRecord<String,
String>> iterator =
consumerRecords.iterator();
    if (iterator.hasNext()) {
      while (iterator.hasNext()) {
        ConsumerRecord<String,
String> record = iterator.next();
        String value = record.value();
        String rvalue =
value.replace("\"","");
        String recordValue
= processRecord(fs, rvalue,
value);
        System.out.println(("
Consumed Record: " + recordValue));
      }
    } else {
      Thread.sleep(1000);
      //stop = true;
    }
}
```

Gets the record and expands individual fields

The application then takes the record and expands fid in the message to path to file using the `getMountPathFidCached()` API and

volid in the message to volume name using the `getVolumeNameCached()` API.

```
while (st.hasMoreTokens()) {
    String field = st.nextToken();
    StringTokenizer st1 = new
StringTokenizer(field, ":");
    while (st1.hasMoreTokens()) {
        String token =
st1.nextToken();
        if (token.endsWith("Fid")) {
           String lfidStr =
st1.nextToken();
           String path= null;
           try {
              path =
fs.getMountPathFidCached(lfidStr); //
Expand FID to path
           } catch (IOException e){
        }
        lfidPath =
"\"FidPath\":\""+path+"\",";
        // System.out.println("\nPAth
for fid " + lfidStr +  "is " +  path);
    }
    if (token.endsWith("volumeId")) {
        String volid =
st1.nextToken();
        String name= null;
        try {
            int volumeId =
Integer.parseInt(volid);
            name =
fs.getVolumeNameCached(volumeId); //
Cached API to convert volume Id to
volume Name
        } catch (IOException e){
        }
        lvolName =
"\"VolumeName\":\""+name+"\",";
        //
System.out.println("\nVolume Name for
volid " + volid +  "is " +  name);
    }
}
```

Returns the record

The application finally returns the record after expanding the fid and volid to file path and volume name respectively.

```
String result = "";
StringTokenizer st2 = new
StringTokenizer(value,",");
while (st2.hasMoreTokens()) {
     String tokens = st2.nextToken();
     result = result + tokens + ",";
     if (tokens.contains("Fid")) {
         result = result + lfidPath;
     }
     if (tokens.contains("volumeId"))
{
         result = result + lvolName;
```

```
    }
}
return result.substring(0,
result.length() - 1);
```

Consumer.java

```
//package com.mapr.examples;
//import com.mapr.fs;

import org.apache.kafka.clients.consumer.ConsumerRecord;
import org.apache.kafka.clients.consumer.ConsumerRecords;
import org.apache.kafka.clients.consumer.KafkaConsumer;

import java.net.*;
import org.apache.hadoop.fs.*;
import org.apache.hadoop.conf.*;
import com.mapr.fs.MapRFileSystem;
import com.mapr.fs.*;

import com.google.common.io.Resources;
import org.apache.kafka.common.PartitionInfo;
import org.apache.kafka.common.TopicPartition;

import java.io.IOException;
import java.util.*;
import java.io.InputStream;
import java.util.regex.*;

public class Consumer {
  // Set the stream and topic to read from.
  private static final String MAPRFS_URI = "maprfs:///";
  public static void main(String[] args) throws
IOException,InterruptedException {
    //configureConsumer(args);
    // and the consumer
    KafkaConsumer<String, String> consumer;
    try (InputStream props =
Resources.getResource("consumer.props").openStream()) {
      Properties properties = new Properties();
      properties.load(props);
      if (properties.getProperty("group.id") == null) {
        properties.setProperty("group.id", "group-" + new
Random().nextInt(100000));
      }

      consumer = new KafkaConsumer<>(properties);
    }

    Configuration conf = new Configuration();
    String uri = MAPRFS_URI;
    uri = uri + "mapr/";
    conf.set("fs.default.name", uri);
    MapRFileSystem fs = new MapRFileSystem();
    fs.initialize(URI.create(uri), conf, true);
    //final String topic = "/var/mapr/auditstream/
auditlogstream:my.cluster.com_atsqa4-130.qa.lab";
    Pattern pattern = Pattern.compile("/var/mapr/auditstream/
auditlogstream:my.cluster.com.auth.+");
    // Subscribe to the topic.
    consumer.subscribe(pattern,null);
```

```
    boolean stop = false;
    int pollTimeout = 1000;
    while (!stop) {
      // Request unread messages from the topic.
      ConsumerRecords<String, String> consumerRecords =
consumer.poll(pollTimeout);
      Iterator<ConsumerRecord<String, String>> iterator =
consumerRecords.iterator();
      if (iterator.hasNext()) {
        while (iterator.hasNext()) {
          ConsumerRecord<String, String> record = iterator.next();
          // Iterate through returned records, extract the value
          // of each message, and print the value to standard output.
          //System.out.println((" Consumed Record: " + record.toString()));
          String value = record.value();
          String rvalue = value.replace("\"","");
          String recordValue = processRecord(fs, rvalue, value);

          System.out.println((" Consumed Record: " + recordValue));
          //System.out.println((" Consumed Record: " + value));
        }
      } else {
    Thread.sleep(1000);
        //stop = true;
      }
    }
    consumer.close();
    System.out.println("All done.");
  }

  /* Get the record and expand individual fields */
  public static String processRecord(MapRFileSystem fs, String rvalue,
String value)
  {
     StringTokenizer st = new StringTokenizer(rvalue, ",");
     String lfidPath = "";
     String lvolName = "";

     while (st.hasMoreTokens())
     {
       String field = st.nextToken();
       StringTokenizer st1 = new StringTokenizer(field, ":");
       while (st1.hasMoreTokens())
       {
         String token = st1.nextToken();
     /* If the field has fid, expand it using Cached API */
         if (token.endsWith("Fid")) {
           String lfidStr = st1.nextToken();
           String path= null;
           try {
               path = fs.getMountPathFidCached(lfidStr); // Expand FID to
path
           } catch (IOException e){
           }
           lfidPath = "\"FidPath\":\""+path+"\",";
           // System.out.println("\nPAth for fid " + lfidStr +  "is " +
path);
         }

         if (token.endsWith("volumeId")) {
           String volid = st1.nextToken();
           String name= null;
           try {
```

```
              int volumeId = Integer.parseInt(volid);
                name = fs.getVolumeNameCached(volumeId); // Cached API to
convert volume Id to volume Name
              }
            catch (IOException e){
            }
            lvolName = "\"VolumeName\":\""+name+"\",";
            //  System.out.println("\nVolume Name for volid " + volid +  "is
" +  name);
          }
        }
      }
      String result = "";
      StringTokenizer st2 = new StringTokenizer(value,",");
      while (st2.hasMoreTokens()) {
        String tokens = st2.nextToken();
        result = result + tokens + ",";
        if (tokens.contains("Fid")) {
          result = result + lfidPath;
        }
        if (tokens.contains("volumeId")) {
          result = result + lvolName;
        }
      }
      //return record after expansion of fid and volume id
      return result.substring(0, result.length() - 1);
  }
}
```

Related tasks

Enabling and Disabling Audit Streaming Using the CLI on page 764
Explains how to enable or disable audit streaming using the CLI.

Related reference

audit cluster on page 1553
Enables and disables auditing of operations that are related to the administration of a MapR cluster.

Related information

Streaming Audit Logs on page 701
Describes the audit streaming feature and how to consume the audit stream messages.

Sample Uncached Consumer Application for Audit Stream

The ConsumerUncached.java application demonstrates how to connect to the MapR Data Platform file system, and consume the messages in a stream topic.

Sample Application

Before running this application, ensure that you have access to a cluster running MapR File System. To build and run this application:

1. Set the classpath as shown below:

```
export CLASSPATH=`hadoop classpath`
```

2. Compile the Java file as shown below:

```
javac -cp .:`mapr classpath` ConsumerUncached.java
```


last-updated: Apr 11, 2022

3. Run the final `ConsumerUncached.class` file. For example:

```
java -cp .:`mapr classpath` ConsumerUncached
```

This application requires the following:

- `org.apache.kafka.clients.consumer.ConsumerRecord`
- `org.apache.kafka.clients.consumer.ConsumerRecords`
- `org.apache.kafka.clients.consumer.KafkaConsumer`
- `org.apache.hadoop.conf.Configuration`
- `com.mapr.fs.MapRFileSystem`
- `com.google.common.io.Resources`
- `java.net.URI`
- `java.io.IOException`
- `java.io.InputStream`
- `java.util.Iterator`
- `java.util.Properties`
- `java.util.Random`
- `java.util.StringTokenizer`
- `java.util.regex.Pattern`

The application performs the actions described in the following sections.

Initializes the consumer properties

The configuration parameters for the consumer are stored in `consumer.props` file. This file should be present in the current directory or mapr classpath. For example, your `consumer.props` file could look similar to the following:

```
#bootstrap.servers=localhost:9092
group.id=test
enable.auto.commit=true
key.deserializer=org.apache.kafka.comm
on.serialization.StringDeserializer
value.deserializer=org.apache.kafka.co
mmon.serialization.StringDeserializer

# fast session timeout makes it more
fun to play with failover
## apps specific ?
session.timeout.ms=10000

# These buffer sizes are needed to
avoid consumer switching to
# a mode where it processes one
bufferful every 5 seconds with
multiple
```

```
# timeouts along the way.
fetch.min.bytes=50000
# receive.buffer.bytes=262144 //
fixed size buffer
max.partition.fetch.bytes=2097152

auto.offset.reset=earliest
```

The application initializes the consumer properties stored in the `consumer.props` file.

```
public static void main(String[]
args) throws
IOException,InterruptedException {
    KafkaConsumer<String, String>
consumer;
    try (InputStream props =
Resources.getResource("consumer.props"
).openStream()) {
          Properties properties = new
Properties();
          properties.load(props);
          if
(properties.getProperty("group.id")
== null) {

properties.setProperty("group.id",
"group-" + new
Random().nextInt(100000));
          }
      consumer = new
KafkaConsumer<>(properties);
      }
}
```

Subscribes to the topic to read from

The application initializes the filesystem object, with the last parameter as `true` so that the audit logs generated by the operations for converting fid to file path and volid to volume name are sent to the `ExpandAudit.json.log` file used by the `expandaudit` on page 2096 utility and not to the stream. It then selects the stream and subscribes to the topic to read at path `/var/mapr/auditstream/auditlogstream:my.cluster.com`.

```
Configuration conf = new
Configuration();
String uri = MAPRFS_URI;
uri = uri + "mapr/";
conf.set("fs.default.name", uri);
MapRFileSystem fs = new
MapRFileSystem();
fs.initialize(URI.create(uri), conf,
true);

Pattern pattern =
Pattern.compile("/var/mapr/
auditstream/
auditlogstream:my.cluster.com.+");
consumer.subscribe(pattern,null);
```

Requests unread messages from the topic

The application requests to read unread messages in the subscribed topic. It then iterates through the returned records, extracts the value of each message, and prints the value to the standard output.

```
boolean stop = false;
int pollTimeout = 1000;
while (!stop) {
      ConsumerRecords<String,
String> consumerRecords =
consumer.poll(pollTimeout);
      Iterator<ConsumerRecord<String,
String>> iterator =
consumerRecords.iterator();
      if (iterator.hasNext()) {
          while (iterator.hasNext()) {
              ConsumerRecord<String,
String> record = iterator.next();
              String value =
record.value();
              String rvalue =
value.replace("\"","");
              String recordValue =
processRecord(fs, rvalue, value);
              System.out.println(("
Consumed Record: " + recordValue));
          }
      } else {
                  //stop = true;
      }
}
```

Gets the record and expands individual fields

The application then takes the record and expands fid in the message to path to file using the `getMountPathFid()` API and volid in the message to volume name using the `getVolumeName()` API.

```
public static String
processRecord(MapRFileSystem fs,
String rvalue, String value)
  {
     StringTokenizer st = new
StringTokenizer(rvalue, ",");
     String lfidPath = "";
     String lvolName = "";

     while (st.hasMoreTokens()) {
          String field =
st.nextToken();
          StringTokenizer st1 = new
StringTokenizer(field, ":");
          while (st1.hasMoreTokens())
{
               String token =
st1.nextToken();
               if
(token.endsWith("Fid")) {
             String lfidStr =
st1.nextToken();
             String path= null;
             try {
                   path =
```

```
fs.getMountPathFid(lfidStr);
            } catch (IOException e)
{ }
            lfidPath =
"\"FidPath\":\""+path+"\",";
             if
(token.endsWith("volumeId")) {
          String volid =
st1.nextToken();
          String name= null;
          try {
              int volumeId =
Integer.parseInt(volid);
                name =
fs.getVolumeName(volumeId);
            }
          catch (IOException e){ }
          lvolName =
"\"VolumeName\":\""+name+"\",";
             }
       }
    }
```

Returns the record

The application finally returns the record after expanding the fid and volid to file path and volume name respectively.

```
String result = "";
StringTokenizer st2 = new
StringTokenizer(value,",");
while (st2.hasMoreTokens()) {
    String tokens = st2.nextToken();
    result = result + tokens + ",";
    if (tokens.contains("Fid")) {
       result = result + lfidPath;
    }
    if (tokens.contains("volumeId")) {
       result = result + lvolName;
    }
    return result.substring(0,
result.length() - 1);
}
```

ConsumerUncached.java

```
//package com.mapr.examples;
//import com.mapr.fs;

import org.apache.kafka.clients.consumer.ConsumerRecord;
import org.apache.kafka.clients.consumer.ConsumerRecords;
import org.apache.kafka.clients.consumer.KafkaConsumer;

import java.net.*;
import org.apache.hadoop.fs.*;
import org.apache.hadoop.conf.*;
import com.mapr.fs.MapRFileSystem;
import com.mapr.fs.*;

import com.google.common.io.Resources;
import org.apache.kafka.common.PartitionInfo;
import org.apache.kafka.common.TopicPartition;
```

```
import java.io.IOException;
import java.util.*;
import java.io.InputStream;
import java.util.regex.*;

public class ConsumerUncached {
  // Set the stream and topic to read from.
  private static final String MAPRFS_URI = "maprfs:///";
  public static void main(String[] args) throws
IOException,InterruptedException {
    //configureConsumer(args);
    // and the consumer
    KafkaConsumer<String, String> consumer;
    try (InputStream props =
Resources.getResource("consumer.props").openStream()) {
      Properties properties = new Properties();
      properties.load(props);
      if (properties.getProperty("group.id") == null) {
        properties.setProperty("group.id", "group-" + new
Random().nextInt(100000));
      }

      consumer = new KafkaConsumer<>(properties);
    }

    Configuration conf = new Configuration();
    String uri = MAPRFS_URI;
    uri = uri + "mapr/";
    conf.set("fs.default.name", uri);
    MapRFileSystem fs = new MapRFileSystem();
    fs.initialize(URI.create(uri), conf, true);
    //final String topic = "/var/mapr/auditstream/
auditlogstream:my.cluster.com_atsqa4-130.qa.lab";
    Pattern pattern = Pattern.compile("/var/mapr/auditstream/
auditlogstream:my.cluster.com.+");
    // Subscribe to the topic.
    consumer.subscribe(pattern,null);

    boolean stop = false;
    int pollTimeout = 1000;
    while (!stop) {
      // Request unread messages from the topic.
      ConsumerRecords<String, String> consumerRecords =
consumer.poll(pollTimeout);
      Iterator<ConsumerRecord<String, String>> iterator =
consumerRecords.iterator();
      if (iterator.hasNext()) {
        while (iterator.hasNext()) {
          ConsumerRecord<String, String> record = iterator.next();
          // Iterate through returned records, extract the value
          // of each message, and print the value to standard output.
          //System.out.println((" Consumed Record: " + record.toString()));
          String value = record.value();
          String rvalue = value.replace("\"","");
          String recordValue = processRecord(fs, rvalue, value);

          System.out.println((" Consumed Record: " + recordValue));
          //System.out.println((" Consumed Record: " + value));
        }
      } else {
    Thread.sleep(1000);
        //stop = true;
      }
```

```
    }
    consumer.close();
    System.out.println("All done.");
  }
  public static String processRecord(MapRFileSystem fs, String rvalue,
String value)
  {
     StringTokenizer st = new StringTokenizer(rvalue, ",");
     String lfidPath = "";
     String lvolName = "";

     while (st.hasMoreTokens())
     {
       String field = st.nextToken();
       StringTokenizer st1 = new StringTokenizer(field, ":");
       while (st1.hasMoreTokens())
       {
         String token = st1.nextToken();
         if (token.endsWith("Fid")) {
           String lfidStr = st1.nextToken();
           String path= null;
           try {
               path = fs.getMountPathFid(lfidStr);
           } catch (IOException e){
           }
           lfidPath = "\"FidPath\":\""+path+"\",";
           // System.out.println("\nPAth for fid " + lfidStr +  "is " +
path);
         }

         if (token.endsWith("volumeId")) {
           String volid = st1.nextToken();
           String name= null;
           try {
             int volumeId = Integer.parseInt(volid);
               name = fs.getVolumeName(volumeId);
             }
           catch (IOException e){
           }
           lvolName = "\"VolumeName\":\""+name+"\",";
           //  System.out.println("\nVolume Name for volid " + volid +  "is
" +  name);
         }
       }
     }
     String result = "";
     StringTokenizer st2 = new StringTokenizer(value,",");
     while (st2.hasMoreTokens()) {
       String tokens = st2.nextToken();
       result = result + tokens + ",";
       if (tokens.contains("Fid")) {
         result = result + lfidPath;
       }
       if (tokens.contains("volumeId")) {
         result = result + lvolName;
       }
     }
     return result.substring(0, result.length() - 1);
  }
}
```

Related tasks

Enabling and Disabling Audit Streaming Using the CLI on page 764

last-updated: Apr 11, 2022

Explains how to enable or disable audit streaming using the CLI.

Related reference

audit cluster on page 1553
Enables and disables auditing of operations that are related to the administration of a MapR cluster.

Related information

Streaming Audit Logs on page 701
Describes the audit streaming feature and how to consume the audit stream messages.

MapR Event Store For Apache Kafka Java Applications

This section contains information on developing client applications with Java including information about the MapR Event Store For Apache Kafka and Apache Kafka Java APIs, configuration parameters, and compiling and running producers and consumers.

Apache Kafka Support

MapR Data Platform supports the following Apache Kafka Java API versions:

Table

Core version	Apache Kafka API
As of 6.2	2.1
As of 6.1	1.1
As of 6.0.1	1.0
6.0.0 and earlier	0.9.0

Log Compaction

As of MapR Data Platform 6.1, log compaction is supported. Log compaction can be enabled for streams created with MapR Data Platform core 6.1 and later. In addition, clients older thanMapR Data Platform 6.1 are prevented from consuming from streams that have had log compaction enabled on them at least once in their lifetime.

When a stream on a source cluster has both log compaction and replication enabled, the replica cluster does not automatically have log compaction enabled. You must explicitly enable log compaction on the replica cluster.

- If a replica cluster has been upgraded and the stream data for a source cluster is compacted (that is, one or more messages have been deleted), then the source cluster replicates the compacted data to the replica cluster.
- If a replica cluster has ***not*** been upgraded, then the source cluster fails the replication and an error is generated that requests an replica cluster upgrade.

In the context of a scan by a client that is **not** upgraded, the (upgraded) server inspects the row header to check if it is serving a compacted row. If it is serving a compacted row, then the server fails the consumer request. This behavior applies both to a stream that is explicitly configured for compaction and a replica that has received a compacted row.

Important: To perform log compaction on older streams, the `-force` option can be used. The `-force` option should only be used when ALL clients have been upgraded to MapR Data Platform 6.1.

last-updated: Apr 11, 2022

Idempotent Producer

As of MapR Data Platform 6.1, the idempotent producer (exactly once) feature is supported. You can implement the idempotent producer with MapR Data Platform core 6.1 and later.

When creating a producer instance, use the following configuration:

```
props.put(ProducerConfig.ENABLE_IDEMPOTENCE_CONFIG, true)
```

The idempotent producer feature is supported by EEP MapR Data Platform 6.0 clients and MapR Data Platform 6.1.0 servers.

- You must upgrade all servers to v6.1.0 and enable all the v6.1.0 features, before you enable the idempotent producer.
- If you use a pre-MapR Data Platform 6.1 client and a MapR Data Platform 6.1 server, and if a group of messages are atomically persisted without a valid producer ID, the server treats the request as a non-idempotent producer.
- If you use a MapR Data Platform 6.1 client and a pre-MapR Data Platform 6.1 server, the idempotent producer is not supported. In this case, the idempotent producer fails to produce to the stream and the following exception is thrown:

```
Exception in thread "main" java.util.concurrent.ExecutionException:
org.apache.kafka.common.errors.UnknownTopicOrPartitionException:
Operation not permitted (1) null
        at
com.mapr.streams.impl.producer.MarlinFuture.valueOrError(MarlinFuture.java
:46)
        at
com.mapr.streams.impl.producer.MarlinFuture.get(MarlinFuture.java:41)
        at
com.mapr.streams.impl.producer.MarlinFuture.get(MarlinFuture.java:17)
        at
com.mapr.qa.marlin.common.StandaloneProducer.main(StandaloneProducer.java:
75)
Caused by:
org.apache.kafka.common.errors.UnknownTopicOrPartitionException:
Operation not permitted (1) null
```

TimestampType Permissions

The following discussion describes the *ACE* permissions that you need when using the timestamp type parameter. See Stream Security on page 664 for general information about MapR Event Store For Apache Kafka streams security.

A MapR Event Store For Apache Kafka stream topic inherits the default timestamp type value from its stream. To override the stream's default value, set the timestamp type for the topic to a different value.

- Setting the value at the stream-level requires `adminperm` permissions. The stream-level timestamp type parameter is `defaulttimestamptype`. See stream create on page 1758 and stream edit on page 1765 for more information on setting this parameter using the `maprcli` command.
- Setting the `timestamptype` at the topic-level requires `topicperm` permissions. The topic-level timestamp type parameter is `timestamptype`. See stream topic create on page 1781 and stream topic edit on page 1784 for more information on setting this parameter using the `maprcli` command.

User Impersonation

As of MapR Data Platform 6.0, user impersonation is supported for MapR Event Store For Apache Kafka.

last-updated: Apr 11, 2022

You can set up user impersonation programmatically. To do so, use the `UserGroupInformation.doAs()` method in the Hadoop documentation. See Class UserGroupInformation for more information.

If you are setting up user impersonation in a secure cluster, you need to generate an impersonation ticket. See the Generating and Printing Service with Impersonation Ticket section in the maprlogin Command Examples on page 2134 topic.

After generating the ticket:

1. Ensure that user `mapruser1` has read permissions on the ticket.
2. If you moved the ticket file to a different location, set the `$MAPR_TICKETFILE_LOCATION` environment variable with the appropriate path.

For more information about impersonation, see:

- How Impersonation Works on page 1478
- Generating a Service with Impersonation Ticket on page 1428
- Managing Impersonation on page 1476

Backward Compatibility

As of MapR Data Platform 6.0.1, along with the support of Apache Kafka, the `java.util.Collection` interface is being used. This impacts applications using certain APIs. See MapR Event Store For Apache Kafka Java API Library on page 2756 for detailed information.

References

- MapR Event Store For Apache Kafka Sample Programs on GitHub.

MapR Event Store For Apache Kafka Java API Library

Use the MapR Event Store For Apache Kafka Admin Java API library as an alternative to `maprcli` commands and the REST APIs for performing administrative tasks on streams and topics. This library can also be used for analysis of the contents of streams.

Javadoc

The following Apache Kafka Java API versions are supported:

Table

Core version	Apache Kafka API
As of 6.1	1.1
As of 6.0.1	1.0
6.0.0 and earlier	0.90

See the following APIs for detailed information:

- MapR Event Store For Apache Kafka Java API Library (6.1)
- Apache Kafka 1.1 APIs used with MapR Event Store For Apache Kafka (6.1)

Java Interfaces, Classes, and Enums

The MapR Event Store For Apache Kafka Java API library consists of the following interfaces and class:

Admin	Provides methods for performing administrative tasks on streams and topics, as well as for obtaining `StreamDescriptor` and `TopicDescriptor` objects.
StreamDescriptor	Provides methods for setting and retrieving values for stream attributes. `StreamDescriptor` is passed into methods when performing operations on streams, for example, creating a new stream and editing an existing stream. `StreamDescriptor` is also used to find attribute values for an existing stream.
TopicDescriptor	**Note:** TopicDescriptor is new as of 6.0.1. Provides methods for setting and retrieving topic values. `TopicDescriptor` is passed into methods when performing operations on topics, for example, setting and retrieving topic partitions and timestamp type. The available timestamp type can be either `createtime` or `logappendtime`. When a producer writes a message to a MapR Event Store For Apache Kafka topic, a timestamp is included that is part of the message record. This timestamp can be used to implement time-based indexing. Time-based indexing enables consumers to find the offsets for messages based on timestamps. The returned message offset corresponds to the *earliest* topic-partition message whose timestamp is equal to or greater than the provided timestamp. **Note:** APIs that support timeout semantics will adhere to the specified timeouts only if the client is configured for soft mount.
Streams Class	Streams class is also required to create an instance of Admin, which is used for all admin operations on streams. It can also be used to create a StreamDescriptor or TopicDescriptor objects. Provides the entry point to accessing MapR Event Store For Apache Kafka streams for analytics purposes.
TimestampType Enum	Provides the timestamp type of the records.
TopicRefreshListListener / TopicRefreshRegexListener	**Note:** Internal; not for public usage.

MapR Event Store For Apache Kafka Java APIs (as of 6.1)

The following MapR Event Store For Apache Kafka Java APIs are available as of MapR Data Platform 6.1:

Table

Interface	Method	Description
`StreamDescriptor`	`void setCompact(boolean compact)`	Sets log compaction on a stream.
`StreamDescriptor`	`boolean getCompact()`	Gets the log compaction on a stream. Returns true if the stream has log compaction on the stream.

last-updated: Apr 11, 2022

Table (Continued)

Interface	Method	Description
`StreamDescriptor`	`void setMinCompactionLagMS(long ts)`	Sets the time in (milliseconds) that a message should remain uncompacted in the topic-partition. Applies only if log compaction is enabled on the stream.
`StreamDescriptor`	`long getMinCompactionLagMS()`	Returns the minimum time (in milliseconds) a message will remain uncompacted in the topic-partition. Applies only if log compaction is enabled on the stream.
`StreamDescriptor`	`void setDeleteRetentionMS(long ts)`	Sets the time (in milliseconds) for which deleted records are retained. Applies only if log compaction is enabled on the stream.
`StreamDescriptor`	`long getDeleteRetentionMS()`	Returns the time (in milliseconds) for which deleted records are retained. Applies only if log compaction is enabled on the stream.
`Producer`	`ProducerConfig` class	The idempotence producer option is set by setting the `enable.idempotence` value of **true** passed through the ProducerConfig class.

MapR Event Store For Apache Kafka Java APIs (as of 6.0.1)

The following table lists the new Interfaces and APIs for MapR Data Platform 6.0.1. They are the delta between MapR Data Platform 6.0.1 and 6.0.0, meaning, they are applicable to MapR Data Platform6.0.1 but not MapR Data Platform 6.0.0.

Table

Interface and Methods	Description
Admin.close	Long duration for TimeUnit.
Admin.createTopic	TopicDescriptor array for topic attributes.
Admin.editTopic	TopicDescriptor array for topic attributes.
Admin.getTopicDescriptor	Method for retrieving topic attributes.
Admin.listTopic	Method for listing all the topics in a stream.
Admin.streamExists	Method for determining whether a stream exists.
StreamDescriptor.getDefaultTimestampType	Method for retrieving the timestamp type.
StreamDescriptor.setDefaultTimestampType	Method for setting the timestamp type.
TopicDescriptor	New MapR Data Platform interface.
TopicDescriptor.getPartitions	Method associated with the new interface.
TopicDescriptor.setPartitions	Method associated with the new interface.
TopicDescriptor.getTimestampType	Method associated with the new interface.
TopicDescriptor.setTimestampType	Method associated with the new interface.
Enum TimestampType	New Enum class and associated methods.

last-updated: Apr 11, 2022

Backward Compatibility

As of MapR Data Platform 6.0.1, Apache Kafka 1.0 is supported. The following `pause`, `resume`, `seekToBeginning`, and `seekToEnd` APIs support the Collection Interface. The deprecated APIs will continue to run unchanged, however, they may be removed in a future release.

Table

Replacement Collection APIs	Deprecated APIs
void pause(Collection<TopicPartition> partitions);	void pause(TopicPartition... partitions);
void resume(Collection<TopicPartition>partitions);	void resume(TopicPartition... partitions);
void seekToBeginning(Collection<TopicPartition>);	void seekToBeginning(TopicPartition... partitions);
void seekToEnd(Collection<TopicPartition>);	void seekToEnd(TopicPartition... partitions);

The following `subscribe` and `assign` APIs support the Collection Interface (which is more generalized) as well as the List Interface. Support for the List Interface has been retained for backward binary compatibility.

Table

Replacement Collection APIs	Retained APIs
void subscribe(Collection<String> topics);	void subscribe(java.util.List<java.lang.String> topics);
void subscribe(Collection<String> topics, ConsumerRebalanceListener);	void subscribe(java.util.List<java.lang.String> topics, ConsumerRebalanceListener listener);
void assign(Collection<TopicPartition> partitions);	void assign(java.util.List<TopicPartition> partitions);

Stream and Topic Operations Summary

Provides a summary of stream topic operations and the interface, class, or method used for the operation.

The following stream and topic operations is not an inclusive list, but a sampling. For detailed information, see the following libraries:

- MapR Event Store For Apache Kafka Java API Library (6.1)
- Apache Kafka 1.1 APIs used with MapR Event Store For Apache Kafka (6.1)

Table

Operation	Interface/Method Used
Creating streams	`StreamDescriptor` is used to set the attributes for streams that you plan to create. `Admin.createStream(String streamPathAndName, StreamDescriptor desc)` - create the stream.
Editing stream attributes	`StreamDescriptor` is used to edit the stream's attribute values. `Admin.editStream(String streamPathAndName, StreamDescriptor desc)` - set or modify the stream's attribute values.
Retrieving the default timestamp type on a stream	`StreamDescriptor.getDefaultTimestampType()` - retrieves the default timestamp type on the stream. **Note:** This method is new as of 6.0.1
Sets the default timestamp type on a stream	`StreamDescriptor.setDefaultTimestampType(TimestampType logAppendTime)` - sets the default timestamp type on the stream. **Note:** TimestampType Enum is new as of 6.0.1

last-updated: Apr 11, 2022

Table (Continued)

Operation	Interface/Method Used
Deleting streams	`Admin.deleteStream(String streamPathAndName)`
Determining stream existence	`Admin.streamExists(String streamPathAndName)` - determines whether a stream exists or not. Returns: true | false  **Note:** This method is new as of 6.0.1
Creating topics	`Admin.createTopic(String streamPathAndName, String topicName, TopicDescriptor desc)` is used when creating a topic with the defaults for partitions and timestamp type.  **Note:** TopicDescriptor is new as of 6.0.1 `Admin.createTopic(String streamPathAndName, String topicName)` is used when accepting the default number of partitions. `Admin.createTopic(String streamPathAndName, String topicName, int npartitions)` - creates a topic with a specific number of partitions.  **Note:** If you do not specify the number of partitions for a stream topic, the default number of partitions is inherited from the stream.
Editing topics	`Admin.editTopic(String streamPathAndName, String topicName, TopicDescriptor desc)` - sets the partitions and timestamp type attributes of a topic.  **Note:** TopicDescriptor is new as of 6.0.1 `Admin.editTopic(String streamPathAndName, String topicName, int npartitions)`
Retrieving topic attributes	`Admin.getTopicDescriptor(String streamPathAndName, String topicName)` - retrieves topic attributes.  **Note:** This method is new as of 6.0.1
Deleting topics	`Admin.deleteTopic(String streamPathAndName, String topicName)` - deletes topics.
Listing topics	`Admin.listTopics(String streamPathAndName)` - lists all topics in a stream.  **Note:** This method is new as of 6.0.1
Counting topics	`Admin.countTopics(String streamPathAndName)` - counts the number of topics in a stream.
Gets/Sets topic timestamp type	`TopicDescriptor` is used to retrieve the timestamp type attribute value of a topic.  **Note:** TopicDescriptor is new as of 6.0.1 `TopicDescriptor.getTimestampType()` - retrieves the default timestamp type of a topic. `TopicDescriptor.setTimestampType(TimestampType timestampType)` - sets the default timestamp type of a topic.

last-updated: Apr 11, 2022

Table (Continued)

Operation	Interface/Method Used
Gets/Sets topic partitions	`TopicDescriptor` is used to retrieve the partition attribute value of a topic.  **Note:** TopicDescriptor is new as of 6.0.1 `TopicDescriptor.getPartitions()` - retrieves the partitions of a topic. `TopicDescriptor.setParitions(int numPartitions)` - sets the partitions of a topic.
Enabling and tuning log compaction	`TopicDescriptor` is used to enable and tune log compaction at the stream-level. • `setCompact(boolean compact)` - sets log compaction. • `getCompact()` - returns **true** if log compaction is set on the stream. • `setMinCompactionLagMS(long ts)` - set the lag time (in milliseconds) that a message should remain uncompacted. • `getMinCompactionLagMS()` - retrieves the value of the lag time. • `setDeleteRetentionMS(long ts)` - sets the time (in milliseconds) for which deleted records are retained. • `getDeleteRetentionMS()` - returns the time value of which deleted records are retained. In addition, the `Admin` interface is used with the following method to set compaction at the topic-level: • `compactTopic(java.lang.String streamPath, java.lang.String topicName)`
Enabling an Idempotent Producer	The `Producer` interface along with the `ProducerConfig` class is used to enable idempotence (exact-once message delivery semantics) publishing.

Managing Streams with Java

Provides Java code snippets for performing CRUD operations on MapR Event Store For Apache Kafka streams.

Creating Streams

`StreamDescriptor` is used to set attributes for streams that you want to create.

```
public StreamDescriptor createStreamDescriptor(int numPartitions, String
adminUsers, String producerUsers, String consumerUsers, String copyUsers,
String topicUsers) {
        StreamDescriptor desc = Streams.newStreamDescriptor();
        desc.setDefaultPartitions(numPartitions);
        desc.setCompressionAlgo("zlib");
        desc.setAutoCreateTopics(false);
        desc.setAdminPerms(adminUsers);
        desc.setConsumePerms(consumerUsers);
        desc.setCopyPerms(copyUsers);
        desc.setTopicPerms(topicUsers);

        return desc;
}
```


last-updated: Apr 11, 2022

`Admin` is used with the `createStream` method to create the stream with the pre-established attribute values.

```
public void createStreamUtilFunction(String streamPathAndName,
StreamDescriptor desc) throws IllegalArgumentException, IOException{
        Configuration conf = new Configuration();
        Admin streamAdmin = Streams.newAdmin(conf);
        streamAdmin.createStream(streamPathAndName, desc);
        streamAdmin.close();
}
```

Editing Stream Attributes

`StreamDescriptor` is used to retrieve the stream's attribute values. `Admin` with the `editStream` method is used to set or modify the stream's attribute values.

```
Admin.editStream(String streamPathAndName, StreamDescriptor desc)
```

```
Configuration conf = new Configuration();
Admin streamAdmin = Streams.newAdmin(conf);
        StreamDescriptor desc =
streamAdmin.getStreamDescriptor(streamPathAndName);
```

```
public void editStreamUtilFunction(String streamPathAndName,
StreamDescriptor desc) throws IllegalArgumentException, IOException{
        Configuration conf = new Configuration();
        Admin streamAdmin = Streams.newAdmin(conf);
        streamAdmin.editStream(streamPathAndName, desc);
        streamAdmin.close();
}
```

Deleting Streams

```
public void deleteStreamUtilFunction(String streamPathAndName) throws
IllegalArgumentException, IOException{
          Configuration conf = new Configuration();
          Admin streamAdmin = Streams.newAdmin(conf);
          streamAdmin.deleteStream(streamPathAndName);
          streamAdmin.close();
}
```

Managing Topics with Java

Provides Java code snippets for performing CRUD operations on MapR Event Store For Apache Kafka stream topics.

Creating Topics

The `createTopic` API is used to create a topic with the default number of partitions.

```
Admin.createTopic(String streamPathAndName, String topicName)
```

Note: If you do not specify the number of partitions for a stream topic, the default number of partitions is inherited from the stream.

```
public void createTopicUtilFunction(String streamPathAndName, String
topicName) throws IOException{
        Configuration conf = new Configuration();
```

```
        Admin streamAdmin = Streams.newAdmin(conf);
        streamAdmin.createTopic(streamPathAndName, topicName);
        streamAdmin.close();
}
```

The `createTopic` API is used to create a topic with a specific number of partitions.

```
Admin.createTopic(String streamPathAndName, String topicName, int
npartitions)
```

```
public void createTopicWithPartitionsUtilFunction(String streamPathAndName,
String topicName, int npartitions) throws IOException{
        Configuration conf = new Configuration();
        Admin streamAdmin = Streams.newAdmin(conf);
        streamAdmin.createTopic(streamPathAndName, topicName, npartitions);
        streamAdmin.close();
}
```

Editing Topics

The `editTopic` API is used to change timestamp type and the number of partitions for a topic.

```
Admin.editTopic(String streamPathAndName, String topicName, int npartitions)
```

```
public void editTopicUtilFunction(String streamPathAndName, String
topicName, int npartitions) throws IOException{
        Configuration conf = new Configuration();
        Admin streamAdmin = Streams.newAdmin(conf);
        streamAdmin.editTopic(streamPathAndName, topicName, npartitions);
        streamAdmin.close();
}
```

Retrieving Topic Attributes

The `getTopicDescriptor` API is used to get or set the topic's attribute values. `TopicDescriptor` is passed into methods to set and retrieve topic partitions and timestamp type. The `Enum TimestampType` values are CREATE_TIME and LOG_APPEND_TIME.

Note: `TopicDescriptor` is available as of MapR 6.0.1.

Deleting Topics

The `deleteTopic` API is used to delete a topic from a stream.

```
Admin.deleteTopic(String streamPathAndName, String topicName)
```

```
public void deleteTopicUtilFunction(String streamPathAndName, String
topicName) throws IOException{
        Configuration conf = new Configuration();
        Admin streamAdmin = Streams.newAdmin(conf);
        streamAdmin.deleteTopic(streamPathAndName, topicName);
        streamAdmin.close();
}
```

Counting Topics

The `countTopics` API is used to count the number of topics in a stream. See the mapr streamanalyzer on page 5348 utility for a sample application that counts and queries topic messages.

```
Admin.countTopics(String streamPathAndName)
```

```
public int countTopicsUtilFunction(String streamPathAndName){
        Configuration conf = new Configuration();
        Admin streamAdmin = Streams.newAdmin(conf);
        int count = streamAdmin.countTopics(streamPathAndName);
        streamAdmin.close();

        return count;
}
```

Using Timestamps on Streams and Topics

Provides a code example for using timestamps on MapR Event Store For Apache Kafka streams and topics.

Passing Timestamp Value

The timestamp value can be passed as part of the `ProducerRecord`, for example:

```
ProducerRecord<String, String> producerRecord =
        new ProducerRecord<String, String>(topicName, partition, timestamp,
key, value);
```

Note: The timestamp value is retained if the timestamp type is `createtime`. If the timestamp type is `logappendtime`, then the timestamp value is ignored and instead the server timestamp is used.

Retrieving Timestamp Type

This example sets and retrieves the timestamp type. The following code example performs the following:

- Creates a stream with a default timestamp type of LogAppendTime.
- Creates a topic with a specific timestamp type of CreateTime.
- Retrieves the topics's timestamp type.

```
// Create stream with default timestamp type as "LogAppendTime"
// Create a topic with timestamp type as "CreateTime"
        Configuration conf = new Configuration();
        Admin streamAdmin = Streams.newAdmin(conf);

// Create a stream
        StreamDescriptor sDesc = Streams.newStreamDescriptor();
        sDesc.setDefaultTimestampType(TimestampType.LOG_APPEND_TIME);
        streamAdmin.createStream(streamName, sDesc);

// Create a topic
        TopicDescriptor tDesc = Streams.newTopicDescriptor();
        tDesc.setTimestampType(TimestampType.CREATE_TIME);
        streamAdmin.createTopic(streamName, topicName, tDesc);

// Get topic timestamp type
        TopicDescriptor rDesc = streamAdmin.getTopicDescriptor(streamName,
```

```
topicName);
        System.out.println(rDesc.getTimestampType().name);
```

Enabling Log Compaction
Provides a code example for using timestamps on MapR Event Store For Apache Kafka streams and topics.

Log compaction is enabled through the MapR Event Store For Apache Kafka `StreamDescriptor` interface with the `setCompact` method where the compact value is set to **true**. Additionally, use the `setDeleteRetentionMS` and `setMinCompactionLagMS` methods to set the time delay before compacting records and the time that deleted records are retained.

Configuration values include:

- compact - used to set log compaction at the stream-level.
- min.compaction.lag.ms - used to set a **minimum** time delay (milliseconds) before starting to compact records after they are written. Records won't get compacted until after this period. The setting gives consumers time to retrieve every record.
- delete.retention.ms - used to set the **minimum** time (milliseconds) that deleted records are retained.

Note: You can set not set log compaction when creating the stream; only when editing the stream configuration. The configuration parameters, `min.compaction.lag.ms` and `delete.retention.ms` can be set when both creating and editing streams.

Enabling Log Compaction

The following code example performs the following:

- Enables log compaction at the stream-level.
- Sets the minumum time delay before log compaction starts
- Set the minimum time that deleted record are retained.

```
// Creates a stream
// Sets log compaction on the stream
// Sets the minimum time for a message to stay uncompacted
// Sets the time that deleted records are retained.

(Admin streamAdmin = Streams.newAdmin(conf))
    StreamDescriptor streamDescriptor = Streams.newStreamDescriptor();
       streamDescriptor.setCompact(true);
       streamDescriptor.setDeleteRetentionMS(deleteRetentionMs);
    streamDescriptor.setMinCompactionLagMS(minCompactionLagMs);
       streamAdmin.editStream(streamName, streamDesc);
}
```

For More Information

See the following topics for more information:

- Log Compaction on page 641
- maprcli stream create on page 1758 and stream edit on page 1765
- Preparing Clusters for Log Compaction on page 1141

Enabling an Idempotent Producer
Describes how to enable an idempotent producer. Idempotence refers to exactly-once message delivery semantics.

To enable idempotence, the `enable.idempotence` configuration must be set to **true**. When set, the retries configuration defaults to Integer.MAX_VALUE and the Acks configuration defaults to all.

The idempotence producer option is set by setting the `enable.idempotence` value of true passed through the `ProducerConfig` class.

Constant Field Values
`org.apache.kafka.clients.producer.ProducerConfig`

Modifier and Type	Constant Field	Value
`public static final java.lang.String`	`ENABLE_IDEMPOTENCE_CONFIG`	enable.idempotence

Example Code Snippet:

```
props.put(ProducerConfig.ENABLE_IDEMPOTENCE_CONFIG, true);
```

Note: The default is false, which retains at-least-once message delivery semantics.

Tip: There are no API changes for the Idempotent Producer functionality, so existing applications do not need to be modified except to enable the producer configuration property.

Example: Subscribing and Querying with Timestamps
This sample Java consumer application uses the `subscribe` API to subscribe to the input topics and queries offsets upon partition-assignment.

In the query, the `offsetsForTimes` API returns the earliest offset in a topic-partition with a timestamp greater than or equal to the input timestamp. The consumer then seeks to that offset if it is greater than the consumer's current position. Following this, the consumer polls for messages. If there are messages following that offset with timestamps earlier than the input timestamp, then those messages are skipped by the consumer.

```
import java.util.Arrays;
import java.util.Collection;
import java.util.HashMap;
import java.util.Map;
import java.util.Properties;

import org.apache.kafka.clients.consumer.Consumer;
import org.apache.kafka.clients.consumer.ConsumerConfig;
import org.apache.kafka.clients.consumer.ConsumerRebalanceListener;
import org.apache.kafka.clients.consumer.ConsumerRecord;
import org.apache.kafka.clients.consumer.ConsumerRecords;
import org.apache.kafka.clients.consumer.KafkaConsumer;
import org.apache.kafka.clients.consumer.OffsetAndTimestamp;
import org.apache.kafka.common.TopicPartition;

public class TimeBasedConsumer {
  private static long kPollTimeout = 100;
  private static int kNumRecordsToProcess = 10;

  public static void main(String[] args) {
    if (args.length < 2) {
      String usage = "Usage: Program <topicName> <startTimestamp>";
      System.err.println(usage);
```

```
      throw new IllegalArgumentException(usage);
    }
    String topic = args[0];
    Long startTimestamp = Long.parseLong(args[1]);
    Properties properties = new Properties();
    properties.put(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG,
"org.apache.kafka.common.serialization.StringDeserializer");
    properties.put(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG,
"org.apache.kafka.common.serialization.StringDeserializer");
    properties.put(ConsumerConfig.GROUP_ID_CONFIG, "testgroup");
    properties.put(ConsumerConfig.AUTO_OFFSET_RESET_CONFIG, "earliest");
    KafkaConsumer<String, String> consumer = new KafkaConsumer<String,
String>(properties);
    SeekToTimeOnRebalance seekToTimeOnRebalance = new
SeekToTimeOnRebalance(consumer, startTimestamp);

    // subscribe to the input topic and listen for assignments.
    consumer.subscribe(Arrays.asList(topic), seekToTimeOnRebalance);

    int numRecords = 0;
    // poll and process the records.
    while (numRecords < kNumRecordsToProcess) {
      ConsumerRecords<String, String> records =
consumer.poll(kPollTimeout );
      for (ConsumerRecord<String, String> record : records) {
        // The offsetsForTimes API returns the earliest offset in a
topic-partition with a timestamp
        // greater than or equal to the input timestamp. There could be
messages following that offset
        // with timestamps lesser than the input timestamp. Let's skip such
messages.
        if (record.timestamp() < startTimestamp) {
          System.out.println("Skipping out of order record with key " +
record.key() +
                             " timestamp " + record.timestamp());
          continue;
        }
        numRecords++;
        System.out.println("record key " + record.key() +
                           "record timestamp " + record.timestamp() +
                           "record offset " + record.offset());
      }
    }
    consumer.close();
  }

  public static class SeekToTimeOnRebalance implements
ConsumerRebalanceListener {
    private Consumer<?, ?> consumer;
    private final Long startTimestamp;

    public SeekToTimeOnRebalance(Consumer<?, ?> consumer, Long
startTimestamp) {
      this.consumer = consumer;
      this.startTimestamp = startTimestamp;
    }

    @Override
    public void onPartitionsAssigned(Collection<TopicPartition> partitions)
{
      Map<TopicPartition, Long> timestampsToSearch = new HashMap<>();
      for (TopicPartition partition : partitions) {
        timestampsToSearch.put(partition,  startTimestamp);
      }
```

```
      // for each assigned partition, find the earliest offset in that
partition with a timestamp
      // greater than or equal to the input timestamp
      Map<TopicPartition, OffsetAndTimestamp> outOffsets =
consumer.offsetsForTimes(timestampsToSearch);
      for (TopicPartition partition : partitions) {
        Long seekOffset = outOffsets.get(partition).offset();
        Long currentPosition = consumer.position(partition);
        // seek to the offset returned by the offsetsForTimes API
        // if it is beyond the current position
        if (seekOffset.compareTo(currentPosition) > 0) {
          consumer.seek(partition, seekOffset);
        }
      }
    }

    @Override
    public void onPartitionsRevoked(Collection<TopicPartition> partitions) {
    }

  }

}
```

Querying Topic Messages
Describes how MapR Event Store For Apache Kafka topic messages can be queried.

Time-based Querying

The **consumer.offsetsForTimes**API is used to get offsets in a topic-partition. This API takes in a `Map` of `TopicPartition` and timestamp. The offset is returned in an `OffsetAndTimestamp` object when `offsetsForTime` is called.

The following shows how the Map is constructed:

```
Long timestamp = 1522195205L;
TopicPartition topicPartition = new TopicPartition(topic,partition);

HashMap<TopicPartition, Long> offsetsForTimesMap = new
HashMap<TopicPartition, Long>();
offsetsForTimesMap.put(topicPartition, timestamp);

// Invocation to offsetsForTimes
Map<TopicPartition, OffsetAndTimestamp> offsetForTimesResultMap =
consumer.offsetsForTimes(offsetsForTimesMap);
```

Direct Querying

The `Streams` class is used to directly query topic messages. See the mapr streamanalyzer on page 5348 utility for a sample application that counts and queries topic messages.

- The `getMessageStore()` APIs are used to get the `DocumentStore` object which represents the underlying topic messages for a specified stream.
- The `DocumentStore.find()` APIs are used to query the messages that are in the `DocumentStore` object. While running `find()` on the returned DocumentStore object, message fields can be projected based on the specified field name.

Note: DocumentStore is a part of the open-source OJAI API.

The logical schema of each message is the same, where analytics applications can run queries on these fields. See Logical Schema of Messages on page 635 for more information.

```
{
        "_id":<STRING>,
        "topic":<STRING>,
        "partition":<SHORT>,
        "offset":<LONG>,
        "timestamp":<LONG>,
        "producer":<VARCHAR>,
        "key":<BINARY>,
        "value":<VARBINARY>
}
```

Apache Kafka Java APIs

MapR Event Store For Apache Kafka supports these Apache Kafka Java APIs.

Javadoc

As of MapR Data Platform 6.1.0, Apache Kafka 1.1 is supported.

See the following APIs for detailed information:

- MapR Event Store For Apache Kafka Java API Library
- Apache Kafka 1.1 APIs used with MapR Event Store For Apache Kafka

Admin APIs

The following Admin APIs, `org.apache.kafka.clients.admin` package, are applicable to MapR Data Platform support of Apache Kafka. These APIs are supported as of MapR Data Platform 6.1.0.

- AdminClient

Note: For MapR 6.1, the following Apache Kafka API is not supported:

- deleteRecords()

Note: The AdminClient API options (CreateTopicsOptions, DeleteTopicsOptions, DescribeTopicsOptions, ListTopicsOptions, CreatePartitionsOptions, and DescribeTopicsOptions), are ignored. All of the methods assume that a topic belongs to the default stream unless a stream path is specified in the topic name.

If a default stream name is not specified and the topic path does not contain a stream name, the an exception is reported via the Result object. For example:

- If the topic name is specified as `topic1`, then the API assumes the full topic path as `/defaultStream:topic1`.
- If the topic name is specified as `/defaultStream:topic1`, then that will be the full topic path.

Note: The `AdminClient` default stream configuration parameter is `streams.admin.default.stream`. See Configuration Parameters on page 2772 for more information.

Table

Modifier and Type	Method
static AdminClient	create(java.util.Properties props)

last-updated: Apr 11, 2022

Table (Continued)

Modifier and Type	Method
static AdminClient	create(java.util.Map<java.lang.String,java.lang.Object> conf)
void	close()
CreateTopicsResult	createTopics(java.util.Collection<NewTopic> newTopics)
DeleteTopicsResult	deleteTopics(java.util.Collection<NewTopic> newTopics)
DescribeTopicsResult	describeTopics(java.util.Collection<java.lang.String> topicNames)
ListTopicsResult	listTopics()
ListTopicsResult	listTopics((java.lang.String streamPath))
CreatePartitionsResult	createPartitions(java.util.Map<java.lang.String,NewPartitions> newPartitions)
DescribeClusterResult	describeCluster()

Note: For a complete list of supported APIs, see Apache Kafka 1.1 APIs used with MapR Event Store For Apache Kafka

Consumer APIs

The following Consumer APIs, `org.apache.kafka.clients.consumer` package, are applicable to MapR Data Platform support of Apache Kafka 1.1. These APIs are supported as of MapR Data Platform 6.1.0..

- Interface ConsumerInterceptor<K,V>
- Class ConsumerRecord
- Class KafkaConsumer

Table

Modifier and Type	Method
long	timestamp()
long	timestamptype()

Table

Modifier and Type	Method
void	pause(Collection<TopicPartition> partitions)
void	resume(Collection<TopicPartition>partitions)
void	seekToBeginning(Collection<TopicPartition>)
void	seekToEnd(Collection<TopicPartition>)
void	subscribe(Collection<String> topics);
void	subscribe(Collection<String> topics, ConsumerRebalanceListener)
void	assign(Collection<TopicPartition> partitions)

Table (Continued)

Modifier and Type	Method
java.util.Map<TopicPartition,OffsetAndTimestamp>	offsetsForTimes(java.util.Map<TopicPartition,java.lang.Long> timestampsToSearch)
java.util.Map<TopicPartition,java.lang.Long>	beginningOffsets(Collection<TopicPartition>)
java.util.Map<TopicPartition,java.lang.Long>	endOffsets(Collection<TopicPartition> partitions)
ConsumerRecords<K,V>	poll(long timeout)
void	commitSync()
void	commitAsync()

The following consumer interface and classes are applicable to MapR Data Platform support of Apache Kafka.

- org.apache.kafka.clients.consumer.ConsumerConfig
- org.apache.kafka.clients.consumer.ConsumerRebalanceCallback (interface)
- org.apache.kafka.clients.consumer.ConsumerRecord<K,V>
- org.apache.kafka.clients.consumer.ConsumerRecords<K,V>
- org.apache.kafka.clients.consumer.KafkaConsumer<K, V> implements Consumer<K, V>

Note: For a complete list of supported APIs, see Apache Kafka 1.1 APIs used with MapR Event Store For Apache Kafka

Producer APIs

The following producer interface and classes, `org.apache.kafka.clients.producer` package, are applicable to MapR Data Platform support of Apache Kafka 1.1. These APIs are supported as of MapR Data Platform 6.1.0..

- Interface ProducerInterceptor<K,V>
- Class RecordMetadata
- Class ProducerRecord

Table

Modifier and Type	Method
java.util.concurrent.Future<RecordMetadata>	send(ProducerRecord<K,V> record)
void	flush()
void	close()

The following producer interface and classes are applicable to MapR Data Platform support of Apache Kafka.

- org.apache.kafka.clients.producer.Callback (Interface)
- org.apache.kafka.clients.producer.KafkaProducer<K,V>

last-updated: Apr 11, 2022

- org.apache.kafka.clients.producer.ProducerConfig
- org.apache.kafka.clients.producer.ProducerRecord<K,V>
- org.apache.kafka.clients.producer.RecordMetadata

Note: For a complete list of supported APIs, see Apache Kafka 1.1 APIs used with MapR Event Store For Apache Kafka

Common APIs

The following common APIs, `org.apache.kafka.clients.common` packages, are applicable toMapR Data Platform support of Apache Kafka 1.1. These APIs are supported as of MapR Data Platform 6.1.0..

- Interface Header

Table

Modifier and Type	Method
java.lang.String	key()
byte[]	value()

The following APIs are applicable to MapR Data Platform support for Apache Kafka.

- org.apache.kafka.common.PartitionInfo

  Supported methods in PartitionInfo:

  - int partition()
  - java.lang.String topic()
  - java.lang.String toString()
- org.apache.kafka.common.serialization.Serializer<T> (Interface)
- org.apache.kafka.common.serialization.Deserializer<T> (interface)
- org.apache.kafka.common.TopicPartition

Note: For a complete list of supported APIs, see Apache Kafka 1.1 APIs used with MapR Event Store For Apache Kafka

Configuration Parameters
This topic describes configuration parameters that are either specific to MapR Event Store For Apache Kafka or supported from Apache Kafka.

AdminClient

Table

Parameter	Description
`streams.admin.default.stream`	This parameter, when set during creation of the AdminClient instance, ensures that the specified stream is using the the AdminClient instance for all administrative operations. Syntax: `/mapr/<cluster name>/<volume name>/<stream name>`
`streams.rpc.timeout.ms`	Specifies the length of time in milliseconds to wait for a response from the MapR Event Store For Apache Kafka server if soft mount is configured (fs.mapr.hardmount is set to false). Default: 120000 Minimum: 30000 **Note:** Applicable as of MapR 6.0.1, is used instead of `fs.mapr.rpc.timeout` For producer and consumer applications, make sure the `streams.rpc.timeout.ms` configuration value for both producers and consumers is set to greater than 50000 to avoid Message Fetch RPC overload.

Consumer

Table

Parameter	Description
`streams.consumer.buffer.memory`	Specifies how much memory to use for caching pre-fetched messages. Messages that are in subscribed topics and partitions are pre-fetched and cached to improve performance. Default 64MB
`streams.consumer.default.stream`	Specifies the path and name of the stream that the consumer subscribes to if, when subscribing to a topic, the consumer does not specify a stream. This default value is also used for the `KafkaConsumer.listTopics() method`.
`streams.rpc.timeout.ms`	Specifies the length of time in milliseconds to wait for a response from the MapR Event Store For Apache Kafka server if a soft mount is configured (fs.mapr.hardmount is set to false). Default: 305000 Minimum: 300000 For producer and consumer applications, make sure the `streams.rpc.timeout.ms` configuration value for both producers and consumers is set to greater than 50000 to avoid Message Fetch RPC overload.

Table

Parameter	Description
`auto.commit.interval.ms`	The frequency in milliseconds that the offsets are committed. Default: 1000ms

Table (Continued)

Parameter	Description
`auto.offset.reset`	Specifies what MapR Event Store For Apache Kafka should do when there is no initial offset, such as when a consumer starts reading from a partition. Default: latest **`earliest`** — Reset the offset to the offset of the earliest message in the partition. **`latest`** — Reset the offset to the offset of the latest message in the partition. **none** — Throws a `NoOffsetForPartitionException` exception when the consumer next polls for messages in its subscription and no offset exists. The consumer must unsubscribe from the partition before polling functions correctly. Any other value throws an error to the consumer.
`enable.auto.commit`	If true, periodically commits the highest offsets of the messages fetched by the consumer in all of the partitions for the topics that the consumer is subscribed to. Default: true
`fetch.min.bytes`	The minimum amount of data the server should return for a fetch request. If insufficient data is available, the server will wait for this minimum amount of data to accumulate before answering the request. This minimum applies to the totality of what a consumer has subscribed to. Works in conjunction with the timeout interval that is specified in the poll function. If the minimum number of bytes is not reached by the time that the interval expires, the poll returns with nothing. For example, suppose the value is set to 6 bytes and the timeout on a poll is set to 100ms. If there are 5 bytes available and no further bytes come in before the 100ms expire, the poll returns with nothing. Default: 1 byte
`fetch.max.bytes`	The maximum amount of data the server should return for a fetch request. If the first record batch in the first non-empty partition of the fetch is larger than this configuration, the record batch is still returned to ensure that the consumer can make progress. **Note:** This parameter is new as of MapR 6.0.1.
`fetch.max.wait.ms`	The maximum amount of time the MapR Event Store For Apache Kafka server will block before answering the fetch request if there isn't sufficient data to satisfy the requirement given by `fetch.min.bytes`.

last-updated: Apr 11, 2022

Table (Continued)

Parameter	Description
`group.id`	A string 2457 up to bytes long that uniquely identifies the group of consumer processes to which this consumer belongs. By setting the same group ID, multiple consumer processes indicate that they are all part of the same consumer group. Putting consumers into groups provides benefits that are described in Consumer Groups. It is possible for a single consumer to be in a group.
`key.deserializer`	The class that implements the `Deserializer` interface for deserializing keys.
`max.poll.records`	Places an upper bound on the number of records returned from each call. **Note:** This parameter is new as of MapR 6.0.1.
`max.partition.fetch.bytes`	The number of bytes of message data to attempt to fetch for each partition in each poll request. These bytes will be read into memory for each partition, so this parameter helps control the memory that the consumer uses. Default: 64KB The size of the poll request must be at least as large as the maximum message size that the server allows or else it is possible for producers to send messages that are larger than the consumer can fetch. If the first record batch in the first non-empty partition of the fetch is larger than this configuration, the record batch is still returned to ensure that the consumer can make progress. **Note:** This is a behavior change as of MapR 6.0.1.
`value.deserializer`	The name of the appropriate deserialization class in the `org.apache.kafka.common.serialization` package or a class that implements the `Deserializer` interface for deserializing values.

Producer

Table

Parameter	Description
`streams.buffer.max.time.ms`	Messages are buffered in the producer for at most the specified time. A thread will flush all the messages that have been buffered for more than the time specified. Default: 3 * 1000 msec create default stream
`streams.parallel.flushers.per.partition`	If enabled, producer may have multiple parallel send requests to the server for each topic partition. If this setting is set to true, it is possible for messages to be sent out of order. Default: true create default stream
`streams.partitioner.class`	The class that implements the `StreamsPartitioner` interface. This interface lets you write custom algorithms for determining which topic and partition to use for messages that match specific criteria. Use this configuration parameter only for producers that are written in Java. create default stream

last-updated: Apr 11, 2022

Table (Continued)

Parameter	Description
`streams.producer.default.stream`	Specifies the stream that the producer will use by default if the producer does not provide the name of a stream when specifying a topic to write to. Syntax: `/mapr/<cluster name>/<volume name>/<stream name>` create default stream
`fs.mapr.hardmount`	Specifies whether to use a hard mount or a soft mount for connections to the MapR Streams server. The default is to use a hard mount and the value is `true`. If a value for this parameter is set in the `core-site.xml` file, the value in that file is ignored. create default stream
`fs.mapr.rpc.timeout`	Specifies the length of time in seconds to wait for a response from the MapR Event Store For Apache Kafka server if the configuration parameter `fs.mapr.hardmount` is set to false. Default: 300. Minimum value: 30. **Note:** Applicable to MapR 6.0.0 and earlier. As of MapR 6.0.1, use `streams.mapr.timeout.ms`. If a soft mount is used, the time expires while a producer waits for a response from the MapR Event Store For Apache Kafka server, and the producer used the `KafkaProducer.send(ProducerRecord<K,V> record, Callback callback)` method, the callback is invoked with the error `EAGAIN`, which means "Resource temporarily unavailable." create default stream
`streams.rpc.timeout.ms`	Specifies the length of time in milliseconds to wait for a response from the MapR Event Store For Apache Kafka server if soft mount is configured (fs.mapr.hardmount is set to false). Default: 30000 Minimum: 30000 For producer and consumer applications, make sure the `streams.rpc.timeout.ms` configuration value for both producers and consumers is set to greater than 50000 to avoid Message Fetch RPC overload.

Table

Parameter	Description
`buffer.memory`	The total bytes of memory the producer can use to buffer records waiting to be sent to the server. If records are generated faster than they can be delivered to the server the producer will block. Default: 33554432
`client.id`	Producers can tag records with a client ID that identifies the producer. Consumers can then be aware of which producer sent a message or set of messages. Apache Drill or other analytic tools querying messages can include this ID in the filters for their queries. Default: No client ID.

Table (Continued)

Parameter	Description
`key.serializer`	The name of the appropriate serialization class in the `org.apache.kafka.common.serialization` package or a class that implements the `Serializer` interface for serializing keys.
`metadata.max.age.ms`	The producer generally refreshes the topic metadata from the server when there is a failure. It will also poll for this data regularly. Default: 300 * 1000 msec
`value.serializer`	The class that implements the `Serializer` interface for serializing values.

Related Links

Configuring Properties for Message Size on page 3029

Compiling and Running MapR Event Store For Apache Kafka Java Apps

For producer and consumer applications that use the MapR Event Store For Apache Kafka Java API, use Maven to compile and determine the application's dependencies. Then, when you run the application, specify those dependencies in the application's classpath.

Compile and Determine Dependencies

See MapR Event Store For Apache Kafka Streams Sample Programs on GitHub for an example pom.xml file.

1. Add MapR's Maven repository to your `pom.xml` file, if it is not already added:

```
<repositories>
    <repository>
      <id>mapr-releases</id>
      <url>https://repository.mapr.com/nexus/content/repositories/
releases</url>
      <snapshots><enabled>true</enabled></snapshots>
      <releases><enabled>true</enabled></releases>
    </repository>
</repositories>
```

2. Add a dependency to the MapR Streams Java client (kakfa-clients) project:

```
<dependency>
      <groupId>org.apache.kafka</groupId>
      <artifactId>kafka-clients</artifactId>
      <version><version selected from the repository></version>
</dependency>
```

Note: The kafka-clients version mentioned above is an example. The actual version that your application requires is based on the current EEP and MapR version that you are running. The versions are listed in the following location: https://repository.mapr.com/nexus/content/groups/mapr-public/org/apache/kafka/kafka-clients/

last-updated: Apr 11, 2022

3. Add a dependency to the MapR Streams project:

```
<dependency>
      <groupId>com.mapr.streams</groupId>
      <artifactId>mapr-streams</artifactId>
      <version><version selected from the repository></version>
</dependency>
```

Note: The MapR Streams project version mentioned above is an example. The actual version that your application requires is based on the current EEP and MapR version that you are running. The versions are listed in the following location:https://repository.mapr.com/nexus/content/groups/mapr-public/com/mapr/streams/mapr-streams/

4. Use Maven to compile the application and resolve dependencies. For example, you can run `mvn clean package`.

Run the Application

When you develop a Java application, you can use a dependency management tool such as Maven to compile your application. However, it is recommended that do the following instead:

1. Compile the Java application without including dependencies
2. Specify the required classpath when you submit the application to the cluster

If you choose to bundle the JAR file, and there is a mismatch between the bundled JAR file and the version that your MapR cluster expects, this can result in failures. The failures differ depending on the version of MapR you are using. For more information, see Using the File System JAR to Connect to the Cluster on page 2367.

When the cluster is secure, the node must also have a mapr ticket configured for the user that runs the application.

You can use the following command to launch MapR Event Store For Apache Kafka applications:

```
java -cp <classpath>:. -Djava.library.path=/opt/mapr/lib <main class JAR>
<command line arguments>
```

References

- MapR Event Store For Apache Kafka Streams Sample Programs on GitHub.
- Getting Started with MapR Streams blog.
- Source on GitHub.

Migrating Apache Kafka Java Applications to MapR Event Store For Apache Kafka
There are only two steps that you need to follow to migrate applications written with the Apache Kafka Java API to MapR Event Store For Apache Kafka.

The following steps assume that migration is from either:

- Apache Kafka 2.1.1 to MapR Event Store For Apache Kafka 6.2 or higher
- Apache Kafka 1.1 to MapR Event Store For Apache Kafka 6.1 or higher
- Apache Kafka 1.0 to MapR Event Store For Apache Kafka 6.0.1 or higher
- Apache Kafka 0.9.0 to MapR Event Store For Apache Kafka 6.0.0 or earlier

Important: For information on backward compatibility, see MapR Event Store For Apache Kafka Java API Library on page 2756

1. Change the names of topics to include the path and name of the MapR Stream stream in which the topic is located.

   Here is the syntax to use:

   ```
   /<path and name of stream>:<name of topic>
   ```

   For example, you might have a stream in a MapR cluster that is named `stream_A`, and the stream might be in a volume named `IoT` and in a directory named `automobile_sensors`. You want to redirect a producer application to a topic in that stream. The syntax of the path to the topic might look like this:

   ```
   /mapr/IoT/automobile_sensors/stream_A:<name of topic>
   ```

2. If a producer application uses the Kafka interface Partitioner to compute which partitions to publish messages to, revise the application so that it uses the Kafka StreamsPartitioner interface instead.

Differences between MapR Event Store For Apache Kafka and Apache Kafka Configuration
Describes the MapR Event Store For Apache Kafka supportability of Apache Kafka configuration parameters for producers and consumers.

Kafka Producer

Name	Description	Supported for producers in MapR Event Store For Apache Kafka?
`bootstrap.servers`	A list of host/port pairs to use for establishing the initial connection to the Kafka cluster. The client will make use of all servers irrespective of which servers are specified here for bootstrapping—this list only impacts the initial hosts used to discover the full set of servers. This list should be in the form `host1:port1,host2:port2,...` Since these servers are just used for the initial connection to discover the full cluster membership (which may change dynamically), this list need not contain the full set of servers (you may want more than one, though, in case a server is down).	No. Cluster details are discovered from the file `mapr-clusters.conf`.
`key.serializer`	`Serializer` class for key that implements the `Serializer` interface.	Yes
`value.serializer`	`Serializer` class for value that implements the `Serializer` interface.	Yes

Name	Description	Supported for producers in MapR Event Store For Apache Kafka?
`acks`	The number of acknowledgments the producer requires the leader to have received before considering a request complete. This controls the durability of records that are sent. The following settings are common:	Ignored, all writes in MapR Event Store For Apache Kafka are synchronous, and number of replicas is determined at the volume level, with a default of 3.
	`acks=0` If set to zero then the producer will not wait for any acknowledgment from the server at all. The record will be immediately added to the socket buffer and considered sent. No guarantee can be made that the server has received the record in this case, and the retries configuration will not take effect (as the client won't generally know of any failures). The offset given back for each record will always be set to -1.	
	`acks=1` This will mean the leader will write the record to its local log but will respond without awaiting full acknowledgement from all followers. In this case should the leader fail immediately after acknowledging the record but before the followers have replicated it then the record will be lost.	
	`acks=all` This means the leader will wait for the full set of in-sync replicas to acknowledge the record. This guarantees that the record will not be lost as long as at least one in-sync replica remains alive. This is the strongest available guarantee.	
`buffer.memory`	The total bytes of memory the producer can use to buffer records waiting to be sent to the server. If records are sent faster than they can be delivered to the server the producer will either block or throw an exception based on the preference specified by `block.on.buffer.full`.	Yes
	This setting should correspond roughly to the total memory the producer will use, but is not a hard bound since not all memory the producer uses is used for buffering. Some additional memory will be used for compression (if compression is enabled) as well as for maintaining in-flight requests.	

Name	Description	Supported for producers in MapR Event Store For Apache Kafka?
`compression.type`	The compression type for all data generated by the producer. The default is none (i.e. no compression). Valid values are `none`, `gzip`, `snappy`, or `lz4`. Compression is of full batches of data, so the efficacy of batching will also impact the compression ratio (more batching means better compression).	Ignored. Compression is configured per stream.
`retries`	Setting a value greater than zero will cause the client to resend any record whose send fails with a potentially transient error. Note that this retry is no different than if the client resent the record upon receiving the error. Allowing retries will potentially change the ordering of records because if two records are sent to a single partition, and the first fails and is retried but the second succeeds, then the second record may appear first.	Ignored. MapR Event Store For Apache Kafka always does automatic retries on transient errors.
`ssl.key.password`	The password of the private key in the key store file. This is optional for client.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`ssl.keystore.location`	The location of the key store file. This is optional for client and can be used for two-way authentication for client.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`ssl.keystore.password`	The store password for the key store file.This is optional for client and only needed if `ssl.keystore.location` is configured.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`ssl.truststore.location`	The location of the trust store file.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`ssl.truststore.password`	The password for the trust store file.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.

Name	Description	Supported for producers in MapR Event Store For Apache Kafka?
`batch.size`	The producer will attempt to batch records together into fewer requests whenever multiple records are being sent to the same partition. This helps performance on both the client and the server. This configuration controls the default batch size in bytes.	Ignored. MapR Data Platform always batches records for optimal performance.
	No attempt will be made to batch records larger than this size.	
	Requests sent to brokers will contain multiple batches, one for each partition with data available to be sent.	
	A small batch size will make batching less common and may reduce throughput (a batch size of zero will disable batching entirely). A very large batch size may use memory a bit more wastefully as we will always allocate a buffer of the specified batch size in anticipation of additional records.	
`client.id`	An id string to pass to the server when making requests. The purpose of this is to be able to track the source of requests beyond just ip/port by allowing a logical application name to be included in server-side request logging.	Yes
`connections.max.idle.ms`	Close idle connections after the number of milliseconds specified by this config.	Ignored.

Name	Description	Supported for producers in MapR Event Store For Apache Kafka?
`linger.ms`	The producer groups together any records that arrive in between request transmissions into a single batched request. Normally this occurs only under load when records arrive faster than they can be sent out. However in some circumstances the client may want to reduce the number of requests even under moderate load. This setting accomplishes this by adding a small amount of artificial delay—that is, rather than immediately sending out a record the producer will wait for up to the given delay to allow other records to be sent so that the sends can be batched together. This can be thought of as analogous to Nagle's algorithm in TCP. This setting gives the upper bound on the delay for batching: once we get batch.size worth of records for a partition it will be sent immediately regardless of this setting, however if we have fewer than this many bytes accumulated for this partition we will 'linger' for the specified time waiting for more records to show up. This setting defaults to 0 (i.e. no delay). Setting linger.ms=5, for example, would have the effect of reducing the number of requests sent but would add up to 5ms of latency to records sent in the absense of load.	Ignored.
`max.block.ms`	The configuration controls how long {@link KafkaProducer#send()} and {@link KafkaProducer#partitionsFor} will block. These methods can be blocked for multiple reasons. For e.g: buffer full, metadata unavailable.This configuration imposes maximum limit on the total time spent in fetching metadata, serialization of key and value, partitioning and allocation of buffer memory when doing a send(). In case of partitionsFor(), this configuration imposes a maximum time threshold on waiting for metadata	Ignored. MapR Event Store For Apache Kafka has a similar parameter: `streams.rpc.timeout.ms`
`max.request.size`	The maximum size of a request. This is also effectively a cap on the maximum record size. Note that the server has its own cap on record size which may be different from this. This setting will limit the number of record batches the producer will send in a single request to avoid sending huge requests.	Ignored.

Name	Description	Supported for producers in MapR Event Store For Apache Kafka?
`partitioner.class`	Partitioner class that implements the Partitioner interface.	Use the Kafka StreamsPartitioner interface.
`receive.buffer.bytes`	The size of the TCP receive buffer (SO_RCVBUF) to use when reading data.	Ignored.
`request.timeout.ms`	The configuration controls the maximum amount of time the client will wait for the response of a request. If the response is not received before the timeout elapses the client will resend the request if necessary or fail the request if retries are exhausted.	Ignored. MapR Event Store For Apache Kafka has a similar parameter: `streams.rpc.timeout.ms`
`sasl.kerberos.service.name`	The Kerberos principal name that Kafka runs as. This can be defined either in Kafka's JAAS config or in Kafka's config.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`security.protocol`	Protocol used to communicate with brokers. Currently only PLAINTEXT and SSL are supported.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`send.buffer.bytes`	The size of the TCP send buffer (SO_SNDBUF) to use when sending data.	Ignored.
`ssl.enabled.protocols`	The list of protocols enabled for SSL connections. TLSv1.2, TLSv1.1 and TLSv1 are enabled by default.	Ignored.
`ssl.keystore.type`	The file format of the key store file. This is optional for client. Default value is JKS	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`ssl.protocol`	The SSL protocol used to generate the SSLContext. Default setting is TLS, which is fine for most cases. Allowed values in recent JVMs are TLS, TLSv1.1 and TLSv1.2. SSL, SSLv2 and SSLv3 may be supported in older JVMs, but their usage is discouraged due to known security vulnerabilities.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`ssl.provider`	The name of the security provider used for SSL connections. Default value is the default security provider of the JVM.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`ssl.truststore.type`	The file format of the trust store file. Default value is JKS.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.

last-updated: Apr 11, 2022

Name	Description	Supported for producers in MapR Event Store For Apache Kafka?
`timeout.ms`	The configuration controls the maximum amount of time the server will wait for acknowledgments from followers to meet the acknowledgment requirements the producer has specified with the acks configuration. If the requested number of acknowledgments are not met when the timeout elapses an error will be returned. This timeout is measured on the server side and does not include the network latency of the request.	Ignored. MapR Event Store For Apache Kafka has a similar parameter: `streams.rpc.timeout.ms`
`block.on.buffer.full`	When our memory buffer is exhausted we must either stop accepting new records (block) or throw errors. By default this setting is true and we block, however in some scenarios blocking is not desirable and it is better to immediately give an error. Setting this to false will accomplish that: the producer will throw a BufferExhaustedException if a recrord is sent and the buffer space is full.	Ignored.
`max.in.flight.requests.per.connection`	The maximum number of unacknowledged requests the client will send on a single connection before blocking. Note that if this setting is set to be greater than 1 and there are failed sends, there is a risk of message re-ordering due to retries (i.e., if retries are enabled).	Ignored. MapR Event Store For Apache Kafka has a similar parameter: `streams.parallel.flushers.per.partition`
`metadata.fetch.timeout.ms`	The first time data is sent to a topic we must fetch metadata about that topic to know which servers host the topic's partitions. This fetch to succeed before throwing an exception back to the client.	Ignored.
`metadata.max.age.ms`	The period of time in milliseconds after which we force a refresh of metadata even if we haven't seen any partition leadership changes to proactively discover any new brokers or partitions.	Yes
`metric.reporters`	A list of classes to use as metrics reporters. Implementing the MetricReporter interface allows plugging in classes that will be notified of new metric creation. The JmxReporter is always included to register JMX statistics.	No
`metrics.num.samples`	The number of samples maintained to compute metrics.	No.
`metrics.sample.window.ms`	The number of samples maintained to compute metrics.	No.

last-updated: Apr 11, 2022

Name	Description	Supported for producers in MapR Event Store For Apache Kafka?
`reconnect.backoff.ms`	The amount of time to wait before attempting to reconnect to a given host. This avoids repeatedly connecting to a host in a tight loop. This backoff applies to all requests sent by the consumer to the broker.	Ignored. MapR Event Store For Apache Kafka has a similar parameter: `streams.rpc.timeout.ms`
`retry.backoff.ms`	The amount of time to wait before attempting to retry a failed fetch request to a given topic partition. This avoids repeated fetching-and-failing in a tight loop.	Ignored. MapR Event Store For Apache Kafka has a similar parameter: `streams.rpc.timeout.ms`
`sasl.kerberos.kinit.cmd`	Kerberos kinit command path. Default is /usr/bin/kinit	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`sasl.kerberos.min.time.before.relogin`	Login thread sleep time between refresh attempts.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`sasl.kerberos.ticket.renew.jitter`	Percentage of random jitter added to the renewal time.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`sasl.kerberos.ticket.renew.window.factor`	Login thread will sleep until the specified window factor of time from last refresh to ticket's expiry has been reached, at which time it will try to renew the ticket.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`ssl.cipher.suites`	A list of cipher suites. This is a named combination of authentication, encryption, MAC and key exchange algorithm used to negotiate the security settings for a network connection using TLS or SSL network protocol.By default all the available cipher suites are supported.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`ssl.endpoint.identification.algorithm`	The endpoint identification algorithm to validate server hostname using server certificate.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`ssl.keymanager.algorithm`	The algorithm used by key manager factory for SSL connections. Default value is the key manager factory algorithm configured for the Java Virtual Machine.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
`ssl.trustmanager.algorithm`	The algorithm used by trust manager factory for SSL connections. Default value is the trust manager factory algorithm configured for the Java Virtual Machine.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.

last-updated: Apr 11, 2022

Kafka Consumer

Name	Description	Supported for consumers in MapR Event Store For Apache Kafka?
bootstrap.servers	A list of host/port pairs to use for establishing the initial connection to the Kafka cluster. The client will make use of all servers irrespective of which servers are specified here for bootstrapping—this list only impacts the initial hosts used to discover the full set of servers. This list should be in the form host1:port1,host2:port2,.... Since these servers are just used for the initial connection to discover the full cluster membership (which may change dynamically), this list need not contain the full set of servers (you may want more than one, though, in case a server is down).	No. Cluster details are discovered from the file `mapr-clusters.conf`.
key.deserializer	Deserializer class for key that implements the Deserializer interface.	Yes
value.deserializer	Deserializer class for value that implements the Deserializer interface.	Yes
fetch.min.bytes	The minimum amount of data the server should return for a fetch request. If insufficient data is available the request will wait for that much data to accumulate before answering the request. The default setting of 1 byte means that fetch requests are answered as soon as a single byte of data is available or the fetch request times out waiting for data to arrive. Setting this to something greater than 1 will cause the server to wait for larger amounts of data to accumulate which can improve server throughput a bit at the cost of some additional latency.	Yes
fetch.max.bytes	The maximum amount of data the server should return for a fetch request. If the first record batch in the first non-empty partition of the fetch is larger than this configuration, the record batch is still returned to ensure that the consumer can make progress.  **Note:** This is new as of MapR 6.0.1.	Yes, as of MapR 6.0.1.
group.id	A unique string that identifies the consumer group this consumer belongs to. This property is required if the consumer uses either the group management functionality by using subscribe(topic) or the Kafka-based offset management strategy.	Yes

last-updated: Apr 11, 2022

Name	Description	Supported for consumers in MapR Event Store For Apache Kafka?
heartbeat.interval.ms	The expected time between heartbeats to the consumer coordinator when using Kafka's group management facilities. Heartbeats are used to ensure that the consumer's session stays active and to facilitate rebalancing when new consumers join or leave the group. The value must be set lower than session.timeout.ms, but typically should be set no higher than 1/3 of that value. It can be adjusted even lower to control the expected time for normal rebalances.	No
max.poll.records	Places an upper bound on the number of records returned from each call.  **Note:** This parameter is new as of MapR 6.0.1.	Yes, as of MapR 6.0.1.
max.partition.fetch.bytes	The maximum amount of data per-partition the server will return. The maximum total memory used for a request will be #partitions * max.partition.fetch.bytes. This size must be at least as large as the maximum message size the server allows or else it is possible for the producer to send messages larger than the consumer can fetch. If that happens, the consumer can get stuck trying to fetch a large message on a certain partition. If the first record batch in the first non-empty partition of the fetch is larger than this configuration, the record batch is still returned to ensure that the consumer can make progress.  **Note:** This is a behavior change as of MapR 6.0.1.	Yes
session.timeout.ms	The timeout used to detect failures when using Kafka's group management facilities.	Ignored
ssl.key.password	The password of the private key in the key store file. This is optional for client.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
ssl.keystore.location	The location of the key store file. This is optional for client and can be used for two-way authentication for client.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.

Name	Description	Supported for consumers in MapR Event Store For Apache Kafka?
ssl.keystore.password	The store password for the key store file.This is optional for client and only needed if ssl.keystore.location is configured.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
ssl.truststore.location	The location of the trust store file.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
ssl.truststore.password	The password for the trust store file.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
auto.offset.reset	What to do when there is no initial offset in Kafka or if the current offset does not exist any more on the server (e.g. because that data has been deleted): **earliest** automatically reset the offset to the earliest offset **latest** automatically reset the offset to the latest offset **none** throw exception to the consumer if no previous offset is found for the consumer's group **anything else** throw exception to the consumer.	Yes
connections.max.idle.ms	Close idle connections after the number of milliseconds specified by this config.	Ignored
enable.auto.commit	If true the consumer's offset will be periodically committed in the background.	Yes
partition.assignment.strategy	The class name of the partition assignment strategy that the client will use to distribute partition ownership amongst consumer instances when group management is used	Ignored. MapR Event Store For Apache Kafka distributes partitions equally among the consumers in a group.
receive.buffer.bytes	The size of the TCP receive buffer (SO_RCVBUF) to use when reading data.	Ignored.

Name	Description	Supported for consumers in MapR Event Store For Apache Kafka?
request.timeout.ms	The configuration controls the maximum amount of time the client will wait for the response of a request. If the response is not received before the timeout elapses the client will resend the request if necessary or fail the request if retries are exhausted.	Ignored. MapR Event Store For Apache Kafka has a similar parameter: `streams.rpc.timeout.ms`
sasl.kerberos.service.name	The Kerberos principal name that Kafka runs as. This can be defined either in Kafka's JAAS config or in Kafka's config.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
security.protocol	Protocol used to communicate with brokers. Currently only PLAINTEXT and SSL are supported.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
send.buffer.bytes	The size of the TCP send buffer (SO_SNDBUF) to use when sending data.	Ignored.
ssl.enabled.protocols	The list of protocols enabled for SSL connections. TLSv1.2, TLSv1.1 and TLSv1 are enabled by default.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
ssl.keystore.type	The file format of the key store file. This is optional for client. Default value is JKS	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
ssl.protocol	The SSL protocol used to generate the SSLContext. Default setting is TLS, which is fine for most cases. Allowed values in recent JVMs are TLS, TLSv1.1 and TLSv1.2. SSL, SSLv2 and SSLv3 may be supported in older JVMs, but their usage is discouraged due to known security vulnerabilities.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
ssl.provider	The name of the security provider used for SSL connections. Default value is the default security provider of the JVM.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
ssl.truststore.type	The file format of the trust store file. Default value is JKS.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
auto.commit.interval.ms	The frequency in milliseconds that the consumer offsets are auto-committed to Kafka if enable.auto.commit is set to true.	Yes

Name	Description	Supported for consumers in MapR Event Store For Apache Kafka?
check.crcs	Automatically check the CRC32 of the records consumed. This ensures no on-the-wire or on-disk corruption to the messages occurred. This check adds some overhead, so it may be disabled in cases seeking extreme performance.	Ignored. MapR Event Store For Apache Kafka always does end-to-end crc computation and verification.
client.id	An id string to pass to the server when making requests. The purpose of this is to be able to track the source of requests beyond just ip/port by allowing a logical application name to be included in server-side request logging.	Yes
fetch.max.wait.ms	The maximum amount of time the server will block before answering the fetch request if there isn't sufficient data to immediately satisfy the requirement given by fetch.min.bytes.	No
metadata.max.age.ms	The period of time in milliseconds after which we force a refresh of metadata even if we haven't seen any partition leadership changes to proactively discover any new brokers or partitions.	Yes
metric.reporters	A list of classes to use as metrics reporters. Implementing the MetricReporter interface allows plugging in classes that will be notified of new metric creation. The JmxReporter is always included to register JMX statistics.	No
metrics.num.samples	The number of samples maintained to compute metrics.	No
metrics.sample.window.ms	The number of samples maintained to compute metrics.	No
reconnect.backoff.ms	The amount of time to wait before attempting to reconnect to a given host. This avoids repeatedly connecting to a host in a tight loop. This backoff applies to all requests sent by the consumer to the broker.	Ignored. MapR Event Store For Apache Kafka has a similar parameter: `streams.rpc.timeout.ms`
retry.backoff.ms	The amount of time to wait before attempting to retry a failed fetch request to a given topic partition. This avoids repeated fetching-and-failing in a tight loop.	Ignored. MapR Event Store For Apache Kafka has a similar parameter: `streams.rpc.timeout.ms`
sasl.kerberos.kinit.cmd	Kerberos kinit command path. Default is /usr/bin/kinit	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.

last-updated: Apr 11, 2022

Name	Description	Supported for consumers in MapR Event Store For Apache Kafka?
sasl.kerberos.min.time.before.relogin	Login thread sleep time between refresh attempts.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
sasl.kerberos.ticket.renew.jitter	Percentage of random jitter added to the renewal time.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
sasl.kerberos.ticket.renew.window.factor	Login thread will sleep until the specified window factor of time from last refresh to ticket's expiry has been reached, at which time it will try to renew the ticket.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
ssl.cipher.suites	A list of cipher suites. This is a named combination of authentication, encryption, MAC and key exchange algorithm used to negotiate the security settings for a network connection using TLS or SSL network protocol.By default all the available cipher suites are supported.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
ssl.endpoint.identification.algorithm	The endpoint identification algorithm to validate server hostname using server certificate.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
ssl.keymanager.algorithm	The algorithm used by key manager factory for SSL connections. Default value is the key manager factory algorithm configured for the Java Virtual Machine.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.
ssl.trustmanager.algorithm	The algorithm used by trust manager factory for SSL connections. Default value is the trust manager factory algorithm configured for the Java Virtual Machine.	Ignored. Authentication and authorization are handled through MapR Data Platform security. See Security on page 683 for more information.

MapR Event Store For Apache Kafka Parameters

The following MapR Event Store For Apache Kafka parameters are for the Admin API.

Table

MapR Event Store For Apache Kafka Parameter	Description	Kafka Parameter Replaced

`streams.rpc.timeout.ms`	Specifies the length of time in milliseconds to wait for a response from the MapR Event Store For Apache Kafka server if soft mount is configured (fs.mapr.hardmount is set to false). Default: 120000 Minimum: 30000
	  **Note:** Applicable as of MapR 6.0.1, is used instead of `fs.mapr.rpc.timeout`
		`request.timeout.ms`
		`reconnect.backoff.ms`
		`retry.backoff.ms`

The following MapR Event Store For Apache Kafka parameter are for the Producer API:

Table

MapR Event Store For Apache Kafka Parameter	Description	Kafka Parameter Replaced
`streams.buffer.max.time.ms`	Messages are buffered in the producer for at most the specified time. A thread will flush all the messages that have been buffered for more than the time specified. Default: 3 * 1000 msec	`linger.ms`
`streams.parallel.flushers.per.partition`	If enabled, producer may have multiple parallel send requests to the server for each topic partition. If this setting is set to true, it is possible for messages to be sent out of order. Default: true	`max.in.flight.requests.per.connection`
`streams.partitioner.class`	The class that implements the `StreamsPartitioner` interface. This interface lets you write custom algorithms for determining which topic and partition to use for messages that match specific criteria. Use this configuration parameter only for producers that are written in Java.	Not applicable.

Table (Continued)

MapR Event Store For Apache Kafka Parameter	Description	Kafka Parameter Replaced
`streams.producer.default.stream`	Specifies the stream that the producer will use by default if the producer does not provide the name of a stream when specifying a topic to write to. For example, the producer can specify the name of a stream together with the name of a topic to write to, like this: /<stream>/<topic> However, if the stream is not specified, the value of this configuration parameter is assumed to be the stream in which the topic is located. If the producer specifies the name of a topic without also providing the path and name of the stream, and there is no value for this configuration parameter, MapR Event Store For Apache Kafka assumes that the topic specified is in Apache Kafka and does nothing.	Not applicable.

`streams.rpc.timeout.ms`	Specifies the length of time in milliseconds to wait for a response from the MapR Event Store For Apache Kafka server if soft mount is configured (fs.mapr.hardmount is set to false). Default: 30000 Minimum: 30000
	If the time expires while a producer waits for a response from the MapR Event Store For Apache Kafka server, and the producer used the `KafkaProducer.send(ProducerRecord<K,V> record, Callback callback)` method, the callback is invoked with the error `EAGAIN`, which means "Resource temporarily unavailable."
	Note: Applicable as of MapR 6.0.1, is used instead of `fs.mapr.rpc.timeout`
		`max.block.ms`
		`request.timeout.ms`
		`timeout.ms`
		`reconnect.backoff.ms`
		`retry.backoff.ms`

The following MapR Event Store For Apache Kafka parameters are for the Consumer API:

Table

MapR Event Store For Apache Kafka Parameter	Description	Kafka Parameter Replaced
`streams.consumer.default.stream`	Specifies the path and name of the stream that the consumer subscribes to if, when subscribing to a topic, the consumer does not specify a stream. This default value is also used for the KafkaConsumer.listTopics() method.	Not applicable.

`streams.rpc.timeout.ms`	Specifies the length of time in milliseconds to wait for a response from the MapR Event Store For Apache Kafka server if a soft mount is configured (fs.mapr.hardmount is set to false). Default: 305000 Minimum: 300000
	Note: Applicable as of MapR 6.0.1, is used instead of `fs.mapr.rpc.timeout`
		`request.timeout.ms`
		`reconnect.backoff.ms`
		`retry.backoff.ms`

MapR Event Store For Apache Kafka C Applications

C applications can be developed for MapR Event Store For Apache Kafka (as of MapR 5.2.1). The MapR Event Store For Apache Kafka C Client is a distribution of librdkafka that works with MapR Event Store For Apache Kafka. The MapR Event Store For Apache Kafka C Client is available in MapR Ecosystem Pack (EEP) 3.0 or higher.

The following Apache Kafka librdkafka versions are supported:

Table

Core release	EEP Release	Kafka librdkafka version
As of 6.0.1	As of 5.0	0.11.3
As of 5.2.1 through 6.0.0	As of 3.0	0.9.0

The MapR Event Store For Apache Kafka C Client supports a majority of the librdkafka C APIs plus additional configuration properties that are available only with MapR Event Store For Apache Kafka. When developing applications for MapR Event Store For Apache Kafka or migrating Kafka applications to MapR Event Store For Apache Kafka, see the list of librdkafka APIs Supported by MapR Event Store For Apache Kafka C Client on page 2811 which also describes API behavior. Reference rdkafka.h for API signatures.

When developing and running MapR Event Store For Apache Kafka C applications, note the following:

- You can create producers and high-level consumers. Low-level consumers are not supported.
- Consuming or producing topics in a Kafka cluster is not supported.
- As of MapR Data Platform 6.0, the MapR Event Store For Apache Kafka offset values start at 0.
- MapR Data Platform Security is supported. Kafka application-level security is not supported. See Security on page 683.
- User impersonation is not supported.

CAUTION: As of MapR Data Platform 6.1, the mapr-core package has a dependency on mapr-librdkafka. If the mapr-librdkafka package is installed, do not remove it manually. Doing so could result in the removal of MapR Data Platform core packages, rendering the node unusable.

last-updated: Apr 11, 2022

Configuring the MapR Event Store For Apache Kafka C Client

After installing the MapR Data Platform Client and before developing applications, you must configure your client C library by setting the library path.

Linux

For Linux installations, add `/opt/mapr/lib` to the end of LD_LIBRARY_PATH.

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/mapr/lib
```

Important: For MapR Data Platform 6.0.1, the libjvm.so configuration is *not* required.

For MapR Data Platform 6.0.0 and earlier, add the /opt/mapr/lib *and the path to the directory that contains libjvm.so* to the end of LD_LIBRARY_PATH.

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/mapr/
lib:<path_to_libjvm.so_directory>
```

The location of the libjvm.so differs based on where you installed Java. You can use `find / -name libjvm*` to determine the file location. For example, if the libjvm.so file is in the following location:

```
/usr/lib/jvm/java-7-openjdk-amd64/jre/lib/amd64/server/libjvm.so
```

Then, you set the library path like this:

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/mapr/lib:/usr/lib/jvm/
java-7-openjdk-amd64/jre/lib/amd64/server/
```

Mac

For Mac installations, add `/opt/mapr/lib` to the end of DYLD_LIBRARY_PATH.

```
export DYLD_LIBRARY_PATH=$DYLD_LIBRARY_PATH:/opt/mapr/lib
```

Important: For MapR Data Platform 6.0.1, the libjvm.so configuration is *not* required.

For MapR Data Platform 6.0.0 and earlier, add /opt/mapr/lib *and the path to the directory that contains libjvm.dylib* to the end of DYLD_LIBRARY_PATH.

```
export DYLD_LIBRARY_PATH=$DYLD_LIBRARY_PATH:/opt/mapr/
lib:<path_to_libjvm.dylib_directory>
```

The location of the libjvm.dylib differs based on where you installed Java. You can use `find / -name libjvm*` to determine the file location. For example, if the libjvm.dylib file is in the following location:

```
/Library/Java/JavaVirtualMachines/jdk1.8.0_121.jdk/Contents/
Home/jre/lib/server/libjvm.dylib
```

Then, you set the library path like this:

```
export DYLD_LIBRARY_PATH=$DYLD_LIBRARY_PATH:/opt/mapr/lib:/Library/Java/
JavaVirtualMachines/jdk1.8.0_121.jdk/Contents/Home/jre/lib/server
```

Windows

Note: As of MapR Data Platform 6.0.1, the MapR Data Platform C client is available on Windows.

For Windows installations, no additional configuration is required. Link your application and run your programs against the MapR Data Platform Client dynampic link libraries (dll) located at: `C:\opt\mapr\lib`. The corresponding librdkafka header is `C:\opt\mapr\include\librdkafka`.

Developing a MapR Event Store For Apache Kafka C Application

This topic includes basic information about how to develop a MapR Event Store For Apache Kafka C application. Sample applications are provided.

Before you Begin

Confirm that your environment meets the following requirements:

- The MapR Data Platform cluster version is 5.2.1 or greater.
- MapR Data Platform core client (mapr-client) packagei s installed on the node and it is configured to access the cluster. Or, it is a MapR Data Platform cluster node. See Installing the MapR Client on page 389 for more information.
- The MapR Event Store For Apache Kafka C Client is installed and configured on the node. See Configuring the MapR Event Store For Apache Kafka C Client on page 2796.
- GNU Compiler Collection (GCC) is installed on the node.

Creating, Compiling and Running C Apps

The following sections describes how to create a producer and consumer in C, compile the source code, generate executables, and run the applications.

Create Producer

This topic describes how to create a MapR Event Store For Apache Kafka streams producer in C. While the code to generate a MapR Event Store For Apache Kafka stream producer varies depending on the use case, in general, the producer code should contain the following:

1. Include the rdkafka.h header file (/opt/mapr/include/librdkafka/rdkafka.h)
2. Use `rd_kafka_conf_new()` to create the producer configuration.
3. Use `rd_kafka_new ()` to create the producer handle.
4. Use `rd_kafka_topic_conf_new()` to create the topic configuration.
5. Use `rd_kafka_topic_new ()` to create a topic handle for the producer.
6. Use `rd_kafka_produce()` to produce messages.
7. Optionally, use `rd_kafka_poll()`to poll for callbacks. This is useful to see if there are messages that have yet to be sent to the server.
8. Use `rd_kafka_topic_destroy ()` to destroy the topic handle destroy
9. Use `rd_kafka_destroy ()` to destroy the producer handle.

Note: For more details on the APIs, see Supported APIs for MapR Event Store For Apache Kafka C Client and rdkafka.h on page 2892

last-updated: Apr 11, 2022

For example, the following source code produces 5 messages to topic /MapR_Streams:MapR-Topic1:

```
/*
 * This file contains the producer function.
 *
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <rdkafka.h>
#include <errno.h>

/* msgDeliveryCB: Is the delivery callback.
 * The delivery report callback will be called once for each message
 * accepted by rd_kafka_produce() with err set to indicate
 * the result of the produce request. An application must call
rd_kafka_poll()
 * at regular intervals to serve queued delivery report callbacks.
 */
static void msgDeliveryCB (rd_kafka_t *rk,
                           const rd_kafka_message_t *rkmessage, void
*opaque) {
    if (rkmessage->err != RD_KAFKA_RESP_ERR_NO_ERROR) {
        printf("FAILURE: Message not delivered to partition.\n");
        printf("ERROR: %s", rd_kafka_err2str(rkmessage->err));
    } else {
        printf("Produced: %.*s\n",(int)rkmessage->len, (const
char*)rkmessage->payload);
    }
}

/*
 * Method      : int producer(int nummsgs_p, const char *fullTopicName)
 * Description : This is a simple producer method. In this method the
producer
 *               produces messages to a topic.
 */

int producer(int nummsgs_p, const char *fullTopicName) {
    printf("********************** PRODUCER **********************\n");
    rd_kafka_t *prodHndle;
    rd_kafka_conf_t *prodCfg;
    char errstr[1000];
    int totalMsgs = nummsgs_p;

    printf("Create producer configuration object\n");
    /*
     * rd_kafka_conf_new(): This API creates default rd_kafka_conf_t object
to
     * be passed at the time of producer object creation using rd_kafka_new
call.
     */
    prodCfg = rd_kafka_conf_new();
    if (prodCfg == NULL) {
        printf("Failed to create conf\n");
        return (EXIT_FAILURE);
    }
    /* rd_kafka_conf_set_dr_msg_cb(): This API sets the producer callback
     * 'msgDeliveryCB' in producer config 'prodCfg'
     * The delivery report callback will be called once for each message
     * accepted by rd_kafka_produce() with err set to indicate
     * the result of the produce request. An application must call
```

```
rd_kafka_poll()
     * at regular intervals to serve queued delivery report callbacks.
     */
    rd_kafka_conf_set_dr_msg_cb(prodCfg, msgDeliveryCB);

    printf("Create Producer Kafka handle\n");
    /*
     * rd_kafka_new():Creates a new Kafka handle and starts its operation
     * according to the specified type (RD_KAFKA_CONSUMER or
RD_KAFKA_PRODUCER).
     * prodCfg object passed here is freed by this function and must not be
used
     * or destroyed by the application subsequently. errstr must be a
pointer to
     * memory of at least size errstr_size where
     * `rd_kafka_new()` may write a human readable error message in case the
     * creation of a new handle fails. In which case the function returns
NULL.
     */
    prodHndle = rd_kafka_new(RD_KAFKA_PRODUCER, prodCfg, errstr,
sizeof(errstr));
    if (prodHndle == NULL) {
        printf("Failed to create producer: %s\n", errstr);
        return (EXIT_FAILURE);
    }

    /*
     * Following code does following:
     * 1. Create a topic handle for each producer-topic combination
     * 2. Produce 'totalMsgs' # of messages using topic handle created in
step 1
     * 3. Wait for all messages to be produced and callback to be delivered.
     * 4. Move on to next topic and repeat.
     */

    int totalTopics = 1;
    for (int nTopics = 0; nTopics < totalTopics ; nTopics++) {
        printf("Create topic handle\n");
        rd_kafka_topic_conf_t *prodTopicCfg;
        /*
         * rd_kafka_topic_conf_new(): This API Creates topic conf object
         * required to create topic handle which then will be used for each
         * producer-topic combination
         */

        prodTopicCfg = rd_kafka_topic_conf_new();
        if (prodTopicCfg == NULL) {
            printf("Failed to create new topic conf\n");
            return (EXIT_FAILURE);
        }

            rd_kafka_topic_t *prodTopicHndl;
            /*
             * rd_kafka_topic_new(): This API Creates topic handle for a
given
             * producer, topic name and topic config. Topic handles are
refcounted
             * internally and calling rd_kafka_topic_new()
             * again with the same topic name will return the previous
topic handle
             * without updating the original handle's configuration.
             * Applications must eventually call rd_kafka_topic_destroy()
for each
             * succesfull call to rd_kafka_topic_new() to clear up
```

```
resources.
             */
            prodTopicHndl = rd_kafka_topic_new(prodHndle, fullTopicName,
prodTopicCfg);
            if (prodTopicHndl == NULL) {
                printf("Failed to create new topic handle\n");
                return (EXIT_FAILURE);
            }
            prodTopicCfg = NULL; /* Now owned by topic */

            const char* key ="Key";
            printf("Send/Produce message to topic: %s\n", fullTopicName);
            for (int i = 0; i < totalMsgs; i++) {
                    char payload[1000];
                    if (i == 0)
                         sprintf(payload, "%s", "Welcome to MapR Streams
CAPI");
                    else
                         sprintf(payload, "MapR Streams CAPI Message
Payload %d", i);
                    /*
                    * rd_kafka_produce(): This API produces a single message
                    * to the cluster. prodTopicHandle must be created using
                    * rd_kafka_topic_new() api. This is an asynch
non-blocking API.
                    * RD_KAFKA_PARTITION_UA is used to indicate automatic
                    * partitioning, using topics partitioner or fixed
partition
                    * can be provided. RD_KAFKA_MSG_F_COPY flag indicates
that
                    * library copies the payload and application manages
its own
                    * payload memory. If API fails to send, errno will be
set
                    * accordingly and will be able to access librdkafka
specific
                    * error using rd_kafka_last_error() api.
                    */
                    if (rd_kafka_produce(prodTopicHndl,
                                         RD_KAFKA_PARTITION_UA,
                                         RD_KAFKA_MSG_F_COPY,
                                         payload,
                                         strlen(payload),
                                         key,
                                         strlen(key),
                                         NULL) == -1) {
                        int errNum = errno;
                        printf("Failed to produce to topic : %s\n",
rd_kafka_topic_name(prodTopicHndl));
                        printf("Error Number: %d ERROR NAME: %s\n"
                                    ,errNum,
rd_kafka_err2str(rd_kafka_last_error()));
                        return (errNum);
                    }
            }

            printf("Wait for messages to be delivered\n");
            /*
            * rd_kafka_outq_len(): This API out queue contains messages
waiting
            * to be sent to, or acknowledged by, server.
            * An application should wait for this queue to reach zero before
            * terminating to make sure outstanding requests are fully
processed.
```


last-updated: Apr 11, 2022

```
             *
             * rd_kafka_poll(): This API polls the producer handle for
events,
             * which will cause application provided callbacks to be called.
             * An application must call rd_kafka_poll() at regular intervals
to
             * serve queued delivery report callbacks. In this case
             * 'msgDeliveryCB' will get called.
             */
             while (rd_kafka_outq_len(prodHndle) > 0)
                 rd_kafka_poll(prodHndle, 100);

             printf("\nDestroy topic handle\n");
             /*
             * Applications must eventually call rd_kafka_topic_destroy()
for each
             * succesfull call to rd_kafka_topic_new() to clear up resources.
             */
             rd_kafka_topic_destroy(prodTopicHndl);
    }
    printf("Destroy producer handle\n");
    /*
     * rd_kafka_destroy(): This API destroys the producer handle created
using
     * rd_kafka_new call and frees resources.
     */
    rd_kafka_destroy(prodHndle);

    return(EXIT_SUCCESS);
}


/* MAIN */
int main(int argc, char *argv[]) {

  /* Number of messages the producer will produce */
  int nummsgs_p = 5;

  /* This is pre created Stream with one topic and one partition*/
  const char* fullTopicName = "/MapR_Streams:MapR-Topic1";
  int ret_val;

  /* Produce Messages */
  ret_val = producer(nummsgs_p, fullTopicName);
  if (EXIT_SUCCESS != ret_val) {
    printf("\nFAIL: producer failed\n");
  } else {
    printf("\nPASS: %d messages produced and sent to topic partition %s
\n", nummsgs_p, fullTopicName);
  }
}
```

Create Consumer

This topic describes how to create a MapR Event Store For Apache Kafka streams consumer in C. While the code to generate a MapR Event Store For Apache Kafka stream consumer varies depends on the use case, in general, the consumer code should contain the following:

1. Include the rdkafka.h header file (/opt/mapr/include/librdkafka/rdkafka.h).
2. Use `rd_kafka_conf_new()` to create the consumer configuration.

3. Use `rd_kafka_conf_set()` to set the configuration parameters. For this API, you must set the "group.id."
4. Use `rd_kafka_new ()` to create the consumer handle.
5. Use `rd_kafka_subscribe()` or `rd_kafka_assign()` to specify which topics to consume.
6. Use `rd_kafka_consumer_poll()` to poll for messages that are ready to be consumed.
7. Use `rd_kafka_consumer_close()` to perform auto commits and prepare to destroy the consumer handle.
8. Use `rd_kafka_destroy ()` to destroy the consumer handle.

For example, the following source code consumes 5 messages from topic /MapR_Streams:MapR-Topic1:

```
/*
 * This file contains the consumer function.
 *
 */

#include <stdio.h>
#include <stdlib.h>
#include <rdkafka.h>
#include <string.h>

/*
 * Method      : int consumer(int expected_nummsgs, const char
*fullTopicName)
 * Description : This is a simple consumer method. In this method the
consumer
 *               consumes messages from a topic.
 */

int consumer(int expected_nummsgs, const char *fullTopicName) {
    printf("*********  CONSUMER START  *********\n");
    rd_kafka_t *consHndle;
    rd_kafka_conf_t *consCfg;
    rd_kafka_topic_conf_t *consTopicCfg;
    char errstr[1000];
    rd_kafka_resp_err_t errCode;

    printf("Create new consumer configuration object\n");
    /*
    * rd_kafka_conf_new(): This API creates default rd_kafka_conf_t object
to
    * be passed at the time of consumer object creation using rd_kafka_new
call.
    */
    consCfg = rd_kafka_conf_new();
    if(consCfg == NULL) {
        printf("Failed to create consumer conf\n");
        return(EXIT_FAILURE);
    }
    /*
    * rd_kafka_conf_set(): This API is used to set config parameters in the
    * rd_kafka_conf_t object. group.id Must be set for all the consumers.
    * All changes to the consCfg must be done before creating consumer
object.
    */
    if(RD_KAFKA_CONF_OK != rd_kafka_conf_set(consCfg,
                      "group.id", "consumerGroup",
```

```
                        errstr, sizeof(errstr))) {
        printf("rd_kafka_conf_set() failed with error: %s\n", errstr);
        return (EXIT_FAILURE);
    }
   /*
    * rd_kafka_topic_conf_new(): This API Creates topic conf object
    * required to set the default topic configuration.
    */
    printf("Set topic configurations\n");
    consTopicCfg = rd_kafka_topic_conf_new();

    /* rd_kafka_topic_conf_set(): This API sets the config property by name.
    * consTopicCfg should have been previously set up with
`rd_kafka_topic_conf_new()`
    * property set in this call is 'auto.offset.reset', when set to
    * earliest will return messages on rd_kafka_consumer_poll from
beginning of
    * time (for the very first time consumption) or from last commited
offset
    * for online consumer. If property is set to 'latest' it will return the
    * messages produced after consumer has started(for first time consumer)
or
    * from the last committed offset for online consumer
    */
    if (RD_KAFKA_CONF_OK != rd_kafka_topic_conf_set(consTopicCfg,
"auto.offset.reset",
                            "earliest" ,errstr, sizeof(errstr))) {
        printf("rd_kafka_topic_conf_set() failed with error: %s\n", errstr);
        return (EXIT_FAILURE);
    }

    /*
    * rd_kafka_conf_set_default_topic_conf(): This API sets the default
topic
    * configuration to use for automatically subscribed topics
    * The topic config object is not usable after this call.
    */
    rd_kafka_conf_set_default_topic_conf(consCfg, consTopicCfg);

    printf("Create consumer Kafka handle\n");
    /*
     * rd_kafka_new():Creates a new Kafka handle and starts its operation
     * according to the specified type (RD_KAFKA_CONSUMER or
RD_KAFKA_PRODUCER).
     * consCfg object passed here is freed by this function and must not be
used
     * or destroyed by the application subsequently. errstr must be a
pointer to
     * memory of at least size errstr_size where
     * `rd_kafka_new()` may write a human readable error message in case the
     * creation of a new handle fails. In which case the function returns
NULL.
     */

    consHndle = rd_kafka_new(RD_KAFKA_CONSUMER, consCfg, errstr,
sizeof(errstr));
    if(consHndle == NULL) {
        printf("Failed to create consumer:%s", errstr);
        return (EXIT_FAILURE);
    }

    /* rd_kafka_poll_set_consumer() is used to redirect the main queue
which is
    *  serviced using rd_kafka_poll() to the rd_kafka_consumer_poll(). With
```

```
one api
    *  'rd_kafka_consumer_poll()' both callbacks and message are serviced.
    *  Once queue is forwarded using this API, it is not permitted to call
    *  rd_kafka_poll to service non message delivery callbacks.
    */
    rd_kafka_poll_set_consumer(consHndle);

    /* Topic partition list (tp_list) is supplied as an input to the
consumer
    *  subscribe(using rd_kafka_subscribe()). The api rd_kafka_subscribe()
expects
    *  that the partition argument to be set to RD_KAFKA_PARTITION_UA and
internally
    *  all partitions are assigned to the consumer.
    *  Note: partition balancing/assignment is done if more consumers are
part
    *  of the same consumer group.
    */

    printf("Create topic partition list for topic: %s\n", fullTopicName);
    rd_kafka_topic_partition_list_t *tp_list =
rd_kafka_topic_partition_list_new(0);
    rd_kafka_topic_partition_t* tpObj =
rd_kafka_topic_partition_list_add(tp_list,
                                                fullTopicName,
RD_KAFKA_PARTITION_UA);
    if (NULL == tpObj) {
        printf("Could not add the topic partition to the list.\n");
        return (EXIT_FAILURE);
    }

    printf("Subscribe consumer to the topic:\n");
    /*
    * rd_kafka_subscribe(): This API subscribes given consumer to the topic
list
    * provided in tp_list, depending upon number of consumers in a consumer
group
    * partitions will be balanced and assigned to each consumer.
    */
    errCode = rd_kafka_subscribe(consHndle, tp_list);
    if (errCode  != RD_KAFKA_RESP_ERR_NO_ERROR) {
        printf("Topic partition subscription failed. ERROR: %d\n", errCode);
        return(errCode);
    }
    printf("Destroy topic partition list:\n");
    /*
    * rd_kafka_topic_partition_list_destroy(): This API is used to free all
    * resources used by the list and the list itself.
    */

    rd_kafka_topic_partition_list_destroy(tp_list);

    printf("\nStart message consumption:\n");
    int msg_count = 0;
    while(1) {
        /*
        * rd_kafka_consumer_poll(): This API returns one message or
callback at
        * a time. An application should make sure to call consumer_poll()
at regular
        * intervals, even if no messages are expected, to serve any
        * queued callbacks waiting to be called.  When the application is
finished
        * with a message it must call rd_kafka_message_destroy() to destroy
```

```
and
        * message.
        */
        rd_kafka_message_t *msg = rd_kafka_consumer_poll(consHndle, 1000);
        if (msg != NULL) {
            if (msg->err == RD_KAFKA_RESP_ERR_NO_ERROR) {
                msg_count++;
                printf("%d Consumed: %.*s\n", msg_count,(int) msg->len,
                   (const char*)msg->payload);
                if (msg_count == expected_nummsgs){
                    rd_kafka_message_destroy(msg);
                    break;
                }
            }
            rd_kafka_message_destroy(msg);
        }
    }

    printf("\nCommit the offsets before closing the consumer\n");
    /*
    * Commit offsets on broker for the provided list of topic partitions.
    * when input is NULL the current partition assignment will be used
instead.
    * If async is false this operation will block until the offset commit
    * is done, returning the resulting success or error code.
    * This call is made to be sure that offsets are committed before
closing
    * consumer.
    */
    int retVal = rd_kafka_commit(consHndle, NULL, false/*async*/);
    if(retVal != RD_KAFKA_RESP_ERR_NO_ERROR) {
        printf("rd_kafka_commit() failed");
        return(EXIT_FAILURE);
    }


    printf("\nClose and destroy consumer handle\n");
    /*
    * Consumer shutdown sequense:
    * 1. rd_kafka_consumer_close(): This is blocking call. It makes sure to
revoke
    *    assignments, commit offsets, leave consumer group.
    *    The application still needs to call rd_kafka_destroy() after
    *    this call finishes to clean up the underlying handle resources.
    * 2. rd_kafka_destroy(): This API destroys the consumer handle created
using
    *    rd_kafka_new call and frees resources
    */

    rd_kafka_consumer_close(consHndle);
    rd_kafka_destroy(consHndle);
    return(EXIT_SUCCESS);
}

/* MAIN */
int main(int argc, char *argv[]) {

  /* Number of expected messages for the consumer */
  int expected_nummsgs = 5;

  /* This is pre created Stream with one topic and one partition*/
  const char* fullTopicName = "/MapR_Streams:MapR-Topic1";
  int ret_val;
```


last-updated: Apr 11, 2022

```
  /* Consume Messages */
  ret_val = consumer(expected_nummsgs, fullTopicName);
  if (EXIT_SUCCESS != ret_val) {
    printf("\nFAIL: consumer failed\n");
  } else {
    printf("\nPASS: %d messages consumed from topic %s\n",
expected_nummsgs, fullTopicName);
  }
}
```

Note: For more details on the APIs, see Supported APIs for MapR Event Store For Apache Kafka C Client and rdkafka.h on page 2892

Compile the Apps

This topic describes how to compile MapR Event Store For Apache Kafka streams producers and consumers in C. When you compile a MapR Event Store For Apache Kafka C application, you must link it with the librdkafka library in the /opt/mapr/lib/ library path and include the header file directory to ensure that your application references the header file included with MapR Event Store For Apache Kafka C Client.

Important: For MapR 6.0.0 and earlier, When you compile a MapR Event Store For Apache Kafka C application, you must link it with the librdkafka library in the /opt/mapr/lib/ library path, *the libjvm library*, and include the header file directory to ensure that your application references the header file included with MapR Event Store For Apache Kafka C Client.

The following steps compile the source code and generate executables in the same directory as the Makefile. For example, in the **librdkafka_example** directory, the `consumer` and `producer` executables are generated from the producer.c and consumer.c source files.

1. On your node, create a directory. For example: **librdkafka_example**.
2. In your directory (**librdkafka_example**), create a producer application. For example, if you are using the provided sample producer application:
   a. Create a file named producer.c.
   b. Copy the contents of the sample producer application into that file.
3. In your directory (**librdkafka_example**), create a consumer application. For example, if you are using the provided sample consumer application:
   a. Create a file named consumer.c.
   b. Copy the contents of the sample consumer application into that file.

4. In your directory (**librdkafka_example**), create a file named Makefile with the following content:

```
CC= g++
HEADERDIR=/opt/mapr/include/librdkafka/
CCFLAGS= -Wall -I$(HEADERDIR) -g -std=c99

export LD_LIBRARY_PATH=/opt/mapr/lib

LIBDIR= /opt/mapr/lib/
%.o: %.c
    gcc $(CCFLAGS) -c $<

consumer: consumer.o
    gcc -o $@ $@.o -lrdkafka -L$(LIBDIR) $(CCFLAGS)

producer: producer.o
    gcc -o $@ $@.o -lrdkafka -L$(LIBDIR) $(CCFLAGS)

all: consumer producer

clean:
    /bin/rm -f *.o consumer producer
```

Important: For MapR 6.0.0 and earlier, use the following Makefile:

```
CC= g++
HEADERDIR=/opt/mapr/include/librdkafka/
CCFLAGS= -Wall -I$(HEADERDIR) -g -std=c99

#Edit JAVA_HOME to be appropriate for your environment
JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64/
export LD_LIBRARY_PATH=/opt/mapr/lib:$(JAVA_HOME)/jre/lib/amd64/
server

LIBDIR= /opt/mapr/lib/
%.o: %.c
    gcc $(CCFLAGS) -c $<

consumer: consumer.o
    gcc -o $@ $@.o -lrdkafka -L$(LIBDIR) $(CCFLAGS)

producer: producer.o
    gcc -o $@ $@.o -lrdkafka -L$(LIBDIR) $(CCFLAGS)

all: consumer producer

clean:
    /bin/rm -f *.o consumer producer
```

5. Complete the following edits to the Makefile:

   For Mac users, locate the following line of code:

```
export LD_LIBRARY_PATH=/opt/mapr/lib
```

   Then, replace this line with the following line of code:

```
export DYLD_LIBRARY_PATH=/opt/mapr/lib
```


last-updated: Apr 11, 2022

Important: For MapR 6.0.0 and earlier, the following steps apply:

a. For Mac users, locate the following line of code:

```
export LD_LIBRARY_PATH=/opt/mapr/lib:$(JAVA_HOME)/jre/lib/amd64/
server
```

Then, replace this line with the following line of code:

```
export DYLD_LIBRARY_PATH=/opt/mapr/lib:$(JAVA_HOME)/jre/lib/server
```

b. Based on your environment, edit JAVA_HOME. This ensures that LD_LIBRARY_PATH or DYLD_LIBRARY_PATH will include the full path to the directory containing the libjvm library.

Note: You can use `find / -name libjvm*` to determine the JAVA_HOME directory on your machine. However, note that the results of this command include the full path to the libjvm file not just the JAVA_HOME directory.

For example, JAVA_HOME may be set to `Library/Java/JavaVirtualMachines/jdk1.8.0_121.jdk/Contents/Home/` on a Mac and JAVA_HOME may be set to `/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.79.x86_64/` on Linux.

6. From your directory (**librdkafka_example**), run the following commands to compile the source code:

```
make clean
```

```
make all
```

Run the Apps

Once you have the application executables, complete the following steps to run the application:

1. On a cluster node, use the maprcli to create a stream. For example, **MapR_Streams**.

```
maprcli stream create -path /MapR_Streams
```

Note: As long as autocreate is enabled for the stream when you run `stream create`,the producer will create the topic. By default, autocreate is enabled. For more information, see stream create on page 1758.

2. At the command line, set the library path to include **/opt/mapr/lib** and the path to the directory that contains the **libjvm** library. For more information, see Configuring the MapR Event Store For Apache Kafka C Client on page 2796.

Note: You must complete this step at the command line even though you already set the library path in the Makefile. If you do not complete the step, an error similar to the following displays when you run the application in the next step: `error while loading shared libraries: librdkafka.so.1: cannot open shared object file: No such file or directory`

last-updated: Apr 11, 2022

3. From your directory (**librdkafka_example**), run the producer application from the command line. For example, if the application is called producer:

```
./producer
```

   The following appears on the console assuming that the stream name is MapR_Streams:

```
********************** PRODUCER ***********************
Create producer configuration object
Create Producer Kafka handle
Create topic handle
Send/Produce message to topic: /MapR_Streams:MapR-Topic1
Wait for messages to be delivered
Produced: Welcome to MapR Streams CAPI
Produced: MapR Streams CAPI Message Payload 1
Produced: MapR Streams CAPI Message Payload 2
Produced: MapR Streams CAPI Message Payload 3
Produced: MapR Streams CAPI Message Payload 4

Destroy topic handle
Destroy producer handle

PASS: 5 messages produced and sent to topic partition /
MapR_Streams:MapR-Topic1
```

4. From your directory (**librdkafka_example**), run the consumer application from the command line. For example, if the application is called consumer:

```
./consumer
```

   The following appears on the console assuming that the stream name is MapR_Streams:

```
*********  CONSUMER START  *********
Create new consumer configuration object
Set topic configurations
Create consumer Kafka handle
Create topic partition list for topic: /MapR_Streams:MapR-Topic1
Subscribe consumer to the topic:
Destroy topic partition list:

Start message consumption:
1 Consumed: Welcome to MapR-ES CAPI
2 Consumed: MapR Streams CAPI Message Payload 1
3 Consumed: MapR Streams CAPI Message Payload 2
4 Consumed: MapR Streams CAPI Message Payload 3
5 Consumed: MapR Streams CAPI Message Payload 4

Commit the offsets before closing the consumer

Close and destroy consumer handle

PASS: 5 messages consumed from topic /MapR_Streams:MapR-Topic1
```

Migrating Kafka C Applications to MapR Event Store For Apache Kafka

With some modification, you can use existing Kafka C applications to consume and produce topics in MapR Event Store For Apache Kafka. The MapR Event Store For Apache Kafka C Client is a distribution of librdkafka that is compatible with MapR Event Store For Apache Kafka.

1. Install and configure the MapR Streams C Client.

last-updated: Apr 11, 2022

2. When you refer to a topic in the application code, include the path and name of the stream in which the topic is located:

   ```
   /<path and name of stream>:<name of topic>
   ```

   For example, you might have a stream in a MapR Data Platform cluster that is named stream_A, and the stream might be in a volume named IoT and in a directory named automobile_sensors. You want to redirect a producer application to a topic in that stream. The syntax of the path to the topic might look like this: `/mapr/IoT/automobile_sensors/stream_A:<name of topic>`.

   Note: Optionally, use the `streams.consumer.default.stream` and `streams.producer.default.stream` configuration parameters. When you configure these parameters, applications can specify just the topic name to write or read from the default stream. To use these MapR Data Platform-specific parameters in your application, compile your application with the rdkafka.h file (/opt/mapr/include/librdkafka/rdkafka.h) that was installed with the MapR Event Store For Apache Kafka C Client. See the Compile the Apps on page 2806 section of Developing a MapR Event Store For Apache Kafka C Application on page 2797.

3. See Configuration Properties for MapR Event Store For Apache Kafka C Client on page 2882 for the list of supported configuration parameters, including a few parameters that are MapR Data Platform-specific. Make changes to your application, as needed.

   Note: SSL-related configuration parameters are ignored. When you set these parameters, the MapR Event Store For Apache Kafka Client issues a warning indicating that the parameters are not supported.

4. Review the list of librdkafka APIs that are **not** supported by the MapR Event Store For Apache Kafka C Client and make changes to your application, as needed.

Simple/low level consumer APIs that are not supported	• rd_kafka_queue_new • rd_kafka_queue_destroy • rd_kafka_consume_start • rd_kafka_consume_start_queue • rd_kafka_consume_stop • rd_kafka_consume • rd_kafka_consume_batch • rd_kafka_consume_callback • rd_kafka_consume_queue • rd_kafka_consume_batch_queue • rd_kafka_consume_callback_queue • rd_kafka_offset_store • rd_kafka_pause_partitions • rd_kafka_resume_partitions
Producer/Consumer common APIs that are not supported	• rd_kafka_conf_set_dr_cb

last-updated: Apr 11, 2022

- rd_kafka_conf_set_throttle_cb
- rd_kafka_conf_set_stats_cb
- rd_kafka_conf_set_socket_cb
- rd_kafka_conf_set_open_cb
- rd_kafka_conf_dump
- rd_kafka_conf_dump_free
- rd_kafka_name
- rd_kafka_set_log_level
- rd_kafka_mem_free
- rd_kafka_set_log_level
- rd_kafka_mem_free

Topic APIs that are not supported

- rd_kafka_query_watermark_offsets

  Note: As of MapR Data Platform 6.0.1, this API *is* supported.

- rd_kafka_get_watermark_offsets

  Note: As of MapR Data Platform 6.0.1, this API *is* supported.

Cluster APIs that are not supported

- rd_kafka_memberid
- rd_kafka_metadata
- rd_kafka_metadata_destroy

Miscellaneous APIs that are not supported

- rd_kafka_version
- rd_kafka_version_str
- rd_kafka_get_debug_contexts
- rd_kafka_dump
- rd_kafka_thread_cnt
- rd_kafka_message_timestamp

librdkafka APIs Supported by MapR Event Store For Apache Kafka C Client

This topic lists the librdkafka APIs supported by the MapR Event Store For Apache Kafka C Client. It also describes behavior differences between librdkafka and the MapR Event Store For Apache Kafka C Client.

Table

Core release	EEP Release	Kafka librdkafka version
As of MapR Data Platform 6.0.1	As of 5.0	0.11.3
As of MapR Data Platform 5.2.1 through 6.0.0	As of 3.0	0.9.0

last-updated: Apr 11, 2022

This topic contains the following supported APIs:

Producer APIs

API	Behavior
rd_kafka_flush	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.
rd_kafka_poll	Same as librdkafka.

last-updated: Apr 11, 2022

API	Behavior
rd_kafka_produce	When this API is called with NULL payload, an invalid argument error is sent to the callback. librdkafka creates a message with NULL payload and key value instead.
rd_kafka_produceva	Same as librdkafka. This API should be used with either RD_KAFKA_V_TOPIC or RD_KAFKA_V_RKT. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.
rd_kafka_produce_batch	When this API is called with NULL payload, an invalid argument error is sent to the callback. librdkafka creates a message with NULL payload and key value instead.

API	Behavior
rd_kafka_outq_len	This API returns a positive number to indicate that messages are waiting to be produced to a streams topic but the value does not indicate the actual number of messages. librdkafka returns the actual number of messages that are waiting to be sent to or acknowledged by the broker.

Consumer APIs

API	Behavior
rd_kafka_assign	If this API is called for a consumer that is already subscribed to topics, no operation is performed.
rd_kafka_assignment	This API returns the number of topic partitions the consumer is assigned to. However, it returns 0 when topic partitions have yet to be created by the producer. librdkafka returns the number of partitions assigned to a consumer even when the partitions have not been created. **Note:** For this API to work, the argument partitions must be explicitly allocated or initialized with either `rd_kafka_topic_partition_list_t *parts = NULL` or `rd_kafka_topic_partition_list_new(0);`. For example: `RD_EXPORT rd_kafka_resp_err_t rd_kafka_assignment (rd_kafka_t *rk, rd_kafka_topic_partition_list_t **partitions);`

last-updated: Apr 11, 2022

API	Behavior
rd_kafka_commit	Same as librdkafka.
rd_kafka_commit_message	Same as librdkafka.
rd_kafka_committed	Same as librdkafka. **Note:** The MapR Event Store For Apache Kafka offset starts at 1.

API	Behavior
rd_kafka_consumer_close	Same as librdkafka.
rd_kafka_consumer_poll	Same as librdkafka. **Note:** For librdkafka 0.9: If the consume callback was set and messages were polled using rd_kafka_consumer_poll(), then the consume callback gets called and the messages can be consumed in the callback. For librdkafka 0.11.3: If the consume callback is set and messages are polled using rd_kafka_consumer_poll(), the consume callback is not called. The result is that you cannot consume messages in the consume callback when using rd_kafka_consumer_poll().

API	Behavior
rd_kafka_group_list_destroy	Same as librdkafka.
rd_kafka_list_groups	This API can only be used by consumers that are subscribed to at least one stream on the cluster and have a default stream configured with the streams.consumer.default.stream parameter. It returns the group list of subscribed consumers associated with the default stream. librdkafka returns all consumer groups from the cluster instead. **Note:** This API returns RD_KAFKA_RESP_ERR__TIMED_OUT when the querying consumer is not subscribed to any topic.

API	Behavior
rd_kafka_message_destroy	Same as librdkafka.
rd_kafka_poll_set_consumer	Same as librdkafka.

API	Behavior
rd_kafka_position	This API returns 0 when the messages have not yet been consumed from partitions. librdkafka returns -1001 instead.
rd_kafka_seek	Same as librdkafka.
rd_kafka_subscribe	Same as librdkafka.

API	Behavior
rd_kafka_subscription	This API allows either a list of topics from one or more streams or a regex expression for topics from a single streams. For example, regex expression `/streamA:^t*a,/streamA:^t*b` is supported but `/streamA:^t*a,/_streamB:^t*a` is not supported. librdkafka accepts both options in the same call. **Note:** You cannot use the rd_kafka_subscribe API to subscribe a consumer to topics when that consumer is already assigned to topics. If you call this API for an assigned consumer, error RD_KAFKA_RESP_ERR__CONFLICT is returned.
rd_kafka_unsubscribe	Same as librdkafka.

Producer/Consumer Common APIs

API	Behavior
rd_kafka_conf_destroy	Same as librdkafka.
rd_kafka_conf_dup	Same as librdkafka.
rd_kafka_conf_get	Same as librdkafka.

API	Behavior
rd_kafka_conf_new	Same as librdkafka.
rd_kafka_conf_set	Same as librdkafka.

API	Behavior
rd_kafka_conf_set_consume_cb	Same as librdkafka.
rd_kafka_conf_set_dr_msg_cb	Same as librdkafka.

API	Behavior
rd_kafka_conf_set_error_cb	Same as librdkafka.
rd_kafka_conf_set_offset_commit_cb	Same as librdkafka.

API	Behavior
rd_kafka_conf_set_opaque	Same as librdkafka.
rd_kafka_conf_set_rebalance_cb	Same as librdkafka.

API	Behavior
rd_kafka_conf_set_throttle_cb	Same as librdkafka.
rd_kafka_destroy	Same as librdkafka.
rd_kafka_new	Same as librdkafka.

API	Behavior
rd_kafka_opaque	Same as librdkafka.
rd_kafka_wait_destroyed	Same as librdkafka.
rd_kafka_yield	Same as librdkafka.

last-updated: Apr 11, 2022

Topic APIs

API	Behavior
rd_kafka_conf_set_default_topic_conf	Same as librdkafka.

API	Behavior
rd_kafka_topic_conf_destroy	Same as librdkafka.
rd_kafka_topic_conf_dup	Same as librdkafka.

last-updated: Apr 11, 2022

API	Behavior
rd_kafka_topic_conf_new	Same as librdkafka.
rd_kafka_topic_conf_set	Same as librdkafka.

API	Behavior
rd_kafka_topic_conf_set_opaque	Same as librdkafka.

API	Behavior
rd_kafka_topic_conf_set_partitioner_cb	Same as librdkafka.
rd_kafka_topic_destroy	Same as librdkafka.

API	Behavior
rd_kafka_topic_name	Same as librdkafka.
rd_kafka_topic_new	Same as librdkafka..

API	Behavior
rd_kafka_topic_partition_list_add	Same as librdkafka.

API	Behavior
rd_kafka_topic_partition_list_add_range	Same as librdkafka.

API	Behavior
rd_kafka_topic_partition_list_copy	Same as librdkafka.

API	Behavior
rd_kafka_topic_partition_list_del	Same as librdkafka.

API	Behavior
rd_kafka_topic_partition_list_del_by_idx	Same as librdkafka.

API	Behavior
rd_kafka_topic_partition_list_destroy	Same as librdkafka.

API	Behavior
rd_kafka_topic_partition_list_find	Same as librdkafka.

API	Behavior
rd_kafka_topic_partition_list_new	Same as librdkafka.

API	Behavior
rd_kafka_topic_partition_list_set_offset	Same as librdkafka.

Queue APIs

API	Behavior
rd_kafka_queue_new	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.
rd_kafka_queue_destroy	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_queue_get_main	For this API, produce events are batched as well as the APIs that use this API, such as, `rd_kafka_event_message_count` and `rd_kafka_event_message_next`. The messages produce events can be consumed together in batches, wheras, opensource librdkafka events are obtained one at a time. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.
rd_kafka_queue_get_consumer	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_queue_length	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.
rd_kafka_queue_poll	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_queue_poll_callback	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.
rd_kafka_commit_queue	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_queue_forward	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.
rd_kafka_set_log_queue	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

last-updated: Apr 11, 2022

API	Behavior
rd_kafka_queue_length	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.
rd_kafka_queue_to_event_enable	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_queue_get_partition	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

Event APIs

API	Behavior
rd_kafka_event_type	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_event_name	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.
rd_kafka_event_destroy	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_event_message_array	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.
rd_kafka_event_message_count	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

last-updated: Apr 11, 2022

API	Behavior
rd_kafka_event_error	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.
rd_kafka_event_error_string	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_event_topic_partition_list	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.
rd_kafka_event_message_next	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_event_opaque	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform6.0.1.

Timestamp APIs

API	Behavior
rd_kafka_offsets_for_times	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_message_timestamp	Same as librdkafka. Available as of librdkafka 0.9.1. Supported as of MapR Data Platform 6.0.1.

Interceptors APIs

Attention: Modifying the message in interceptors is not supported and can result in undefined behavior.

last-updated: Apr 11, 2022

API	Behavior
rd_kafka_interceptor_f_on_conf_set_t	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_interceptor_f_on_conf_dup	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

last-updated: Apr 11, 2022

API	Behavior
rd_kafka_interceptor_f_on_conf_destroy_t	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_interceptor_f_on_new_t	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_interceptor_f_on_destroy_t	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_interceptor_f_on_send_t	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_offset_store_message_interceptor_f_on_acknowledgement_f	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_interceptor_f_on_consume_t	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_interceptor_f_on_commit_t	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_conf_interceptor_add_on_conf_set	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_conf_interceptor_add_on_conf_dup	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_conf_interceptor_add_on_conf_destroy	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_conf_interceptor_add_on_new	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_conf_interceptor_add_on_destroy	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_interceptor_add_on_send	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_interceptor_add_on_acknowledgement	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

last-updated: Apr 11, 2022

API	Behavior
rd_kafka_interceptor_add_on_consume	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

API	Behavior
rd_kafka_interceptor_add_on_commit	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

Cluster Configuration APIs

API	Behavior
rd_kafka_brokers_add	This API has no impact on MapR Event Store For Apache Kafka since MapR Event Store For Apache Kafka does not utilize Kafka brokers. When this API is called, the MapR Event Store For Apache Kafka client may print a `_brokers are down error` message to the console.

last-updated: Apr 11, 2022

Miscellaneous API

API	Behavior
rd_kafka_err2name	Same as librdkafka.
rd_kafka_err2str	Same as librdkafka.
rd_kafka_errno	Same as librdkafka.

API	Behavior
rd_kafka_err_no2err	Same as librdkafka.
rd_kafka_get_err_descs	Same as librdkafka.
rd_kafka_last_error	Same as librdkafka.

last-updated: Apr 11, 2022

API	Behavior
rd_kafka_log_print	Same as librdkafka.
rd_kafka_log_syslog	Same as librdkafka.
rd_kafka_message_errstr	Same as librdkafka.

API	Behavior
rd_kafka_set_logger	Same as librdkafka.
rd_kafka_query_watermark_offsets	When you are querying or retrieving a topic that is non-existent topic/partition (using `rd_kafka_query_watermark_offsets()` and `rd_kafka_get_watermark_offsets()` APIs), the timeout is honored even though you still receive the correct error message. Supported as of MapR Data Platform 6.0.1.

last-updated: Apr 11, 2022

API	Behavior
rd_kafka_get_watermark_offsets	When you are querying or retrieving a topic that is non-existent topic/partition (using `rd_kafka_query_watermark_offsets()` and `rd_kafka_get_watermark_offsets()` APIs), the timeout is honored even though you still receive the correct error message. Supported as of MapR Data Platform 6.0.1.

Additional Information

For more information and API signatures, see rdkafka.h on page 2892.

librdkafka APIs NOT Supported by MapR Event Store For Apache Kafka C Client
This topic lists the librdkafka APIs that are *not* supported by the MapR Event Store For Apache Kafka C Client.

These APIs are also documented in the rdkafka.h on page 2892 as not support by MapR Event Store For Apache Kafka. If you want to see the list of supported librdkafka APIs, see librdkafka APIs Supported by MapR Event Store For Apache Kafka C Client on page 2811.

Note: This list of librdkafka APIs *not* supported is applicable to MapR Data Platform 6.0.1 and librdkafka 0.11.3.

Table

Core release	EEP Release	Kafka librdkafka version
As of MapR Data Platform 6.0.1	As of 5.0	0.11.3
As of MapR Data Platform 5.2.1 through 6.0.0	As of 3.0	0.9.0

```
RD_EXPORT
const char *rd_kafka_version_str (void);

RD_EXPORT
const char *rd_kafka_get_debug_contexts(void);
```

```
RD_EXPORT void
rd_kafka_topic_partition_list_sort (rd_kafka_topic_partition_list_t
*rktparlist,
                                    int (*cmp) (const void *a, const void
*b,
                                                void *opaque),
                                    void *opaque);

RD_EXPORT
int64_t rd_kafka_message_latency (const rd_kafka_message_t *rkmessage);

RD_EXPORT
rd_kafka_conf_t *rd_kafka_conf_dup_filter (const rd_kafka_conf_t *conf,
                                           size_t filter_cnt,
                                           const char **filter);

RD_EXPORT
void rd_kafka_conf_set_dr_cb(rd_kafka_conf_t *conf,
                  void (*dr_cb) (rd_kafka_t *rk,
                         void *payload, size_t len,
                         rd_kafka_resp_err_t err,
                         void *opaque, void *msg_opaque));

RD_EXPORT
void rd_kafka_conf_set_throttle_cb (rd_kafka_conf_t *conf,
                    void (*throttle_cb) (
                        rd_kafka_t *rk,
                        const char *broker_name,
                        int32_t broker_id,
                        int throttle_time_ms,
                        void *opaque));

RD_EXPORT
void rd_kafka_conf_set_log_cb(rd_kafka_conf_t *conf,
              void (*log_cb) (const rd_kafka_t *rk, int level,
                                          const char *fac, const char
*buf));

RD_EXPORT
void rd_kafka_conf_set_stats_cb(rd_kafka_conf_t *conf,
                 int (*stats_cb) (rd_kafka_t *rk,
                          char *json,
                          size_t json_len,
                          void *opaque));

RD_EXPORT
void rd_kafka_conf_set_socket_cb(rd_kafka_conf_t *conf,
                                 int (*socket_cb) (int domain, int type,
                                                   int protocol,
                                                   void *opaque));

RD_EXPORT void
rd_kafka_conf_set_connect_cb (rd_kafka_conf_t *conf,
                              int (*connect_cb) (int sockfd,
                                                 const struct sockaddr
*addr,
                                                 int addrlen,
                                                 const char *id,
                                                 void *opaque));

RD_EXPORT void
rd_kafka_conf_set_closesocket_cb (rd_kafka_conf_t *conf,
                                  int (*closesocket_cb) (int sockfd,
```

```
                                                        void *opaque));

RD_EXPORT
void rd_kafka_conf_set_open_cb (rd_kafka_conf_t *conf,
                                int (*open_cb) (const char *pathname,
                                                int flags, mode_t mode,
                                                void *opaque));

RD_EXPORT
const char **rd_kafka_conf_dump(rd_kafka_conf_t *conf, size_t *cntp);

RD_EXPORT
void rd_kafka_conf_dump_free(const char **arr, size_t cnt);

RD_EXPORT
void rd_kafka_conf_properties_show(FILE *fp);

RD_EXPORT
const char *rd_kafka_name(const rd_kafka_t *rk);

RD_EXPORT
rd_kafka_type_t rd_kafka_type(const rd_kafka_t *rk);

RD_EXPORT
char *rd_kafka_memberid (const rd_kafka_t *rk);

RD_EXPORT
char *rd_kafka_clusterid (rd_kafka_t *rk, int timeout_ms);

RD_EXPORT rd_kafka_resp_err_t
rd_kafka_pause_partitions (rd_kafka_t *rk,
               rd_kafka_topic_partition_list_t *partitions);

RD_EXPORT rd_kafka_resp_err_t
rd_kafka_resume_partitions (rd_kafka_t *rk,
                rd_kafka_topic_partition_list_t *partitions);

RD_EXPORT
void rd_kafka_mem_free (rd_kafka_t *rk, void *ptr);

RD_EXPORT
rd_kafka_queue_t *rd_kafka_queue_get_partition (rd_kafka_t *rk,
                                                const char *topic,
                                                int32_t partition);

RD_EXPORT
rd_kafka_resp_err_t rd_kafka_set_log_queue (rd_kafka_t *rk,
                                            rd_kafka_queue_t *rkqu);

RD_EXPORT
int rd_kafka_consume_start(rd_kafka_topic_t *rkt, int32_t partition,
                int64_t offset);

RD_EXPORT
int rd_kafka_consume_start_queue(rd_kafka_topic_t *rkt, int32_t partition,
                  int64_t offset, rd_kafka_queue_t *rkqu);

RD_EXPORT
int rd_kafka_consume_stop(rd_kafka_topic_t *rkt, int32_t partition);

RD_EXPORT
rd_kafka_message_t *rd_kafka_consume(rd_kafka_topic_t *rkt, int32_t
partition,
                      int timeout_ms);
```

```
RD_EXPORT
ssize_t rd_kafka_consume_batch(rd_kafka_topic_t *rkt, int32_t partition,
                int timeout_ms,
                rd_kafka_message_t **rkmessages,
                size_t rkmessages_size);

RD_EXPORT
int rd_kafka_consume_callback(rd_kafka_topic_t *rkt, int32_t partition,
                   int timeout_ms,
                   void (*consume_cb) (rd_kafka_message_t
                           *rkmessage,
                           void *opaque),
                   void *opaque);

RD_EXPORT
rd_kafka_message_t *rd_kafka_consume_queue(rd_kafka_queue_t *rkqu,
                        int timeout_ms);

RD_EXPORT
ssize_t rd_kafka_consume_batch_queue(rd_kafka_queue_t *rkqu,
                      int timeout_ms,
                      rd_kafka_message_t **rkmessages,
                      size_t rkmessages_size);

RD_EXPORT
int rd_kafka_consume_callback_queue(rd_kafka_queue_t *rkqu,
                     int timeout_ms,
                     void (*consume_cb) (rd_kafka_message_t
                             *rkmessage,
                             void *opaque),
                     void *opaque);

RD_EXPORT
rd_kafka_resp_err_t rd_kafka_offset_store(rd_kafka_topic_t *rkt,
                       int32_t partition, int64_t offset);

RD_EXPORT rd_kafka_resp_err_t
rd_kafka_offsets_store(rd_kafka_t *rk,
                       rd_kafka_topic_partition_list_t *offsets);

RD_EXPORT
rd_kafka_resp_err_t
rd_kafka_metadata (rd_kafka_t *rk, int all_topics,
                   rd_kafka_topic_t *only_rkt,
                   const struct rd_kafka_metadata **metadatap,
                   int timeout_ms);

RD_EXPORT
void rd_kafka_metadata_destroy(const struct rd_kafka_metadata *metadata);

RD_EXPORT
void rd_kafka_dump(FILE *fp, rd_kafka_t *rk);

RD_EXPORT
int rd_kafka_thread_cnt(void)

RD_EXPORT
int rd_kafka_unittest (void);

RD_EXPORT
int rd_kafka_event_log (rd_kafka_event_t *rkev,
            const char **fac, const char **str, int *level);
```

```
RD_EXPORT
const char *rd_kafka_event_stats (rd_kafka_event_t *rkev);
```

Configuration Properties for MapR Event Store For Apache Kafka C Client

This topic describes the configuration properties supported by the MapR Event Store For Apache Kafka C Client. This includes librdkafka configuration properties that MapR Event Store For Apache Kafka supports and additional properties that are specific to MapR Event Store For Apache Kafka.

Global Configuration Properties

Property Name	Behavior
client.id	Same as librdkafka.
message.max.bytes	See Configuring Properties for Message Size on page 3029.

Property Name	Behavior
receive.message.max.bytes	See Configuring Properties for Message Size on page 3029.
topic.blacklist	Same as librdkafka.
error_cb	Same as librdkafka.

Property Name	Behavior
opaque	Same as librdkafka.
queue.buffering.max.ms	Same as librdkafka.
enable.partition.eof	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

Property Name	Behavior
queue.buffering.max.kbytes	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

Consumer Configuration Properties

Property Name	Behavior
default_topic_conf	Same as librdkafka.
group.id	Same as librdkafka.
enable.auto.commit	Same as librdkafka.

Property Name	Behavior
auto.commit.interval.ms	Same as librdkafka.
rebalance_cb	Same as librdkafka.
offset_commit_cb	Same as librdkafka.

Property Name	Behavior
delivery.report.only.error	Same as librdkafka.
dr_msg_cb	Same as librdkafka.

Topic Configuration

Property Name	Behavior
partitioner_cb	Same as librdkafka.
auto.offset.reset	Supports the following values: beginning, end, earliest, latest, none, smallest, and largest. As of MapR Data Platform 6.0.1, beginning and end are supported. **Note:** librdkafka additionally supports error.

MapR Data Platform-Specific Configurations

Property Name	Behavior
streams.consumer.default.stream	Specifies the path and name of the stream that the consumer subscribes to if, when subscribing to a topic, the consumer does not specify a stream. For example, the consumer can specify the name of a stream together with the name of a topic to write to, like this: /<stream>:<topic>. **Note:** rd_kafka_list groups API uses this consumer configuration to obtain the consumer groups.

Property Name	Behavior
streams.parallel.flushers.per.partition	Enables the producer to have multiple parallel send requests to the server for each topic partition. When this property is set to true, the default value, it is possible for messages to be sent out of order.

Property Name	Behavior
streams.producer.default.stream	Specifies the stream that the producer will use by default if the producer does not provide the name of a stream when specifying a topic to write to. For example, the producer can specify the name of a stream together with the name of a topic to write to, like this: /<stream>:<topic>. However, if the stream is not specified, the value of this configuration parameter is assumed to be the stream in which the topic is located. If the producer specifies the name of a topic without also providing the path and name of the stream, and there is no value for this configuration parameter, MapR Event Store For Apache Kafka assumes that the topic specified is in Apache Kafka and does nothing.

Additional Information

For more information, see rdkafka.h on page 2892.

rdkafka.h
This rdkafka header file has been updated to be compatible with MapR Event Store For Apache Kafka. After you install the MapR Event Store For Apache Kafka C Client, this file is available in the following directory: /opt/mapr/include/librdkafka/

librdkafka 0.11.3

Apache librdkafka 0.11.3 is supported as of MapR Data Platform 6.0.1/EEP5.0.

Important: With this release, the RD_KAFKA_MSG_F_BLOCK call provides *blocking* behavior, whereas, the RD_KAFKA_MSG_F_COPY and RD_KAFKA_MSG_F_FREE calls are *non-blocking* (this is a behavior change from previous releases.

```
/*
 * librdkafka - Apache Kafka C library
```

```
 *
 * Copyright (c) 2012-2013 Magnus Edenhill
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are
met:
 *
 * 1. Redistributions of source code must retain the above copyright notice,
 *    this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
notice,
 *    this list of conditions and the following disclaimer in the
documentation
 *    and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS"
 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
 * LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
 * CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
 * SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
 * INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
 * CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
 * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF
THE
 * POSSIBILITY OF SUCH DAMAGE.
 */

/**
 * @file rdkafka.h
 * @brief Apache Kafka C/C++ consumer and producer client library.
 *
 * rdkafka.h contains the public API for librdkafka.
 * The API is documented in this file as comments prefixing the function,
type,
 * enum, define, etc.
 *
 * @sa For the C++ interface see rdkafkacpp.h
 *
 * @tableofcontents
 */


/* @cond NO_DOC */
#pragma once

#include <stdio.h>
#include <inttypes.h>
#include <sys/types.h>
#include "streams_util.h"

#ifdef __cplusplus
extern "C" {
#if 0
} /* Restore indent */
#endif
#endif

#ifdef _MSC_VER
```

```
#define strtok_r strtok_s
#include <basetsd.h>
#ifndef WIN32_MEAN_AND_LEAN
#define WIN32_MEAN_AND_LEAN
#endif
#include <Winsock2.h>  /* for sockaddr, .. */
typedef SSIZE_T ssize_t;
#define RD_UNUSED
#define RD_INLINE __inline
#define RD_DEPRECATED __declspec(deprecated)
#undef RD_EXPORT
#ifdef LIBRDKAFKA_STATICLIB
#define RD_EXPORT
#else
#ifdef LIBRDKAFKA_EXPORTS
#define RD_EXPORT __declspec(dllexport)
#else
#define RD_EXPORT __declspec(dllimport)
#endif
#ifndef LIBRDKAFKA_TYPECHECKS
#define LIBRDKAFKA_TYPECHECKS 0
#endif
#endif

#else
#include <sys/socket.h> /* for sockaddr, .. */

#define RD_UNUSED __attribute__((unused))
#define RD_INLINE inline
#define RD_EXPORT
#define RD_DEPRECATED __attribute__((deprecated))

#ifndef LIBRDKAFKA_TYPECHECKS
#define LIBRDKAFKA_TYPECHECKS 1
#endif
#endif


/**
 * @brief Type-checking macros
 * Compile-time checking that \p ARG is of type \p TYPE.
 * @returns \p RET
 */
#if LIBRDKAFKA_TYPECHECKS
#define _LRK_TYPECHECK(RET,TYPE,ARG)                    \
        ({ if (0) { TYPE __t RD_UNUSED = (ARG); } RET; })

#define _LRK_TYPECHECK2(RET,TYPE,ARG,TYPE2,ARG2)        \
        ({                                              \
                if (0) {                                \
                        TYPE __t RD_UNUSED = (ARG);     \
                        TYPE2 __t2 RD_UNUSED = (ARG2);  \
                }                                       \
                RET; })
#else
#define _LRK_TYPECHECK(RET,TYPE,ARG)  (RET)
#define _LRK_TYPECHECK2(RET,TYPE,ARG,TYPE2,ARG2) (RET)
#endif

/* @endcond */


/**
 * @name librdkafka version
```

```
 * @{
 *
 *
 */

/**
 * @brief librdkafka version
 *
 * Interpreted as hex \c MM.mm.rr.xx:
 *  - MM = Major
 *  - mm = minor
 *  - rr = revision
 *  - xx = pre-release id (0xff is the final release)
 *
 * E.g.: \c 0x000801ff = 0.8.1
 *
 * @remark This value should only be used during compile time,
 *         for runtime checks of version use rd_kafka_version()
 */
#define RD_KAFKA_VERSION  0x000b03ff
#define STREAMS_MIN_VERSION "5.2.1"

/**
 * @brief Returns the librdkafka version as integer.
 *
 * @returns Version integer.
 *
 * @sa See RD_KAFKA_VERSION for how to parse the integer format.
 * @sa Use rd_kafka_version_str() to retreive the version as a string.
 */
RD_EXPORT
int rd_kafka_version(void);

/**
 * @brief Returns the librdkafka version as string.
 *
 * @returns Version string
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
const char *rd_kafka_version_str (void);

/**@}*/


/**
 * @name Constants, errors, types
 * @{
 *
 *
 */


/**
 * @enum rd_kafka_type_t
 *
 * @brief rd_kafka_t handle type.
 *
 * @sa rd_kafka_new()
 */
typedef enum rd_kafka_type_t {
    RD_KAFKA_PRODUCER, /**< Producer client */
    RD_KAFKA_CONSUMER,  /**< Consumer client */
```

```
    RD_KAFKA_UNKNOWN /**< Error case, unknown client */
} rd_kafka_type_t;


/**
 * @enum Timestamp types
 *
 * @sa rd_kafka_message_timestamp()
 */
typedef enum rd_kafka_timestamp_type_t {
    RD_KAFKA_TIMESTAMP_NOT_AVAILABLE,   /**< Timestamp not available */
    RD_KAFKA_TIMESTAMP_CREATE_TIME,     /**< Message creation time */
    RD_KAFKA_TIMESTAMP_LOG_APPEND_TIME  /**< Log append time */
} rd_kafka_timestamp_type_t;



/**
 * @brief Retrieve supported debug contexts for use with the \c \"debug\"
 *        configuration property. (runtime)
 *
 * @returns Comma-separated list of available debugging contexts.
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
const char *rd_kafka_get_debug_contexts(void);

/**
 * @brief Supported debug contexts. (compile time)
 *
 * @deprecated This compile time value may be outdated at runtime due to
 *             linking another version of the library.
 *             Use rd_kafka_get_debug_contexts() instead.
 */
#define RD_KAFKA_DEBUG_CONTEXTS \

"all,generic,broker,topic,metadata,queue,msg,protocol,cgrp,security,fetch,fe
ature"


/* @cond NO_DOC */
/* Private types to provide ABI compatibility */
typedef struct rd_kafka_s rd_kafka_t;
typedef struct rd_kafka_topic_s rd_kafka_topic_t;
typedef struct rd_kafka_conf_s rd_kafka_conf_t;
typedef struct rd_kafka_topic_conf_s rd_kafka_topic_conf_t;
typedef struct rd_kafka_queue_s rd_kafka_queue_t;
/* @endcond */


/**
 * @enum rd_kafka_resp_err_t
 * @brief Error codes.
 *
 * The negative error codes delimited by two underscores
 * (\c RD_KAFKA_RESP_ERR__..) denotes errors internal to librdkafka and are
 * displayed as \c \"Local: \<error string..\>\", while the error codes
 * delimited by a single underscore (\c RD_KAFKA_RESP_ERR_..) denote broker
 * errors and are displayed as \c \"Broker: \<error string..\>\".
 *
 * @sa Use rd_kafka_err2str() to translate an error code a human readable
string
 */
```


last-updated: Apr 11, 2022

```
typedef enum {
    /* Internal errors to rdkafka: */
    /** Begin internal error codes */
    RD_KAFKA_RESP_ERR__BEGIN = -200,
    /** Received message is incorrect */
    RD_KAFKA_RESP_ERR__BAD_MSG = -199,
    /** Bad/unknown compression */
    RD_KAFKA_RESP_ERR__BAD_COMPRESSION = -198,
    /** Broker is going away */
    RD_KAFKA_RESP_ERR__DESTROY = -197,
    /** Generic failure */
    RD_KAFKA_RESP_ERR__FAIL = -196,
    /** Broker transport failure */
    RD_KAFKA_RESP_ERR__TRANSPORT = -195,
    /** Critical system resource */
    RD_KAFKA_RESP_ERR__CRIT_SYS_RESOURCE = -194,
    /** Failed to resolve broker */
    RD_KAFKA_RESP_ERR__RESOLVE = -193,
    /** Produced message timed out*/
    RD_KAFKA_RESP_ERR__MSG_TIMED_OUT = -192,
    /** Reached the end of the topic+partition queue on
     * the broker. Not really an error. */
    RD_KAFKA_RESP_ERR__PARTITION_EOF = -191,
    /** Permanent: Partition does not exist in cluster. */
    RD_KAFKA_RESP_ERR__UNKNOWN_PARTITION = -190,
    /** File or filesystem error */
    RD_KAFKA_RESP_ERR__FS = -189,
     /** Permanent: Topic does not exist in cluster. */
    RD_KAFKA_RESP_ERR__UNKNOWN_TOPIC = -188,
    /** All broker connections are down. */
    RD_KAFKA_RESP_ERR__ALL_BROKERS_DOWN = -187,
    /** Invalid argument, or invalid configuration */
    RD_KAFKA_RESP_ERR__INVALID_ARG = -186,
    /** Operation timed out */
    RD_KAFKA_RESP_ERR__TIMED_OUT = -185,
    /** Queue is full */
    RD_KAFKA_RESP_ERR__QUEUE_FULL = -184,
    /** ISR count < required.acks */
        RD_KAFKA_RESP_ERR__ISR_INSUFF = -183,
    /** Broker node update */
        RD_KAFKA_RESP_ERR__NODE_UPDATE = -182,
    /** SSL error */
    RD_KAFKA_RESP_ERR__SSL = -181,
    /** Waiting for coordinator to become available. */
        RD_KAFKA_RESP_ERR__WAIT_COORD = -180,
    /** Unknown client group */
        RD_KAFKA_RESP_ERR__UNKNOWN_GROUP = -179,
    /** Operation in progress */
        RD_KAFKA_RESP_ERR__IN_PROGRESS = -178,
     /** Previous operation in progress, wait for it to finish. */
        RD_KAFKA_RESP_ERR__PREV_IN_PROGRESS = -177,
     /** This operation would interfere with an existing subscription */
        RD_KAFKA_RESP_ERR__EXISTING_SUBSCRIPTION = -176,
    /** Assigned partitions (rebalance_cb) */
        RD_KAFKA_RESP_ERR__ASSIGN_PARTITIONS = -175,
    /** Revoked partitions (rebalance_cb) */
        RD_KAFKA_RESP_ERR__REVOKE_PARTITIONS = -174,
    /** Conflicting use */
        RD_KAFKA_RESP_ERR__CONFLICT = -173,
    /** Wrong state */
        RD_KAFKA_RESP_ERR__STATE = -172,
    /** Unknown protocol */
        RD_KAFKA_RESP_ERR__UNKNOWN_PROTOCOL = -171,
    /** Not implemented */
```

```
        RD_KAFKA_RESP_ERR__NOT_IMPLEMENTED = -170,
    /** Authentication failure*/
    RD_KAFKA_RESP_ERR__AUTHENTICATION = -169,
    /** No stored offset */
    RD_KAFKA_RESP_ERR__NO_OFFSET = -168,
    /** Outdated */
    RD_KAFKA_RESP_ERR__OUTDATED = -167,
    /** Timed out in queue */
    RD_KAFKA_RESP_ERR__TIMED_OUT_QUEUE = -166,
        /** Feature not supported by broker */
        RD_KAFKA_RESP_ERR__UNSUPPORTED_FEATURE = -165,
        /** Awaiting cache update */
        RD_KAFKA_RESP_ERR__WAIT_CACHE = -164,
        /** Operation interrupted (e.g., due to yield)) */
        RD_KAFKA_RESP_ERR__INTR = -163,
        /** Key serialization error */
        RD_KAFKA_RESP_ERR__KEY_SERIALIZATION = -162,
        /** Value serialization error */
        RD_KAFKA_RESP_ERR__VALUE_SERIALIZATION = -161,
        /** Key deserialization error */
        RD_KAFKA_RESP_ERR__KEY_DESERIALIZATION = -160,
        /** Value deserialization error */
        RD_KAFKA_RESP_ERR__VALUE_DESERIALIZATION = -159,
        /** Partial response */
        RD_KAFKA_RESP_ERR__PARTIAL = -158,

    /** End internal error codes */
    RD_KAFKA_RESP_ERR__END = -100,

    /* Kafka broker errors: */
    /** Unknown broker error */
    RD_KAFKA_RESP_ERR_UNKNOWN = -1,
    /** Success */
    RD_KAFKA_RESP_ERR_NO_ERROR = 0,
    /** Offset out of range */
    RD_KAFKA_RESP_ERR_OFFSET_OUT_OF_RANGE = 1,
    /** Invalid message */
    RD_KAFKA_RESP_ERR_INVALID_MSG = 2,
    /** Unknown topic or partition */
    RD_KAFKA_RESP_ERR_UNKNOWN_TOPIC_OR_PART = 3,
    /** Invalid message size */
    RD_KAFKA_RESP_ERR_INVALID_MSG_SIZE = 4,
    /** Leader not available */
    RD_KAFKA_RESP_ERR_LEADER_NOT_AVAILABLE = 5,
    /** Not leader for partition */
    RD_KAFKA_RESP_ERR_NOT_LEADER_FOR_PARTITION = 6,
    /** Request timed out */
    RD_KAFKA_RESP_ERR_REQUEST_TIMED_OUT = 7,
    /** Broker not available */
    RD_KAFKA_RESP_ERR_BROKER_NOT_AVAILABLE = 8,
    /** Replica not available */
    RD_KAFKA_RESP_ERR_REPLICA_NOT_AVAILABLE = 9,
    /** Message size too large */
    RD_KAFKA_RESP_ERR_MSG_SIZE_TOO_LARGE = 10,
    /** StaleControllerEpochCode */
    RD_KAFKA_RESP_ERR_STALE_CTRL_EPOCH = 11,
    /** Offset metadata string too large */
    RD_KAFKA_RESP_ERR_OFFSET_METADATA_TOO_LARGE = 12,
    /** Broker disconnected before response received */
    RD_KAFKA_RESP_ERR_NETWORK_EXCEPTION = 13,
    /** Group coordinator load in progress */
        RD_KAFKA_RESP_ERR_GROUP_LOAD_IN_PROGRESS = 14,
     /** Group coordinator not available */
        RD_KAFKA_RESP_ERR_GROUP_COORDINATOR_NOT_AVAILABLE = 15,
```


last-updated: Apr 11, 2022

```
    /** Not coordinator for group */
        RD_KAFKA_RESP_ERR_NOT_COORDINATOR_FOR_GROUP = 16,
    /** Invalid topic */
        RD_KAFKA_RESP_ERR_TOPIC_EXCEPTION = 17,
    /** Message batch larger than configured server segment size */
        RD_KAFKA_RESP_ERR_RECORD_LIST_TOO_LARGE = 18,
    /** Not enough in-sync replicas */
        RD_KAFKA_RESP_ERR_NOT_ENOUGH_REPLICAS = 19,
    /** Message(s) written to insufficient number of in-sync replicas */
        RD_KAFKA_RESP_ERR_NOT_ENOUGH_REPLICAS_AFTER_APPEND = 20,
    /** Invalid required acks value */
        RD_KAFKA_RESP_ERR_INVALID_REQUIRED_ACKS = 21,
    /** Specified group generation id is not valid */
        RD_KAFKA_RESP_ERR_ILLEGAL_GENERATION = 22,
    /** Inconsistent group protocol */
        RD_KAFKA_RESP_ERR_INCONSISTENT_GROUP_PROTOCOL = 23,
    /** Invalid group.id */
    RD_KAFKA_RESP_ERR_INVALID_GROUP_ID = 24,
    /** Unknown member */
        RD_KAFKA_RESP_ERR_UNKNOWN_MEMBER_ID = 25,
    /** Invalid session timeout */
        RD_KAFKA_RESP_ERR_INVALID_SESSION_TIMEOUT = 26,
    /** Group rebalance in progress */
    RD_KAFKA_RESP_ERR_REBALANCE_IN_PROGRESS = 27,
    /** Commit offset data size is not valid */
        RD_KAFKA_RESP_ERR_INVALID_COMMIT_OFFSET_SIZE = 28,
    /** Topic authorization failed */
        RD_KAFKA_RESP_ERR_TOPIC_AUTHORIZATION_FAILED = 29,
    /** Group authorization failed */
    RD_KAFKA_RESP_ERR_GROUP_AUTHORIZATION_FAILED = 30,
    /** Cluster authorization failed */
    RD_KAFKA_RESP_ERR_CLUSTER_AUTHORIZATION_FAILED = 31,
    /** Invalid timestamp */
    RD_KAFKA_RESP_ERR_INVALID_TIMESTAMP = 32,
    /** Unsupported SASL mechanism */
    RD_KAFKA_RESP_ERR_UNSUPPORTED_SASL_MECHANISM = 33,
    /** Illegal SASL state */
    RD_KAFKA_RESP_ERR_ILLEGAL_SASL_STATE = 34,
    /** Unuspported version */
    RD_KAFKA_RESP_ERR_UNSUPPORTED_VERSION = 35,
    /** Topic already exists */
    RD_KAFKA_RESP_ERR_TOPIC_ALREADY_EXISTS = 36,
    /** Invalid number of partitions */
    RD_KAFKA_RESP_ERR_INVALID_PARTITIONS = 37,
    /** Invalid replication factor */
    RD_KAFKA_RESP_ERR_INVALID_REPLICATION_FACTOR = 38,
    /** Invalid replica assignment */
    RD_KAFKA_RESP_ERR_INVALID_REPLICA_ASSIGNMENT = 39,
    /** Invalid config */
    RD_KAFKA_RESP_ERR_INVALID_CONFIG = 40,
    /** Not controller for cluster */
    RD_KAFKA_RESP_ERR_NOT_CONTROLLER = 41,
    /** Invalid request */
    RD_KAFKA_RESP_ERR_INVALID_REQUEST = 42,
    /** Message format on broker does not support request */
    RD_KAFKA_RESP_ERR_UNSUPPORTED_FOR_MESSAGE_FORMAT = 43,
        /** Isolation policy volation */
        RD_KAFKA_RESP_ERR_POLICY_VIOLATION = 44,
        /** Broker received an out of order sequence number */
        RD_KAFKA_RESP_ERR_OUT_OF_ORDER_SEQUENCE_NUMBER = 45,
        /** Broker received a duplicate sequence number */
        RD_KAFKA_RESP_ERR_DUPLICATE_SEQUENCE_NUMBER = 46,
        /** Producer attempted an operation with an old epoch */
        RD_KAFKA_RESP_ERR_INVALID_PRODUCER_EPOCH = 47,
```


last-updated: Apr 11, 2022

```
        /** Producer attempted a transactional operation in an invalid
state */
        RD_KAFKA_RESP_ERR_INVALID_TXN_STATE = 48,
        /** Producer attempted to use a producer id which is not
         *  currently assigned to its transactional id */
        RD_KAFKA_RESP_ERR_INVALID_PRODUCER_ID_MAPPING = 49,
        /** Transaction timeout is larger than the maximum
         *  value allowed by the broker's max.transaction.timeout.ms */
        RD_KAFKA_RESP_ERR_INVALID_TRANSACTION_TIMEOUT = 50,
        /** Producer attempted to update a transaction while another
         *  concurrent operation on the same transaction was ongoing */
        RD_KAFKA_RESP_ERR_CONCURRENT_TRANSACTIONS = 51,
        /** Indicates that the transaction coordinator sending a
         *  WriteTxnMarker is no longer the current coordinator for a
         *  given producer */
        RD_KAFKA_RESP_ERR_TRANSACTION_COORDINATOR_FENCED = 52,
        /** Transactional Id authorization failed */
        RD_KAFKA_RESP_ERR_TRANSACTIONAL_ID_AUTHORIZATION_FAILED = 53,
        /** Security features are disabled */
        RD_KAFKA_RESP_ERR_SECURITY_DISABLED = 54,
        /** Operation not attempted */
        RD_KAFKA_RESP_ERR_OPERATION_NOT_ATTEMPTED = 55,

    RD_KAFKA_RESP_ERR_END_ALL,
} rd_kafka_resp_err_t;


/**
 * @brief Error code value, name and description.
 *        Typically for use with language bindings to automatically expose
 *        the full set of librdkafka error codes.
 */
struct rd_kafka_err_desc {
    rd_kafka_resp_err_t code;/**< Error code */
    const char *name;      /**< Error name, same as code enum sans prefix */
    const char *desc;      /**< Human readable error description. */
};


/**
 * @brief Returns the full list of error codes.
 */
RD_EXPORT
void rd_kafka_get_err_descs (const struct rd_kafka_err_desc **errdescs,
                 size_t *cntp);




/**
 * @brief Returns a human readable representation of a kafka error.
 *
 * @param err Error code to translate
 */
RD_EXPORT
const char *rd_kafka_err2str (rd_kafka_resp_err_t err);



/**
 * @brief Returns the error code name (enum name).
 *
 * @param err Error code to translate
 */
```

```
RD_EXPORT
const char *rd_kafka_err2name (rd_kafka_resp_err_t err);


/**
 * @brief Returns the last error code generated by a legacy API call
 *        in the current thread.
 *
 * The legacy APIs are the ones using errno to propagate error value,
namely:
 *  - rd_kafka_topic_new()
 *  - rd_kafka_consume_start()
 *  - rd_kafka_consume_stop()
 *  - rd_kafka_consume()
 *  - rd_kafka_consume_batch()
 *  - rd_kafka_consume_callback()
 *  - rd_kafka_consume_queue()
 *  - rd_kafka_produce()
 *
 * The main use for this function is to avoid converting system \p errno
 * values to rd_kafka_resp_err_t codes for legacy APIs.
 *
 * @remark The last error is stored per-thread, if multiple rd_kafka_t
handles
 *         are used in the same application thread the developer needs to
 *         make sure rd_kafka_last_error() is called immediately after
 *         a failed API call.
 *
 * @remark errno propagation from librdkafka is not safe on Windows
 *         and should not be used, use rd_kafka_last_error() instead.
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_last_error (void);


/**
 * @brief Converts the system errno value \p errnox to a rd_kafka_resp_err_t
 *        error code upon failure from the following functions:
 *  - rd_kafka_topic_new()
 *  - rd_kafka_consume_start()
 *  - rd_kafka_consume_stop()
 *  - rd_kafka_consume()
 *  - rd_kafka_consume_batch()
 *  - rd_kafka_consume_callback()
 *  - rd_kafka_consume_queue()
 *  - rd_kafka_produce()
 *
 * @param errnox  System errno value to convert
 *
 * @returns Appropriate error code for \p errnox
 *
 * @remark A better alternative is to call rd_kafka_last_error() immediately
 *         after any of the above functions return -1 or NULL.
 *
 * @deprecated Use rd_kafka_last_error() to retrieve the last error code
 *             set by the legacy librdkafka APIs.
 *
 * @sa rd_kafka_last_error()
 */
RD_EXPORT RD_DEPRECATED
rd_kafka_resp_err_t rd_kafka_errno2err(int errnox);


/**
```

```
 * @brief Returns the thread-local system errno
 *
 * On most platforms this is the same as \p errno but in case of different
 * runtimes between library and application (e.g., Windows static DLLs)
 * this provides a means for exposing the errno librdkafka uses.
 *
 * @remark The value is local to the current calling thread.
 *
 * @deprecated Use rd_kafka_last_error() to retrieve the last error code
 *             set by the legacy librdkafka APIs.
 */
RD_EXPORT RD_DEPRECATED
int rd_kafka_errno (void);



/**
 * @brief Topic+Partition place holder
 *
 * Generic place holder for a Topic+Partition and its related information
 * used for multiple purposes:
 *   - consumer offset (see rd_kafka_commit(), et.al.)
 *   - group rebalancing callback (rd_kafka_conf_set_rebalance_cb())
 *   - offset commit result callback (rd_kafka_conf_set_offset_commit_cb())
 */

/**
 * @brief Generic place holder for a specific Topic+Partition.
 *
 * @sa rd_kafka_topic_partition_list_new()
 */
typedef struct rd_kafka_topic_partition_s {
        char        *topic;             /**< Topic name */
        int32_t      partition;         /**< Partition */
    int64_t      offset;            /**< Offset */
        void        *metadata;          /**< Metadata */
        size_t       metadata_size;     /**< Metadata size */
        void        *opaque;            /**< Application opaque */
        rd_kafka_resp_err_t err;        /**< Error code, depending on use.
*/
        void       *_private;           /**< INTERNAL USE ONLY,
                                          *   INITIALIZE TO ZERO, DO NOT
TOUCH */
} rd_kafka_topic_partition_t;


/**
 * @brief Destroy a rd_kafka_topic_partition_t.
 * @remark This must not be called for elements in a topic partition list.
 */
RD_EXPORT
void rd_kafka_topic_partition_destroy (rd_kafka_topic_partition_t *rktpar);


/**
 * @brief A growable list of Topic+Partitions.
 *
 */
typedef struct rd_kafka_topic_partition_list_s {
        int cnt;               /**< Current number of elements */
        int size;              /**< Current allocated size */
        rd_kafka_topic_partition_t *elems; /**< Element array[] */
} rd_kafka_topic_partition_list_t;
```

```
/**
 * @brief Create a new list/vector Topic+Partition container.
 *
 * @param size  Initial allocated size used when the expected number of
 *              elements is known or can be estimated.
 *              Avoids reallocation and possibly relocation of the
 *              elems array.
 *
 * @returns A newly allocated Topic+Partition list.
 *
 * @remark Use rd_kafka_topic_partition_list_destroy() to free all resources
 *         in use by a list and the list itself.
 * @sa     rd_kafka_topic_partition_list_add()
 */
RD_EXPORT
rd_kafka_topic_partition_list_t *rd_kafka_topic_partition_list_new (int
size);


/**
 * @brief Free all resources used by the list and the list itself.
 */
RD_EXPORT
void
rd_kafka_topic_partition_list_destroy (rd_kafka_topic_partition_list_t
*rkparlist);

/**
 * @brief Add topic+partition to list
 *
 * @param rktparlist List to extend
 * @param topic      Topic name (copied)
 * @param partition  Partition id
 *
 * @returns The object which can be used to fill in additionals fields.
 */
RD_EXPORT
rd_kafka_topic_partition_t *
rd_kafka_topic_partition_list_add (rd_kafka_topic_partition_list_t
*rktparlist,
                                   const char *topic, int32_t partition);


/**
 * @brief Add range of partitions from \p start to \p stop inclusive.
 *
 * @param rktparlist List to extend
 * @param topic      Topic name (copied)
 * @param start      Start partition of range
 * @param stop       Last partition of range (inclusive)
 */
RD_EXPORT
void
rd_kafka_topic_partition_list_add_range (rd_kafka_topic_partition_list_t
                                         *rktparlist,
                                         const char *topic,
                                         int32_t start, int32_t stop);



/**
 * @brief Delete partition from list.
 *
```


last-updated: Apr 11, 2022

```
 * @param rktparlist List to modify
 * @param topic      Topic name to match
 * @param partition  Partition to match
 *
 * @returns 1 if partition was found (and removed), else 0.
 *
 * @remark Any held indices to elems[] are unusable after this call returns
1.
 */
RD_EXPORT
int
rd_kafka_topic_partition_list_del (rd_kafka_topic_partition_list_t
*rktparlist,
                   const char *topic, int32_t partition);


/**
 * @brief Delete partition from list by elems[] index.
 *
 * @returns 1 if partition was found (and removed), else 0.
 *
 * @sa rd_kafka_topic_partition_list_del()
 */
RD_EXPORT
int
rd_kafka_topic_partition_list_del_by_idx (
    rd_kafka_topic_partition_list_t *rktparlist,
    int idx);


/**
 * @brief Make a copy of an existing list.
 *
 * @param src   The existing list to copy.
 *
 * @returns A new list fully populated to be identical to \p src
 */
RD_EXPORT
rd_kafka_topic_partition_list_t *
rd_kafka_topic_partition_list_copy (const rd_kafka_topic_partition_list_t
*src);




/**
 * @brief Set offset to \p offset for \p topic and \p partition
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or
 *          RD_KAFKA_RESP_ERR__UNKNOWN_PARTITION if \p partition was not
found
 *          in the list.
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_topic_partition_list_set_offset (
    rd_kafka_topic_partition_list_t *rktparlist,
    const char *topic, int32_t partition, int64_t offset);



/**
 * @brief Find element by \p topic and \p partition.
 *
 * @returns a pointer to the first matching element, or NULL if not found.
```

```
 */
RD_EXPORT
rd_kafka_topic_partition_t *
rd_kafka_topic_partition_list_find (rd_kafka_topic_partition_list_t
*rktparlist,
                    const char *topic, int32_t partition);


/**
 * @brief Sort list using comparator \p cmp.
 *
 * If \p cmp is NULL the default comparator will be used that
 * sorts by ascending topic name and partition.
 *
 * Not supported on MapR streams.
 *
 */
RD_EXPORT void
rd_kafka_topic_partition_list_sort (rd_kafka_topic_partition_list_t
*rktparlist,
                                    int (*cmp) (const void *a, const void
*b,
                                                void *opaque),
                                    void *opaque);


/**@}*/



/**
 * @name Var-arg tag types
 * @{
 *
 */

/**
 * @enum rd_kafka_vtype_t
 *
 * @brief Var-arg tag types
 *
 * @sa rd_kafka_producev()
 */
typedef enum rd_kafka_vtype_t {
        RD_KAFKA_VTYPE_END,       /**< va-arg sentinel */
        RD_KAFKA_VTYPE_TOPIC,     /**< (const char *) Topic name */
        RD_KAFKA_VTYPE_RKT,       /**< (rd_kafka_topic_t *) Topic handle */
        RD_KAFKA_VTYPE_PARTITION, /**< (int32_t) Partition */
        RD_KAFKA_VTYPE_VALUE,     /**< (void *, size_t) Message value
(payload)*/
        RD_KAFKA_VTYPE_KEY,       /**< (void *, size_t) Message key */
        RD_KAFKA_VTYPE_OPAQUE,    /**< (void *) Application opaque */
        RD_KAFKA_VTYPE_MSGFLAGS,  /**< (int) RD_KAFKA_MSG_F_.. flags */
        RD_KAFKA_VTYPE_TIMESTAMP, /**< (int64_t) Milliseconds since epoch
UTC */
} rd_kafka_vtype_t;


/**
 * @brief Convenience macros for rd_kafka_vtype_t that takes the
 *        correct arguments for each vtype.
 */

/*!
```


last-updated: Apr 11, 2022

```
 * va-arg end sentinel used to terminate the variable argument list
 */
#define RD_KAFKA_V_END RD_KAFKA_VTYPE_END

/*!
 * Topic name (const char *)
 */
#define RD_KAFKA_V_TOPIC(topic)                                         \
        _LRK_TYPECHECK(RD_KAFKA_VTYPE_TOPIC, const char *, topic),      \
        (const char *)topic
/*!
 * Topic object (rd_kafka_topic_t *)
 */
#define RD_KAFKA_V_RKT(rkt)                                             \
        _LRK_TYPECHECK(RD_KAFKA_VTYPE_RKT, rd_kafka_topic_t *, rkt),    \
        (rd_kafka_topic_t *)rkt
/*!
 * Partition (int32_t)
 */
#define RD_KAFKA_V_PARTITION(partition)                                 \
        _LRK_TYPECHECK(RD_KAFKA_VTYPE_PARTITION, int32_t, partition),   \
        (int32_t)partition
/*!
 * Message value/payload pointer and length (void *, size_t)
 */
#define RD_KAFKA_V_VALUE(VALUE,LEN)                                     \
        _LRK_TYPECHECK2(RD_KAFKA_VTYPE_VALUE, void *, VALUE, size_t, LEN), \
        (void *)VALUE, (size_t)LEN
/*!
 * Message key pointer and length (const void *, size_t)
 */
#define RD_KAFKA_V_KEY(KEY,LEN)                                         \
        _LRK_TYPECHECK2(RD_KAFKA_VTYPE_KEY, const void *, KEY, size_t,
LEN), \
        (void *)KEY, (size_t)LEN
/*!
 * Opaque pointer (void *)
 */
#define RD_KAFKA_V_OPAQUE(opaque)                                   \
        _LRK_TYPECHECK(RD_KAFKA_VTYPE_OPAQUE, void *, opaque),    \
        (void *)opaque
/*!
 * Message flags (int)
 * @sa RD_KAFKA_MSG_F_COPY, et.al.
 */
#define RD_KAFKA_V_MSGFLAGS(msgflags)                                 \
        _LRK_TYPECHECK(RD_KAFKA_VTYPE_MSGFLAGS, int, msgflags),       \
        (int)msgflags
/*!
 * Timestamp (int64_t)
 */
#define RD_KAFKA_V_TIMESTAMP(timestamp)                                 \
        _LRK_TYPECHECK(RD_KAFKA_VTYPE_TIMESTAMP, int64_t, timestamp),   \
        (int64_t)timestamp

/**@}*/


/**
 * @name Kafka messages
 * @{
 *
 */
```

```
// FIXME: This doesn't show up in docs for some reason
// "Compound rd_kafka_message_t is not documented."

/**
 * @brief A Kafka message as returned by the \c rd_kafka_consume*() family
 *        of functions as well as provided to the Producer \c dr_msg_cb().
 *
 * For the consumer this object has two purposes:
 *  - provide the application with a consumed message. (\c err == 0)
 *  - report per-topic+partition consumer errors (\c err != 0)
 *
 * The application must check \c err to decide what action to take.
 *
 * When the application is finished with a message it must call
 * rd_kafka_message_destroy() unless otherwise noted.
 */
typedef struct rd_kafka_message_s {
    rd_kafka_resp_err_t err;   /**< Non-zero for error signaling. */
    rd_kafka_topic_t *rkt;     /**< Topic */
    int32_t partition;         /**< Partition */
    void   *payload;           /**< Producer: original message payload.
                    * Consumer: Depends on the value of \c err :
                    * - \c err==0: Message payload.
                    * - \c err!=0: Error string */
    size_t  len;               /**< Depends on the value of \c err :
                    * - \c err==0: Message payload length
                    * - \c err!=0: Error string length */
    void   *key;               /**< Depends on the value of \c err :
                    * - \c err==0: Optional message key */
    size_t  key_len;           /**< Depends on the value of \c err :
                    * - \c err==0: Optional message key length*/
    int64_t offset;            /**< Consume:
                                    * - Message offset (or offset for error
                    *   if \c err!=0 if applicable).
                                    * - dr_msg_cb:
                                    *   Message offset assigned by broker.
                                    *   If \c produce.offset.report is set
then
                                    *   each message will have this field
set,
                                    *   otherwise only the last message in
                                    *   each produced internal batch will
                                    *   have this field set, otherwise 0. */
    void  *_private;           /**< Consume:
                    *  - rdkafka private pointer: DO NOT MODIFY
                    *  - dr_msg_cb:
                                    *    msg_opaque from produce() call */
  bool is_streams_message;
  bool is_dummy_message;           /** To be used only to report error*/
  streams_consumer_record_t *_streams_consumer_record;  /**< Streams record
                                  * associated with this message */
} rd_kafka_message_t;


/**
 * @brief Frees resources for \p rkmessage and hands ownership back to
rdkafka.
 */
RD_EXPORT
void rd_kafka_message_destroy(rd_kafka_message_t *rkmessage);
```


last-updated: Apr 11, 2022

```
/**
 * @brief Returns the error string for an errored rd_kafka_message_t or
NULL if
 *        there was no error.
 *
 * @remark This function MUST NOT be used with the producer.
 */
static RD_INLINE const char *
RD_UNUSED
rd_kafka_message_errstr(const rd_kafka_message_t *rkmessage) {
    if (!rkmessage->err)
        return NULL;

    if (rkmessage->payload)
        return (const char *)rkmessage->payload;

    return rd_kafka_err2str(rkmessage->err);
}



/**
 * @brief Returns the message timestamp for a consumed message.
 *
 * The timestamp is the number of milliseconds since the epoch (UTC).
 *
 * \p tstype (if not NULL) is updated to indicate the type of timestamp.
 *
 * @returns message timestamp, or -1 if not available.
 *
 * @remark Message timestamps require broker version 0.10.0 or later.
 */
RD_EXPORT
int64_t rd_kafka_message_timestamp (const rd_kafka_message_t *rkmessage,
                    rd_kafka_timestamp_type_t *tstype);



/**
 * @brief Returns the latency for a produced message measured from
 *        the produce() call.
 *
 * @returns the latency in microseconds, or -1 if not available.
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
int64_t rd_kafka_message_latency (const rd_kafka_message_t *rkmessage);


/**@}*/


/**
 * @name Configuration interface
 * @{
 *
 * @brief Main/global configuration property interface
 *
 */

/**
```

```
 * @enum rd_kafka_conf_res_t
 * @brief Configuration result type
 */
typedef enum {
    RD_KAFKA_CONF_UNKNOWN = -2, /**< Unknown configuration name. */
    RD_KAFKA_CONF_INVALID = -1, /**< Invalid configuration value. */
    RD_KAFKA_CONF_OK = 0        /**< Configuration okay */
} rd_kafka_conf_res_t;


/**
 * @brief Create configuration object.
 *
 * When providing your own configuration to the \c rd_kafka_*_new_*() calls
 * the rd_kafka_conf_t objects needs to be created with this function
 * which will set up the defaults.
 * I.e.:
 * @code
 *   rd_kafka_conf_t *myconf;
 *   rd_kafka_conf_res_t res;
 *
 *   myconf = rd_kafka_conf_new();
 *   res = rd_kafka_conf_set(myconf, "socket.timeout.ms", "600",
 *                           errstr, sizeof(errstr));
 *   if (res != RD_KAFKA_CONF_OK)
 *      die("%s\n", errstr);
 *
 *   rk = rd_kafka_new(..., myconf);
 * @endcode
 *
 * Please see CONFIGURATION.md for the default settings or use
 * rd_kafka_conf_properties_show() to provide the information at runtime.
 *
 * The properties are identical to the Apache Kafka configuration properties
 * whenever possible.
 *
 * @returns A new rd_kafka_conf_t object with defaults set.
 *
 * @sa rd_kafka_conf_set(), rd_kafka_conf_destroy()
 */
RD_EXPORT
rd_kafka_conf_t *rd_kafka_conf_new(void);


/**
 * @brief Destroys a conf object.
 */
RD_EXPORT
void rd_kafka_conf_destroy(rd_kafka_conf_t *conf);


/**
 * @brief Creates a copy/duplicate of configuration object \p conf
 *
 * @remark Interceptors are NOT copied to the new configuration object.
 * @sa rd_kafka_interceptor_f_on_conf_dup
 */
RD_EXPORT
rd_kafka_conf_t *rd_kafka_conf_dup(const rd_kafka_conf_t *conf);


/**
 * @brief Same as rd_kafka_conf_dup() but with an array of property name
 *        prefixes to filter out (ignore) when copying.
```

```
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
rd_kafka_conf_t *rd_kafka_conf_dup_filter (const rd_kafka_conf_t *conf,
                                           size_t filter_cnt,
                                           const char **filter);



/**
 * @brief Sets a configuration property.
 *
 * \p conf must have been previously created with rd_kafka_conf_new().
 *
 * Fallthrough:
 * Topic-level configuration properties may be set using this interface
 * in which case they are applied on the \c default_topic_conf.
 * If no \c default_topic_conf has been set one will be created.
 * Any sub-sequent rd_kafka_conf_set_default_topic_conf() calls will
 * replace the current default topic configuration.
 *
 * @returns \c rd_kafka_conf_res_t to indicate success or failure.
 * In case of failure \p errstr is updated to contain a human readable
 * error string.
 */
RD_EXPORT
rd_kafka_conf_res_t rd_kafka_conf_set(rd_kafka_conf_t *conf,
                      const char *name,
                      const char *value,
                      char *errstr, size_t errstr_size);


/**
 * @brief Enable event sourcing.
 * \p events is a bitmask of \c RD_KAFKA_EVENT_* of events to enable
 * for consumption by `rd_kafka_queue_poll()`.
 */
RD_EXPORT
void rd_kafka_conf_set_events(rd_kafka_conf_t *conf, int events);


/**
 @deprecated See rd_kafka_conf_set_dr_msg_cb()
 Not supported on MapR streams.
*/
RD_EXPORT
void rd_kafka_conf_set_dr_cb(rd_kafka_conf_t *conf,
                  void (*dr_cb) (rd_kafka_t *rk,
                        void *payload, size_t len,
                        rd_kafka_resp_err_t err,
                        void *opaque, void *msg_opaque));

/**
 * @brief \b Producer: Set delivery report callback in provided \p conf
object.
 *
 * The delivery report callback will be called once for each message
 * accepted by rd_kafka_produce() (et.al) with \p err set to indicate
 * the result of the produce request.
 *
 * The callback is called when a message is succesfully produced or
 * if librdkafka encountered a permanent failure, or the retry counter for
 * temporary errors has been exhausted.
```

```
 *
 * An application must call rd_kafka_poll() at regular intervals to
 * serve queued delivery report callbacks.
 */
RD_EXPORT
void rd_kafka_conf_set_dr_msg_cb(rd_kafka_conf_t *conf,
                                 void (*dr_msg_cb) (rd_kafka_t *rk,
                                                    const
rd_kafka_message_t *
                                                    rkmessage,
                                                    void *opaque));


/**
 * @brief \b Consumer: Set consume callback for use with
rd_kafka_consumer_poll()
 *
 */
RD_EXPORT
void rd_kafka_conf_set_consume_cb (rd_kafka_conf_t *conf,
                                   void (*consume_cb) (rd_kafka_message_t *
                                                       rkmessage,
                                                       void *opaque));

/**
 * @brief \b Consumer: Set rebalance callback for use with
 *                     coordinated consumer group balancing.
 *
 * The \p err field is set to either RD_KAFKA_RESP_ERR__ASSIGN_PARTITIONS
 * or RD_KAFKA_RESP_ERR__REVOKE_PARTITIONS and 'partitions'
 * contains the full partition set that was either assigned or revoked.
 *
 * Registering a \p rebalance_cb turns off librdkafka's automatic
 * partition assignment/revocation and instead delegates that responsibility
 * to the application's \p rebalance_cb.
 *
 * The rebalance callback is responsible for updating librdkafka's
 * assignment set based on the two events:
RD_KAFKA_RESP_ERR__ASSIGN_PARTITIONS
 * and RD_KAFKA_RESP_ERR__REVOKE_PARTITIONS but should also be able to
handle
 * arbitrary rebalancing failures where \p err is neither of those.
 * @remark In this latter case (arbitrary error), the application must
 *         call rd_kafka_assign(rk, NULL) to synchronize state.
 *
 * Without a rebalance callback this is done automatically by librdkafka
 * but registering a rebalance callback gives the application flexibility
 * in performing other operations along with the assinging/revocation,
 * such as fetching offsets from an alternate location (on assign)
 * or manually committing offsets (on revoke).
 *
 * @remark The \p partitions list is destroyed by librdkafka on return
 *         return from the rebalance_cb and must not be freed or
 *         saved by the application.
 *
 * The following example shows the application's responsibilities:
 * @code
 *    static void rebalance_cb (rd_kafka_t *rk, rd_kafka_resp_err_t err,
 *                              rd_kafka_topic_partition_list_t *partitions,
 *                              void *opaque) {
 *
 *        switch (err)
 *        {
 *          case RD_KAFKA_RESP_ERR__ASSIGN_PARTITIONS:
```

```
 *              // application may load offets from arbitrary external
 *              // storage here and update \p partitions
 *
 *              rd_kafka_assign(rk, partitions);
 *              break;
 *
 *          case RD_KAFKA_RESP_ERR__REVOKE_PARTITIONS:
 *              if (manual_commits) // Optional explicit manual commit
 *                  rd_kafka_commit(rk, partitions, 0); // sync commit
 *
 *              rd_kafka_assign(rk, NULL);
 *              break;
 *
 *          default:
 *              handle_unlikely_error(err);
 *              rd_kafka_assign(rk, NULL); // sync state
 *              break;
 *         }
 *    }
 * @endcode
 */
RD_EXPORT
void rd_kafka_conf_set_rebalance_cb (
        rd_kafka_conf_t *conf,
        void (*rebalance_cb) (rd_kafka_t *rk,
                              rd_kafka_resp_err_t err,
                              rd_kafka_topic_partition_list_t *partitions,
                              void *opaque));



/**
 * @brief \b Consumer: Set offset commit callback for use with consumer
groups.
 *
 * The results of automatic or manual offset commits will be scheduled
 * for this callback and is served by rd_kafka_consumer_poll().
 *
 * If no partitions had valid offsets to commit this callback will be called
 * with \p err == RD_KAFKA_RESP_ERR__NO_OFFSET which is not to be considered
 * an error.
 *
 * The \p offsets list contains per-partition information:
 *   - \c offset: committed offset (attempted)
 *   - \c err:    commit error
 */
RD_EXPORT
void rd_kafka_conf_set_offset_commit_cb (
        rd_kafka_conf_t *conf,
        void (*offset_commit_cb) (rd_kafka_t *rk,
                                  rd_kafka_resp_err_t err,
                                  rd_kafka_topic_partition_list_t *offsets,
                                  void *opaque));


/**
 * @brief Set error callback in provided conf object.
 *
 * The error callback is used by librdkafka to signal critical errors
 * back to the application.
 *
 * If no \p error_cb is registered then the errors will be logged instead.
 */
RD_EXPORT
```

```
void rd_kafka_conf_set_error_cb(rd_kafka_conf_t *conf,
                 void  (*error_cb) (rd_kafka_t *rk, int err,
                           const char *reason,
                           void *opaque));

/**
 * @brief Set throttle callback.
 *
 * The throttle callback is used to forward broker throttle times to the
 * application for Produce and Fetch (consume) requests.
 *
 * Callbacks are triggered whenever a non-zero throttle time is returned by
 * the broker, or when the throttle time drops back to zero.
 *
 * An application must call rd_kafka_poll() or rd_kafka_consumer_poll() at
 * regular intervals to serve queued callbacks.
 *
 * @remark Requires broker version 0.9.0 or later.
 * Not supported on MapR streams.
 */
RD_EXPORT
void rd_kafka_conf_set_throttle_cb (rd_kafka_conf_t *conf,
                    void (*throttle_cb) (
                        rd_kafka_t *rk,
                        const char *broker_name,
                        int32_t broker_id,
                        int throttle_time_ms,
                        void *opaque));


/**
 * @brief Set logger callback.
 *
 * The default is to print to stderr, but a syslog logger is also available,
 * see rd_kafka_log_print and rd_kafka_log_syslog for the builtin
alternatives.
 * Alternatively the application may provide its own logger callback.
 * Or pass \p func as NULL to disable logging.
 *
 * This is the configuration alternative to the deprecated
rd_kafka_set_logger()
 *
 * @remark The log_cb will be called spontaneously from librdkafka's
internal
 *         threads unless logs have been forwarded to a poll queue through
 *         \c rd_kafka_set_log_queue().
 *         An application MUST NOT call any librdkafka APIs or do any
prolonged
 *         work in a non-forwarded \c log_cb.
 * Not supported on MapR streams.
 */
RD_EXPORT
void rd_kafka_conf_set_log_cb(rd_kafka_conf_t *conf,
              void (*log_cb) (const rd_kafka_t *rk, int level,
                                          const char *fac, const char
*buf));


/**
 * @brief Set statistics callback in provided conf object.
 *
 * The statistics callback is triggered from rd_kafka_poll() every
 * \c statistics.interval.ms (needs to be configured separately).
 * Function arguments:
```

```
 *   - \p rk - Kafka handle
 *   - \p json - String containing the statistics data in JSON format
 *   - \p json_len - Length of \p json string.
 *   - \p opaque - Application-provided opaque.
 *
 * If the application wishes to hold on to the \p json pointer and free
 * it at a later time it must return 1 from the \p stats_cb.
 * If the application returns 0 from the \p stats_cb then librdkafka
 * will immediately free the \p json pointer.
 * Not supported on MapR streams.
 */
RD_EXPORT
void rd_kafka_conf_set_stats_cb(rd_kafka_conf_t *conf,
                  int (*stats_cb) (rd_kafka_t *rk,
                           char *json,
                           size_t json_len,
                           void *opaque));



/**
 * @brief Set socket callback.
 *
 * The socket callback is responsible for opening a socket
 * according to the supplied \p domain, \p type and \p protocol.
 * The socket shall be created with \c CLOEXEC set in a racefree fashion, if
 * possible.
 *
 * Default:
 *  - on linux: racefree CLOEXEC
 *  - others  : non-racefree CLOEXEC
 *
 * @remark The callback will be called from an internal librdkafka thread.
 * Not supported on MapR streams.
 */
RD_EXPORT
void rd_kafka_conf_set_socket_cb(rd_kafka_conf_t *conf,
                                 int (*socket_cb) (int domain, int type,
                                                   int protocol,
                                                   void *opaque));



/**
 * @brief Set connect callback.
 *
 * The connect callback is responsible for connecting socket \p sockfd
 * to peer address \p addr.
 * The \p id field contains the broker identifier.
 *
 * \p connect_cb shall return 0 on success (socket connected) or an error
 * number (errno) on error.
 *
 * @remark The callback will be called from an internal librdkafka thread.
 * Not supported on MapR streams.
 */
RD_EXPORT void
rd_kafka_conf_set_connect_cb (rd_kafka_conf_t *conf,
                              int (*connect_cb) (int sockfd,
                                                 const struct sockaddr
*addr,
                                                 int addrlen,
                                                 const char *id,
                                                 void *opaque));
```

```
/**
 * @brief Set close socket callback.
 *
 * Close a socket (optionally opened with socket_cb()).
 *
 * @remark The callback will be called from an internal librdkafka thread.
 * Not supported on MapR streams.
 */
RD_EXPORT void
rd_kafka_conf_set_closesocket_cb (rd_kafka_conf_t *conf,
                                  int (*closesocket_cb) (int sockfd,
                                                         void *opaque));



#ifndef _MSC_VER
/**
 * @brief Set open callback.
 *
 * The open callback is responsible for opening the file specified by
 * pathname, flags and mode.
 * The file shall be opened with \c CLOEXEC set in a racefree fashion, if
 * possible.
 *
 * Default:
 *  - on linux: racefree CLOEXEC
 *  - others  : non-racefree CLOEXEC
 *
 * @remark The callback will be called from an internal librdkafka thread.
 * Not supported on MapR streams.
 */
RD_EXPORT
void rd_kafka_conf_set_open_cb (rd_kafka_conf_t *conf,
                                int (*open_cb) (const char *pathname,
                                                int flags, mode_t mode,
                                                void *opaque));
#endif

/**
 * @brief Sets the application's opaque pointer that will be passed to
callbacks
 */
RD_EXPORT
void rd_kafka_conf_set_opaque(rd_kafka_conf_t *conf, void *opaque);

/**
 * @brief Retrieves the opaque pointer previously set with
rd_kafka_conf_set_opaque()
 */
RD_EXPORT
void *rd_kafka_opaque(const rd_kafka_t *rk);



/**
 * Sets the default topic configuration to use for automatically
 * subscribed topics (e.g., through pattern-matched topics).
 * The topic config object is not usable after this call.
 */
RD_EXPORT
void rd_kafka_conf_set_default_topic_conf (rd_kafka_conf_t *conf,
                                           rd_kafka_topic_conf_t *tconf);
```

```
/**
 * @brief Retrieve configuration value for property \p name.
 *
 * If \p dest is non-NULL the value will be written to \p dest with at
 * most \p dest_size.
 *
 * \p *dest_size is updated to the full length of the value, thus if
 * \p *dest_size initially is smaller than the full length the application
 * may reallocate \p dest to fit the returned \p *dest_size and try again.
 *
 * If \p dest is NULL only the full length of the value is returned.
 *
 * Fallthrough:
 * Topic-level configuration properties from the \c default_topic_conf
 * may be retrieved using this interface.
 *
 * @returns \p RD_KAFKA_CONF_OK if the property name matched, else
 * \p RD_KAFKA_CONF_UNKNOWN.
 */
RD_EXPORT
rd_kafka_conf_res_t rd_kafka_conf_get (const rd_kafka_conf_t *conf,
                                       const char *name,
                                       char *dest, size_t *dest_size);


/**
 * @brief Retrieve topic configuration value for property \p name.
 *
 * @sa rd_kafka_conf_get()
 */
RD_EXPORT
rd_kafka_conf_res_t rd_kafka_topic_conf_get (const rd_kafka_topic_conf_t
*conf,
                                             const char *name,
                                             char *dest, size_t *dest_size);


/**
 * @brief Dump the configuration properties and values of \p conf to an
array
 *        with \"key\", \"value\" pairs.
 *
 * The number of entries in the array is returned in \p *cntp.
 *
 * The dump must be freed with `rd_kafka_conf_dump_free()`.
 * Not supported on MapR streams.
 */
RD_EXPORT
const char **rd_kafka_conf_dump(rd_kafka_conf_t *conf, size_t *cntp);


/**
 * @brief Dump the topic configuration properties and values of \p conf
 *        to an array with \"key\", \"value\" pairs.
 *
 * The number of entries in the array is returned in \p *cntp.
 *
 * The dump must be freed with `rd_kafka_conf_dump_free()`.
 */
RD_EXPORT
const char **rd_kafka_topic_conf_dump(rd_kafka_topic_conf_t *conf,
                      size_t *cntp);
```

```
/**
 * @brief Frees a configuration dump returned from `rd_kafka_conf_dump()` or
 *        `rd_kafka_topic_conf_dump().
 * Not supported on MapR streams.
 */
RD_EXPORT
void rd_kafka_conf_dump_free(const char **arr, size_t cnt);

/**
 * @brief Prints a table to \p fp of all supported configuration properties,
 *        their default values as well as a description.
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
void rd_kafka_conf_properties_show(FILE *fp);

/**@}*/


/**
 * @name Topic configuration
 * @{
 *
 * @brief Topic configuration property interface
 *
 */


/**
 * @brief Create topic configuration object
 *
 * @sa Same semantics as for rd_kafka_conf_new().
 */
RD_EXPORT
rd_kafka_topic_conf_t *rd_kafka_topic_conf_new(void);


/**
 * @brief Creates a copy/duplicate of topic configuration object \p conf.
 */
RD_EXPORT
rd_kafka_topic_conf_t *rd_kafka_topic_conf_dup(const rd_kafka_topic_conf_t
                        *conf);


/**
 * @brief Destroys a topic conf object.
 */
RD_EXPORT
void rd_kafka_topic_conf_destroy(rd_kafka_topic_conf_t *topic_conf);


/**
 * @brief Sets a single rd_kafka_topic_conf_t value by property name.
 *
 * \p topic_conf should have been previously set up
 * with `rd_kafka_topic_conf_new()`.
 *
 * @returns rd_kafka_conf_res_t to indicate success or failure.
 */
RD_EXPORT
rd_kafka_conf_res_t rd_kafka_topic_conf_set(rd_kafka_topic_conf_t *conf,
```

```
                          const char *name,
                          const char *value,
                          char *errstr, size_t errstr_size);

/**
 * @brief Sets the application's opaque pointer that will be passed to all
topic
 * callbacks as the \c rkt_opaque argument.
 */
RD_EXPORT
void rd_kafka_topic_conf_set_opaque(rd_kafka_topic_conf_t *conf, void
*opaque);


/**
 * @brief \b Producer: Set partitioner callback in provided topic conf
object.
 *
 * The partitioner may be called in any thread at any time,
 * it may be called multiple times for the same message/key.
 *
 * Partitioner function constraints:
 *   - MUST NOT call any rd_kafka_*() functions except:
 *       rd_kafka_topic_partition_available()
 *   - MUST NOT block or execute for prolonged periods of time.
 *   - MUST return a value between 0 and partition_cnt-1, or the
 *     special \c RD_KAFKA_PARTITION_UA value if partitioning
 *     could not be performed.
 */
RD_EXPORT
void
rd_kafka_topic_conf_set_partitioner_cb (rd_kafka_topic_conf_t *topic_conf,
                    int32_t (*partitioner) (
                        const rd_kafka_topic_t *rkt,
                        const void *keydata,
                        size_t keylen,
                        int32_t partition_cnt,
                        void *rkt_opaque,
                        void *msg_opaque));

/**
 * @brief Check if partition is available (has a leader broker).
 *
 * @returns 1 if the partition is available, else 0.
 *
 * @warning This function must only be called from inside a partitioner
function
 */
RD_EXPORT
int rd_kafka_topic_partition_available(const rd_kafka_topic_t *rkt,
                    int32_t partition);


/*******************************************************************
 *                                                                 *
 * Partitioners provided by rdkafka                                *
 *                                                                 *
 *******************************************************************/

/**
 * @brief Random partitioner.
 *
 * Will try not to return unavailable partitions.
 *
```

```
 * @returns a random partition between 0 and \p partition_cnt - 1.
 *
 */
RD_EXPORT
int32_t rd_kafka_msg_partitioner_random(const rd_kafka_topic_t *rkt,
                     const void *key, size_t keylen,
                     int32_t partition_cnt,
                     void *opaque, void *msg_opaque);

/**
 * @brief Consistent partitioner.
 *
 * Uses consistent hashing to map identical keys onto identical partitions.
 *
 * @returns a \"random\" partition between 0 and \p partition_cnt - 1 based
on
 *          the CRC value of the key
 */
RD_EXPORT
int32_t rd_kafka_msg_partitioner_consistent (const rd_kafka_topic_t *rkt,
                     const void *key, size_t keylen,
                     int32_t partition_cnt,
                     void *opaque, void *msg_opaque);

/**
 * @brief Consistent-Random partitioner.
 *
 * This is the default partitioner.
 * Uses consistent hashing to map identical keys onto identical partitions,
and
 * messages without keys will be assigned via the random partitioner.
 *
 * @returns a \"random\" partition between 0 and \p partition_cnt - 1 based
on
 *          the CRC value of the key (if provided)
 */
RD_EXPORT
int32_t rd_kafka_msg_partitioner_consistent_random (const rd_kafka_topic_t
*rkt,
           const void *key, size_t keylen,
           int32_t partition_cnt,
           void *opaque, void *msg_opaque);


/**@}*/



/**
 * @name Main Kafka and Topic object handles
 * @{
 *
 *
 */




/**
 * @brief Creates a new Kafka handle and starts its operation according to
the
 *        specified \p type (\p RD_KAFKA_CONSUMER or \p RD_KAFKA_PRODUCER).
 *
 * \p conf is an optional struct created with `rd_kafka_conf_new()` that
```

```
will
 * be used instead of the default configuration.
 * The \p conf object is freed by this function on success and must not be
used
 * or destroyed by the application sub-sequently.
 * See `rd_kafka_conf_set()` et.al for more information.
 *
 * \p errstr must be a pointer to memory of at least size \p errstr_size
where
 * `rd_kafka_new()` may write a human readable error message in case the
 * creation of a new handle fails. In which case the function returns NULL.
 *
 * @remark \b RD_KAFKA_CONSUMER: When a new \p RD_KAFKA_CONSUMER
 *           rd_kafka_t handle is created it may either operate in the
 *           legacy simple consumer mode using the rd_kafka_consume_start()
 *           interface, or the High-level KafkaConsumer API.
 * @remark An application must only use one of these groups of APIs on a
given
 *         rd_kafka_t RD_KAFKA_CONSUMER handle.
 *
 * @returns The Kafka handle on success or NULL on error (see \p errstr)
 *
 * @sa To destroy the Kafka handle, use rd_kafka_destroy().
 */
RD_EXPORT
rd_kafka_t *rd_kafka_new(rd_kafka_type_t type, rd_kafka_conf_t *conf,
              char *errstr, size_t errstr_size);


/**
 * @brief Destroy Kafka handle.
 *
 * @remark This is a blocking operation.
 */
RD_EXPORT
void        rd_kafka_destroy(rd_kafka_t *rk);



/**
 * @brief Returns Kafka handle name.
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
const char *rd_kafka_name(const rd_kafka_t *rk);


/**
 * @brief Returns Kafka handle type.
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
rd_kafka_type_t rd_kafka_type(const rd_kafka_t *rk);


/**
 * @brief Returns this client's broker-assigned group member id
 *
 * @remark This currently requires the high-level KafkaConsumer
 *
 * @returns An allocated string containing the current broker-assigned group
 *           member id, or NULL if not available.
```

```
 *          The application must free the string with \p free() or
 *          rd_kafka_mem_free()
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
char *rd_kafka_memberid (const rd_kafka_t *rk);



/**
 * @brief Returns the ClusterId as reported in broker metadata.
 *
 * @param timeout_ms If there is no cached value from metadata retrieval
 *                   then this specifies the maximum amount of time
 *                   (in milliseconds) the call will block waiting
 *                   for metadata to be retrieved.
 *                   Use 0 for non-blocking calls.
 * @remark Requires broker version >=0.10.0 and api.version.request=true.
 *
 * @remark The application must free the returned pointer
 *         using rd_kafka_mem_free().
 *
 * @returns a newly allocated string containing the ClusterId, or NULL
 *          if no ClusterId could be retrieved in the allotted timespan.
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
char *rd_kafka_clusterid (rd_kafka_t *rk, int timeout_ms);


/**
 * @brief Creates a new topic handle for topic named \p topic.
 *
 * \p conf is an optional configuration for the topic created with
 * `rd_kafka_topic_conf_new()` that will be used instead of the default
 * topic configuration.
 * The \p conf object is freed by this function and must not be used or
 * destroyed by the application sub-sequently.
 * See `rd_kafka_topic_conf_set()` et.al for more information.
 *
 * Topic handles are refcounted internally and calling rd_kafka_topic_new()
 * again with the same topic name will return the previous topic handle
 * without updating the original handle's configuration.
 * Applications must eventually call rd_kafka_topic_destroy() for each
 * succesfull call to rd_kafka_topic_new() to clear up resources.
 *
 * @returns the new topic handle or NULL on error (use rd_kafka_errno2err()
 *          to convert system \p errno to an rd_kafka_resp_err_t error code.
 *
 * @sa rd_kafka_topic_destroy()
 */
RD_EXPORT
rd_kafka_topic_t *rd_kafka_topic_new(rd_kafka_t *rk, const char *topic,
                     rd_kafka_topic_conf_t *conf);



/**
 * @brief Loose application's topic handle refcount as previously created
 *        with `rd_kafka_topic_new()`.
 *
 * @remark Since topic objects are refcounted (both internally and for the
```

```
app)
 *         the topic object might not actually be destroyed by this call,
 *         but the application must consider the object destroyed.
 */
RD_EXPORT
void rd_kafka_topic_destroy(rd_kafka_topic_t *rkt);


/**
 * @brief Returns the topic name.
 */
RD_EXPORT
const char *rd_kafka_topic_name(const rd_kafka_topic_t *rkt);


/**
 * @brief Get the \p rkt_opaque pointer that was set in the topic
configuration.
 */
RD_EXPORT
void *rd_kafka_topic_opaque (const rd_kafka_topic_t *rkt);


/**
 * @brief Unassigned partition.
 *
 * The unassigned partition is used by the producer API for messages
 * that should be partitioned using the configured or default partitioner.
 */
#define RD_KAFKA_PARTITION_UA  ((int32_t)-1)


/**
 * @brief Polls the provided kafka handle for events.
 *
 * Events will cause application provided callbacks to be called.
 *
 * The \p timeout_ms argument specifies the maximum amount of time
 * (in milliseconds) that the call will block waiting for events.
 * For non-blocking calls, provide 0 as \p timeout_ms.
 * To wait indefinately for an event, provide -1.
 *
 * @remark  An application should make sure to call poll() at regular
 *          intervals to serve any queued callbacks waiting to be called.
 *
 * Events:
 *   - delivery report callbacks  (if dr_cb/dr_msg_cb is configured)
[producer]
 *   - error callbacks (rd_kafka_conf_set_error_cb()) [all]
 *   - stats callbacks (rd_kafka_conf_set_stats_cb()) [all]
 *   - throttle callbacks (rd_kafka_conf_set_throttle_cb()) [all]
 *
 * @returns the number of events served.
 */
RD_EXPORT
int rd_kafka_poll(rd_kafka_t *rk, int timeout_ms);


/**
 * @brief Cancels the current callback dispatcher (rd_kafka_poll(),
 *        rd_kafka_consume_callback(), etc).
 *
 * A callback may use this to force an immediate return to the calling
 * code (caller of e.g. rd_kafka_poll()) without processing any further
```

```
 * events.
 *
 * @remark This function MUST ONLY be called from within a librdkafka
callback.
 */
RD_EXPORT
void rd_kafka_yield (rd_kafka_t *rk);



/**
 * @brief Pause producing or consumption for the provided list of
partitions.
 *
 * Success or error is returned per-partition \p err in the \p partitions
list.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR
 * RD_KAFKA_RESP_ERR__UNSUPPORTED_FEATURE on MapR streams.
 *
 * Not supported on MapR streams.
 *
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_pause_partitions (rd_kafka_t *rk,
               rd_kafka_topic_partition_list_t *partitions);



/**
 * @brief Resume producing consumption for the provided list of partitions.
 *
 * Success or error is returned per-partition \p err in the \p partitions
list.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR
 * RD_KAFKA_RESP_ERR__UNSUPPORTED_FEATURE on MapR streams.
 *
 * Not supported on MapR streams.
 *
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_resume_partitions (rd_kafka_t *rk,
                rd_kafka_topic_partition_list_t *partitions);



/**
 * @brief Query broker for low (oldest/beginning) and high (newest/end)
offsets
 *        for partition.
 *
 * Offsets are returned in \p *low and \p *high respectively.
 * For Mapr Streams this function will block for at most \p timeout_ms
milliseconds.
 * Min timeout_ms is 30 sec and this api adjusts it if provided timeout_ms
is
 * less than 30 sec
 * This API supports streams.consumer.default.stream config
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or an error code on
failure.
 */
```

```
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_query_watermark_offsets (rd_kafka_t *rk,
              const char *topic, int32_t partition,
              int64_t *low, int64_t *high, int timeout_ms);


/**
 * @brief Get last known low (oldest/beginning) and high (newest/end)
offsets
 *        for partition.
 *
 * The low offset is updated periodically (if statistics.interval.ms is set)
 * while the high offset is updated on each fetched message set from the
broker.
 *
 * If there is no cached offset (either low or high, or both) then
 * RD_KAFKA_OFFSET_INVALID will be returned for the respective offset.
 *
 * For Mapr Streams this function will block for at most 30sec (RPC
timeout).
 * Offsets are returned in \p *low and \p *high respectively.
 * This API supports streams.consumer.default.stream config
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or an error code on
failure.
 *
 * @remark Shall only be used with an active consumer instance.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_get_watermark_offsets (rd_kafka_t *rk,
                const char *topic, int32_t partition,
                int64_t *low, int64_t *high);



/**
 * @brief Look up the offsets for the given partitions by timestamp.
 *
 * The returned offset for each partition is the earliest offset whose
 * timestamp is greater than or equal to the given timestamp in the
 * corresponding partition.
 *
 * The timestamps to query are represented as \c offset in \p offsets
 * on input, and \c offset will contain the offset on output.
 *
 * The function will block for at most \p timeout_ms milliseconds.
 * For mapr streams min timeout_ms is 30 sec and this api adjusts it
 * if provided timeout_ms is less than 30 sec
 *
 * @remark Duplicate Topic+Partitions are not supported.
 * @remark Per-partition errors may be returned in \c
rd_kafka_topic_partition_t.err
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR if offsets were be queried (do note
 *          that per-partition errors might be set),
 *          RD_KAFKA_RESP_ERR__TIMED_OUT if not all offsets could be fetched
 *          within \p timeout_ms,
 *          RD_KAFKA_RESP_ERR__INVALID_ARG if the \p offsets list is empty,
 *          RD_KAFKA_RESP_ERR__UNKNOWN_PARTITION if all partitions are
unknown,
 *          RD_KAFKA_RESP_ERR_LEADER_NOT_AVAILABLE if unable to query
leaders
 *          for the given partitions.
 */
```

```
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_offsets_for_times (rd_kafka_t *rk,
                            rd_kafka_topic_partition_list_t *offsets,
                            int timeout_ms);


/**
 * @brief Free pointer returned by librdkafka
 *
 * This is typically an abstraction for the free(3) call and makes sure
 * the application can use the same memory allocator as librdkafka for
 * freeing pointers returned by librdkafka.
 *
 * In standard setups it is usually not necessary to use this interface
 * rather than the free(3) functione.
 *
 * @remark rd_kafka_mem_free() must only be used for pointers returned by
APIs
 *         that explicitly mention using this function for freeing.
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
void rd_kafka_mem_free (rd_kafka_t *rk, void *ptr);


/**@}*/




/**
 * @name Queue API
 * @{
 *
 * Message queues allows the application to re-route consumed messages
 * from multiple topic+partitions into one single queue point.
 * This queue point containing messages from a number of topic+partitions
 * may then be served by a single rd_kafka_consume*_queue() call,
 * rather than one call per topic+partition combination.
 */


/**
 * @brief Create a new message queue.
 *
 * See rd_kafka_consume_start_queue(), rd_kafka_consume_queue(), et.al.
 */
RD_EXPORT
rd_kafka_queue_t *rd_kafka_queue_new(rd_kafka_t *rk);

/**
 * Destroy a queue, purging all of its enqueued messages.
 */
RD_EXPORT
void rd_kafka_queue_destroy(rd_kafka_queue_t *rkqu);


/**
 * @returns a reference to the main librdkafka event queue.
 * This is the queue served by rd_kafka_poll().
 *
 * Use rd_kafka_queue_destroy() to loose the reference.
```

```
 */
RD_EXPORT
rd_kafka_queue_t *rd_kafka_queue_get_main (rd_kafka_t *rk);


/**
 * @returns a reference to the librdkafka consumer queue.
 * This is the queue served by rd_kafka_consumer_poll().
 *
 * Use rd_kafka_queue_destroy() to loose the reference.
 *
 * @remark rd_kafka_queue_destroy() MUST be called on this queue
 *         prior to calling rd_kafka_consumer_close().
 */
RD_EXPORT
rd_kafka_queue_t *rd_kafka_queue_get_consumer (rd_kafka_t *rk);

/**
 * @returns a reference to the partition's queue, or NULL if
 *          partition is invalid.
 *
 * Use rd_kafka_queue_destroy() to loose the reference.
 *
 * @remark rd_kafka_queue_destroy() MUST be called on this queue
 *
 * @remark This function only works on consumers.
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
rd_kafka_queue_t *rd_kafka_queue_get_partition (rd_kafka_t *rk,
                                                const char *topic,
                                                int32_t partition);

/**
 * @brief Forward/re-route queue \p src to \p dst.
 * If \p dst is \c NULL the forwarding is removed.
 *
 * The internal refcounts for both queues are increased.
 *
 * @remark Regardless of whether \p dst is NULL or not, after calling this
 *         function, \p src will not forward it's fetch queue to the
consumer
 *         queue.
 */
RD_EXPORT
void rd_kafka_queue_forward (rd_kafka_queue_t *src, rd_kafka_queue_t *dst);

/**
 * @brief Forward librdkafka logs (and debug) to the specified queue
 *        for serving with one of the ..poll() calls.
 *
 *        This allows an application to serve log callbacks (\c log_cb)
 *        in its thread of choice.
 *
 * @param rkqu Queue to forward logs to. If the value is NULL the logs
 *        are forwarded to the main queue.
 *
 * @remark The configuration property \c log.queue MUST also be set to true.
 *
 * @remark librdkafka maintains its own reference to the provided queue.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or an error code on error.
 *
```

```
 * Not supported on MapR streams.
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_set_log_queue (rd_kafka_t *rk,
                                            rd_kafka_queue_t *rkqu);


/**
 * @returns the current number of elements in queue.
 */
RD_EXPORT
size_t rd_kafka_queue_length (rd_kafka_queue_t *rkqu);


/**
 * @brief Enable IO event triggering for queue.
 *
 * To ease integration with IO based polling loops this API
 * allows an application to create a separate file-descriptor
 * that librdkafka will write \p payload (of size \p size) to
 * whenever a new element is enqueued on a previously empty queue.
 *
 * To remove event triggering call with \p fd = -1.
 *
 * librdkafka will maintain a copy of the \p payload.
 *
 * @remark When using forwarded queues the IO event must only be enabled
 *         on the final forwarded-to (destination) queue.
 */
RD_EXPORT
void rd_kafka_queue_io_event_enable (rd_kafka_queue_t *rkqu, int fd,
                     const void *payload, size_t size);

/**@}*/

/**
 *
 * @name Simple Consumer API (legacy)
 * @{
 *
 */


#define RD_KAFKA_OFFSET_BEGINNING -2  /**< Start consuming from beginning of
                        *   kafka partition queue: oldest msg */
#define RD_KAFKA_OFFSET_END       -1  /**< Start consuming from end of kafka
                        *   partition queue: next msg */
#define RD_KAFKA_OFFSET_STORED -1000  /**< Start consuming from offset
retrieved
                        *   from offset store */
#define RD_KAFKA_OFFSET_INVALID -1001 /**< Invalid offset */


/** @cond NO_DOC */
#define RD_KAFKA_OFFSET_TAIL_BASE -2000 /* internal: do not use */
/** @endcond */

/**
 * @brief Start consuming \p CNT messages from topic's current end offset.
 *
 * That is, if current end offset is 12345 and \p CNT is 200, it will start
 * consuming from offset \c 12345-200 = \c 12145. */
#define RD_KAFKA_OFFSET_TAIL(CNT)  (RD_KAFKA_OFFSET_TAIL_BASE - (CNT))
```

```
/**
 * @brief Start consuming messages for topic \p rkt and \p partition
 * at offset \p offset which may either be an absolute \c (0..N)
 * or one of the logical offsets:
 *  - RD_KAFKA_OFFSET_BEGINNING
 *  - RD_KAFKA_OFFSET_END
 *  - RD_KAFKA_OFFSET_STORED
 *  - RD_KAFKA_OFFSET_TAIL
 *
 * rdkafka will attempt to keep \c queued.min.messages (config property)
 * messages in the local queue by repeatedly fetching batches of messages
 * from the broker until the threshold is reached.
 *
 * The application shall use one of the `rd_kafka_consume*()` functions
 * to consume messages from the local queue, each kafka message being
 * represented as a `rd_kafka_message_t *` object.
 *
 * `rd_kafka_consume_start()` must not be called multiple times for the same
 * topic and partition without stopping consumption first with
 * `rd_kafka_consume_stop()`.
 *
 * @returns 0 on success or -1 on error in which case errno is set
accordingly:
 *  - EBUSY    - Conflicts with an existing or previous subscription
 *               (RD_KAFKA_RESP_ERR__CONFLICT)
 *  - EINVAL   - Invalid offset, or incomplete configuration (lacking
group.id)
 *               (RD_KAFKA_RESP_ERR__INVALID_ARG)
 *  - ESRCH    - requested \p partition is invalid.
 *               (RD_KAFKA_RESP_ERR__UNKNOWN_PARTITION)
 *  - ENOENT   - topic is unknown in the Kafka cluster.
 *               (RD_KAFKA_RESP_ERR__UNKNOWN_TOPIC)
 *  - ENOSYS   - This API is not supported.
 *               (RD_KAFKA_RESP_ERR__UNSUPPORTED_FEATURE)
 *
 * Use `rd_kafka_errno2err()` to convert sytem \c errno to
`rd_kafka_resp_err_t`
 * Not supported on MapR streams.
 */
RD_EXPORT
int rd_kafka_consume_start(rd_kafka_topic_t *rkt, int32_t partition,
                int64_t offset);

/**
 * @brief Same as rd_kafka_consume_start() but re-routes incoming messages
to
 * the provided queue \p rkqu (which must have been previously allocated
 * with `rd_kafka_queue_new()`.
 *
 * The application must use one of the `rd_kafka_consume_*_queue()`
functions
 * to receive fetched messages.
 *
 * `rd_kafka_consume_start_queue()` must not be called multiple times for
the
 * same topic and partition without stopping consumption first with
 * `rd_kafka_consume_stop()`.
 * `rd_kafka_consume_start()` and `rd_kafka_consume_start_queue()` must not
 * be combined for the same topic and partition.
 *
 * Not supported on MapR streams.
 *
 */
RD_EXPORT
```

```
int rd_kafka_consume_start_queue(rd_kafka_topic_t *rkt, int32_t partition,
                  int64_t offset, rd_kafka_queue_t *rkqu);

/**
 * @brief Stop consuming messages for topic \p rkt and \p partition, purging
 * all messages currently in the local queue.
 *
 * NOTE: To enforce synchronisation this call will block until the internal
 *       fetcher has terminated and offsets are committed to configured
 *       storage method.
 *
 * The application needs to be stop all consumers before calling
 * `rd_kafka_destroy()` on the main object handle.
 *
 * @returns 0 on success or -1 on error (see `errno`).
 *
 * Not supported on MapR streams.
 *
 */
RD_EXPORT
int rd_kafka_consume_stop(rd_kafka_topic_t *rkt, int32_t partition);



/**
 * @brief Seek consumer for topic+partition to \p offset which is either an
 *        absolute or logical offset.
 *
 * If \p timeout_ms is not 0 the call will wait this long for the
 * seek to be performed. If the timeout is reached the internal state
 * will be unknown and this function returns `RD_KAFKA_RESP_ERR__TIMED_OUT`.
 * If \p timeout_ms is 0 it will initiate the seek but return
 * immediately without any error reporting (e.g., async).
 *
 * This call triggers a fetch queue barrier flush.
 *
 * @returns `RD_KAFKA_RESP_ERR__NO_ERROR` on success else an error code.
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_seek (rd_kafka_topic_t *rkt,
                                   int32_t partition,
                                   int64_t offset,
                                   int timeout_ms);


/**
 * @brief Consume a single message from topic \p rkt and \p partition
 *
 * \p timeout_ms is maximum amount of time to wait for a message to be
received.
 * Consumer must have been previously started with
`rd_kafka_consume_start()`.
 *
 * @returns a message object on success or \c NULL on error.
 * The message object must be destroyed with `rd_kafka_message_destroy()`
 * when the application is done with it.
 *
 * Errors (when returning NULL):
 *  - ETIMEDOUT - \p timeout_ms was reached with no new messages fetched.
 *  - ENOENT    - \p rkt + \p partition is unknown.
 *                 (no prior `rd_kafka_consume_start()` call)
 *  - ENOSYS   - This API is not supported.
 *               (RD_KAFKA_RESP_ERR__UNSUPPORTED_FEATURE)
 *
```

```
 * NOTE: The returned message's \c ..->err must be checked for errors.
 * NOTE: \c ..->err \c == \c RD_KAFKA_RESP_ERR__PARTITION_EOF signals that
the
 *       end of the partition has been reached, which should typically not
be
 *       considered an error. The application should handle this case
 *       (e.g., ignore).
 *
 * @remark on_consume() interceptors may be called from this function prior
to
 *         passing message to application.
 *
 * Not supported on MapR streams.
 *
 */
RD_EXPORT
rd_kafka_message_t *rd_kafka_consume(rd_kafka_topic_t *rkt, int32_t
partition,
                     int timeout_ms);



/**
 * @brief Consume up to \p rkmessages_size from topic \p rkt and \p
partition
 *        putting a pointer to each message in the application provided
 *        array \p rkmessages (of size \p rkmessages_size entries).
 *
 * `rd_kafka_consume_batch()` provides higher throughput performance
 * than `rd_kafka_consume()`.
 *
 * \p timeout_ms is the maximum amount of time to wait for all of
 * \p rkmessages_size messages to be put into \p rkmessages.
 * If no messages were available within the timeout period this function
 * returns 0 and \p rkmessages remains untouched.
 * This differs somewhat from `rd_kafka_consume()`.
 *
 * The message objects must be destroyed with `rd_kafka_message_destroy()`
 * when the application is done with it.
 *
 * @returns the number of rkmessages added in \p rkmessages,
 * or -1 on error (same error codes as for `rd_kafka_consume()`.
 *
 * @sa rd_kafka_consume()
 *
 * @remark on_consume() interceptors may be called from this function prior
to
 *         passing message to application.
 *
 * Not supported on MapR streams.
 *
 */
RD_EXPORT
ssize_t rd_kafka_consume_batch(rd_kafka_topic_t *rkt, int32_t partition,
                int timeout_ms,
                rd_kafka_message_t **rkmessages,
                size_t rkmessages_size);



/**
 * @brief Consumes messages from topic \p rkt and \p partition, calling
 * the provided callback for each consumed messsage.
 *
```

```
 * `rd_kafka_consume_callback()` provides higher throughput performance
 * than both `rd_kafka_consume()` and `rd_kafka_consume_batch()`.
 *
 * \p timeout_ms is the maximum amount of time to wait for one or more
messages
 * to arrive.
 *
 * The provided \p consume_cb function is called for each message,
 * the application \b MUST \b NOT call `rd_kafka_message_destroy()` on the
 * provided \p rkmessage.
 *
 * The \p opaque argument is passed to the 'consume_cb' as \p opaque.
 *
 * @returns the number of messages processed or -1 on error.
 *
 * @sa rd_kafka_consume()
 *
 * @remark on_consume() interceptors may be called from this function prior
to
 *         passing message to application.
 *
 * Not supported on MapR streams.
 *
 */
RD_EXPORT
int rd_kafka_consume_callback(rd_kafka_topic_t *rkt, int32_t partition,
                  int timeout_ms,
                  void (*consume_cb) (rd_kafka_message_t
                          *rkmessage,
                          void *opaque),
                  void *opaque);


/**
 * @name Simple Consumer API (legacy): Queue consumers
 * @{
 *
 * The following `..._queue()` functions are analogue to the functions above
 * but reads messages from the provided queue \p rkqu instead.
 * \p rkqu must have been previously created with `rd_kafka_queue_new()`
 * and the topic consumer must have been started with
 * `rd_kafka_consume_start_queue()` utilising the the same queue.
 */

/**
 * @brief Consume from queue
 *
 * @sa rd_kafka_consume()
 *
 * Not supported on MapR streams.
 *
 */
RD_EXPORT
rd_kafka_message_t *rd_kafka_consume_queue(rd_kafka_queue_t *rkqu,
                        int timeout_ms);

/**
 * @brief Consume batch of messages from queue
 *
 * @sa rd_kafka_consume_batch()
 *
 * Not supported on MapR streams.
 *
 */
```

```
RD_EXPORT
ssize_t rd_kafka_consume_batch_queue(rd_kafka_queue_t *rkqu,
                     int timeout_ms,
                     rd_kafka_message_t **rkmessages,
                     size_t rkmessages_size);

/**
 * @brief Consume multiple messages from queue with callback
 *
 * @sa rd_kafka_consume_callback()
 *
 * Not supported on MapR streams.
 *
 */
RD_EXPORT
int rd_kafka_consume_callback_queue(rd_kafka_queue_t *rkqu,
                    int timeout_ms,
                    void (*consume_cb) (rd_kafka_message_t
                            *rkmessage,
                            void *opaque),
                    void *opaque);


/**@}*/




/**
 * @name Simple Consumer API (legacy): Topic+partition offset store.
 * @{
 *
 * If \c auto.commit.enable is true the offset is stored automatically
prior to
 * returning of the message(s) in each of the rd_kafka_consume*() functions
 * above.
 */


/**
 * @brief Store offset \p offset for topic \p rkt partition \p partition.
 *
 * The offset will be committed (written) to the offset store according
 * to \c `auto.commit.interval.ms` or manual offset-less commit()
 *
 * @remark \c `enable.auto.offset.store` must be set to "false" when using
this API.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or an error code on error.
 * RD_KAFKA_RESP_ERR__UNSUPPORTED_FEATURE on MapR streams.
 *
 * Not supported on MapR streams.
 *
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_offset_store(rd_kafka_topic_t *rkt,
                      int32_t partition, int64_t offset);


/**
 * @brief Store offsets for next auto-commit for one or more partitions.
 *
 * The offset will be committed (written) to the offset store according
 * to \c `auto.commit.interval.ms` or manual offset-less commit().
```

```
 *
 * Per-partition success/error status propagated through each partition's
 * \c .err field.
 *
 * @remark \c `enable.auto.offset.store` must be set to "false" when using
this API.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success, or
 *          RD_KAFKA_RESP_ERR__UNKNOWN_PARTITION if none of the
 *          offsets could be stored, or
 *          RD_KAFKA_RESP_ERR__INVALID_ARG if \c enable.auto.offset.store
is true.
 *          RD_KAFKA_RESP_ERR__UNSUPPORTED_FEATURE on MapR streams.
 *
 * Not supported on MapR streams.
 *
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_offsets_store(rd_kafka_t *rk,
                       rd_kafka_topic_partition_list_t *offsets);
/**@}*/




/**
 * @name KafkaConsumer (C)
 * @{
 * @brief High-level KafkaConsumer C API
 *
 *
 *
 */

/**
 * @brief Subscribe to topic set using balanced consumer groups.
 *
 * Wildcard (regex) topics are supported by the librdkafka assignor:
 * any topic name in the \p topics list that is prefixed with \c \"^\" will
 * be regex-matched to the full list of topics in the cluster and matching
 * topics will be added to the subscription list.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or
 *          RD_KAFKA_RESP_ERR__INVALID_ARG if list is empty, contains
invalid
 *          topics or regexes.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_subscribe (rd_kafka_t *rk,
                    const rd_kafka_topic_partition_list_t *topics);


/**
 * @brief Unsubscribe from the current subscription set.
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_unsubscribe (rd_kafka_t *rk);


/**
 * @brief Returns the current topic subscription
 *
 * @returns An error code on failure, otherwise \p topic is updated
 *          to point to a newly allocated topic list (possibly empty).
```

```
 *
 * @remark The application is responsible for calling
 *         rd_kafka_topic_partition_list_destroy on the returned list.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_subscription (rd_kafka_t *rk,
                       rd_kafka_topic_partition_list_t **topics);



/**
 * @brief Poll the consumer for messages or events.
 *
 * Will block for at most \p timeout_ms milliseconds.
 *
 * @remark  An application should make sure to call consumer_poll() at
regular
 *          intervals, even if no messages are expected, to serve any
 *          queued callbacks waiting to be called. This is especially
 *          important when a rebalance_cb has been registered as it needs
 *          to be called and handled properly to synchronize internal
 *          consumer state.
 *
 * @returns A message object which is a proper message if \p ->err is
 *          RD_KAFKA_RESP_ERR_NO_ERROR, or an event or error for any other
 *          value.
 *
 * @remark on_consume() interceptors may be called from this function prior
to
 *         passing message to application.
 *
 * @sa rd_kafka_message_t
 */
RD_EXPORT
rd_kafka_message_t *rd_kafka_consumer_poll (rd_kafka_t *rk, int timeout_ms);

/**
 * @brief Close down the KafkaConsumer.
 *
 * @remark This call will block until the consumer has revoked its
assignment,
 *         calling the \c rebalance_cb if it is configured, committed
offsets
 *         to broker, and left the consumer group.
 *         The maximum blocking time is roughly limited to
session.timeout.ms.
 *
 * @returns An error code indicating if the consumer close was succesful
 *          or not.
 *
 * @remark The application still needs to call rd_kafka_destroy() after
 *         this call finishes to clean up the underlying handle resources.
 *
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_consumer_close (rd_kafka_t *rk);



/**
 * @brief Atomic assignment of partitions to consume.
 *
 * The new \p partitions will replace the existing assignment.
 *
```


last-updated: Apr 11, 2022

```
 * When used from a rebalance callback the application shall pass the
 * partition list passed to the callback (or a copy of it) (even if the list
 * is empty) rather than NULL to maintain internal join state.
 * A zero-length \p partitions will treat the partitions as a valid,
 * albeit empty, assignment, and maintain internal state, while a \c NULL
 * value for \p partitions will reset and clear the internal state.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_assign (rd_kafka_t *rk,
                 const rd_kafka_topic_partition_list_t *partitions);

/**
 * @brief Returns the current partition assignment
 *
 * @returns An error code on failure, otherwise \p partitions is updated
 *          to point to a newly allocated partition list (possibly empty).
 *
 * @remark The application is responsible for calling
 *         rd_kafka_topic_partition_list_destroy on the returned list.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_assignment (rd_kafka_t *rk,
                     rd_kafka_topic_partition_list_t **partitions);




/**
 * @brief Commit offsets on broker for the provided list of partitions.
 *
 * \p offsets should contain \c topic, \c partition, \c offset and possibly
 * \c metadata.
 * If \p offsets is NULL the current partition assignment will be used
instead.
 *
 * If \p async is false this operation will block until the broker offset
commit
 * is done, returning the resulting success or error code.
 *
 * If a rd_kafka_conf_set_offset_commit_cb() offset commit callback has been
 * configured the callback will be enqueued for a future call to
 * rd_kafka_poll(), rd_kafka_consumer_poll() or similar.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_commit (rd_kafka_t *rk, const rd_kafka_topic_partition_list_t
*offsets,
                 int async);


/**
 * @brief Commit message's offset on broker for the message's partition.
 *
 * @sa rd_kafka_commit
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_commit_message (rd_kafka_t *rk, const rd_kafka_message_t
*rkmessage,
                         int async);


/**
 * @brief Commit offsets on broker for the provided list of partitions.
 *
 * See rd_kafka_commit for \p offsets semantics.
```

```
 *
 * The result of the offset commit will be posted on the provided \p rkqu
queue.
 *
 * If the application uses one of the poll APIs (rd_kafka_poll(),
 * rd_kafka_consumer_poll(), rd_kafka_queue_poll(), ..) to serve the queue
 * the \p cb callback is required. \p opaque is passed to the callback.
 *
 * If using the event API the callback is ignored and the offset commit
result
 * will be returned as an RD_KAFKA_EVENT_COMMIT event. The \p opaque
 * value will be available with rd_kafka_event_opaque()
 *
 * If \p rkqu is NULL a temporary queue will be created and the callback
will
 * be served by this call.
 *
 * @sa rd_kafka_commit()
 * @sa rd_kafka_conf_set_offset_commit_cb()
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_commit_queue (rd_kafka_t *rk,
               const rd_kafka_topic_partition_list_t *offsets,
               rd_kafka_queue_t *rkqu,
               void (*cb) (rd_kafka_t *rk,
                   rd_kafka_resp_err_t err,
                   rd_kafka_topic_partition_list_t *offsets,
                   void *opaque),
               void *opaque);


/**
 * @brief Retrieve committed offsets for topics+partitions.
 *
 * The \p offset field of each requested partition will either be set to
 * stored offset or to RD_KAFKA_OFFSET_INVALID in case there was no stored
 * offset for that partition.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success in which case the
 *          \p offset or \p err field of each \p partitions' element is
filled
 *          in with the stored offset, or a partition specific error.
 *          Else returns an error code.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_committed (rd_kafka_t *rk,
            rd_kafka_topic_partition_list_t *partitions,
            int timeout_ms);



/**
 * @brief Retrieve current positions (offsets) for topics+partitions.
 *
 * The \p offset field of each requested partition will be set to the offset
 * of the last consumed message + 1, or RD_KAFKA_OFFSET_INVALID in case
there was
 * no previous message.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success in which case the
 *          \p offset or \p err field of each \p partitions' element is
filled
 *          in with the stored offset, or a partition specific error.
 *          Else returns an error code.
```

```
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_position (rd_kafka_t *rk,
           rd_kafka_topic_partition_list_t *partitions);


/**@}*/



/**
 * @name Producer API
 * @{
 *
 *
 */


/**
 * @brief Producer message flags
 */
#define RD_KAFKA_MSG_F_FREE  0x1 /**< Delegate freeing of payload to
rdkafka. */
#define RD_KAFKA_MSG_F_COPY  0x2 /**< rdkafka will make a copy of the
payload. */
#define RD_KAFKA_MSG_F_BLOCK 0x4 /**< Block produce*() on message queue
full.
                  *   WARNING: If a delivery report callback
                  *            is used the application MUST
                  *            call rd_kafka_poll() (or equiv.)
                  *            to make sure delivered messages
                  *            are drained from the internal
                  *            delivery report queue.
                  *            Failure to do so will result
                  *            in indefinately blocking on
                  *            the produce() call when the
                  *            message queue is full.
                  */



/**
 * @brief Produce and send a single message to broker.
 *
 * \p rkt is the target topic which must have been previously created with
 * `rd_kafka_topic_new()`.
 *
 * `rd_kafka_produce()` is an asynch non-blocking API.
 *
 * \p partition is the target partition, either:
 *   - RD_KAFKA_PARTITION_UA (unassigned) for
 *     automatic partitioning using the topic's partitioner function, or
 *   - a fixed partition (0..N)
 *
 * \p msgflags is zero or more of the following flags OR:ed together:
 *    RD_KAFKA_MSG_F_BLOCK - block \p produce*() call if
 *                           \p queue.buffering.max.messages or
 *                           \p queue.buffering.max.kbytes are exceeded.
 *                           Messages are considered in-queue from the
point they
 *                           are accepted by produce() until their
corresponding
 *                           delivery report callback/event returns.
 *                           It is thus a requirement to call
```

```
 *                            rd_kafka_poll() (or equiv.) from a separate
 *                            thread when F_BLOCK is used.
 *                            See WARNING on \c RD_KAFKA_MSG_F_BLOCK above.
 *
 *    RD_KAFKA_MSG_F_FREE - rdkafka will free(3) \p payload when it is done
 *                          with it.
 *    RD_KAFKA_MSG_F_COPY - the \p payload data will be copied and the
 *                          \p payload pointer will not be used by rdkafka
 *                          after the call returns.
 *
 *    ..._F_FREE and ..._F_COPY are mutually exclusive.
 *
 *    If the function returns -1 and RD_KAFKA_MSG_F_FREE was specified, then
 *    the memory associated with the payload is still the caller's
 *    responsibility.
 *
 * \p payload is the message payload of size \p len bytes.
 *
 * \p key is an optional message key of size \p keylen bytes, if non-NULL it
 * will be passed to the topic partitioner as well as be sent with the
 * message to the broker and passed on to the consumer.
 *
 * \p msg_opaque is an optional application-provided per-message opaque
 * pointer that will provided in the delivery report callback (`dr_cb`) for
 * referencing this message.
 *
 * @remark on_send() and on_acknowledgement() interceptors may be called
 *         from this function. on_acknowledgement() will only be called if
the
 *         message fails partitioning.
 *
 * @returns 0 on success or -1 on error in which case errno is set
accordingly:
 *  - ENOBUFS  - maximum number of outstanding messages has been reached:
 *               "queue.buffering.max.messages"
 *               (RD_KAFKA_RESP_ERR__QUEUE_FULL)
 *  - EMSGSIZE - message is larger than configured max size:
 *               "messages.max.bytes".
 *               (RD_KAFKA_RESP_ERR_MSG_SIZE_TOO_LARGE)
 *  - ESRCH    - requested \p partition is unknown in the Kafka cluster.
 *               (RD_KAFKA_RESP_ERR__UNKNOWN_PARTITION)
 *  - ENOENT   - topic is unknown in the Kafka cluster.
 *               (RD_KAFKA_RESP_ERR__UNKNOWN_TOPIC)
 *
 * @sa Use rd_kafka_errno2err() to convert `errno` to rdkafka error code.
 */
RD_EXPORT
int rd_kafka_produce(rd_kafka_topic_t *rkt, int32_t partition,
             int msgflags,
             void *payload, size_t len,
             const void *key, size_t keylen,
             void *msg_opaque);


/**
 * @brief Produce and send a single message to broker.
 *
 * The message is defined by a va-arg list using \c rd_kafka_vtype_t
 * tag tuples which must be terminated with a single \c RD_KAFKA_V_END.
 *
 * @returns \c RD_KAFKA_RESP_ERR_NO_ERROR on success, else an error code.
 *
 * @sa rd_kafka_produce, RD_KAFKA_V_END
 */
```

```
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_producev (rd_kafka_t *rk, ...);


/**
 * @brief Produce multiple messages.
 *
 * If partition is RD_KAFKA_PARTITION_UA the configured partitioner will
 * be run for each message (slower), otherwise the messages will be enqueued
 * to the specified partition directly (faster).
 *
 * The messages are provided in the array \p rkmessages of count \p
message_cnt
 * elements.
 * The \p partition and \p msgflags are used for all provided messages.
 *
 * Honoured \p rkmessages[] fields are:
 *  - payload,len    Message payload and length
 *  - key,key_len    Optional message key
 *  - _private       Message opaque pointer (msg_opaque)
 *  - err            Will be set according to success or failure.
 *                   Application only needs to check for errors if
 *                   return value != \p message_cnt.
 *
 * @returns the number of messages succesfully enqueued for producing.
 */
RD_EXPORT
int rd_kafka_produce_batch(rd_kafka_topic_t *rkt, int32_t partition,
                           int msgflags,
                           rd_kafka_message_t *rkmessages, int
message_cnt);




/**
 * @brief Wait until all outstanding produce requests, et.al, are completed.
 *        This should typically be done prior to destroying a producer
instance
 *        to make sure all queued and in-flight produce requests are
completed
 *        before terminating.
 *
 * @remark This function will call rd_kafka_poll() and thus trigger
callbacks.
 *
 * @returns RD_KAFKA_RESP_ERR__TIMED_OUT if \p timeout_ms was reached
before all
 *          outstanding requests were completed, else
RD_KAFKA_RESP_ERR_NO_ERROR
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_flush (rd_kafka_t *rk, int timeout_ms);


/**@}*/


/**
* @name Metadata API
* @{
*
*
*/
```

```
/**
 * @brief Broker information
 */
typedef struct rd_kafka_metadata_broker {
        int32_t     id;             /**< Broker Id */
        char       *host;           /**< Broker hostname */
        int         port;           /**< Broker listening port */
} rd_kafka_metadata_broker_t;

/**
 * @brief Partition information
 */
typedef struct rd_kafka_metadata_partition {
        int32_t     id;             /**< Partition Id */
        rd_kafka_resp_err_t err;    /**< Partition error reported by broker
*/
        int32_t     leader;         /**< Leader broker */
        int         replica_cnt;    /**< Number of brokers in \p replicas */
        int32_t    *replicas;       /**< Replica brokers */
        int         isr_cnt;        /**< Number of ISR brokers in \p isrs */
        int32_t    *isrs;           /**< In-Sync-Replica brokers */
} rd_kafka_metadata_partition_t;

/**
 * @brief Topic information
 */
typedef struct rd_kafka_metadata_topic {
        char       *topic;          /**< Topic name */
        int         partition_cnt;  /**< Number of partitions in \p
partitions*/
        struct rd_kafka_metadata_partition *partitions; /**< Partitions */
        rd_kafka_resp_err_t err;    /**< Topic error reported by broker */
} rd_kafka_metadata_topic_t;


/**
 * @brief Metadata container
 */
typedef struct rd_kafka_metadata {
        int         broker_cnt;     /**< Number of brokers in \p brokers */
        struct rd_kafka_metadata_broker *brokers;  /**< Brokers */

        int         topic_cnt;      /**< Number of topics in \p topics */
        struct rd_kafka_metadata_topic *topics;    /**< Topics */

        int32_t     orig_broker_id;   /**< Broker originating this metadata
*/
        char       *orig_broker_name; /**< Name of originating broker */
} rd_kafka_metadata_t;


/**
 * @brief Request Metadata from broker.
 *
 * Parameters:
 *  - \p all_topics  if non-zero: request info about all topics in cluster,
 *                   if zero: only request info about locally known topics.
 *  - \p only_rkt    only request info about this topic
 *  - \p metadatap   pointer to hold metadata result.
 *                   The \p *metadatap pointer must be released
 *                   with rd_kafka_metadata_destroy().
 *  - \p timeout_ms  maximum response time before failing.
```

```
 *
 * Returns RD_KAFKA_RESP_ERR_NO_ERROR on success (in which case *metadatap)
 * will be set, else RD_KAFKA_RESP_ERR__TIMED_OUT on timeout or
 * other error code on error.
 * Not supported on MapR streams.
 */
RD_EXPORT
rd_kafka_resp_err_t
rd_kafka_metadata (rd_kafka_t *rk, int all_topics,
                   rd_kafka_topic_t *only_rkt,
                   const struct rd_kafka_metadata **metadatap,
                   int timeout_ms);

/**
 * @brief Release metadata memory.
 * Not supported on MapR streams.
 */
RD_EXPORT
void rd_kafka_metadata_destroy(const struct rd_kafka_metadata *metadata);


/**@}*/



/**
* @name Client group information
* @{
*
*
*/


/**
 * @brief Group member information
 *
 * For more information on \p member_metadata format, see
 * https://cwiki.apache.org/confluence/display/KAFKA/
A+Guide+To+The+Kafka+Protocol#AGuideToTheKafkaProtocol-GroupMembershipAPI
 *
 */
struct rd_kafka_group_member_info {
        char *member_id;           /**< Member id (generated by broker) */
        char *client_id;           /**< Client's \p client.id */
        char *client_host;         /**< Client's hostname */
        void *member_metadata;     /**< Member metadata (binary),
                                    *   format depends on \p
protocol_type. */
        int   member_metadata_size; /**< Member metadata size in bytes */
        void *member_assignment;    /**< Member assignment (binary),
                                     *    format depends on \p
protocol_type. */
        int   member_assignment_size; /**< Member assignment size in bytes
*/
};

/**
 * @brief Group information
 */
struct rd_kafka_group_info {
        struct rd_kafka_metadata_broker broker; /**< Originating broker
info */
        char *group;                            /**< Group name */
        rd_kafka_resp_err_t err;                /**< Broker-originated
```

```
error */
        char *state;                              /**< Group state */
        char *protocol_type;                      /**< Group protocol type */
        char *protocol;                           /**< Group protocol */
        struct rd_kafka_group_member_info *members; /**< Group members */
        int   member_cnt;                         /**< Group member count */
};

/**
 * @brief List of groups
 *
 * @sa rd_kafka_group_list_destroy() to release list memory.
 */
struct rd_kafka_group_list {
        struct rd_kafka_group_info *groups;   /**< Groups */
        int    group_cnt;                     /**< Group count */
        bool   is_streams_list;               /* List contains consumer gr
                                               * on mapr streams
                                               */
};


/**
 * @brief List and describe client groups in cluster.
 *
 * \p group is an optional group name to describe, otherwise (\p NULL) all
 * groups are returned.
 *
 * \p timeout_ms is the (approximate) maximum time to wait for response
 * from brokers and must be a positive value.
 *
 * @returns \c RD_KAFKA_RESP_ERR__NO_ERROR on success and \p grplistp is
 *           updated to point to a newly allocated list of groups.
 *           \c RD_KAFKA_RESP_ERR__PARTIAL if not all brokers responded
 *           in time but at least one group is returned in  \p grplistlp.
 *           \c RD_KAFKA_RESP_ERR__TIMED_OUT if no groups were returned in
the
 *           given timeframe but not all brokers have yet responded, or
 *           if the list of brokers in the cluster could not be obtained
within
 *           the given timeframe.
 *           \c RD_KAFKA_RESP_ERR__TRANSPORT if no brokers were found.
 *           Other error codes may also be returned from the request layer.
 *
 *           The \p grplistp remains untouched if any error code is
returned,
 *           with the exception of RD_KAFKA_RESP_ERR__PARTIAL which behaves
 *           as RD_KAFKA_RESP_ERR__NO_ERROR (success) but with an incomplete
 *           group list.
 *
 * @sa Use rd_kafka_group_list_destroy() to release list memory.
 */
RD_EXPORT
rd_kafka_resp_err_t
rd_kafka_list_groups (rd_kafka_t *rk, const char *group,
                      const struct rd_kafka_group_list **grplistp,
                      int timeout_ms);

/**
 * @brief Release list memory
 */
RD_EXPORT
void rd_kafka_group_list_destroy (const struct rd_kafka_group_list
*grplist);
```

```
/**@}*/



/**
 * @name Miscellaneous APIs
 * @{
 *
 */


/**
 * @brief Adds one or more brokers to the kafka handle's list of initial
 *        bootstrap brokers.
 *
 * Additional brokers will be discovered automatically as soon as rdkafka
 * connects to a broker by querying the broker metadata.
 *
 * If a broker name resolves to multiple addresses (and possibly
 * address families) all will be used for connection attempts in
 * round-robin fashion.
 *
 * \p brokerlist is a ,-separated list of brokers in the format:
 *   \c \<broker1\>,\<broker2\>,..
 * Where each broker is in either the host or URL based format:
 *   \c \<host\>[:\<port\>]
 *   \c \<proto\>://\<host\>[:port]
 * \c \<proto\> is either \c PLAINTEXT, \c SSL, \c SASL, \c SASL_PLAINTEXT
 * The two formats can be mixed but ultimately the value of the
 * `security.protocol` config property decides what brokers are allowed.
 *
 * Example:
 *    brokerlist = "broker1:10000,broker2"
 *    brokerlist = "SSL://broker3:9000,ssl://broker2"
 *
 * @returns the number of brokers successfully added.
 *
 * @remark Brokers may also be defined with the \c metadata.broker.list or
 *         \c bootstrap.servers configuration property (preferred method).
 */
RD_EXPORT
int rd_kafka_brokers_add(rd_kafka_t *rk, const char *brokerlist);




/**
 * @brief Set logger function.
 *
 * The default is to print to stderr, but a syslog logger is also available,
 * see rd_kafka_log_(print|syslog) for the builtin alternatives.
 * Alternatively the application may provide its own logger callback.
 * Or pass 'func' as NULL to disable logging.
 *
 * @deprecated Use rd_kafka_conf_set_log_cb()
 *
 * @remark \p rk may be passed as NULL in the callback.
 */
RD_EXPORT RD_DEPRECATED
void rd_kafka_set_logger(rd_kafka_t *rk,
             void (*func) (const rd_kafka_t *rk, int level,
                   const char *fac, const char *buf));
```

```
/**
 * @brief Specifies the maximum logging level produced by
 *        internal kafka logging and debugging.
 *
 * If the \p \"debug\" configuration property is set the level is
automatically
 * adjusted to \c LOG_DEBUG (7).
 */
RD_EXPORT
void rd_kafka_set_log_level(rd_kafka_t *rk, int level);


/**
 * @brief Builtin (default) log sink: print to stderr
 */
RD_EXPORT
void rd_kafka_log_print(const rd_kafka_t *rk, int level,
             const char *fac, const char *buf);


/**
 * @brief Builtin log sink: print to syslog.
 */
RD_EXPORT
void rd_kafka_log_syslog(const rd_kafka_t *rk, int level,
              const char *fac, const char *buf);


/**
 * @brief Returns the current out queue length.
 *
 * The out queue contains messages waiting to be sent to, or acknowledged
by,
 * the broker.
 *
 * An application should wait for this queue to reach zero before
terminating
 * to make sure outstanding requests (such as offset commits) are fully
 * processed.
 *
 * @returns number of messages in the out queue.
 */
RD_EXPORT
int         rd_kafka_outq_len(rd_kafka_t *rk);



/**
 * @brief Dumps rdkafka's internal state for handle \p rk to stream \p fp
 *
 * This is only useful for debugging rdkafka, showing state and statistics
 * for brokers, topics, partitions, etc.
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
void rd_kafka_dump(FILE *fp, rd_kafka_t *rk);



/**
 * @brief Retrieve the current number of threads in use by librdkafka.
```

```
 *
 * Used by regression tests.
 * Not supported on MapR streams.
 */
RD_EXPORT
int rd_kafka_thread_cnt(void);


/**
 * @brief Wait for all rd_kafka_t objects to be destroyed.
 *
 * Returns 0 if all kafka objects are now destroyed, or -1 if the
 * timeout was reached.
 *
 * @remark This function is deprecated.
 */
RD_EXPORT
int rd_kafka_wait_destroyed(int timeout_ms);


/**
 * @brief Run librdkafka's built-in unit-tests.
 *
 * @returns the number of failures, or 0 if all tests passed.
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
int rd_kafka_unittest (void);


/**@}*/




/**
 * @name Experimental APIs
 * @{
 */

/**
 * @brief Redirect the main (rd_kafka_poll()) queue to the KafkaConsumer's
 *        queue (rd_kafka_consumer_poll()).
 *
 * @warning It is not permitted to call rd_kafka_poll() after directing the
 *          main queue with rd_kafka_poll_set_consumer().
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_poll_set_consumer (rd_kafka_t *rk);


/**@}*/

/**
 * @name Event interface
 *
 * @brief The event API provides an alternative pollable non-callback
interface
 *        to librdkafka's message and event queues.
 *
 * @{
 */
```

```
/**
 * @brief Event types
 */
typedef int rd_kafka_event_type_t;
#define RD_KAFKA_EVENT_NONE          0x0
#define RD_KAFKA_EVENT_DR            0x1  /**< Producer Delivery report
batch */
#define RD_KAFKA_EVENT_FETCH         0x2  /**< Fetched message (consumer) */
#define RD_KAFKA_EVENT_LOG           0x4  /**< Log message */
#define RD_KAFKA_EVENT_ERROR         0x8  /**< Error */
#define RD_KAFKA_EVENT_REBALANCE     0x10 /**< Group rebalance (consumer) */
#define RD_KAFKA_EVENT_OFFSET_COMMIT 0x20 /**< Offset commit result */
#define RD_KAFKA_EVENT_STATS         0x40 /**< Stats */


typedef struct rd_kafka_op_s rd_kafka_event_t;


/**
 * @returns the event type for the given event.
 *
 * @remark As a convenience it is okay to pass \p rkev as NULL in which case
 *         RD_KAFKA_EVENT_NONE is returned.
 */
RD_EXPORT
rd_kafka_event_type_t rd_kafka_event_type (const rd_kafka_event_t *rkev);

/**
 * @returns the event type's name for the given event.
 *
 * @remark As a convenience it is okay to pass \p rkev as NULL in which case
 *         the name for RD_KAFKA_EVENT_NONE is returned.
 */
RD_EXPORT
const char *rd_kafka_event_name (const rd_kafka_event_t *rkev);


/**
 * @brief Destroy an event.
 *
 * @remark Any references to this event, such as extracted messages,
 *         will not be usable after this call.
 *
 * @remark As a convenience it is okay to pass \p rkev as NULL in which case
 *         no action is performed.
 */
RD_EXPORT
void rd_kafka_event_destroy (rd_kafka_event_t *rkev);


/**
 * @returns the next message from an event.
 *
 * Call repeatedly until it returns NULL.
 *
 * Event types:
 *  - RD_KAFKA_EVENT_FETCH  (1 message)
 *  - RD_KAFKA_EVENT_DR     (>=1 message(s))
 *
 * @remark The returned message(s) MUST NOT be
 *         freed with rd_kafka_message_destroy().
 *
 * @remark on_consume() interceptor may be called
```


last-updated: Apr 11, 2022

```
 *         from this function prior to passing message to application.
 */
RD_EXPORT
const rd_kafka_message_t *rd_kafka_event_message_next (rd_kafka_event_t
*rkev);


/**
 * @brief Extacts \p size message(s) from the event into the
 *        pre-allocated array \p rkmessages.
 *
 * Event types:
 *  - RD_KAFKA_EVENT_FETCH  (1 message)
 *  - RD_KAFKA_EVENT_DR     (>=1 message(s))
 *
 * @returns the number of messages extracted.
 *
 * @remark on_consume() interceptor may be called
 *         from this function prior to passing message to application.
 */
RD_EXPORT
size_t rd_kafka_event_message_array (rd_kafka_event_t *rkev,
                     const rd_kafka_message_t **rkmessages,
                     size_t size);


/**
 * @returns the number of remaining messages in the event.
 *
 * Event types:
 *  - RD_KAFKA_EVENT_FETCH  (1 message)
 *  - RD_KAFKA_EVENT_DR     (>=1 message(s))
 */
RD_EXPORT
size_t rd_kafka_event_message_count (rd_kafka_event_t *rkev);


/**
 * @returns the error code for the event.
 *
 * Event types:
 *  - all
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_event_error (rd_kafka_event_t *rkev);


/**
 * @returns the error string (if any).
 *          An application should check that rd_kafka_event_error() returns
 *          non-zero before calling this function.
 *
 * Event types:
 *  - all
 */
RD_EXPORT
const char *rd_kafka_event_error_string (rd_kafka_event_t *rkev);



/**
 * @returns the user opaque (if any)
 *
 * Event types:
```

```
 *  - RD_KAFKA_OFFSET_COMMIT
 */
RD_EXPORT
void *rd_kafka_event_opaque (rd_kafka_event_t *rkev);


/**
 * @brief Extract log message from the event.
 *
 * Event types:
 *  - RD_KAFKA_EVENT_LOG
 *
 * @returns 0 on success or -1 if unsupported event type.
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
int rd_kafka_event_log (rd_kafka_event_t *rkev,
            const char **fac, const char **str, int *level);


/**
 * @brief Extract stats from the event.
 *
 * Event types:
 *  - RD_KAFKA_EVENT_STATS
 *
 * @returns stats json string.
 *
 * @remark the returned string will be freed automatically along with the
event object
 *
 * Not supported on MapR streams.
 */
RD_EXPORT
const char *rd_kafka_event_stats (rd_kafka_event_t *rkev);


/**
 * @returns the topic partition list from the event.
 *
 * @remark The list MUST NOT be freed with
rd_kafka_topic_partition_list_destroy()
 *
 * Event types:
 *  - RD_KAFKA_EVENT_REBALANCE
 *  - RD_KAFKA_EVENT_OFFSET_COMMIT
 */
RD_EXPORT rd_kafka_topic_partition_list_t *
rd_kafka_event_topic_partition_list (rd_kafka_event_t *rkev);


/**
 * @returns a newly allocated topic_partition container, if applicable for
the event type,
 *          else NULL.
 *
 * @remark The returned pointer MUST be freed with
rd_kafka_topic_partition_destroy().
 *
 * Event types:
 *   RD_KAFKA_EVENT_ERROR  (for partition level errors)
 */
RD_EXPORT rd_kafka_topic_partition_t *
```

```
rd_kafka_event_topic_partition (rd_kafka_event_t *rkev);


/**
 * @brief Poll a queue for an event for max \p timeout_ms.
 *
 * @returns an event, or NULL.
 *
 * @remark Use rd_kafka_event_destroy() to free the event.
 */
RD_EXPORT
rd_kafka_event_t *rd_kafka_queue_poll (rd_kafka_queue_t *rkqu, int
timeout_ms);

/**
* @brief Poll a queue for events served through callbacks for max \p
timeout_ms.
*
* @returns the number of events served.
*
* @remark This API must only be used for queues with callbacks registered
*         for all expected event types. E.g., not a message queue.
*/
RD_EXPORT
int rd_kafka_queue_poll_callback (rd_kafka_queue_t *rkqu, int timeout_ms);


/**@}*/


/**
 * @name Plugin interface
 *
 * @brief A plugin interface that allows external runtime-loaded libraries
 *        to integrate with a client instance without modifications to
 *        the application code.
 *
 *        Plugins are loaded when referenced through the
`plugin.library.paths`
 *        configuration property and operates on the \c rd_kafka_conf_t
 *        object prior \c rd_kafka_t instance creation.
 *
 * @warning Plugins require the application to link librdkafka dynamically
 *          and not statically. Failure to do so will lead to missing
symbols
 *          or finding symbols in another librdkafka library than the
 *          application was linked with.
 */


/**
 * @brief Plugin's configuration initializer method called each time the
 *        library is referenced from configuration (even if previously
loaded by
 *        another client instance).
 *
 * @remark This method MUST be implemented by plugins and have the symbol
name
 *         \c conf_init
 *
 * @param conf Configuration set up to this point.
 * @param plug_opaquep Plugin can set this pointer to a per-configuration
 *                     opaque pointer.
 * @param errstr String buffer of size \p errstr_size where plugin must
```

```
write
 *               a human readable error string in the case the initializer
 *               fails (returns non-zero).
 *
 * @remark A plugin may add an on_conf_destroy() interceptor to clean up
 *         plugin-specific resources created in the plugin's conf_init()
method.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or an error code on error.
 */
typedef rd_kafka_resp_err_t
(rd_kafka_plugin_f_conf_init_t) (rd_kafka_conf_t *conf,
                                 void **plug_opaquep,
                                 char *errstr, size_t errstr_size);

/**@}*/



/**
 * @name Interceptors
 *
 * @{
 *
 * @brief A callback interface that allows message interception for both
 *        producer and consumer data pipelines.
 *
 * Except for the on_new(), on_conf_set(), on_conf_dup() and
on_conf_destroy()
 * interceptors, interceptors are added to the
 * newly created rd_kafka_t client instance. These interceptors MUST only
 * be added from on_new() and MUST NOT be added after rd_kafka_new()
returns.
 *
 * The on_new(), on_conf_set(), on_conf_dup() and on_conf_destroy()
interceptors
 * are added to the configuration object which is later passed to
 * rd_kafka_new() where on_new() is called to allow addition of
 * other interceptors.
 *
 * Each interceptor reference consists of a display name (ic_name),
 * a callback function, and an application-specified opaque value that is
 * passed as-is to the callback.
 * The ic_name must be unique for the interceptor implementation and is used
 * to reject duplicate interceptor methods.
 *
 * Any number of interceptors can be added and they are called in the order
 * they were added, unless otherwise noted.
 * The list of registered interceptor methods are referred to as
 * interceptor chains.
 *
 * @remark Contrary to the Java client the librdkafka interceptor interface
 *         does not support message modification. Message mutability is
 *         discouraged in the Java client and the combination of
 *         serializers and headers cover most use-cases.
 *
 * @remark Interceptors are NOT copied to the new configuration on
 *         rd_kafka_conf_dup() since it would be hard for interceptors to
 *         track usage of the interceptor's opaque value.
 *         An interceptor should rely on the plugin, which will be copied
 *         in rd_kafka_conf_conf_dup(), to set up the initial interceptors.
 *         An interceptor should implement the on_conf_dup() method
 *         to manually set up its internal configuration on the newly
created
```

```
 *         configuration object that is being copied-to based on the
 *         interceptor-specific configuration properties.
 *         conf_dup() should thus be treated the same as conf_init().
 *
 * @remark Interceptors are keyed by the interceptor type (on_..()), the
 *         interceptor name (ic_name) and the interceptor method function.
 *         Duplicates are not allowed and the .._add_on_..() method will
 *         return RD_KAFKA_RESP_ERR__CONFLICT if attempting to add a
duplicate
 *         method.
 *         The only exception is on_conf_destroy() which may be added
multiple
 *         times by the same interceptor to allow proper cleanup of
 *         interceptor configuration state.
 */


/**
 * @brief on_conf_set() is called from rd_kafka_*_conf_set() in the order
 *        the interceptors were added.
 *
 * @param ic_opaque The interceptor's opaque pointer specified in ..add..().
 * @param name The configuration property to set.
 * @param val The configuration value to set, or NULL for reverting to
default
 *            in which case the previous value should be freed.
 * @param errstr A human readable error string in case the interceptor
fails.
 * @param errstr_size Maximum space (including \0) in \p errstr.
 *
 * @returns RD_KAFKA_CONF_RES_OK if the property was known and successfully
 *          handled by the interceptor, RD_KAFKA_CONF_RES_INVALID if the
 *          property was handled by the interceptor but the value was
invalid,
 *          or RD_KAFKA_CONF_RES_UNKNOWN if the interceptor did not handle
 *          this property, in which case the property is passed on on the
 *          interceptor in the chain, finally ending up at the built-in
 *          configuration handler.
 */
typedef rd_kafka_conf_res_t
(rd_kafka_interceptor_f_on_conf_set_t) (rd_kafka_conf_t *conf,
                                        const char *name, const char *val,
                                        char *errstr, size_t errstr_size,
                                        void *ic_opaque);


/**
 * @brief on_conf_dup() is called from rd_kafka_conf_dup() in the
 *        order the interceptors were added and is used to let
 *        an interceptor re-register its conf interecptors with a new
 *        opaque value.
 *        The on_conf_dup() method is called prior to the configuration from
 *        \p old_conf being copied to \p new_conf.
 *
 * @param ic_opaque The interceptor's opaque pointer specified in ..add..().
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or an error code
 *          on failure (which is logged but otherwise ignored).
 *
 * @remark No on_conf_* interceptors are copied to the new configuration
 *         object on rd_kafka_conf_dup().
 */
typedef rd_kafka_resp_err_t
(rd_kafka_interceptor_f_on_conf_dup_t) (rd_kafka_conf_t *new_conf,
```

```
                                        const rd_kafka_conf_t *old_conf,
                                        size_t filter_cnt,
                                        const char **filter,
                                        void *ic_opaque);


/**
 * @brief on_conf_destroy() is called from rd_kafka_*_conf_destroy() in the
 *        order the interceptors were added.
 *
 * @param ic_opaque The interceptor's opaque pointer specified in ..add..().
 */
typedef rd_kafka_resp_err_t
(rd_kafka_interceptor_f_on_conf_destroy_t) (void *ic_opaque);


/**
 * @brief on_new() is called from rd_kafka_new() prior toreturning
 *        the newly created client instance to the application.
 *
 * @param rk The client instance.
 * @param conf The client instance's final configuration.
 * @param ic_opaque The interceptor's opaque pointer specified in ..add..().
 * @param errstr A human readable error string in case the interceptor
fails.
 * @param errstr_size Maximum space (including \0) in \p errstr.
 *
 * @returns an error code on failure, the error is logged but otherwise
ignored.
 *
 * @warning The \p rk client instance will not be fully set up when this
 *          interceptor is called and the interceptor MUST NOT call any
 *          other rk-specific APIs than rd_kafka_interceptor_add..().
 *
 */
typedef rd_kafka_resp_err_t
(rd_kafka_interceptor_f_on_new_t) (rd_kafka_t *rk, const rd_kafka_conf_t
*conf,
                                   void *ic_opaque,
                                   char *errstr, size_t errstr_size);


/**
 * @brief on_destroy() is called from rd_kafka_destroy() or (rd_kafka_new()
 *        if rd_kafka_new() fails during initialization).
 *
 * @param rk The client instance.
 * @param ic_opaque The interceptor's opaque pointer specified in ..add..().
 */
typedef rd_kafka_resp_err_t
(rd_kafka_interceptor_f_on_destroy_t) (rd_kafka_t *rk, void *ic_opaque);




/**
 * @brief on_send() is called from rd_kafka_produce*() (et.al) prior to
 *        the partitioner being called.
 *
 * @param rk The client instance.
 * @param rkmessage The message being produced. Immutable.
 * @param ic_opaque The interceptor's opaque pointer specified in ..add..().
 *
 * @remark This interceptor is only used by producer instances.
```

```
 *
 * @remark The \p rkmessage object is NOT mutable and MUST NOT be modified
 *         by the interceptor.
 *
 * @remark If the partitioner fails or an unknown partition was specified,
 *         the on_acknowledgement() interceptor chain will be called from
 *         within the rd_kafka_produce*() call to maintain
send-acknowledgement
 *         symmetry.
 *
 * @returns an error code on failure, the error is logged but otherwise
ignored.
 */
typedef rd_kafka_resp_err_t
(rd_kafka_interceptor_f_on_send_t) (rd_kafka_t *rk,
                                    rd_kafka_message_t *rkmessage,
                                    void *ic_opaque);

/**
 * @brief on_acknowledgement() is called to inform interceptors that a
message
 *        was succesfully delivered or permanently failed delivery.
 *        The interceptor chain is called from internal librdkafka
background
 *        threads, or rd_kafka_produce*() if the partitioner failed.
 *
 * @param rk The client instance.
 * @param rkmessage The message being produced. Immutable.
 * @param ic_opaque The interceptor's opaque pointer specified in ..add..().
 *
 * @remark This interceptor is only used by producer instances.
 *
 * @remark The \p rkmessage object is NOT mutable and MUST NOT be modified
 *         by the interceptor.
 *
 * @warning The on_acknowledgement() method may be called from internal
 *         librdkafka threads. An on_acknowledgement() interceptor MUST NOT
 *         call any librdkafka API's associated with the \p rk, or perform
 *         any blocking or prolonged work.
 *
 * @returns an error code on failure, the error is logged but otherwise
ignored.
 */
typedef rd_kafka_resp_err_t
(rd_kafka_interceptor_f_on_acknowledgement_t) (rd_kafka_t *rk,
                                               rd_kafka_message_t
*rkmessage,
                                               void *ic_opaque);


/**
 * @brief on_consume() is called just prior to passing the message to the
 *        application in rd_kafka_consumer_poll(), rd_kafka_consume*(),
 *        the event interface, etc.
 *
 * @param rk The client instance.
 * @param rkmessage The message being consumed. Immutable.
 * @param ic_opaque The interceptor's opaque pointer specified in ..add..().
 *
 * @remark This interceptor is only used by consumer instances.
 *
 * @remark The \p rkmessage object is NOT mutable and MUST NOT be modified
 *         by the interceptor.
 *
```

```
 * @returns an error code on failure, the error is logged but otherwise
ignored.
 */
typedef rd_kafka_resp_err_t
(rd_kafka_interceptor_f_on_consume_t) (rd_kafka_t *rk,
                                       rd_kafka_message_t *rkmessage,
                                       void *ic_opaque);

/**
 * @brief on_commit() is called on completed or failed offset commit.
 *        It is called from internal librdkafka threads.
 *
 * @param rk The client instance.
 * @param offsets List of topic+partition+offset+error that were committed.
 *                The error message of each partition should be checked for
 *                error.
 * @param ic_opaque The interceptor's opaque pointer specified in ..add..().
 *
 * @remark This interceptor is only used by consumer instances.
 *
 * @warning The on_commit() interceptor is called from internal
 *          librdkafka threads. An on_commit() interceptor MUST NOT
 *          call any librdkafka API's associated with the \p rk, or perform
 *          any blocking or prolonged work.
 *
 *
 * @returns an error code on failure, the error is logged but otherwise
ignored.
 */
typedef rd_kafka_resp_err_t
(rd_kafka_interceptor_f_on_commit_t) (
        rd_kafka_t *rk,
        const rd_kafka_topic_partition_list_t *offsets,
        rd_kafka_resp_err_t err, void *ic_opaque);



/**
 * @brief Append an on_conf_set() interceptor.
 *
 * @param conf Configuration object.
 * @param ic_name Interceptor name, used in logging.
 * @param on_conf_set Function pointer.
 * @param ic_opaque Opaque value that will be passed to the function.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or
RD_KAFKA_RESP_ERR__CONFLICT
 *          if an existing intercepted with the same \p ic_name and function
 *          has already been added to \p conf.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_conf_interceptor_add_on_conf_set (
        rd_kafka_conf_t *conf, const char *ic_name,
        rd_kafka_interceptor_f_on_conf_set_t *on_conf_set,
        void *ic_opaque);


/**
 * @brief Append an on_conf_dup() interceptor.
 *
 * @param conf Configuration object.
 * @param ic_name Interceptor name, used in logging.
 * @param on_conf_dup Function pointer.
 * @param ic_opaque Opaque value that will be passed to the function.
```

```
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or
RD_KAFKA_RESP_ERR__CONFLICT
 *          if an existing intercepted with the same \p ic_name and function
 *          has already been added to \p conf.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_conf_interceptor_add_on_conf_dup (
        rd_kafka_conf_t *conf, const char *ic_name,
        rd_kafka_interceptor_f_on_conf_dup_t *on_conf_dup,
        void *ic_opaque);

/**
 * @brief Append an on_conf_destroy() interceptor.
 *
 * @param conf Configuration object.
 * @param ic_name Interceptor name, used in logging.
 * @param on_conf_destroy Function pointer.
 * @param ic_opaque Opaque value that will be passed to the function.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR
 *
 * @remark Multiple on_conf_destroy() interceptors are allowed to be added
 *         to the same configuration object.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_conf_interceptor_add_on_conf_destroy (
        rd_kafka_conf_t *conf, const char *ic_name,
        rd_kafka_interceptor_f_on_conf_destroy_t *on_conf_destroy,
        void *ic_opaque);


/**
 * @brief Append an on_new() interceptor.
 *
 * @param conf Configuration object.
 * @param ic_name Interceptor name, used in logging.
 * @param on_send Function pointer.
 * @param ic_opaque Opaque value that will be passed to the function.
  *
 * @remark Since the on_new() interceptor is added to the configuration
object
 *         it may be copied by rd_kafka_conf_dup().
 *         An interceptor implementation must thus be able to handle
 *         the same interceptor,ic_opaque tuple to be used by multiple
 *         client instances.
 *
 * @remark An interceptor plugin should check the return value to make sure
it
 *         has not already been added.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or
RD_KAFKA_RESP_ERR__CONFLICT
 *          if an existing intercepted with the same \p ic_name and function
 *          has already been added to \p conf.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_conf_interceptor_add_on_new (
        rd_kafka_conf_t *conf, const char *ic_name,
        rd_kafka_interceptor_f_on_new_t *on_new,
        void *ic_opaque);
```


last-updated: Apr 11, 2022

```
/**
 * @brief Append an on_destroy() interceptor.
 *
 * @param rk Client instance.
 * @param ic_name Interceptor name, used in logging.
 * @param on_destroy Function pointer.
 * @param ic_opaque Opaque value that will be passed to the function.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or
RD_KAFKA_RESP_ERR__CONFLICT
 *          if an existing intercepted with the same \p ic_name and function
 *          has already been added to \p conf.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_interceptor_add_on_destroy (
        rd_kafka_t *rk, const char *ic_name,
        rd_kafka_interceptor_f_on_destroy_t *on_destroy,
        void *ic_opaque);


/**
 * @brief Append an on_send() interceptor.
 *
 * @param rk Client instance.
 * @param ic_name Interceptor name, used in logging.
 * @param on_send Function pointer.
 * @param ic_opaque Opaque value that will be passed to the function.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or
RD_KAFKA_RESP_ERR__CONFLICT
 *          if an existing intercepted with the same \p ic_name and function
 *          has already been added to \p conf.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_interceptor_add_on_send (
        rd_kafka_t *rk, const char *ic_name,
        rd_kafka_interceptor_f_on_send_t *on_send,
        void *ic_opaque);

/**
 * @brief Append an on_acknowledgement() interceptor.
 *
 * @param rk Client instance.
 * @param ic_name Interceptor name, used in logging.
 * @param on_acknowledgement Function pointer.
 * @param ic_opaque Opaque value that will be passed to the function.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or
RD_KAFKA_RESP_ERR__CONFLICT
 *          if an existing intercepted with the same \p ic_name and function
 *          has already been added to \p conf.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_interceptor_add_on_acknowledgement (
        rd_kafka_t *rk, const char *ic_name,
        rd_kafka_interceptor_f_on_acknowledgement_t *on_acknowledgement,
        void *ic_opaque);


/**
 * @brief Append an on_consume() interceptor.
 *
 * @param rk Client instance.
 * @param ic_name Interceptor name, used in logging.
```

```
 * @param on_consume Function pointer.
 * @param ic_opaque Opaque value that will be passed to the function.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or
RD_KAFKA_RESP_ERR__CONFLICT
 *          if an existing intercepted with the same \p ic_name and function
 *          has already been added to \p conf.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_interceptor_add_on_consume (
        rd_kafka_t *rk, const char *ic_name,
        rd_kafka_interceptor_f_on_consume_t *on_consume,
        void *ic_opaque);


/**
 * @brief Append an on_commit() interceptor.
 *
 * @param rk Client instance.
 * @param ic_name Interceptor name, used in logging.
 * @param on_commit() Function pointer.
 * @param ic_opaque Opaque value that will be passed to the function.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or
RD_KAFKA_RESP_ERR__CONFLICT
 *          if an existing intercepted with the same \p ic_name and function
 *          has already been added to \p conf.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_interceptor_add_on_commit (
        rd_kafka_t *rk, const char *ic_name,
        rd_kafka_interceptor_f_on_commit_t *on_commit,
        void *ic_opaque);




/**@}*/


#ifdef __cplusplus
}
#endif
```

librdkafka 0.9.0

Apache librdkafka 0.9.0 is supported as of MapR 5.2.1 through MapR 6.0.0.

```
/*
 * librdkafka - Apache Kafka C library
 *
 * Copyright (c) 2012-2013 Magnus Edenhill
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are
met:
 *
 * 1. Redistributions of source code must retain the above copyright notice,
 *    this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
notice,
 *    this list of conditions and the following disclaimer in the
```

```
documentation
 *    and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS"
 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
 * LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
 * CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
 * SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
 * INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
 * CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
 * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF
THE
 * POSSIBILITY OF SUCH DAMAGE.
 */

/**
 * @file rdkafka.h
 * @brief Apache Kafka C/C++ consumer and producer client library.
 *
 * rdkafka.h contains the public API for librdkafka.
 * The API is documented in this file as comments prefixing the function,
type,
 * enum, define, etc.
 *
 * @sa For the C++ interface see rdkafkacpp.h
 *
 * @tableofcontents
 */


/* @cond NO_DOC */
#pragma once

#include <stdio.h>
#include <inttypes.h>
#include <sys/types.h>
#include "streams_util.h"

#ifdef __cplusplus
extern "C" {
#if 0
} /* Restore indent */
#endif
#endif

#ifdef _MSC_VER
#include <basetsd.h>
typedef SSIZE_T ssize_t;
#define RD_UNUSED
#define RD_INLINE __inline
#define RD_DEPRECATED
#undef RD_EXPORT
#ifdef LIBRDKAFKA_EXPORTS
#define RD_EXPORT __declspec(dllexport)
#else
#define RD_EXPORT __declspec(dllimport)
#endif

#else
```

```
#define RD_UNUSED __attribute__((unused))
#define RD_INLINE inline
#define RD_EXPORT
#define RD_DEPRECATED __attribute__((deprecated))
#endif
/* @endcond */



/**
 * @name librdkafka version
 * @{
 *
 *
 */

/**
 * @brief librdkafka version
 *
 * Interpreted as hex \c MM.mm.rr.xx:
 *  - MM = Major
 *  - mm = minor
 *  - rr = revision
 *  - xx = pre-release id (0xff is the final release)
 *
 * E.g.: \c 0x000801ff = 0.8.1
 *
 * @remark This value should only be used during compile time,
 *         for runtime checks of version use rd_kafka_version()
 */
#define RD_KAFKA_VERSION  0x000901ff
#define STREAMS_MIN_VERSION "5.2.1"

/**
 * @brief Returns the librdkafka version as integer.
 *
 * @returns Version integer.
 *
 * @sa See RD_KAFKA_VERSION for how to parse the integer format.
 * @sa Use rd_kafka_version_str() to retreive the version as a string.
 */
RD_EXPORT
int rd_kafka_version(void);

/**
 * @brief Returns the librdkafka version as string.
 *
 * @returns Version string
 */
RD_EXPORT
const char *rd_kafka_version_str (void);

/**@}*/


/**
 * @name Constants, errors, types
 * @{
 *
 *
 */


/**
```

```
 * @enum rd_kafka_type_t
 *
 * @brief rd_kafka_t handle type.
 *
 * @sa rd_kafka_new()
 */
typedef enum rd_kafka_type_t {
    RD_KAFKA_PRODUCER, /**< Producer client */
    RD_KAFKA_CONSUMER  /**< Consumer client */
} rd_kafka_type_t;


/**
 * @enum Timestamp types
 *
 * @sa rd_kafka_message_timestamp()
 */
typedef enum rd_kafka_timestamp_type_t {
    RD_KAFKA_TIMESTAMP_NOT_AVAILABLE,   /**< Timestamp not available */
    RD_KAFKA_TIMESTAMP_CREATE_TIME,     /**< Message creation time */
    RD_KAFKA_TIMESTAMP_LOG_APPEND_TIME  /**< Log append time */
} rd_kafka_timestamp_type_t;



/**
 * @brief Retrieve supported debug contexts for use with the \c \"debug\"
 *        configuration property. (runtime)
 *
 * @returns Comma-separated list of available debugging contexts.
 */
RD_EXPORT
const char *rd_kafka_get_debug_contexts(void);

/**
 * @brief Supported debug contexts. (compile time)
 *
 * @deprecated This compile time value may be outdated at runtime due to
 *             linking another version of the library.
 *             Use rd_kafka_get_debug_contexts() instead.
 */
#define RD_KAFKA_DEBUG_CONTEXTS \

"all,generic,broker,topic,metadata,producer,queue,msg,protocol,cgrp,security
,fetch"


/* @cond NO_DOC */
/* Private types to provide ABI compatibility */
typedef struct rd_kafka_s rd_kafka_t;
typedef struct rd_kafka_topic_s rd_kafka_topic_t;
typedef struct rd_kafka_conf_s rd_kafka_conf_t;
typedef struct rd_kafka_topic_conf_s rd_kafka_topic_conf_t;
typedef struct rd_kafka_queue_s rd_kafka_queue_t;
/* @endcond */


/**
 * @enum rd_kafka_resp_err_t
 * @brief Error codes.
 *
 * The negative error codes delimited by two underscores
 * (\c RD_KAFKA_RESP_ERR__..) denotes errors internal to librdkafka and are
 * displayed as \c \"Local: \<error string..\>\", while the error codes
```


last-updated: Apr 11, 2022

```
 * delimited by a single underscore (\c RD_KAFKA_RESP_ERR_..) denote broker
 * errors and are displayed as \c \"Broker: \<error string..\>\".
 *
 * @sa Use rd_kafka_err2str() to translate an error code a human readable
string
 */
typedef enum {
    /* Internal errors to rdkafka: */
    /** Begin internal error codes */
    RD_KAFKA_RESP_ERR__BEGIN = -200,
    /** Received message is incorrect */
    RD_KAFKA_RESP_ERR__BAD_MSG = -199,
    /** Bad/unknown compression */
    RD_KAFKA_RESP_ERR__BAD_COMPRESSION = -198,
    /** Broker is going away */
    RD_KAFKA_RESP_ERR__DESTROY = -197,
    /** Generic failure */
    RD_KAFKA_RESP_ERR__FAIL = -196,
    /** Broker transport failure */
    RD_KAFKA_RESP_ERR__TRANSPORT = -195,
    /** Critical system resource */
    RD_KAFKA_RESP_ERR__CRIT_SYS_RESOURCE = -194,
    /** Failed to resolve broker */
    RD_KAFKA_RESP_ERR__RESOLVE = -193,
    /** Produced message timed out*/
    RD_KAFKA_RESP_ERR__MSG_TIMED_OUT = -192,
    /** Reached the end of the topic+partition queue on
     * the broker. Not really an error. */
    RD_KAFKA_RESP_ERR__PARTITION_EOF = -191,
    /** Permanent: Partition does not exist in cluster. */
    RD_KAFKA_RESP_ERR__UNKNOWN_PARTITION = -190,
    /** File or filesystem error */
    RD_KAFKA_RESP_ERR__FS = -189,
     /** Permanent: Topic does not exist in cluster. */
    RD_KAFKA_RESP_ERR__UNKNOWN_TOPIC = -188,
    /** All broker connections are down. */
    RD_KAFKA_RESP_ERR__ALL_BROKERS_DOWN = -187,
    /** Invalid argument, or invalid configuration */
    RD_KAFKA_RESP_ERR__INVALID_ARG = -186,
    /** Operation timed out */
    RD_KAFKA_RESP_ERR__TIMED_OUT = -185,
    /** Queue is full */
    RD_KAFKA_RESP_ERR__QUEUE_FULL = -184,
    /** ISR count < required.acks */
        RD_KAFKA_RESP_ERR__ISR_INSUFF = -183,
    /** Broker node update */
        RD_KAFKA_RESP_ERR__NODE_UPDATE = -182,
    /** SSL error */
    RD_KAFKA_RESP_ERR__SSL = -181,
    /** Waiting for coordinator to become available. */
        RD_KAFKA_RESP_ERR__WAIT_COORD = -180,
    /** Unknown client group */
        RD_KAFKA_RESP_ERR__UNKNOWN_GROUP = -179,
    /** Operation in progress */
        RD_KAFKA_RESP_ERR__IN_PROGRESS = -178,
     /** Previous operation in progress, wait for it to finish. */
        RD_KAFKA_RESP_ERR__PREV_IN_PROGRESS = -177,
     /** This operation would interfere with an existing subscription */
        RD_KAFKA_RESP_ERR__EXISTING_SUBSCRIPTION = -176,
    /** Assigned partitions (rebalance_cb) */
        RD_KAFKA_RESP_ERR__ASSIGN_PARTITIONS = -175,
    /** Revoked partitions (rebalance_cb) */
        RD_KAFKA_RESP_ERR__REVOKE_PARTITIONS = -174,
    /** Conflicting use */
```

```
        RD_KAFKA_RESP_ERR__CONFLICT = -173,
    /** Wrong state */
        RD_KAFKA_RESP_ERR__STATE = -172,
    /** Unknown protocol */
        RD_KAFKA_RESP_ERR__UNKNOWN_PROTOCOL = -171,
    /** Not implemented */
        RD_KAFKA_RESP_ERR__NOT_IMPLEMENTED = -170,
    /** Authentication failure*/
    RD_KAFKA_RESP_ERR__AUTHENTICATION = -169,
    /** No stored offset */
    RD_KAFKA_RESP_ERR__NO_OFFSET = -168,
    /** Outdated */
    RD_KAFKA_RESP_ERR__OUTDATED = -167,
    /** End internal error codes */
    RD_KAFKA_RESP_ERR__END = -100,

    /* Kafka broker errors: */
    /** Unknown broker error */
    RD_KAFKA_RESP_ERR_UNKNOWN = -1,
    /** Success */
    RD_KAFKA_RESP_ERR_NO_ERROR = 0,
    /** Offset out of range */
    RD_KAFKA_RESP_ERR_OFFSET_OUT_OF_RANGE = 1,
    /** Invalid message */
    RD_KAFKA_RESP_ERR_INVALID_MSG = 2,
    /** Unknown topic or partition */
    RD_KAFKA_RESP_ERR_UNKNOWN_TOPIC_OR_PART = 3,
    /** Invalid message size */
    RD_KAFKA_RESP_ERR_INVALID_MSG_SIZE = 4,
    /** Leader not available */
    RD_KAFKA_RESP_ERR_LEADER_NOT_AVAILABLE = 5,
    /** Not leader for partition */
    RD_KAFKA_RESP_ERR_NOT_LEADER_FOR_PARTITION = 6,
    /** Request timed out */
    RD_KAFKA_RESP_ERR_REQUEST_TIMED_OUT = 7,
    /** Broker not available */
    RD_KAFKA_RESP_ERR_BROKER_NOT_AVAILABLE = 8,
    /** Replica not available */
    RD_KAFKA_RESP_ERR_REPLICA_NOT_AVAILABLE = 9,
    /** Message size too large */
    RD_KAFKA_RESP_ERR_MSG_SIZE_TOO_LARGE = 10,
    /** StaleControllerEpochCode */
    RD_KAFKA_RESP_ERR_STALE_CTRL_EPOCH = 11,
    /** Offset metadata string too large */
    RD_KAFKA_RESP_ERR_OFFSET_METADATA_TOO_LARGE = 12,
    /** Broker disconnected before response received */
    RD_KAFKA_RESP_ERR_NETWORK_EXCEPTION = 13,
    /** Group coordinator load in progress */
        RD_KAFKA_RESP_ERR_GROUP_LOAD_IN_PROGRESS = 14,
     /** Group coordinator not available */
        RD_KAFKA_RESP_ERR_GROUP_COORDINATOR_NOT_AVAILABLE = 15,
    /** Not coordinator for group */
        RD_KAFKA_RESP_ERR_NOT_COORDINATOR_FOR_GROUP = 16,
    /** Invalid topic */
        RD_KAFKA_RESP_ERR_TOPIC_EXCEPTION = 17,
    /** Message batch larger than configured server segment size */
        RD_KAFKA_RESP_ERR_RECORD_LIST_TOO_LARGE = 18,
    /** Not enough in-sync replicas */
        RD_KAFKA_RESP_ERR_NOT_ENOUGH_REPLICAS = 19,
    /** Message(s) written to insufficient number of in-sync replicas */
        RD_KAFKA_RESP_ERR_NOT_ENOUGH_REPLICAS_AFTER_APPEND = 20,
    /** Invalid required acks value */
        RD_KAFKA_RESP_ERR_INVALID_REQUIRED_ACKS = 21,
    /** Specified group generation id is not valid */
```

```
        RD_KAFKA_RESP_ERR_ILLEGAL_GENERATION = 22,
    /** Inconsistent group protocol */
        RD_KAFKA_RESP_ERR_INCONSISTENT_GROUP_PROTOCOL = 23,
    /** Invalid group.id */
    RD_KAFKA_RESP_ERR_INVALID_GROUP_ID = 24,
    /** Unknown member */
        RD_KAFKA_RESP_ERR_UNKNOWN_MEMBER_ID = 25,
    /** Invalid session timeout */
        RD_KAFKA_RESP_ERR_INVALID_SESSION_TIMEOUT = 26,
    /** Group rebalance in progress */
    RD_KAFKA_RESP_ERR_REBALANCE_IN_PROGRESS = 27,
    /** Commit offset data size is not valid */
        RD_KAFKA_RESP_ERR_INVALID_COMMIT_OFFSET_SIZE = 28,
    /** Topic authorization failed */
        RD_KAFKA_RESP_ERR_TOPIC_AUTHORIZATION_FAILED = 29,
    /** Group authorization failed */
    RD_KAFKA_RESP_ERR_GROUP_AUTHORIZATION_FAILED = 30,
    /** Cluster authorization failed */
    RD_KAFKA_RESP_ERR_CLUSTER_AUTHORIZATION_FAILED = 31,
    /** Invalid timestamp */
    RD_KAFKA_RESP_ERR_INVALID_TIMESTAMP = 32,
    /** Unsupported SASL mechanism */
    RD_KAFKA_RESP_ERR_UNSUPPORTED_SASL_MECHANISM = 33,
    /** Illegal SASL state */
    RD_KAFKA_RESP_ERR_ILLEGAL_SASL_STATE = 34,
    /** Unuspported version */
    RD_KAFKA_RESP_ERR_UNSUPPORTED_VERSION = 35,

    RD_KAFKA_RESP_ERR_END_ALL,
} rd_kafka_resp_err_t;


/**
 * @brief Error code value, name and description.
 *        Typically for use with language bindings to automatically expose
 *        the full set of librdkafka error codes.
 */
struct rd_kafka_err_desc {
    rd_kafka_resp_err_t code;/**< Error code */
    const char *name;      /**< Error name, same as code enum sans prefix */
    const char *desc;      /**< Human readable error description. */
};


/**
 * @brief Returns the full list of error codes.
 */
RD_EXPORT
void rd_kafka_get_err_descs (const struct rd_kafka_err_desc **errdescs,
                 size_t *cntp);




/**
 * @brief Returns a human readable representation of a kafka error.
 *
 * @param err Error code to translate
 */
RD_EXPORT
const char *rd_kafka_err2str (rd_kafka_resp_err_t err);
```

```
/**
 * @brief Returns the error code name (enum name).
 *
 * @param err Error code to translate
 */
RD_EXPORT
const char *rd_kafka_err2name (rd_kafka_resp_err_t err);


/**
 * @brief Returns the last error code generated by a legacy API call
 *        in the current thread.
 *
 * The legacy APIs are the ones using errno to propagate error value,
namely:
 *  - rd_kafka_topic_new()
 *  - rd_kafka_consume_start()
 *  - rd_kafka_consume_stop()
 *  - rd_kafka_consume()
 *  - rd_kafka_consume_batch()
 *  - rd_kafka_consume_callback()
 *  - rd_kafka_consume_queue()
 *  - rd_kafka_produce()
 *
 * The main use for this function is to avoid converting system \p errno
 * values to rd_kafka_resp_err_t codes for legacy APIs.
 *
 * @remark The last error is stored per-thread, if multiple rd_kafka_t
handles
 *         are used in the same application thread the developer needs to
 *         make sure rd_kafka_last_error() is called immediately after
 *         a failed API call.
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_last_error (void);


/**
 * @brief Converts the system errno value \p errnox to a rd_kafka_resp_err_t
 *        error code upon failure from the following functions:
 *  - rd_kafka_topic_new()
 *  - rd_kafka_consume_start()
 *  - rd_kafka_consume_stop()
 *  - rd_kafka_consume()
 *  - rd_kafka_consume_batch()
 *  - rd_kafka_consume_callback()
 *  - rd_kafka_consume_queue()
 *  - rd_kafka_produce()
 *
 * @param errnox  System errno value to convert
 *
 * @returns Appropriate error code for \p errnox
 *
 * @remark A better alternative is to call rd_kafka_last_error() immediately
 *         after any of the above functions return -1 or NULL.
 *
 * @sa rd_kafka_last_error()
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_errno2err(int errnox);


/**
 * @brief Returns the thread-local system errno
```

```
 *
 * On most platforms this is the same as \p errno but in case of different
 * runtimes between library and application (e.g., Windows static DLLs)
 * this provides a means for expsing the errno librdkafka uses.
 *
 * @remark The value is local to the current calling thread.
 */
RD_EXPORT
int rd_kafka_errno (void);



/**
 * @brief Topic+Partition place holder
 *
 * Generic place holder for a Topic+Partition and its related information
 * used for multiple purposes:
 *   - consumer offset (see rd_kafka_commit(), et.al.)
 *   - group rebalancing callback (rd_kafka_conf_set_rebalance_cb())
 *   - offset commit result callback (rd_kafka_conf_set_offset_commit_cb())
 */

/**
 * @brief Generic place holder for a specific Topic+Partition.
 *
 * @sa rd_kafka_topic_partition_list_new()
 */
typedef struct rd_kafka_topic_partition_s {
        char        *topic;             /**< Topic name */
        int32_t      partition;         /**< Partition */
    int64_t      offset;            /**< Offset */
        void        *metadata;          /**< Metadata */
        size_t       metadata_size;     /**< Metadata size */
        void        *opaque;            /**< Application opaque */
        rd_kafka_resp_err_t err;        /**< Error code, depending on use.
*/
        void       *_private;           /**< INTERNAL USE ONLY,
                                         *   INITIALIZE TO ZERO, DO NOT
TOUCH */
} rd_kafka_topic_partition_t;



/**
 * @brief A growable list of Topic+Partitions.
 *
 */
typedef struct rd_kafka_topic_partition_list_s {
        int cnt;               /**< Current number of elements */
        int size;              /**< Current allocated size */
        rd_kafka_topic_partition_t *elems; /**< Element array[] */
} rd_kafka_topic_partition_list_t;


/**
 * @brief Create a new list/vector Topic+Partition container.
 *
 * @param size  Initial allocated size used when the expected number of
 *              elements is known or can be estimated.
 *              Avoids reallocation and possibly relocation of the
 *              elems array.
 *
 * @returns A newly allocated Topic+Partition list.
 *
```

```
 * @remark Use rd_kafka_topic_partition_list_destroy() to free all resources
 *         in use by a list and the list itself.
 * @sa     rd_kafka_topic_partition_list_add()
 */
RD_EXPORT
rd_kafka_topic_partition_list_t *rd_kafka_topic_partition_list_new (int
size);


/**
 * @brief Free all resources used by the list and the list itself.
 */
RD_EXPORT
void
rd_kafka_topic_partition_list_destroy (rd_kafka_topic_partition_list_t
*rkparlist);

/**
 * @brief Add topic+partition to list
 *
 * @param rktparlist List to extend
 * @param topic      Topic name (copied)
 * @param partition  Partition id
 *
 * @returns The object which can be used to fill in additionals fields.
 */
RD_EXPORT
rd_kafka_topic_partition_t *
rd_kafka_topic_partition_list_add (rd_kafka_topic_partition_list_t
*rktparlist,
                                   const char *topic, int32_t partition);


/**
 * @brief Add range of partitions from \p start to \p stop inclusive.
 *
 * @param rktparlist List to extend
 * @param topic      Topic name (copied)
 * @param start      Start partition of range
 * @param stop       Last partition of range (inclusive)
 */
RD_EXPORT
void
rd_kafka_topic_partition_list_add_range (rd_kafka_topic_partition_list_t
                                         *rktparlist,
                                         const char *topic,
                                         int32_t start, int32_t stop);



/**
 * @brief Delete partition from list.
 *
 * @param rktparlist List to modify
 * @param topic      Topic name to match
 * @param partition  Partition to match
 *
 * @returns 1 if partition was found (and removed), else 0.
 *
 * @remark Any held indices to elems[] are unusable after this call returns
1.
 */
RD_EXPORT
int
```

```
rd_kafka_topic_partition_list_del (rd_kafka_topic_partition_list_t
*rktparlist,
                   const char *topic, int32_t partition);


/**
 * @brief Delete partition from list by elems[] index.
 *
 * @returns 1 if partition was found (and removed), else 0.
 *
 * @sa rd_kafka_topic_partition_list_del()
 */
RD_EXPORT
int
rd_kafka_topic_partition_list_del_by_idx (
    rd_kafka_topic_partition_list_t *rktparlist,
    int idx);


/**
 * @brief Make a copy of an existing list.
 *
 * @param src   The existing list to copy.
 *
 * @returns A new list fully populated to be identical to \p src
 */
RD_EXPORT
rd_kafka_topic_partition_list_t *
rd_kafka_topic_partition_list_copy (const rd_kafka_topic_partition_list_t
*src);




/**
 * @brief Set offset to \p offset for \p topic and \p partition
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or
 *          RD_KAFKA_RESP_ERR__UNKNOWN_PARTITION if \p partition was not
found
 *          in the list.
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_topic_partition_list_set_offset (
    rd_kafka_topic_partition_list_t *rktparlist,
    const char *topic, int32_t partition, int64_t offset);



/**
 * @brief Find element by \p topic and \p partition.
 *
 * @returns a pointer to the first matching element, or NULL if not found.
 */
RD_EXPORT
rd_kafka_topic_partition_t *
rd_kafka_topic_partition_list_find (rd_kafka_topic_partition_list_t
*rktparlist,
                    const char *topic, int32_t partition);

/**@}*/
```

```
/**
 * @name Kafka messages
 * @{
 *
 */



// FIXME: This doesn't show up in docs for some reason
// "Compound rd_kafka_message_t is not documented."

/**
 * @brief A Kafka message as returned by the \c rd_kafka_consume*() family
 *        of functions as well as provided to the Producer \c dr_msg_cb().
 *
 * For the consumer this object has two purposes:
 *  - provide the application with a consumed message. (\c err == 0)
 *  - report per-topic+partition consumer errors (\c err != 0)
 *
 * The application must check \c err to decide what action to take.
 *
 * When the application is finished with a message it must call
 * rd_kafka_message_destroy() unless otherwise noted.
 */
typedef struct rd_kafka_message_s {
    rd_kafka_resp_err_t err;   /**< Non-zero for error signaling. */
    rd_kafka_topic_t *rkt;     /**< Topic */
    int32_t partition;         /**< Partition */
    void   *payload;           /**< Producer: original message payload.
                    * Consumer: Depends on the value of \c err :
                    * - \c err==0: Message payload.
                    * - \c err!=0: Error string */
    size_t  len;               /**< Depends on the value of \c err :
                    * - \c err==0: Message payload length
                    * - \c err!=0: Error string length */
    void   *key;               /**< Depends on the value of \c err :
                    * - \c err==0: Optional message key */
    size_t  key_len;           /**< Depends on the value of \c err :
                    * - \c err==0: Optional message key length*/
    int64_t offset;            /**< Consume:
                                    * - Message offset (or offset for error
                    *   if \c err!=0 if applicable).
                                    * - dr_msg_cb:
                                    *   Message offset assigned by broker.
                                    *   If \c produce.offset.report is set
then
                                    *   each message will have this field
set,
                                    *   otherwise only the last message in
                                    *   each produced internal batch will
                                    *   have this field set, otherwise 0. */
    void  *_private;           /**< Consume:
                    *  - rdkafka private pointer: DO NOT MODIFY
                    *  - dr_msg_cb:
                                    *    msg_opaque from produce() call */
    bool is_streams_message;

    streams_consumer_record_t *_streams_consumer_record;  /**< Streams
record
                                    * associated with this message */

} rd_kafka_message_t;
```

```
/**
 * @brief Frees resources for \p rkmessage and hands ownership back to
rdkafka.
 */
RD_EXPORT
void rd_kafka_message_destroy(rd_kafka_message_t *rkmessage);



/**
 * @brief Returns the error string for an errored rd_kafka_message_t or
NULL if
 *        there was no error.
 */
static RD_INLINE const char *
RD_UNUSED
rd_kafka_message_errstr(const rd_kafka_message_t *rkmessage) {
    if (!rkmessage || !rkmessage->err)
        return NULL;

    if (rkmessage->payload)
        return (const char *)rkmessage->payload;

    return rd_kafka_err2str(rkmessage->err);
}



/**
 * @brief Returns the message timestamp for a consumed message.
 *
 * The timestamp is the number of milliseconds since the epoch (UTC).
 *
 * \p tstype is updated to indicate the type of timestamp.
 *
 * @returns message timestamp, or -1 if not available.
 *
 * @remark Message timestamps require broker version 0.10.0 or later.
 */
RD_EXPORT
int64_t rd_kafka_message_timestamp (const rd_kafka_message_t *rkmessage,
                    rd_kafka_timestamp_type_t *tstype);



/**@}*/


/**
 * @name Configuration interface
 * @{
 *
 * @brief Main/global configuration property interface
 *
 */

/**
 * @enum rd_kafka_conf_res_t
 * @brief Configuration result type
 */
typedef enum {
    RD_KAFKA_CONF_UNKNOWN = -2, /**< Unknown configuration name. */
    RD_KAFKA_CONF_INVALID = -1, /**< Invalid configuration value. */
```

```
    RD_KAFKA_CONF_OK = 0        /**< Configuration okay */
} rd_kafka_conf_res_t;


/**
 * @brief Create configuration object.
 *
 * When providing your own configuration to the \c rd_kafka_*_new_*() calls
 * the rd_kafka_conf_t objects needs to be created with this function
 * which will set up the defaults.
 * I.e.:
 * @code
 *   rd_kafka_conf_t *myconf;
 *   rd_kafka_conf_res_t res;
 *
 *   myconf = rd_kafka_conf_new();
 *   res = rd_kafka_conf_set(myconf, "socket.timeout.ms", "600",
 *                           errstr, sizeof(errstr));
 *   if (res != RD_KAFKA_CONF_OK)
 *      die("%s\n", errstr);
 *
 *   rk = rd_kafka_new(..., myconf);
 * @endcode
 *
 * Please see CONFIGURATION.md for the default settings or use
 * rd_kafka_conf_properties_show() to provide the information at runtime.
 *
 * The properties are identical to the Apache Kafka configuration properties
 * whenever possible.
 *
 * @returns A new rd_kafka_conf_t object with defaults set.
 *
 * @sa rd_kafka_conf_set(), rd_kafka_conf_destroy()
 */
RD_EXPORT
rd_kafka_conf_t *rd_kafka_conf_new(void);


/**
 * @brief Destroys a conf object.
 */
RD_EXPORT
void rd_kafka_conf_destroy(rd_kafka_conf_t *conf);


/**
 * @brief Creates a copy/duplicate of configuration object \p conf
 */
RD_EXPORT
rd_kafka_conf_t *rd_kafka_conf_dup(const rd_kafka_conf_t *conf);


/**
 * @brief Sets a configuration property.
 *
 * \p must have been previously created with rd_kafka_conf_new().
 *
 * Returns \c rd_kafka_conf_res_t to indicate success or failure.
 * In case of failure \p errstr is updated to contain a human readable
 * error string.
 */
RD_EXPORT
rd_kafka_conf_res_t rd_kafka_conf_set(rd_kafka_conf_t *conf,
                      const char *name,
```

```
                      const char *value,
                      char *errstr, size_t errstr_size);


/**
 @deprecated See rd_kafka_conf_set_dr_msg_cb()
*/
RD_EXPORT
void rd_kafka_conf_set_dr_cb(rd_kafka_conf_t *conf,
                  void (*dr_cb) (rd_kafka_t *rk,
                        void *payload, size_t len,
                        rd_kafka_resp_err_t err,
                        void *opaque, void *msg_opaque));

/**
 * @brief \b Producer: Set delivery report callback in provided \p conf
object.
 *
 * The delivery report callback will be called once for each message
 * accepted by rd_kafka_produce() (et.al) with \p err set to indicate
 * the result of the produce request.
 *
 * The callback is called when a message is succesfully produced or
 * if librdkafka encountered a permanent failure, or the retry counter for
 * temporary errors has been exhausted.
 *
 * An application must call rd_kafka_poll() at regular intervals to
 * serve queued delivery report callbacks.
 */
RD_EXPORT
void rd_kafka_conf_set_dr_msg_cb(rd_kafka_conf_t *conf,
                                 void (*dr_msg_cb) (rd_kafka_t *rk,
                                                    const
rd_kafka_message_t *
                                                    rkmessage,
                                                    void *opaque));


/**
 * @brief \b Consumer: Set consume callback for use with
rd_kafka_consumer_poll()
 *
 */
RD_EXPORT
void rd_kafka_conf_set_consume_cb (rd_kafka_conf_t *conf,
                                   void (*consume_cb) (rd_kafka_message_t *
                                                       rkmessage,
                                                       void *opaque));

/**
 * @brief \b Consumer: Set rebalance callback for use with
 *                     coordinated consumer group balancing.
 *
 * The \p err field is set to either RD_KAFKA_RESP_ERR__ASSIGN_PARTITIONS
 * or RD_KAFKA_RESP_ERR__REVOKE_PARTITIONS and 'partitions'
 * contains the full partition set that was either assigned or revoked.
 *
 * Registering a \p rebalance_cb turns off librdkafka's automatic
 * partition assignment/revocation and instead delegates that responsibility
 * to the application's \p rebalance_cb.
 *
 * The rebalance callback is responsible for updating librdkafka's
 * assignment set based on the two events:
RD_KAFKA_RESP_ERR__ASSIGN_PARTITIONS
```

```
 * and RD_KAFKA_RESP_ERR__REVOKE_PARTITIONS but should also be able to
handle
 * arbitrary rebalancing failures where \p err is neither of those.
 * @remark In this latter case (arbitrary error), the application must
 *         call rd_kafka_assign(rk, NULL) to synchronize state.
 *
 * Without a rebalance callback this is done automatically by librdkafka
 * but registering a rebalance callback gives the application flexibility
 * in performing other operations along with the assinging/revocation,
 * such as fetching offsets from an alternate location (on assign)
 * or manually committing offsets (on revoke).
 *
 * @remark The \p partitions list is destroyed by librdkafka on return
 *         return from the rebalance_cb and must not be freed or
 *         saved by the application.
 *
 * The following example shows the application's responsibilities:
 * @code
 *    static void rebalance_cb (rd_kafka_t *rk, rd_kafka_resp_err_t err,
 *                              rd_kafka_topic_partition_list_t *partitions,
 *                              void *opaque) {
 *
 *        switch (err)
 *        {
 *          case RD_KAFKA_RESP_ERR__ASSIGN_PARTITIONS:
 *             // application may load offets from arbitrary external
 *             // storage here and update \p partitions
 *
 *             rd_kafka_assign(rk, partitions);
 *             break;
 *
 *          case RD_KAFKA_RESP_ERR__REVOKE_PARTITIONS:
 *             if (manual_commits) // Optional explicit manual commit
 *                 rd_kafka_commit(rk, partitions, 0); // sync commit
 *
 *             rd_kafka_assign(rk, NULL);
 *             break;
 *
 *          default:
 *             handle_unlikely_error(err);
 *             rd_kafka_assign(rk, NULL); // sync state
 *             break;
 *         }
 *    }
 * @endcode
 */
RD_EXPORT
void rd_kafka_conf_set_rebalance_cb (
        rd_kafka_conf_t *conf,
        void (*rebalance_cb) (rd_kafka_t *rk,
                              rd_kafka_resp_err_t err,
                              rd_kafka_topic_partition_list_t *partitions,
                              void *opaque));



/**
 * @brief \b Consumer: Set offset commit callback for use with consumer
groups.
 *
 * The results of automatic or manual offset commits will be scheduled
 * for this callback and is served by rd_kafka_consumer_poll().
 *
 * If no partitions had valid offsets to commit this callback will be called
```

```
 * with \p err == RD_KAFKA_RESP_ERR__NO_OFFSET which is not to be considered
 * an error.
 *
 * The \p offsets list contains per-partition information:
 *   - \c offset: committed offset (attempted)
 *   - \c err:    commit error
 */
RD_EXPORT
void rd_kafka_conf_set_offset_commit_cb (
        rd_kafka_conf_t *conf,
        void (*offset_commit_cb) (rd_kafka_t *rk,
                                  rd_kafka_resp_err_t err,
                                  rd_kafka_topic_partition_list_t *offsets,
                                  void *opaque));


/**
 * @brief Set error callback in provided conf object.
 *
 * The error callback is used by librdkafka to signal critical errors
 * back to the application.
 *
 * If no \p error_cb is registered then the errors will be logged instead.
 */
RD_EXPORT
void rd_kafka_conf_set_error_cb(rd_kafka_conf_t *conf,
                 void  (*error_cb) (rd_kafka_t *rk, int err,
                           const char *reason,
                           void *opaque));

/**
 * @brief Set throttle callback.
 *
 * The throttle callback is used to forward broker throttle times to the
 * application for Produce and Fetch (consume) requests.
 *
 * Callbacks are triggered whenever a non-zero throttle time is returned by
 * the broker, or when the throttle time drops back to zero.
 *
 * An application must call rd_kafka_poll() or rd_kafka_consumer_poll() at
 * regular intervals to serve queued callbacks.
 *
 * @remark Requires broker version 0.9.0 or later.
 */
RD_EXPORT
void rd_kafka_conf_set_throttle_cb (rd_kafka_conf_t *conf,
                    void (*throttle_cb) (
                        rd_kafka_t *rk,
                        const char *broker_name,
                        int32_t broker_id,
                        int throttle_time_ms,
                        void *opaque));


/**
 * @brief Set logger callback.
 *
 * The default is to print to stderr, but a syslog logger is also available,
 * see rd_kafka_log_print and rd_kafka_log_syslog for the builtin
alternatives.
 * Alternatively the application may provide its own logger callback.
 * Or pass \p func as NULL to disable logging.
 *
 * This is the configuration alternative to the deprecated
```

```
rd_kafka_set_logger()
 */
RD_EXPORT
void rd_kafka_conf_set_log_cb(rd_kafka_conf_t *conf,
              void (*log_cb) (const rd_kafka_t *rk, int level,
                                          const char *fac, const char
*buf));


/**
 * @brief Set statistics callback in provided conf object.
 *
 * The statistics callback is triggered from rd_kafka_poll() every
 * \c statistics.interval.ms (needs to be configured separately).
 * Function arguments:
 *   - \p rk - Kafka handle
 *   - \p json - String containing the statistics data in JSON format
 *   - \p json_len - Length of \p json string.
 *   - \p opaque - Application-provided opaque.
 *
 * If the application wishes to hold on to the \p json pointer and free
 * it at a later time it must return 1 from the \p stats_cb.
 * If the application returns 0 from the \p stats_cb then librdkafka
 * will immediately free the \p json pointer.
 */
RD_EXPORT
void rd_kafka_conf_set_stats_cb(rd_kafka_conf_t *conf,
                 int (*stats_cb) (rd_kafka_t *rk,
                          char *json,
                          size_t json_len,
                          void *opaque));



/**
 * @brief Set socket callback.
 *
 * The socket callback is responsible for opening a socket
 * according to the supplied \p domain, \p type and \p protocol.
 * The socket shall be created with \c CLOEXEC set in a racefree fashion, if
 * possible.
 *
 * Default:
 *  - on linux: racefree CLOEXEC
 *  - others  : non-racefree CLOEXEC
 */
RD_EXPORT
void rd_kafka_conf_set_socket_cb(rd_kafka_conf_t *conf,
                                 int (*socket_cb) (int domain, int type,
                                                   int protocol,
                                                   void *opaque));


#ifndef _MSC_VER
/**
 * @brief Set open callback.
 *
 * The open callback is responsible for opening the file specified by
 * pathname, flags and mode.
 * The file shall be opened with \c CLOEXEC set in a racefree fashion, if
 * possible.
 *
 * Default:
 *  - on linux: racefree CLOEXEC
```

```
 *  - others  : non-racefree CLOEXEC
 */
RD_EXPORT
void rd_kafka_conf_set_open_cb (rd_kafka_conf_t *conf,
                                int (*open_cb) (const char *pathname,
                                                int flags, mode_t mode,
                                                void *opaque));
#endif

/**
 * @brief Sets the application's opaque pointer that will be passed to
callbacks
 */
RD_EXPORT
void rd_kafka_conf_set_opaque(rd_kafka_conf_t *conf, void *opaque);

/**
 * @brief Retrieves the opaque pointer previously set with
rd_kafka_conf_set_opaque()
 */
RD_EXPORT
void *rd_kafka_opaque(const rd_kafka_t *rk);



/**
 * Sets the default topic configuration to use for automatically
 * subscribed topics (e.g., through pattern-matched topics).
 * The topic config object is not usable after this call.
 */
RD_EXPORT
void rd_kafka_conf_set_default_topic_conf (rd_kafka_conf_t *conf,
                                           rd_kafka_topic_conf_t *tconf);



/**
 * @brief Retrieve configuration value for property \p name.
 *
 * If \p dest is non-NULL the value will be written to \p dest with at
 * most \p dest_size.
 *
 * \p *dest_size is updated to the full length of the value, thus if
 * \p *dest_size initially is smaller than the full length the application
 * may reallocate \p dest to fit the returned \p *dest_size and try again.
 *
 * If \p dest is NULL only the full length of the value is returned.
 *
 * Returns \p RD_KAFKA_CONF_OK if the property name matched, else
 * \p RD_KAFKA_CONF_UNKNOWN.
 */
RD_EXPORT
rd_kafka_conf_res_t rd_kafka_conf_get (const rd_kafka_conf_t *conf,
                                       const char *name,
                                       char *dest, size_t *dest_size);


/**
 * @brief Retrieve topic configuration value for property \p name.
 *
 * @sa rd_kafka_conf_get()
 */
RD_EXPORT
rd_kafka_conf_res_t rd_kafka_topic_conf_get (const rd_kafka_topic_conf_t
```

```
*conf,
                                                const char *name,
                                                char *dest, size_t *dest_size);


/**
 * @brief Dump the configuration properties and values of \p conf to an
array
 *        with \"key\", \"value\" pairs.
 *
 * The number of entries in the array is returned in \p *cntp.
 *
 * The dump must be freed with `rd_kafka_conf_dump_free()`.
 */
RD_EXPORT
const char **rd_kafka_conf_dump(rd_kafka_conf_t *conf, size_t *cntp);


/**
 * @brief Dump the topic configuration properties and values of \p conf
 *        to an array with \"key\", \"value\" pairs.
 *
 * The number of entries in the array is returned in \p *cntp.
 *
 * The dump must be freed with `rd_kafka_conf_dump_free()`.
 */
RD_EXPORT
const char **rd_kafka_topic_conf_dump(rd_kafka_topic_conf_t *conf,
                      size_t *cntp);

/**
 * @brief Frees a configuration dump returned from `rd_kafka_conf_dump()` or
 *        `rd_kafka_topic_conf_dump().
 */
RD_EXPORT
void rd_kafka_conf_dump_free(const char **arr, size_t cnt);

/**
 * @brief Prints a table to \p fp of all supported configuration properties,
 *        their default values as well as a description.
 */
RD_EXPORT
void rd_kafka_conf_properties_show(FILE *fp);

/**@}*/


/**
 * @name Topic configuration
 * @{
 *
 * @brief Topic configuration property interface
 *
 */


/**
 * @brief Create topic configuration object
 *
 * @sa Same semantics as for rd_kafka_conf_new().
 */
RD_EXPORT
rd_kafka_topic_conf_t *rd_kafka_topic_conf_new(void);
```

```
/**
 * @brief Creates a copy/duplicate of topic configuration object \p conf.
 */
RD_EXPORT
rd_kafka_topic_conf_t *rd_kafka_topic_conf_dup(const rd_kafka_topic_conf_t
                        *conf);


/**
 * @brief Destroys a topic conf object.
 */
RD_EXPORT
void rd_kafka_topic_conf_destroy(rd_kafka_topic_conf_t *topic_conf);


/**
 * @brief Sets a single rd_kafka_topic_conf_t value by property name.
 *
 * \p topic_conf should have been previously set up
 * with `rd_kafka_topic_conf_new()`.
 *
 * @returns rd_kafka_conf_res_t to indicate success or failure.
 */
RD_EXPORT
rd_kafka_conf_res_t rd_kafka_topic_conf_set(rd_kafka_topic_conf_t *conf,
                         const char *name,
                         const char *value,
                         char *errstr, size_t errstr_size);

/**
 * @brief Sets the application's opaque pointer that will be passed to all
topic
 * callbacks as the \c rkt_opaque argument.
 */
RD_EXPORT
void rd_kafka_topic_conf_set_opaque(rd_kafka_topic_conf_t *conf, void
*opaque);


/**
 * @brief \b Producer: Set partitioner callback in provided topic conf
object.
 *
 * The partitioner may be called in any thread at any time,
 * it may be called multiple times for the same message/key.
 *
 * Partitioner function constraints:
 *   - MUST NOT call any rd_kafka_*() functions except:
 *       rd_kafka_topic_partition_available()
 *   - MUST NOT block or execute for prolonged periods of time.
 *   - MUST return a value between 0 and partition_cnt-1, or the
 *     special \c RD_KAFKA_PARTITION_UA value if partitioning
 *     could not be performed.
 */
RD_EXPORT
void
rd_kafka_topic_conf_set_partitioner_cb (rd_kafka_topic_conf_t *topic_conf,
                     int32_t (*partitioner) (
                         const rd_kafka_topic_t *rkt,
                         const void *keydata,
                         size_t keylen,
                         int32_t partition_cnt,
                         void *rkt_opaque,
```

```
                        void *msg_opaque));

/**
 * @brief Check if partition is available (has a leader broker).
 *
 * @returns 1 if the partition is available, else 0.
 *
 * @warning This function must only be called from inside a partitioner
function
 */
RD_EXPORT
int rd_kafka_topic_partition_available(const rd_kafka_topic_t *rkt,
                    int32_t partition);


/*****************************************************************
 *                                  *
 * Partitioners provided by rdkafka                                *
 *                                  *
 *****************************************************************/

/**
 * @brief Random partitioner.
 *
 * Will try not to return unavailable partitions.
 *
 * @returns a random partition between 0 and \p partition_cnt - 1.
 *
 */
RD_EXPORT
int32_t rd_kafka_msg_partitioner_random(const rd_kafka_topic_t *rkt,
                     const void *key, size_t keylen,
                     int32_t partition_cnt,
                     void *opaque, void *msg_opaque);

/**
 * @brief Consistent partitioner.
 *
 * Uses consistent hashing to map identical keys onto identical partitions.
 *
 * @returns a \"random\" partition between 0 and \p partition_cnt - 1 based
on
 *          the CRC value of the key
 */
RD_EXPORT
int32_t rd_kafka_msg_partitioner_consistent (const rd_kafka_topic_t *rkt,
                     const void *key, size_t keylen,
                     int32_t partition_cnt,
                     void *opaque, void *msg_opaque);

/**
 * @brief Consistent-Random partitioner.
 *
 * This is the default partitioner.
 * Uses consistent hashing to map identical keys onto identical partitions,
and
 * messages without keys will be assigned via the random partitioner.
 *
 * @returns a \"random\" partition between 0 and \p partition_cnt - 1 based
on
 *          the CRC value of the key (if provided)
 */
RD_EXPORT
int32_t rd_kafka_msg_partitioner_consistent_random (const rd_kafka_topic_t
```

```
*rkt,
           const void *key, size_t keylen,
           int32_t partition_cnt,
           void *opaque, void *msg_opaque);


/**@}*/



/**
 * @name Main Kafka and Topic object handles
 * @{
 *
 *
 */




/**
 * @brief Creates a new Kafka handle and starts its operation according to
the
 *        specified \p type (\p RD_KAFKA_CONSUMER or \p RD_KAFKA_PRODUCER).
 *
 * \p conf is an optional struct created with `rd_kafka_conf_new()` that
will
 * be used instead of the default configuration.
 * The \p conf object is freed by this function and must not be used or
 * destroyed by the application sub-sequently.
 * See `rd_kafka_conf_set()` et.al for more information.
 *
 * \p errstr must be a pointer to memory of at least size \p errstr_size
where
 * `rd_kafka_new()` may write a human readable error message in case the
 * creation of a new handle fails. In which case the function returns NULL.
 *
 * @remark \b RD_KAFKA_CONSUMER: When a new \p RD_KAFKA_CONSUMER
 *           rd_kafka_t handle is created it may either operate in the
 *           legacy simple consumer mode using the rd_kafka_consume_start()
 *           interface, or the High-level KafkaConsumer API.
 * @remark An application must only use one of these groups of APIs on a
given
 *         rd_kafka_t RD_KAFKA_CONSUMER handle.

 *
 * @returns The Kafka handle on success or NULL on error (see \p errstr)
 *
 * @sa To destroy the Kafka handle, use rd_kafka_destroy().
 */
RD_EXPORT
rd_kafka_t *rd_kafka_new(rd_kafka_type_t type, rd_kafka_conf_t *conf,
              char *errstr, size_t errstr_size);


/**
 * @brief Destroy Kafka handle.
 *
 * @remark This is a blocking operation.
 */
RD_EXPORT
void        rd_kafka_destroy(rd_kafka_t *rk);
```

```
/**
 * @brief Returns Kafka handle name.
 */
RD_EXPORT
const char *rd_kafka_name(const rd_kafka_t *rk);


/**
 * @brief Returns this client's broker-assigned group member id
 *
 * @remark This currently requires the high-level KafkaConsumer
 *
 * @returns An allocated string containing the current broker-assigned group
 *          member id, or NULL if not available.
 *          The application must free the string with \p free() or
 *          rd_kafka_mem_free()
 */
RD_EXPORT
char *rd_kafka_memberid (const rd_kafka_t *rk);


/**
 * @brief Creates a new topic handle for topic named \p topic.
 *
 * \p conf is an optional configuration for the topic created with
 * `rd_kafka_topic_conf_new()` that will be used instead of the default
 * topic configuration.
 * The \p conf object is freed by this function and must not be used or
 * destroyed by the application sub-sequently.
 * See `rd_kafka_topic_conf_set()` et.al for more information.
 *
 * Topic handles are refcounted internally and calling rd_kafka_topic_new()
 * again with the same topic name will return the previous topic handle
 * without updating the original handle's configuration.
 * Applications must eventually call rd_kafka_topic_destroy() for each
 * succesfull call to rd_kafka_topic_new() to clear up resources.
 *
 * @returns the new topic handle or NULL on error (use rd_kafka_errno2err()
 *          to convert system \p errno to an rd_kafka_resp_err_t error code.
 *
 * @sa rd_kafka_topic_destroy()
 */
RD_EXPORT
rd_kafka_topic_t *rd_kafka_topic_new(rd_kafka_t *rk, const char *topic,
                      rd_kafka_topic_conf_t *conf);



/**
 * @brief Destroy topic handle previously created with
`rd_kafka_topic_new()`.
 */
RD_EXPORT
void rd_kafka_topic_destroy(rd_kafka_topic_t *rkt);


/**
 * @brief Returns the topic name.
 */
RD_EXPORT
const char *rd_kafka_topic_name(const rd_kafka_topic_t *rkt);
```

```
/**
 * @brief Get the \p rkt_opaque pointer that was set in the topic
configuration.
 */
RD_EXPORT
void *rd_kafka_topic_opaque (const rd_kafka_topic_t *rkt);


/**
 * @brief Unassigned partition.
 *
 * The unassigned partition is used by the producer API for messages
 * that should be partitioned using the configured or default partitioner.
 */
#define RD_KAFKA_PARTITION_UA  ((int32_t)-1)


/**
 * @brief Polls the provided kafka handle for events.
 *
 * Events will cause application provided callbacks to be called.
 *
 * The \p timeout_ms argument specifies the maximum amount of time
 * (in milliseconds) that the call will block waiting for events.
 * For non-blocking calls, provide 0 as \p timeout_ms.
 * To wait indefinately for an event, provide -1.
 *
 * @remark  An application should make sure to call poll() at regular
 *          intervals to serve any queued callbacks waiting to be called.
 *
 * Events:
 *   - delivery report callbacks  (if dr_cb/dr_msg_cb is configured)
[producer]
 *   - error callbacks (rd_kafka_conf_set_error_cb()) [all]
 *   - stats callbacks (rd_kafka_conf_set_stats_cb()) [all]
 *   - throttle callbacks (rd_kafka_conf_set_throttle_cb()) [all]
 *
 * @returns the number of events served.
 */
RD_EXPORT
int rd_kafka_poll(rd_kafka_t *rk, int timeout_ms);


/**
 * @brief Cancels the current callback dispatcher (rd_kafka_poll(),
 *        rd_kafka_consume_callback(), etc).
 *
 * A callback may use this to force an immediate return to the calling
 * code (caller of e.g. rd_kafka_poll()) without processing any further
 * events.
 *
 * @remark This function MUST ONLY be called from within a librdkafka
callback.
 */
RD_EXPORT
void rd_kafka_yield (rd_kafka_t *rk);




/**
 * @brief Pause producing or consumption for the provided list of
partitions.
 *
```

```
 * Success or error is returned per-partition \p err in the \p partitions
list.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_pause_partitions (rd_kafka_t *rk,
               rd_kafka_topic_partition_list_t *partitions);



/**
 * @brief Resume producing consumption for the provided list of partitions.
 *
 * Success or error is returned per-partition \p err in the \p partitions
list.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_resume_partitions (rd_kafka_t *rk,
                rd_kafka_topic_partition_list_t *partitions);




/**
 * @brief Query broker for low (oldest/beginning) and high (newest/end)
offsets
 *        for partition.
 *
 * Offsets are returned in \p *low and \p *high respectively.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or an error code on
failure.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_query_watermark_offsets (rd_kafka_t *rk,
              const char *topic, int32_t partition,
              int64_t *low, int64_t *high, int timeout_ms);


/**
 * @brief Get last known low (oldest/beginning) and high (newest/end)
offsets
 *        for partition.
 *
 * The low offset is updated periodically (if statistics.interval.ms is set)
 * while the high offset is updated on each fetched message set from the
broker.
 *
 * If there is no cached offset (either low or high, or both) then
 * RD_KAFKA_OFFSET_INVALID will be returned for the respective offset.
 *
 * Offsets are returned in \p *low and \p *high respectively.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or an error code on
failure.
 *
 * @remark Shall only be used with an active consumer instance.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_get_watermark_offsets (rd_kafka_t *rk,
                const char *topic, int32_t partition,
```

```
                int64_t *low, int64_t *high);



/**
 * @brief Free pointer returned by librdkafka
 *
 * This is typically an abstraction for the free(3) call and makes sure
 * the application can use the same memory allocator as librdkafka for
 * freeing pointers returned by librdkafka.
 *
 * In standard setups it is usually not necessary to use this interface
 * rather than the free(3) functione.
 *
 * @remark rd_kafka_mem_free() must only be used for pointers returned by
APIs
 *         that explicitly mention using this function for freeing.
 */
RD_EXPORT
void rd_kafka_mem_free (rd_kafka_t *rk, void *ptr);


/**@}*/




/**
 * @name Queue API
 * @{
 *
 * Message queues allows the application to re-route consumed messages
 * from multiple topic+partitions into one single queue point.
 * This queue point containing messages from a number of topic+partitions
 * may then be served by a single rd_kafka_consume*_queue() call,
 * rather than one call per topic+partition combination.
 */


/**
 * @brief Create a new message queue.
 *
 * See rd_kafka_consume_start_queue(), rd_kafka_consume_queue(), et.al.
 */
RD_EXPORT
rd_kafka_queue_t *rd_kafka_queue_new(rd_kafka_t *rk);

/**
 * Destroy a queue, purging all of its enqueued messages.
 */
RD_EXPORT
void rd_kafka_queue_destroy(rd_kafka_queue_t *rkqu);


/**@}*/

/**
 *
 * @name Simple Consumer API (legacy)
 * @{
 *
 */
```

```
#define RD_KAFKA_OFFSET_BEGINNING -2  /**< Start consuming from beginning of
                      *   kafka partition queue: oldest msg */
#define RD_KAFKA_OFFSET_END       -1  /**< Start consuming from end of kafka
                      *   partition queue: next msg */
#define RD_KAFKA_OFFSET_STORED -1000  /**< Start consuming from offset
retrieved
                      *   from offset store */
#define RD_KAFKA_OFFSET_INVALID -1001 /**< Invalid offset */


/** @cond NO_DOC */
#define RD_KAFKA_OFFSET_TAIL_BASE -2000 /* internal: do not use */
/** @endcond */

/**
 * @brief Start consuming \p CNT messages from topic's current end offset.
 *
 * That is, if current end offset is 12345 and \p CNT is 200, it will start
 * consuming from offset \c 12345-200 = \c 12145. */
#define RD_KAFKA_OFFSET_TAIL(CNT)  (RD_KAFKA_OFFSET_TAIL_BASE - (CNT))

/**
 * @brief Start consuming messages for topic \p rkt and \p partition
 * at offset \p offset which may either be an absolute \c (0..N)
 * or one of the logical offsets:
 *  - RD_KAFKA_OFFSET_BEGINNING
 *  - RD_KAFKA_OFFSET_END
 *  - RD_KAFKA_OFFSET_STORED
 *  - RD_KAFKA_OFFSET_TAIL
 *
 * rdkafka will attempt to keep \c queued.min.messages (config property)
 * messages in the local queue by repeatedly fetching batches of messages
 * from the broker until the threshold is reached.
 *
 * The application shall use one of the `rd_kafka_consume*()` functions
 * to consume messages from the local queue, each kafka message being
 * represented as a `rd_kafka_message_t *` object.
 *
 * `rd_kafka_consume_start()` must not be called multiple times for the same
 * topic and partition without stopping consumption first with
 * `rd_kafka_consume_stop()`.
 *
 * @returns 0 on success or -1 on error in which case errno is set
accordingly:
 *  - EBUSY    - Conflicts with an existing or previous subscription
 *               (RD_KAFKA_RESP_ERR__CONFLICT)
 *  - EINVAL   - Invalid offset, or incomplete configuration (lacking
group.id)
 *               (RD_KAFKA_RESP_ERR__INVALID_ARG)
 *  - ESRCH    - requested \p partition is invalid.
 *               (RD_KAFKA_RESP_ERR__UNKNOWN_PARTITION)
 *  - ENOENT   - topic is unknown in the Kafka cluster.
 *               (RD_KAFKA_RESP_ERR__UNKNOWN_TOPIC)
 *
 * Use `rd_kafka_errno2err()` to convert sytem \c errno to
`rd_kafka_resp_err_t`
 */
RD_EXPORT
int rd_kafka_consume_start(rd_kafka_topic_t *rkt, int32_t partition,
               int64_t offset);

/**
 * @brief Same as rd_kafka_consume_start() but re-routes incoming messages
```

```
to
 * the provided queue \p rkqu (which must have been previously allocated
 * with `rd_kafka_queue_new()`.
 *
 * The application must use one of the `rd_kafka_consume_*_queue()`
functions
 * to receive fetched messages.
 *
 * `rd_kafka_consume_start_queue()` must not be called multiple times for
the
 * same topic and partition without stopping consumption first with
 * `rd_kafka_consume_stop()`.
 * `rd_kafka_consume_start()` and `rd_kafka_consume_start_queue()` must not
 * be combined for the same topic and partition.
 */
RD_EXPORT
int rd_kafka_consume_start_queue(rd_kafka_topic_t *rkt, int32_t partition,
                  int64_t offset, rd_kafka_queue_t *rkqu);

/**
 * @brief Stop consuming messages for topic \p rkt and \p partition, purging
 * all messages currently in the local queue.
 *
 * NOTE: To enforce synchronisation this call will block until the internal
 *       fetcher has terminated and offsets are committed to configured
 *       storage method.
 *
 * The application needs to be stop all consumers before calling
 * `rd_kafka_destroy()` on the main object handle.
 *
 * @returns 0 on success or -1 on error (see `errno`).
 */
RD_EXPORT
int rd_kafka_consume_stop(rd_kafka_topic_t *rkt, int32_t partition);



/**
 * @brief Seek consumer for topic+partition to \p offset which is either an
 *        absolute or logical offset.
 *
 * If \p timeout_ms is not 0 the call will wait this long for the
 * seek to be performed. If the timeout is reached the internal state
 * will be unknown and this function returns `RD_KAFKA_RESP_ERR__TIMED_OUT`.
 * If \p timeout_ms is 0 it will initiate the seek but return
 * immediately without any error reporting (e.g., async).
 *
 * This call triggers a fetch queue barrier flush.
 *
 * @returns `RD_KAFKA_RESP_ERR__NO_ERROR` on success else an error code.
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_seek (rd_kafka_topic_t *rkt,
                                   int32_t partition,
                                   int64_t offset,
                                   int timeout_ms);


/**
 * @brief Consume a single message from topic \p rkt and \p partition
 *
 * \p timeout_ms is maximum amount of time to wait for a message to be
received.
 * Consumer must have been previously started with
```

```
`rd_kafka_consume_start()`.
 *
 * Returns a message object on success or \c NULL on error.
 * The message object must be destroyed with `rd_kafka_message_destroy()`
 * when the application is done with it.
 *
 * Errors (when returning NULL):
 *  - ETIMEDOUT - \p timeout_ms was reached with no new messages fetched.
 *  - ENOENT    - \p rkt + \p partition is unknown.
 *                (no prior `rd_kafka_consume_start()` call)
 *
 * NOTE: The returned message's \c ..->err must be checked for errors.
 * NOTE: \c ..->err \c == \c RD_KAFKA_RESP_ERR__PARTITION_EOF signals that
the
 *       end of the partition has been reached, which should typically not
be
 *       considered an error. The application should handle this case
 *       (e.g., ignore).
 */
RD_EXPORT
rd_kafka_message_t *rd_kafka_consume(rd_kafka_topic_t *rkt, int32_t
partition,
                     int timeout_ms);



/**
 * @brief Consume up to \p rkmessages_size from topic \p rkt and \p
partition
 *        putting a pointer to each message in the application provided
 *        array \p rkmessages (of size \p rkmessages_size entries).
 *
 * `rd_kafka_consume_batch()` provides higher throughput performance
 * than `rd_kafka_consume()`.
 *
 * \p timeout_ms is the maximum amount of time to wait for all of
 * \p rkmessages_size messages to be put into \p rkmessages.
 * If no messages were available within the timeout period this function
 * returns 0 and \p rkmessages remains untouched.
 * This differs somewhat from `rd_kafka_consume()`.
 *
 * The message objects must be destroyed with `rd_kafka_message_destroy()`
 * when the application is done with it.
 *
 * @returns the number of rkmessages added in \p rkmessages,
 * or -1 on error (same error codes as for `rd_kafka_consume()`.
 *
 * @sa rd_kafka_consume()
 */
RD_EXPORT
ssize_t rd_kafka_consume_batch(rd_kafka_topic_t *rkt, int32_t partition,
                int timeout_ms,
                rd_kafka_message_t **rkmessages,
                size_t rkmessages_size);



/**
 * @brief Consumes messages from topic \p rkt and \p partition, calling
 * the provided callback for each consumed messsage.
 *
 * `rd_kafka_consume_callback()` provides higher throughput performance
 * than both `rd_kafka_consume()` and `rd_kafka_consume_batch()`.
 *
```

```
 * \p timeout_ms is the maximum amount of time to wait for one or more
messages
 * to arrive.
 *
 * The provided \p consume_cb function is called for each message,
 * the application \b MUST \b NOT call `rd_kafka_message_destroy()` on the
 * provided \p rkmessage.
 *
 * The \p opaque argument is passed to the 'consume_cb' as \p opaque.
 *
 * @returns the number of messages processed or -1 on error.
 *
 * @sa rd_kafka_consume()
 */
RD_EXPORT
int rd_kafka_consume_callback(rd_kafka_topic_t *rkt, int32_t partition,
                  int timeout_ms,
                  void (*consume_cb) (rd_kafka_message_t
                         *rkmessage,
                         void *opaque),
                  void *opaque);


/**
 * @name Simple Consumer API (legacy): Queue consumers
 * @{
 *
 * The following `..._queue()` functions are analogue to the functions above
 * but reads messages from the provided queue \p rkqu instead.
 * \p rkqu must have been previously created with `rd_kafka_queue_new()`
 * and the topic consumer must have been started with
 * `rd_kafka_consume_start_queue()` utilising the the same queue.
 */

/**
 * @brief Consume from queue
 *
 * @sa rd_kafka_consume()
 */
RD_EXPORT
rd_kafka_message_t *rd_kafka_consume_queue(rd_kafka_queue_t *rkqu,
                       int timeout_ms);

/**
 * @brief Consume batch of messages from queue
 *
 * @sa rd_kafka_consume_batch()
 */
RD_EXPORT
ssize_t rd_kafka_consume_batch_queue(rd_kafka_queue_t *rkqu,
                     int timeout_ms,
                     rd_kafka_message_t **rkmessages,
                     size_t rkmessages_size);

/**
 * @brief Consume multiple messages from queue with callback
 *
 * @sa rd_kafka_consume_callback()
 */
RD_EXPORT
int rd_kafka_consume_callback_queue(rd_kafka_queue_t *rkqu,
                    int timeout_ms,
                    void (*consume_cb) (rd_kafka_message_t
                           *rkmessage,
```

```
                              void *opaque),
                     void *opaque);


/**@}*/




/**
 * @name Simple Consumer API (legacy): Topic+partition offset store.
 * @{
 *
 * If \c auto.commit.enable is true the offset is stored automatically
prior to
 * returning of the message(s) in each of the rd_kafka_consume*() functions
 * above.
 */


/**
 * @brief Store offset \p offset for topic \p rkt partition \p partition.
 *
 * The offset will be committed (written) to the offset store according
 * to \c `auto.commit.interval.ms`.
 *
 * @remark \c `auto.commit.enable` must be set to "false" when using this
API.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success or an error code on error.
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_offset_store(rd_kafka_topic_t *rkt,
                       int32_t partition, int64_t offset);
/**@}*/




/**
 * @name KafkaConsumer (C)
 * @{
 * @brief High-level KafkaConsumer C API
 *
 *
 *
 */

/**
 * @brief Subscribe to topic set using balanced consumer groups.
 *
 * Wildcard (regex) topics are supported by the librdkafka assignor:
 * any topic name in the \p topics list that is prefixed with \c \"^\" will
 * be regex-matched to the full list of topics in the cluster and matching
 * topics will be added to the subscription list.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_subscribe (rd_kafka_t *rk,
                    const rd_kafka_topic_partition_list_t *topics);


/**
 * @brief Unsubscribe from the current subscription set.
 */
```

```
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_unsubscribe (rd_kafka_t *rk);


/**
 * @brief Returns the current topic subscription
 *
 * @returns An error code on failure, otherwise \p topic is updated
 *          to point to a newly allocated topic list (possibly empty).
 *
 * @remark The application is responsible for calling
 *         rd_kafka_topic_partition_list_destroy on the returned list.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_subscription (rd_kafka_t *rk,
                       rd_kafka_topic_partition_list_t **topics);



/**
 * @brief Poll the consumer for messages or events.
 *
 * Will block for at most \p timeout_ms milliseconds.
 *
 * @remark  An application should make sure to call consumer_poll() at
regular
 *          intervals, even if no messages are expected, to serve any
 *          queued callbacks waiting to be called. This is especially
 *          important when a rebalance_cb has been registered as it needs
 *          to be called and handled properly to synchronize internal
 *          consumer state.
 *
 * @returns A message object which is a proper message if \p ->err is
 *          RD_KAFKA_RESP_ERR_NO_ERROR, or an event or error for any other
 *          value.
 *
 * @sa rd_kafka_message_t
 */
RD_EXPORT
rd_kafka_message_t *rd_kafka_consumer_poll (rd_kafka_t *rk, int timeout_ms);

/**
 * @brief Close down the KafkaConsumer.
 *
 * @remark This call will block until the consumer has revoked its
assignment,
 *         calling the \c rebalance_cb if it is configured, committed
offsets
 *         to broker, and left the consumer group.
 *         The maximum blocking time is roughly limited to
session.timeout.ms.
 *
 * @returns An error code indicating if the consumer close was succesful
 *          or not.
 *
 * @remark The application still needs to call rd_kafka_destroy() after
 *         this call finishes to clean up the underlying handle resources.
 *
 *
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_consumer_close (rd_kafka_t *rk);
```

```
/**
 * @brief Atomic assignment of partitions to consume.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_assign (rd_kafka_t *rk,
                 const rd_kafka_topic_partition_list_t *partitions);

/**
 * @brief Returns the current partition assignment
 *
 * @returns An error code on failure, otherwise \p partitions is updated
 *          to point to a newly allocated partition list (possibly empty).
 *
 * @remark The application is responsible for calling
 *         rd_kafka_topic_partition_list_destroy on the returned list.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_assignment (rd_kafka_t *rk,
                     rd_kafka_topic_partition_list_t **partitions);




/**
 * @brief Commit offsets on broker for the provided list of partitions.
 *
 * \p offsets should contain \c topic, \c partition, \c offset and possibly
 * \c metadata.
 * If \p offsets is NULL the current partition assignment will be used
instead.
 *
 * If \p async is false this operation will block until the broker offset
commit
 * is done, returning the resulting success or error code.
 *
 * If a rd_kafka_conf_set_offset_commit_cb() offset commit callback has been
 * configured a callback will be enqueued for a future call to
rd_kafka_poll().
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_commit (rd_kafka_t *rk, const rd_kafka_topic_partition_list_t
*offsets,
                 int async);


/**
 * @brief Commit message's offset on broker for the message's partition.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_commit_message (rd_kafka_t *rk, const rd_kafka_message_t
*rkmessage,
                         int async);




/**
 * @brief Retrieve committed offsets for topics+partitions.
 *
 * The \p offset field of each requested partition will either be set to
 * stored offset or to RD_KAFKA_OFFSET_INVALID in case there was no stored
 * offset for that partition.
 *
```

```
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success in which case the
 *          \p offset or \p err field of each \p partitions' element is
filled
 *          in with the stored offset, or a partition specific error.
 *          Else returns an error code.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_committed (rd_kafka_t *rk,
            rd_kafka_topic_partition_list_t *partitions,
            int timeout_ms);



/**
 * @brief Retrieve current positions (offsets) for topics+partitions.
 *
 * The \p offset field of each requested partition will be set to the offset
 * of the last consumed message + 1, or RD_KAFKA_OFFSET_INVALID in case
there was
 * no previous message.
 *
 * @returns RD_KAFKA_RESP_ERR_NO_ERROR on success in which case the
 *          \p offset or \p err field of each \p partitions' element is
filled
 *          in with the stored offset, or a partition specific error.
 *          Else returns an error code.
 */
RD_EXPORT rd_kafka_resp_err_t
rd_kafka_position (rd_kafka_t *rk,
           rd_kafka_topic_partition_list_t *partitions);


/**@}*/



/**
 * @name Producer API
 * @{
 *
 *
 */


/**
 * @brief Producer message flags
 */
#define RD_KAFKA_MSG_F_FREE  0x1 /**< Delegate freeing of payload to
rdkafka. */
#define RD_KAFKA_MSG_F_COPY  0x2 /**< rdkafka will make a copy of the
payload. */



/**
 * @brief Produce and send a single message to broker.
 *
 * \p rkt is the target topic which must have been previously created with
 * `rd_kafka_topic_new()`.
 *
 * `rd_kafka_produce()` is an asynch non-blocking API.
 *
 * \p partition is the target partition, either:
 *   - RD_KAFKA_PARTITION_UA (unassigned) for
```

```
 *     automatic partitioning using the topic's partitioner function, or
 *   - a fixed partition (0..N)
 *
 * \p msgflags is zero or more of the following flags OR:ed together:
 *    RD_KAFKA_MSG_F_FREE - rdkafka will free(3) \p payload when it is done
 *                          with it.
 *    RD_KAFKA_MSG_F_COPY - the \p payload data will be copied and the
 *                          \p payload pointer will not be used by rdkafka
 *                          after the call returns.
 *
 *    ..._F_FREE and ..._F_COPY are mutually exclusive.
 *
 *    If the function returns -1 and RD_KAFKA_MSG_F_FREE was specified, then
 *    the memory associated with the payload is still the caller's
 *    responsibility.
 *
 * \p payload is the message payload of size \p len bytes.
 *
 * \p key is an optional message key of size \p keylen bytes, if non-NULL it
 * will be passed to the topic partitioner as well as be sent with the
 * message to the broker and passed on to the consumer.
 *
 * \p msg_opaque is an optional application-provided per-message opaque
 * pointer that will provided in the delivery report callback (`dr_cb`) for
 * referencing this message.
 *
 * Returns 0 on success or -1 on error in which case errno is set
accordingly:
 *  - ENOBUFS  - maximum number of outstanding messages has been reached:
 *               "queue.buffering.max.messages"
 *               (RD_KAFKA_RESP_ERR__QUEUE_FULL)
 *  - EMSGSIZE - message is larger than configured max size:
 *               "messages.max.bytes".
 *               (RD_KAFKA_RESP_ERR_MSG_SIZE_TOO_LARGE)
 *  - ESRCH    - requested \p partition is unknown in the Kafka cluster.
 *               (RD_KAFKA_RESP_ERR__UNKNOWN_PARTITION)
 *  - ENOENT   - topic is unknown in the Kafka cluster.
 *               (RD_KAFKA_RESP_ERR__UNKNOWN_TOPIC)
 *
 * @sa Use rd_kafka_errno2err() to convert `errno` to rdkafka error code.
 */
RD_EXPORT
int rd_kafka_produce(rd_kafka_topic_t *rkt, int32_t partition,
             int msgflags,
             void *payload, size_t len,
             const void *key, size_t keylen,
             void *msg_opaque);



/**
 * @brief Produce multiple messages.
 *
 * If partition is RD_KAFKA_PARTITION_UA the configured partitioner will
 * be run for each message (slower), otherwise the messages will be enqueued
 * to the specified partition directly (faster).
 *
 * The messages are provided in the array \p rkmessages of count \p
message_cnt
 * elements.
 * The \p partition and \p msgflags are used for all provided messages.
 *
 * Honoured \p rkmessages[] fields are:
 *  - payload,len    Message payload and length
```

```
 *  - key,key_len    Optional message key
 *  - _private       Message opaque pointer (msg_opaque)
 *  - err            Will be set according to success or failure.
 *                   Application only needs to check for errors if
 *                   return value != \p message_cnt.
 *
 * @returns the number of messages succesfully enqueued for producing.
 */
RD_EXPORT
int rd_kafka_produce_batch(rd_kafka_topic_t *rkt, int32_t partition,
                           int msgflags,
                           rd_kafka_message_t *rkmessages, int
message_cnt);




/**@}*/


/**
* @name Metadata API
* @{
*
*
*/


/**
 * @brief Broker information
 */
typedef struct rd_kafka_metadata_broker {
        int32_t     id;             /**< Broker Id */
        char       *host;           /**< Broker hostname */
        int         port;           /**< Broker listening port */
} rd_kafka_metadata_broker_t;

/**
 * @brief Partition information
 */
typedef struct rd_kafka_metadata_partition {
        int32_t     id;             /**< Partition Id */
        rd_kafka_resp_err_t err;    /**< Partition error reported by broker
*/
        int32_t     leader;         /**< Leader broker */
        int         replica_cnt;    /**< Number of brokers in \p replicas */
        int32_t    *replicas;       /**< Replica brokers */
        int         isr_cnt;        /**< Number of ISR brokers in \p isrs */
        int32_t    *isrs;           /**< In-Sync-Replica brokers */
} rd_kafka_metadata_partition_t;

/**
 * @brief Topic information
 */
typedef struct rd_kafka_metadata_topic {
        char       *topic;          /**< Topic name */
        int         partition_cnt;  /**< Number of partitions in \p
partitions*/
        struct rd_kafka_metadata_partition *partitions; /**< Partitions */
        rd_kafka_resp_err_t err;    /**< Topic error reported by broker */
} rd_kafka_metadata_topic_t;


/**
```

```
 * @brief Metadata container
 */
typedef struct rd_kafka_metadata {
        int         broker_cnt;     /**< Number of brokers in \p brokers */
        struct rd_kafka_metadata_broker *brokers;  /**< Brokers */

        int         topic_cnt;      /**< Number of topics in \p topics */
        struct rd_kafka_metadata_topic *topics;    /**< Topics */

        int32_t     orig_broker_id;   /**< Broker originating this metadata
*/
        char       *orig_broker_name; /**< Name of originating broker */
} rd_kafka_metadata_t;


/**
 * @brief Request Metadata from broker.
 *
 * Parameters:
 *  - \p all_topics  if non-zero: request info about all topics in cluster,
 *                   if zero: only request info about locally known topics.
 *  - \p only_rkt    only request info about this topic
 *  - \p metadatap   pointer to hold metadata result.
 *                   The \p *metadatap pointer must be released
 *                   with rd_kafka_metadata_destroy().
 *  - \p timeout_ms  maximum response time before failing.
 *
 * Returns RD_KAFKA_RESP_ERR_NO_ERROR on success (in which case *metadatap)
 * will be set, else RD_KAFKA_RESP_ERR__TIMED_OUT on timeout or
 * other error code on error.
 */
RD_EXPORT
rd_kafka_resp_err_t
rd_kafka_metadata (rd_kafka_t *rk, int all_topics,
                   rd_kafka_topic_t *only_rkt,
                   const struct rd_kafka_metadata **metadatap,
                   int timeout_ms);

/**
 * @brief Release metadata memory.
 */
RD_EXPORT
void rd_kafka_metadata_destroy(const struct rd_kafka_metadata *metadata);


/**@}*/



/**
* @name Client group information
* @{
*
*
*/


/**
 * @brief Group member information
 *
 * For more information on \p member_metadata format, see
 * https://cwiki.apache.org/confluence/display/KAFKA/
A+Guide+To+The+Kafka+Protocol#AGuideToTheKafkaProtocol-GroupMembershipAPI
 *
```

```
 */
struct rd_kafka_group_member_info {
        char *member_id;            /**< Member id (generated by broker) */
        char *client_id;            /**< Client's \p client.id */
        char *client_host;          /**< Client's hostname */
        void *member_metadata;      /**< Member metadata (binary),
                                     *   format depends on \p
protocol_type. */
        int   member_metadata_size; /**< Member metadata size in bytes */
        void *member_assignment;    /**< Member assignment (binary),
                                     *    format depends on \p
protocol_type. */
        int   member_assignment_size; /**< Member assignment size in bytes
*/
};

/**
 * @brief Group information
 */
struct rd_kafka_group_info {
        struct rd_kafka_metadata_broker broker; /**< Originating broker
info */
        char *group;                            /**< Group name */
        rd_kafka_resp_err_t err;                /**< Broker-originated
error */
        char *state;                            /**< Group state */
        char *protocol_type;                    /**< Group protocol type */
        char *protocol;                         /**< Group protocol */
        struct rd_kafka_group_member_info *members; /**< Group members */
        int   member_cnt;                       /**< Group member count */
};

/**
 * @brief List of groups
 *
 * @sa rd_kafka_group_list_destroy() to release list memory.
 */
struct rd_kafka_group_list {
        struct rd_kafka_group_info *groups;   /**< Groups */
        int    group_cnt;                     /**< Group count */
        bool   is_streams_list;               /* List contains consumer gr
                                               * on mapr streams
                                               */
};


/**
 * @brief List and describe client groups in cluster.
 *
 * \p group is an optional group name to describe, otherwise (\p NULL) all
 * groups are returned.
 *
 * \p timeout_ms is the (approximate) maximum time to wait for response
 * from brokers and must be a positive value.
 *
 * @returns \p RD_KAFKA_RESP_ERR__NO_ERROR on success and \p grplistp is
 *           updated to point to a newly allocated list of groups.
 *           Else returns an error code on failure and \p grplistp remains
 *           untouched.
 *
 * @sa Use rd_kafka_group_list_destroy() to release list memory.
 */
RD_EXPORT
rd_kafka_resp_err_t
```

```
rd_kafka_list_groups (rd_kafka_t *rk, const char *group,
                      const struct rd_kafka_group_list **grplistp,
                      int timeout_ms);

/**
 * @brief Release list memory
 */
RD_EXPORT
void rd_kafka_group_list_destroy (const struct rd_kafka_group_list
*grplist);


/**@}*/



/**
 * @name Miscellaneous APIs
 * @{
 *
 */


/**
 * @brief Adds one or more brokers to the kafka handle's list of initial
 *        bootstrap brokers.
 *
 * Additional brokers will be discovered automatically as soon as rdkafka
 * connects to a broker by querying the broker metadata.
 *
 * If a broker name resolves to multiple addresses (and possibly
 * address families) all will be used for connection attempts in
 * round-robin fashion.
 *
 * \p brokerlist is a ,-separated list of brokers in the format:
 *   \c \<broker1\>,\<broker2\>,..
 * Where each broker is in either the host or URL based format:
 *   \c \<host\>[:\<port\>]
 *   \c \<proto\>://\<host\>[:port]
 * \c \<proto\> is either \c PLAINTEXT, \c SSL, \c SASL, \c SASL_PLAINTEXT
 * The two formats can be mixed but ultimately the value of the
 * `security.protocol` config property decides what brokers are allowed.
 *
 * Example:
 *    brokerlist = "broker1:10000,broker2"
 *    brokerlist = "SSL://broker3:9000,ssl://broker2"
 *
 * @returns the number of brokers successfully added.
 *
 * @remark Brokers may also be defined with the \c metadata.broker.list or
 *         \c bootstrap.servers configuration property (preferred method).
 */
RD_EXPORT
int rd_kafka_brokers_add(rd_kafka_t *rk, const char *brokerlist);




/**
 * @brief Set logger function.
 *
 * The default is to print to stderr, but a syslog logger is also available,
 * see rd_kafka_log_(print|syslog) for the builtin alternatives.
 * Alternatively the application may provide its own logger callback.
```

```
 * Or pass 'func' as NULL to disable logging.
 *
 * @deprecated Use rd_kafka_conf_set_log_cb()
 *
 * @remark \p rk may be passed as NULL in the callback.
 */
RD_EXPORT RD_DEPRECATED
void rd_kafka_set_logger(rd_kafka_t *rk,
              void (*func) (const rd_kafka_t *rk, int level,
                    const char *fac, const char *buf));


/**
 * @brief Specifies the maximum logging level produced by
 *        internal kafka logging and debugging.
 *
 * If the \p \"debug\" configuration property is set the level is
automatically
 * adjusted to \c LOG_DEBUG (7).
 */
RD_EXPORT
void rd_kafka_set_log_level(rd_kafka_t *rk, int level);


/**
 * @brief Builtin (default) log sink: print to stderr
 */
RD_EXPORT
void rd_kafka_log_print(const rd_kafka_t *rk, int level,
             const char *fac, const char *buf);


/**
 * @brief Builtin log sink: print to syslog.
 */
RD_EXPORT
void rd_kafka_log_syslog(const rd_kafka_t *rk, int level,
              const char *fac, const char *buf);


/**
 * @brief Returns the current out queue length.
 *
 * The out queue contains messages waiting to be sent to, or acknowledged
by,
 * the broker.
 *
 * An application should wait for this queue to reach zero before
terminating
 * to make sure outstanding requests (such as offset commits) are fully
 * processed.
 *
 * @returns number of messages in the out queue.
 */
RD_EXPORT
int         rd_kafka_outq_len(rd_kafka_t *rk);



/**
 * @brief Dumps rdkafka's internal state for handle \p rk to stream \p fp
 *
 * This is only useful for debugging rdkafka, showing state and statistics
 * for brokers, topics, partitions, etc.
```

```
 */
RD_EXPORT
void rd_kafka_dump(FILE *fp, rd_kafka_t *rk);



/**
 * @brief Retrieve the current number of threads in use by librdkafka.
 *
 * Used by regression tests.
 */
RD_EXPORT
int rd_kafka_thread_cnt(void);


/**
 * @brief Wait for all rd_kafka_t objects to be destroyed.
 *
 * Returns 0 if all kafka objects are now destroyed, or -1 if the
 * timeout was reached.
 * Since `rd_kafka_destroy()` is an asynch operation the
 * `rd_kafka_wait_destroyed()` function can be used for applications where
 * a clean shutdown is required.
 */
RD_EXPORT
int rd_kafka_wait_destroyed(int timeout_ms);


/**@}*/




/**
 * @name Experimental APIs
 * @{
 */

/**
 * @brief Redirect the main (rd_kafka_poll()) queue to the KafkaConsumer's
 *        queue (rd_kafka_consumer_poll()).
 *
 * @warning It is not permitted to call rd_kafka_poll() after directing the
 *          main queue with rd_kafka_poll_set_consumer().
 */
RD_EXPORT
rd_kafka_resp_err_t rd_kafka_poll_set_consumer (rd_kafka_t *rk);

/**@}*/

#ifdef __cplusplus
}
#endif
```

MapR Event Store For Apache Kafka Python Applications

As of MapR Data Platform 5.2.1, you can create python applications for MapR Event Store For Apache Kafka using the MapR Event Store For Apache Kafka Python client. The MapR Event Store For Apache Kafka Python client is a binding for librdkafka and the MapR Event Store For Apache Kafka C Client is a distribution of librdkafka that works with MapR Event Store For Apache Kafka.

The MapR Event Store For Apache Kafka Python client is available in a MapR Ecosystem Pack (EEP) starting with EEP 3.0.

The following Apache Kafka librdkafka versions are supported:

Table

Core release	EEP Release	Kafka librdkafka version
As of MapR Data Platform 6.0.1	As of 5.0	0.11.3
As of MapR Data Platform 5.2.1 through 6.0.0	As of 3.0	0.9.0

Note: Because the MapR Event Store For Apache Kafka Python Client is dependent on the MapR Event Store For Apache Kafka C Client, the MapR Event Store For Apache Kafka C Client must be configured before using the MapR Event Store For Apache Kafka Python Client.

When developing and running MapR Event Store For Apache Kafka Python applications, note the following points:

- You can create producers and high-level consumers. Low-level consumers are not supported.
- Consuming or producing topics in a Kafka cluster is not supported.
- MapR Event Store For Apache Kafka offset values start at 1, not 0.
- MapR Data Platform security is supported including ACLs and ACEs for authorization. The unique Kafka security features that are part of Apache Kafka are not supported. See Security on page 683 for more information about MapR Data Platform security features.
- User impersonation is not supported.

Developing MapR Event Store For Apache Kafka Python Applications

This topic includes basic information about how to develop a MapR Event Store For Apache Kafka Python application and an example program that you can run.

Before you Begin

Confirm that your environment meets the following requirements:

- MapR Data Platform cluster version 5.2.1 or greater.
- MapR Data Platform core client (mapr-client) package. See Installing the MapR Client on page 389 for more information.
- MapR Event Store For Apache Kafka C Client (mapr-librdkafka) is installed and configured on the node. See Configuring the MapR Event Store For Apache Kafka C Client on page 2796.
- MapR Event Store For Apache Kafka Python Client (mapr-streams-python) is installed on the node. See Installing MapR Event Store For Apache Kafka Python Client on page 198.
- Python installed on the node (Python version 2.7.x and above, up to version 3.6.x).

Create a MapR Event Store For Apache Kafka Producer Application

In general, you want to create a producer that performs the following steps:

1. Import the producer class.
2. Define the producer and its configuration.
3. Produce data.
4. Wait for all messages to be sent to consumer.

last-updated: Apr 11, 2022

As of EEP 5.0 MapR Event Store For Apache Kafka Python Client: In the following example code, three messages are produced to a topic named `mytopic` in a stream named `my_stream`.

```
from confluent_kafka import Producer
p = Producer({'streams.producer.default.stream': '/my_stream'})
some_data_source= ["msg1", "msg2", "msg3"]
for data in some_data_source:
     p.produce('mytopic', data.encode('utf-8'))
     p.flush()
```

As of EEP 3.0 MapR Event Store For Apache Kafka Python Client: In the following example code, three messages are produced to a topic named `mytopic` in a stream named `my_stream`.

```
from mapr_streams_python import Producer
p = Producer({'streams.producer.default.stream': '/my_stream'})
some_data_source= ["msg1", "msg2", "msg3"]
for data in some_data_source:
    p.produce('mytopic', data.encode('utf-8'))
    p.flush()
```

Create a MapR Event Store For Apache Kafka Consumer Application

In general, you want to create a consumer that performs the following steps:

1. Import the consumer class.
2. Define the consumer and its configuration.
3. Consume data.
4. Wait for all messages to be consumed.

As of EEP 5.0 MapR Event Store For Apache Kafka Python Client: In following example code, the MapR Event Store For Apache Kafka consumer is subscribed to `my_stream/mytopic` and it prints the content of each message that it reads.

```
from confluent_kafka import Consumer, KafkaError
c = Consumer({'group.id': 'mygroup',
              'default.topic.config': {'auto.offset.reset': 'earliest'}})
c.subscribe(['/my_stream:mytopic'])
running = True
while running:
  msg = c.poll(timeout=1.0)
  if msg is None: continue
  if not msg.error():
    print('Received message: %s' % msg.value().decode('utf-8'))
  elif msg.error().code() != KafkaError._PARTITION_EOF:
    print(msg.error())
    running = False
c.close()
```

As of EEP 3.0 MapR Event Store For Apache Kafka Python Client: In following example code, the MapR Event Store For Apache Kafka consumer is subscribed to `my_stream/mytopic` and it prints the content of each message that it reads.

```
from mapr_streams_python import Consumer, KafkaError
c = Consumer({'group.id': 'mygroup',
              'default.topic.config': {'auto.offset.reset': 'earliest'}})
c.subscribe(['/my_stream:mytopic'])
running = True
```

```
while running:
  msg = c.poll(timeout=1.0)
  if msg is None: continue
  if not msg.error():
    print('Received message: %s' % msg.value().decode('utf-8'))
  elif msg.error().code() != KafkaError._PARTITION_EOF:
    print(msg.error())
    running = False
c.close()
```

Run the Example Applications

To run the sample producer and consumer applications:

1. Create a stream named `mystream`.
2. Create a file named producer.py.
3. Add the producer example code into the producer.py file.
4. Create a file named consumer.py.
5. Add the consumer example code into the consumer.py file.
6. Verify that you have completed the steps to configure the MapR Event Store For Apache Kafka C client or complete the steps now. See Configuring the MapR Event Store For Apache Kafka C Client on page 2796.

   Note: The MapR Event Store For Apache Kafka Python Client is dependent on the MapR Event Store For Apache Kafka C Client. Therefore, the MapR Event Store For Apache Kafka C Client must be configured before you can run the application.

7. Run producer.py from the command line to generate messages.

   ```
   $ python producer.py
   ```

8. Run consumer.py from the command line:

   ```
   $ python consumer.py
   ```

Migrating Kafka Python Applications to MapR Event Store For Apache Kafka

With some modification, you can use existing confluent-kafka python applications to consume and produce topics in MapR Event Store For Apache Kafka. The MapR Event Store For Apache Kafka Python Client is a binding for Apache librdkafka that works with MapR Event Store For Apache Kafka.

1. Install the MapR Event Store For Apache Kafka Python Client.

   Note: This required that you also install and configure the MapR Event Store For Apache Kafka C Client. See Configuring the MapR Event Store For Apache Kafka C Client on page 2796.

2. Do one of the following depending on whether you are using the EEP 5.0 (or higher) MapR Event Store For Apache Kafka Python Client or the EEP 3.0 (or higher) MapR Event Store For Apache Kafka Python Client.
   - If you are using MapR Event Store For Apache Kafka Python EEP 5.0 (or higher), skip this step. The references to `confluent_kafka` should be retained.

last-updated: Apr 11, 2022

- If you are using MapR Event Store For Apache Kafka Python EEP 3.0 (or higher), update import statements to refer to the MapR Stream Python API. References to `confluent_kafka` should be updated to `mapr_streams_python`.

  Note: For example, update `from confluent_kafka import Consumer` to `from mapr_streams_python import Consumer`.

3. When you refer to a topic in the application code, include the path and name of the stream in which the topic is located:

   ```
   /<path and name of stream>:<name of topic>
   ```

   For example, you might have a stream in a MapR Data Platform cluster that is named stream_A, and the stream might be in a volume named IoT and in a directory named automobile_sensors. You want to redirect a producer application to a topic in that stream. The syntax of the path to the topic might look like this: `/mapr/IoT/automobile_sensors/stream_A:<name of topic>`.

   Note: Optionally, use the `streams.consumer.default.stream` and `streams.producer.default.stream` configuration parameters. When you configure these parameters, applications can specify just the topic name to write or read from the default stream.

4. Review the APIs that are supported and make changes to your application, as needed. See API for MapR Event Store For Apache Kafka Python Client on page 3002.

5. See Configuration Properties for MapR Event Store For Apache Kafka Python Client on page 3008 for the list of supported configuration parameters and make changes to your application, as needed.

   Note: SSL-related configuration parameters are ignored. When you set these parameters, the MapR Event Store For Apache Kafka Client issues a warning indicating that the parameters are not supported.

API for MapR Event Store For Apache Kafka Python Client

MapR Event Store For Apache Kafka Python Client is a binding for librdkafka and it supports the following APIs.

As of MapR Data Platform 5.2.1, you can create python applications for MapR Event Store For Apache Kafka using the MapR Event Store For Apache Kafka Python client. The MapR Event Store For Apache Kafka Python client is a binding for librdkafka and the MapR Event Store For Apache Kafka C Client is a distribution of librdkafka that works with MapR Event Store For Apache Kafka.

Table

Core release	EEP Release	Kafka librdkafka version
As of MapR Data Platform 6.0.1	As of 5.0	0.11.3
As of MapR Data Platform 5.2.1 through 6.0.0	As of 3.0	0.9.0

class mapr_streams_python.Consumer

A high-level Kafka Consumer.

Method	Behavior
Consumer(**kwargs)	Create new Consumer instance using provided configuration dictionary.

Method	Behavior
assign(partitions)	Set consumer partition assignment to the provided list of TopicPartition and starts consuming. Parameters(s): • partitions (list(TopicPartition)) – List of topic+partitions and optionally initial offsets to start consuming
unassign()	Unassign from all TopicPartitions that have been assigned with the `.assign(*topic_partition_list) method.`  **Note:** This method is applicable as of MapR Event Store For Apache Kafka Python Client EEP 5.0 which is associated with librdkafka 0.11.3.
assignment()	Return a list of assignments for a consumer object.  **Note:** This method is applicable as of MapR Event Store For Apache Kafka Python Client EEP 5.0 which is associated with librdkafka 0.11.3.
close()	Close down and terminate the Kafka Consumer. Actions(s): • Stops consuming • Commits offsets • Leave consumer group
commit([message=None][, offsets=None][, async=True])	Commit a message or a list of offsets. Message and offsets are mutually exclusive, if neither is set the current partition assignment's offsets are used instead. Parameters(s): • message (confluent_kafka.Message) – Commit message's offset+1. • offsets (list(TopicPartition)) – List of topic+partitions+offsets to commit. • async (bool) – Asynchronous commit, return immediately.

last-updated: Apr 11, 2022

Method	Behavior
committed(partitions[, timeout=None])	Retrieve committed offsets for the list of partitions. Parameters(s): • partitions (list(TopicPartition)) - List of topic+partitions to query for stored offsets. • timeout (float) – Request timeout Returns: List of topic+partitions with offset and possibly error set. Return type: list(TopicPartition) Raises: KafkaException **Note:** As of MapR Data Platform 6.0, the message offset in a partition starts from zero (0). If you are upgrading and do not enable the MapR Database/MapR Event Store For Apache Kafka feature, **mfs.feature.db.streams.v6.support**, the message offset in a partition starts from one (1).
on_commit(err, partitions)	A callback for Consumer.commit() that triggers custom actions when a commit request completes. Parameters(s): • err (KafkaError) – Commit error object, or None on success. • Partitions (list(TopicPartition)) – List of partitions with their committed offsets or per-partition errors
poll([timeout=None])	Consume messages, calls callbacks and returns events. The application must check the returned Message object's Message.error() method to distinguish between proper messages (error() returns None), or an event or error (see error().code() for specifics). Parameter(s): timeout (*float*) – Maximum time to block waiting for message, event or callback Returns: A Message object or None on timeout Return type: Message or None
position(partitions[, timeout=None])	Retrieve current positions (offsets) for the list of partitions. Parameter(s): partitions (list(TopicPartition)) – List of topic+partitions to return current offsets for. The current offset is the offset of the last consumed message + 1 Returns: List of topic+partitions with offset and possibly error set. Return type: list(TopicPartition) Raises: KafkaException This function returns 0 when the messages have not yet been consumed from partitions. librdkafka returns -1001 instead.

Method	Behavior
subscribe(topics[, listener=None])	Set subscription to supplied list of topics This replaces a previous subscription. Parameters: • topics (list(str)) – List of topics (strings) to subscribe to. • on_assign (callable) – callback to provide handling of customized offsets on completion of a successful partition re-assignment. • on_revoke (callable) – callback to provide handling of offset commits to a customized store on the start of a rebalance operation. Raises: KafkaException **Note:** You cannot use the rd_kafka_subscribe API to subscribe a consumer to topics when that consumer is already assigned to topics. If you call this API for an assigned consumer, error RD_KAFKA_RESP_ERR__CONFLICT is returned.
on_assign(consumer, partitions)	Same as librdkafka.
unsubscribe()	Same as librdkafka.
on_revoke(consumer, partitions)	Parameter(s): • consumer (Consumer) – Consumer instance. • partitions (list(TopicPartition)) – Absolute list of partitions being assigned or revoked.
get_watermark_offsets(confluent_kafka.TopicPartition)	Get WatermarkOffsets for a given Topic Partition. Parameter(s): TopicPartition - Gets the watermark offset **Note:** This method is applicable as of MapR Event Store For Apache Kafka Python Client EEP 5.0 which is associated with librdkafka 0.11.3.

class mapr_streams_python.Producer

Asynchronous Kafka Producer.

Method	Behavior
Producer(**kwargs)	Create new Producer instance using provided configuration dict.
len()	This API returns a positive number to indicate that messages are waiting to be produced to a streams topic but the value does not indicate the actual number of messages. librdkafka returns the actual number of messages that are waiting to be sent to or acknowledged by the broker. Return type: int

Method	Behavior
flush()	Wait for all messages in the Producer queue to be delivered. This is a convenience method that calls poll() until len()is zero.
poll([timeout])	Polls the producer for events and calls the corresponding callbacks (if registered). Parameter(s): • timeout (float) – Maximum time to block waiting for events Returns: Number of events processed (callbacks served). Return type: int
produce(topic[, value][, key][, partition][, callback])	Produce message to topic. This is an asynchronous operation, an application may use the callback(alias on_delivery) argument to pass a function (or lambda) that will be called from poll() when the message has been successfully delivered or permanently fails delivery. Parameters: • topic (str) – Topic to produce message to • value (str|bytes) – Message payload • key (str|bytes) – Message key • partition (int) – Partition to produce to, else uses the configured partitioner. • on_delivery(err,msg) (func) – Delivery report callback to call (from poll() or flush()) on successful or failed deliver Raises: • BufferError – if the internal producer message queue is full (queue.buffering.max.messages exceeded) • KafkaException – for other errors, see exception code **Note:** When this function is called with NULL payload, an invalid argument error is sent to the callback. librdkafka creates a message with NULL payload and key value instead.

class mapr_streams_python.Message

The Message object represents either a single consumed or produced message, or an event . An application must check with error() to see if the object is a proper message (error() returns None) or an error/event. This class is not user-instantiable.

Method	Behavior
len()	Returns: Message value (payload) size in bytes. Return type: int

Method	Behavior
error()	The message object is also used to propagate errors and events. Applications must check error() to determine if the Message is a proper message (error() returns None) or an error or event (error() returns a KafkaError object) Return type: None or KafkaError
key()	Returns: message key or None if not available Return type: str|bytes or None
offset()	Returns: message offset or None if not available Return type: int or None
partition()	Returns: partition number or None if not available Return type: int or None
topic()	Returns: topic name or None if not available Return type: str or None
value()	Returns: message value (payload) or None if not available Return type: str|bytes or None
timestamp()	Returns: message timestamp  **Note:** This method is applicable as of MapR Event Store For Apache Kafka Python Client EEP 5.0 which is associated with librdkafka 0.11.3.

class mapr_streams_python.TopicPartition

TopicPartition is a generic type to hold a single partition and various information about it. It is typically used to provide a list of topics or partitions for various operations, such as Consumer.assign().

Method	Behavior
TopicPartition(topic[, partition][, offset])	Instantiate a TopicPartition object. Parameter(s) • topic (string) – Topic name • partition (int) – Partition id • offset (int) – Initial partition offset Return type: TopicPartition
error	Attribute that indicates an error (with KafkaError) unless None.
offset	Attribute for offset.
partition	Attribute for partition number.
topic	Attribute for topic name.

last-updated: Apr 11, 2022

class mapr_streams_python.KafkaError

Kafka error and event object.

The KafkaError class serves multiple purposes:

- Propagation of errors
- Propagation of events
- Exceptions

This class is not user-instantiable.

Method	Behavior
code()	Returns the error/event code for comparison toKafkaError.<ERR_CONSTANTS>. Returns: error/event code Return type: int
name()	Returns the enum name for error/event. Returns: error/event enum name string Return type: str
str()	Returns the human-readable error/event string. Returns: error/event enum message string Return type: str

Configuration Properties for MapR Event Store For Apache Kafka Python Client
In the instance constructor of a MapR Event Store For Apache Kafka Python application, you can use a dictionary to set the following configuration properties. MapR Event Store For Apache Kafka Python client supports a superset of the configuration properties supported by the MapR Event Store For Apache Kafka C client.

Global Configuration Properties

Property Name	Behavior
client.id	Same as librdkafka
default.topic.config	A dictionary of topic-level configuration properties that are applied to all used topics for the instance.
message.max.bytes	Supports a value less than or equal to 10MB (10000000). If this property is set to a value that is higher than 10MB, the client issues a warning and sets the configuration to 10MB. Produce calls fail when the message size is greater than 10MB.
receive.message.max.bytes	Same as librdkafka
topic.blacklist	Same as librdkafka
error_cb	A callback for generic/global error events. This callback is served by poll().
opaque	Same as librdkafka.

Consumer Configuration Properties

Property Name	Behavior
group.id	Same as librdkafka.
enable.auto.commit	Same as librdkafka.
auto.commit.interval.ms	Same as librdkafka.
rebalance_cb	Same as librdkafka.
offset_commit_cb	Same as librdkafka.
delivery.report.only.error	Same as librdkafka.
dr_msg_cb	Same as librdkafka.
on_commit	A callback used to indicate success or failure of commit requests.

Topic Configuration Properties

Property Name	Behavior
partitioner_cb	Same as librdkafka.
auto.offset.reset	Supports the following values: earliest, latest, none, smallest, and largest. librdkafka also supports biggest, end and error.

Producer Configuration Properties

Property Name	Behavior
on_delivery(kafka.KafkaError, kafka.Message)	A Python function reference that is called once for each produced message to indicate the final delivery result (success or failure). This property may also be set per-message by passing callback=callable (or on_delivery=callable) to the confluent_kafka.Producer.produce() function.

MapR Data Platform-Specific Configuration Properties

Property Name	Behavior
streams.consumer.default.stream	Specifies the path and name of the stream that the consumer subscribes to if, when subscribing to a topic, the consumer does not specify a stream. For example, the consumer can specify the name of a stream together with the name of a topic to write to, like this: /<stream>:<topic>. **Note:** rd_kafka_list groups API uses this consumer configuration to obtain the consumer groups.
streams.parallel.flushers.per.partition	Enables the producer may have multiple parallel send requests to the server for each topic partition. If this setting is set to true, it is possible for messages to be sent out of order.

last-updated: Apr 11, 2022

Property Name	Behavior
streams.producer.default.stream	Specifies the stream that the producer will use by default if the producer does not provide the name of a stream when specifying a topic to write to. For example, the producer can specify the name of a stream together with the name of a topic to write to, like this: /<stream>:<topic>. However, if the stream is not specified, the value of this configuration parameter is assumed to be the stream in which the topic is located. If the producer specifies the name of a topic without also providing the path and name of the stream, and there is no value for this configuration parameter, MapR Event Store For Apache Kafka assumes that the topic specified is in Apache Kafka and does nothing.

Additional Information

Here is a consumer configuration example:

```
conf = {'group.id': 'mygroup',

'session.timeout.ms': 6000,

'on_commit': my_commit_callback,

'default.topic.config': {'auto.offset.reset': 'smallest'}}

consumer = mapr_streams_python.Consumer(**conf)
```

Related Links

- rdkafka.h on page 2892
- Configuring Properties for Message Size on page 3029

MapR Event Store For Apache Kafka C#/.NET Applications

As of MapR Data Platform 6.0.1/EEP5.0, you can create C#/.NET applications for MapR Event Store For Apache Kafka using the MapR Event Store For Apache Kafka C#/.NET client. The MapR Event Store For Apache Kafka C#/.NET client is a binding for librdkafka and the MapR Event Store For Apache Kafka C Client is a distribution of librdkafka that works with MapR Event Store For Apache Kafka.

Requirements

- MapR Data Platform Client on Windows 7 (or higher) x64 operating systems
- MapR Data Platform cluster version 6.0.1 or greater
- Java 8 SDK and set Java HOME
- MapR Event Store For Apache Kafka C Client (mapr-librdkafka 0.11.3)
- MapR Event Store For Apache Kafka C#/.NET Client (mapr-streams-dotnet)
- .NET SDK 4.5.x or 4.6.x or .NET Core SDK 1.1
- nuget.exe

See Installing MapR Event Store For Apache Kafka C#/.NET Client on page 200 for installation information.

General Information

When developing and running MapR Event Store For Apache Kafka C#/.NET applications, note the following points:

- You can create producers and high-level consumers. Low-level consumers are not supported.
- Consuming or producing topics in a Kafka cluster is not supported.
- MapR Event Store For Apache Kafka offset values start at 1, not 0.
- MapR Data Platform security is supported including ACLs and ACEs for authorization. The unique Kafka security features that are part of Apache Kafka are not supported. See Security on page 683 for more information about MapR Data Platform security features.
- User impersonation is not supported.

Developing MapR Event Store For Apache Kafka C#/.NET Applications

Describes general tasks for developing C#/.NET applications.

Before Your Begin

Confirm that your environment meets the following requirements:

- MapR Data Platform cluster version 6.0.1 or greater.
- MapR Event Store For Apache Kafka C Client (mapr-librdkafka 0.11.3) is installed and configured on the node. See Configuring the MapR Event Store For Apache Kafka C Client on page 2796.
- MapR Event Store For Apache Kafka C#/.NET Client (mapr-streams-dotnet) is installed on the node.
- .NET SKD 4.5.x or 4.6.x
- .NET Core SDK 1.1
- nuget.exe

Create a Producer Application

In general, you want to create a producer that performs the following steps:

1. Import the producer class.
2. Define the producer and its configuration.
3. Produce data.
4. Wait for all messages to be sent to consumer.

In the following example code, three messages are produced to a topic named mytopic in a stream named my_stream.

```
class Producer
    {
        public static async void Produce()
        {
            string stream = "/my_stream";
            string topicName = "mytopic";

            var config = new Dictionary<string, object>
{ { "streams.producer.default.stream", stream } };
            var messages = new string[] { "Msg1", "Msg2", "Msg3" };
```


last-updated: Apr 11, 2022

```
            using (var producer = new Producer<Null, string>(config, null,
new StringSerializer(Encoding.UTF8)))
            {
                foreach (var msg in messages)
                {
                    var deliveryReport = await
producer.ProduceAsync(topicName, null, msg);
                    Console.WriteLine($"Delivery report:
{deliveryReport.TopicPartitionOffset}");
                }

                producer.Flush(TimeSpan.FromSeconds(1));
            }
        }
    }
```

Create a Consumer Application

In general, you want to create a consumer that performs the following steps:

1. Import the consumer class.
2. Define the consumer and its configuration.
3. Consume data.
4. Wait for all messages to be consumed.

In following example code, the MapR Event Store For Apache Kafka consumer is subscribed to `my_stream/mytopic` and it prints the content of each message that it reads.

```
using Confluent.Kafka;
using Confluent.Kafka.Serialization

    class Consumer
    {
        public static void Consume()
        {
            var stream = "/mystream";
            var topic = "mytopic";

            var config = new Dictionary<string, object>
            {
                { "group.id", "simple-csharp-consumer" },
                { "streams.consumer.default.stream", stream }
            };

            bool running = true;

            using (var consumer = new Consumer<Ignore, string>(config,
null, new StringDeserializer(Encoding.UTF8)))
            {
                var l = new List<TopicPartitionOffset> { new
TopicPartitionOffset(topic, 0, 0) };
                consumer.Assign(l);


                // Raised on critical errors, e.g. connection failures.
                consumer.OnError += (_, error) =>
                {
                    Console.WriteLine($"Error: {error}");
```

```
                    running = false;
                };

                // Raised on deserialization errors or when a consumed
message has an error != NoError.
                consumer.OnConsumeError += (_, error) =>
                {
                    Console.WriteLine($"Consume error: {error}");
                    running = false;
                };

                while (running)
                {
                    Message<Ignore, string> msg;
                    if (consumer.Consume(out msg, TimeSpan.FromSeconds(10)))
                    {
                        Console.WriteLine($"Topic: {msg.Topic} Partition:
{msg.Partition} Offset: {msg.Offset} {msg.Value}");
                    }
                }
            }
        }
    }
```

Run the Example Applications

To run the sample producer and consumer applications:

1. Create a stream named **mystream**.
2. Create a folder application.
3. Create a file named **example.cs**.
4. Add producer example code into the **example.cs** file.
5. Add consumer example code into the **example.cs** file.
6. Add an entry point for your application:

```
    class Demo
              {
              public static void Main(string[] args)
              {
              Producer.Produce();
              Consumer.Consume();
              }
              }
```

7. Create a project file named **example.csproj**.

8. Add the following dependency properties into the **example.csproj** file:

```
<?xml version="1.0" encoding="utf-8"?>
<Project ToolsVersion="15.0" xmlns="http://schemas.microsoft.com/
developer/msbuild/2003">
  <Import Project="$(MSBuildExtensionsPath)\$(MSBuildToolsVersion)
\Microsoft.Common.props" Condition="Exists('$(MSBuildExtensionsPath)\$
(MSBuildToolsVersion)\Microsoft.Common.props')" />
  <PropertyGroup>
    <Configuration Condition=" '$(Configuration)' == '' ">Debug</
Configuration>
    <Platform Condition=" '$(Platform)' == '' ">AnyCPU</Platform>
    <ProjectGuid>{99EDBA4B-D7DA-48BB-8D0C-AF4B12387935}</ProjectGuid>
    <OutputType>Exe</OutputType>
    <RuntimeIdentifiers>win10-x64</RuntimeIdentifiers>
    <RootNamespace>app</RootNamespace>
    <AssemblyName>app</AssemblyName>
    <TargetFrameworkVersion>v4.6.1</TargetFrameworkVersion>
    <FileAlignment>512</FileAlignment>
    <AutoGenerateBindingRedirects>true</AutoGenerateBindingRedirects>
  </PropertyGroup>
  <PropertyGroup Condition=" '$(Configuration)|$(Platform)' == 'Debug|
AnyCPU' ">
    <PlatformTarget>AnyCPU</PlatformTarget>
    <DebugSymbols>true</DebugSymbols>
    <DebugType>full</DebugType>
    <Optimize>false</Optimize>
    <OutputPath>bin\Debug\</OutputPath>
    <DefineConstants>DEBUG;TRACE</DefineConstants>
    <ErrorReport>prompt</ErrorReport>
    <WarningLevel>4</WarningLevel>
  </PropertyGroup>
  <ItemGroup>
    <Compile Include="app.cs" />
  </ItemGroup>
  <ItemGroup>
    <PackageReference Include="mapr-streams-dotnet" Version="0.11.3" />
  </ItemGroup>
    <Import Project="$(MSBuildToolsPath)\Microsoft.CSharp.targets" />
</Project>
```

9. Verify that you have completed the steps to configure the MapR Event Store For Apache Kafka C client or complete the steps now. See Configuring the MapR Event Store For Apache Kafka C Client on page 2796.

   Note: The MapR Event Store For Apache Kafka C#/.NET Client is dependent on the MapR Event Store For Apache Kafka C Client. Therefore, the MapR Event Store For Apache Kafka C Client must be configured before you can run the application.

10. Open your project folder on the command line and run:

```
dotnet run
```

Migrating Kafka C#/.NET Applications to MapR Event Store For Apache Kafka

With some modification, you can use existing confluent-kafka C#/.NET applications to consume and produce topics in MapR Event Store For Apache Kafka. The MapR Event Store For Apache Kafka C#/.NET Client is a binding for Apache librdkafka that works with MapR Event Store For Apache Kafka.

Migrating a .NET 4.5 or 4.6 Application

Note: This migration information is applicable for Windows (Win7-x64) platform *only*.

To migrate an existing .NET 4.5 or 4.6 application:

1. Install and configure the MapR Event Store For Apache Kafka C Client. See Configuring the MapR Event Store For Apache Kafka C Client on page 2796.
2. Replace the librdkafka.dll with the MapR Data Platform librdkafka 0.11.3 from **/bin/.../runtimes/<win7-x64>/<native folder>**.
3. Add a symlink from the **MapRClient.dll** to the **librdkafka.dll**.
4. Restart the application.

Migrating a .NET Core Application1

Note: This migration information is applicable for Windows (Win7-x64) and Linux platforms.

To migrate an existing .NET Core application:

1. Install and configure the MapR Event Store For Apache Kafka C Client. See Configuring the MapR Event Store For Apache Kafka C Client on page 2796.
2. Replace the librdkafka.dll with the MapR librdkafka 0.11.3 from **USER_HOME/.NUGET/PACKAGES/LIBRDKAFKA.REDIST/0.11.3/runtimes/<platform>/<native folder>**.
3. Add a symlink from the **MapRClient.dll** to the **librdkafka.dll**.
4. Restart the application.

Migrating a .NET Core Application2

Note: This migration information is applicable for Linux platforms *only*.

To migrate an existing .NET Core application:

1. Remove all **.so** files from the **~/.NUGET/PACKAGES/LIBRDKAFKA.REDIST/0.11.3/runtimes/<platform>/<native folder>** directory.
2. Install and configure the MapR Event Store For Apache Kafka C Client. See Configuring the MapR Event Store For Apache Kafka C Client on page 2796.
3. Replace the librdkafka.dll with the MapR librdkafka 0.11.3 from **USER_HOME/.NUGET/PACKAGES/LIBRDKAFKA.REDIST/0.11.3/runtimes/** directory.
4. If the MapR Client doesn not install into the ID search path, add a symlink from the **MapRClient.dll** to the **l/usr/local/lib**.
5. Restart the application.

last-updated: Apr 11, 2022

General Migration Information

- When you refer to a topic in the application code, include the path and name of the stream in which the topic is located:

  ```
  /<path and name of stream>:<name of topic>
  ```

  For example, you might have a stream in a MapR cluster that is named stream_A, and the stream might be in a volume named IoT and in a directory named automobile_sensors. You want to redirect a producer application to a topic in that stream. The syntax of the path to the topic might look like this: `/mapr/IoT/automobile_sensors/stream_A:<name of topic>`.

  Note: Optionally, use the `streams.consumer.default.stream` and `streams.producer.default.stream` configuration parameters. When you configure these parameters, applications can specify just the topic name to write or read from the default stream.

- Review the APIs that are supported and make changes to your application, as needed. See API for MapR Event Store For Apache Kafka C#/.NET on page 3016.

- See Configuration Properties for MapR Event Store For Apache Kafka C#/.NET Client on page 3018 for the list of supported configuration parameters and make changes to your application, as needed.

  Note: SSL-related configuration parameters are ignored. When you set these parameters, the MapR Event Store For Apache Kafka Client issues a warning indicating that the parameters are not supported.

API for MapR Event Store For Apache Kafka C#/.NET

MapR Event Store For Apache Kafka C#/.NET Client is a binding for librdkafka and the MapR Event Store For Apache Kafka C Client is a distribution of librdkafka that works with MapR Event Store For Apache Kafka.

Table

Core release	EEP Release	Kafka librdkafka version
As of MapR Data Platform 6.0.1	As of 5.0	0.11.3

Classes

Classes	Descripton
CommittedOffsets	Encapsulates information provided to a Consumer's OnOffsetsCommitted event - per-partition offsets and success/error together with overall success/error of the commit operation.
Consumer	Implements a high-level Apache Kafka consumer (without deserialization).
Consumer<TKey, TValue>	Implements a high-level Apache Kafka consumer (with key and value deserialization).
Error	Represents an error that occured when interacting with a Kafka broker or the librdkafka library.
ErrorCodeExtensions	Provides extension methods on the ErrorCode enumeration.
GroupInfo	Encapsulates information describing a particular Kafka group.

last-updated: Apr 11, 2022

Classes	Descripton
GroupMemberInfo	Encapsulates information describing a particular member of a Kafka group.
Ignore	A type for use in conjunction with that enables message keys or values to be read as null, regardless of their value.
KafkaException	Represents an error that occured during an interaction with Kafka.
Library	Methods that relate to the native librdkafka library itself (do not require a Producer or Consumer broker connection).
Loggers	OnLog callback event handler implementations.
LogMessage	Encapsulates information provided to the Producer/ Consumer OnLog event.
Message	Represents a message stored in Kafka.
Message<TKey, TValue>	Represents a (deserialized) message stored in Kafka.
Metadata	Kafka cluster metadata.
Null	A type for use in conjunction with and that enables null key or values to be enforced when producing or consuming messages.
PartitionMetadata	Metadata pertaining to a single Kafka topic partition.
Producer	Implements a high-level Apache Kafka producer (without serialization).
Producer<TKey, TValue>	Implements a high-level Apache Kafka producer with key and value serialization.
TopicMetadata	Metadata pertaining to a single Kafka topic.
TopicPartition	Represents a Kafka (topic, partition) tuple.
TopicPartitionError	Represents a Kafka (topic, partition, error) tuple.
TopicPartitionOffset	Represents a Kafka (topic, partition, offset) tuple.
TopicPartitionOffsetError	Represents a Kafka (topic, partition, offset, error) tuple.
TopicPartitionTimestamp	Represents a Kafka (topic, partition, timestamp) tuple.
WatermarkOffsets	Represents the low and high watermark offsets of a Kafka topic/partition.

Interfaces

Interface	Description
IDeliveryHandler	This interface is implemented by types that handle delivery report callbacks as a result of calls to Producer.ProduceAsync().
IDeliveryHandler<TKey, TValue>	This interface is implemented by types that handle delivery report callbacks as a result of calls to Producer<TKey,TValue>.ProduceAsync().

last-updated: Apr 11, 2022

Interface	Description
ISerializingProducer<TKey, TValue>	This interface describes the minimum functionality to be provided by a high level (serializing) Kafka producer.

Structs

Struct	Descripton
Offset	Represents a Kafka partition offset value.
Timestamp	Encapsulates a Kafka timestamp and its type.

Enums

Enum	Descripton
ErrorCode	Enumeration of local and broker generated error codes.
TimestampType	Enumerates the different meanings of a message timestamp value.

Configuration Properties for MapR Event Store For Apache Kafka C#/.NET Client

Describes the C#/.NET client configuration properties.

Global Configuration Properties

Property Name	Behavior
client.id	Same as librdkafka.

Property Name	Behavior
message.max.bytes	Supports a value less than or equal to 10MB (10000000). If this property is set to a value that is higher than 10MB, the client issues a warning and sets the configuration to 10MB. Produce calls fail when the message size is greater than 10MB.
receive.message.max.bytes	Same as librdkafka.

Property Name	Behavior
topic.blacklist	Same as librdkafka.
error_cb	Same as librdkafka.
opaque	Same as librdkafka.
enable.partition.eof	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

Property Name	Behavior
queue.buffering.max.kbytes	Same as librdkafka. Available as of librdkafka 0.11.3. Supported as of MapR Data Platform 6.0.1.

Consumer Configuration Properties

Property Name	Behavior
default_topic_conf	Same as librdkafka.
group.id	Same as librdkafka.
enable.auto.commit	Same as librdkafka.

Property Name	Behavior
auto.commit.interval.ms	Same as librdkafka.
rebalance_cb	Same as librdkafka.
offset_commit_cb	Same as librdkafka.

Property Name	Behavior
delivery.report.only.error	Same as librdkafka.
dr_msg_cb	Same as librdkafka.

Topic Configuration

Property Name	Behavior
partitioner_cb	Same as librdkafka.
auto.offset.reset	Supports the following values: beginning, end, earliest, latest, none, smallest, and largest. As of MapR Data Platform 6.0.1, beginning and end are supported.  **Note:** librdkafka additionally supports error.

last-updated: Apr 11, 2022

MapR Data Platform-Specific Configurations

Property Name	Behavior
streams.consumer.default.stream	Specifies the path and name of the stream that the consumer subscribes to if, when subscribing to a topic, the consumer does not specify a stream. For example, the consumer can specify the name of a stream together with the name of a topic to write to, like this: /<stream>:<topic>. **Note:** rd_kafka_list groups API uses this consumer configuration to obtain the consumer groups.

last-updated: Apr 11, 2022

Property Name	Behavior
streams.parallel.flushers.per.partition	Enables the producer to have multiple parallel send requests to the server for each topic partition. When this property is set to true, the default value, it is possible for messages to be sent out of order.

Property Name	Behavior
streams.producer.default.stream	Specifies the stream that the producer will use by default if the producer does not provide the name of a stream when specifying a topic to write to. For example, the producer can specify the name of a stream together with the name of a topic to write to, like this: /<stream>:<topic>. However, if the stream is not specified, the value of this configuration parameter is assumed to be the stream in which the topic is located. If the producer specifies the name of a topic without also providing the path and name of the stream, and there is no value for this configuration parameter, MapR Event Store For Apache Kafka assumes that the topic specified is in Apache Kafka and does nothing.

Related Links

- rdkafka.h on page 2892
- Configuring Properties for Message Size on page 3029

Utilities for MapR Event Store For Apache Kafka

MapR Event Store For Apache Kafka provides the utilities for operating on streams and topics.

Note: MapR Event Store For Apache Kafka cannot use MapR Database Shell to perform operations on streams or topics.

`mapr copystream`	This utility copies data from one MapR Stream to another MapR Stream. You can use it, for example, if you want to set up replication manually from one stream to another.
`mapr diffstreams`	This utility compares the message IDs, metadata, and data in two MapR Streams. Then, generates two

	directories that contain sequence files that you can use to merge the rows from the two MapR Streams.
mapr diffstreamswithcrc	This utility uses a cyclic redundancy check to detect differences between sets of messages in the specified MapR Streams. Then, for each set of non-identical messages, it performs a detailed comparison. Finally, it generates one or more directories of sequence files.
mapr exportstream and mapr importstream	Use these utilities together to export data from MapR Streams into binary sequence files, and then import the data from the binary sequence files into other MapR Streams. You can also use the `mapr importstream` utility to import changes that are specified in sequence files output by the `mapr diffstreams` utility.
mapr perfconsumer	This utility runs a consumer reading messages from topics in a MapR Stream. Use this utility to run consumers when you want to estimate the performance of consumers for your MapR Streams applications, given your network configuration.
mapr perfproducer	This utility runs a producer, generating messages and publishing them to a MapR Stream. Use this utility to run producers when you want to estimate the performance of producers for your MapR Streams applications, given your network configuration.
mapr streamanalyzer	This light-weight utility, which is a sample application for the `Streams` Java class for analytics on MapR Streams, lets you count the messages in a stream or a subset of the topics in a stream. The utility also lets you print either whole retrieved messages or a subset of the fields in each message.

Configuring Properties for Message Size

Describes the `message.max.bytes` and `receive.message.max.bytes` properties for configuring message size.

message.max.bytes	For a C producer, the `message.max.bytes` value is 1000000 B by default. The minimum value is 1000 B and maximum value is 32000000 B. The maximum message size produced by a C API is decided by the `message.max.bytes` value set on the C producer. From C, Python, and C# APIs, the maximum message size that can be produced is 32000000 B. If a C consumer needs to consume a message that is greater than 32000000 B in size, which may be produced by a Java client, the consumer needs to update the `message.max.bytes` or `receive.message.max.bytes` properties to a higher value to consume it. If the `message.max.bytes` property is set to greater than 32000000 B, it is by default capped at 32000000 B. Though a consumer can consume messages greater than 32000000 B (produced say by a Java client), only up to 32000000 B is produced from the MapR C client. The maximum message size consumed by a C API is limited by the value that is higher among the `message.max.bytes` and `receive.message.max.bytes` values.

last-updated: Apr 11, 2022

receive.message.max.bytes	For a C consumer, the `receive.message.max.bytes` is 1000000 B by default. The minimum value is 1000 B and maximum value is 1000000000 B.

Using a Java API, a larger message size can be produced if the cluster-side property is changed using the following `maprcli config save` command:

```
Cluster side:

maprcli config save -values
{"mfs.db.max.rowsize.kb":<value in KB>}
```

In this case, the row size is 32 MB by default and the maximum is a little less than 2 GB.

The `mfs.db.max.rowsize.kb` setting is a cluster-wide setting that applies to MapR Database (Binary+JSON) and MapR Event Store for Apache Kafka, and it is not configurable per stream or topic.

MapReduce and Apps

This section contains information associated with developing YARN applications.

External Applications and Classpath

Describes how to configure the class path for external applications.

MapReduce version 2 applications require the hadoop 2.x or the yarn classpath, and other applications that can run on YARN require the yarn classpath.

The method to specify the classpath differs based on how the job or application is submitted:

Method used to Submit the Job	Method to Specify Classpath
The external application uses the `hadoop jar` or the `yarn jar` command.	YARN applications (MapReduce or custom applications) that are submitted using the `yarn jar` command will automatically use the yarn classpath. If the external application has a service that submits the job, you can set the CLASSPATH environment variable to point to a different classpath prior to starting the service. In this case, the hadoop classpath that you set in the CLASSPATH environment variable takes priority over the hadoop classpath for the `hadoop jar` command.
The external application does not use the `hadoop jar` or the `yarn jar` command.	Set the classpath using one of the following options: • If the external application has a service that submits the job or application, you can set the CLASSPATH environment variable to point to the hadoop or yarn classpath prior to starting the service. • Set the classpath within the application. Use one of the following methods to get the classpath: • `hadoop2 classpath` or `hadoop -yarn classpath`: Gets the classpath for MRv2 applications. • `yarn classpath`: Gets the classpath for YARN applications (MRv2 or other applications that can run on YARN).

Important: When you launch a spring boot application, ensure that you prefix the classpath with `/opt/mapr/conf:/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop`. Alternatively, copy the `core-site.xml` file to the `/src/main/resources/` folder.

Classpath Construction

This section describes how the MapReduce classpath is constructed.

The classpath that is used to run a MapReduce program is constructed based on how the program is submitted.

When you submit an application from the command line, the classpath used to process the program is based on the following items in this order of priority:

1. JARs in the program's classpath, such as the hadoop 2.x or yarn classpath.
2. JARs specified with the `-libjar` parameter which can be appended to `hadoop jar` or `yarn jar` commands.

If an external application submits the application, the classpath that is used to process the jar file is based on the following items in this order of priority:

1. JARs in the classpath of the external application.
2. JARs in the program's classpath, such as the hadoop 2.x or yarn classpath.
3. JARs specified with `-libjar` parameter which can be appended to `hadoop jar` or `yarn jar` commands.

Managing Third-Party Libraries

Any third-party library that is required by a MapReduce program must be accessible to the data node that processes the application.

A data node is a node in the cluster that includes the NodeManager role. You can provide the third-party libraries when you submit the program, or you can install the third-party libraries on each node that processes the application.

Include the third-party libraries with each program

Including the third-party libraries with each program is the preferred method.

Perform one the following operations to include the third-party jars when you submit the program:

- Package the third-party libraries with the MapReduce jar file. The benefit of this method is that the node from which you submit the program and the node that runs the program are not required to have the libraries files.
- Use the `-libjars` parameter to specify the third-party libraries on the command line. With this option, the library files are submitted to the data node along with the program. The benefit of this method is that the node that runs the program does not need to have the library files installed. However, the node that submits the program must have the library files installed.

Install the third-party libraries on each node that runs the program

You can also install the third-party libraries on each data node. However, this may not be preferred as there could be conflicts between library versions or library files.

To install the third-party libraries on each data node, perform one of the following operations:

- Install the third-party libraries in the following directory on each Node Manager node: `/opt/mapr/hadoop/hadoop-2.x/share/hadoop/common`

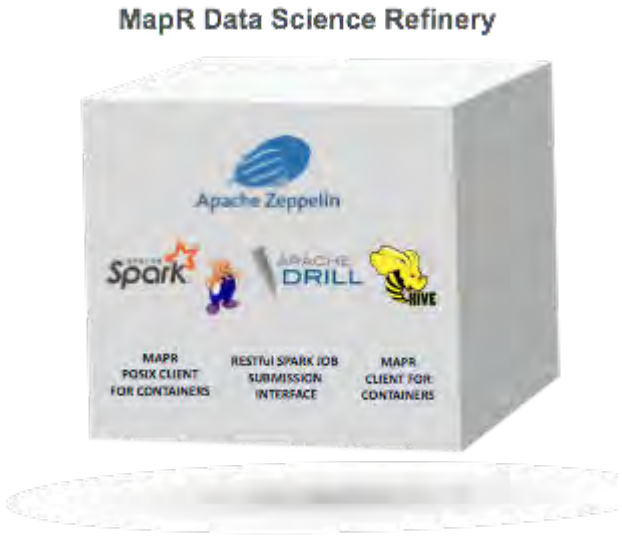
last-updated: Apr 11, 2022

- On each node with the NodeManager role, install the required third-party libraries and then specify the location(s) of the third-party libraries with the HADOOP_CLASSPATH env variable in the `env_override.sh` file. The `env_override.sh` file is located in the following directory: `/opt/mapr/conf`. For more information about the file, see About env_override.sh on page 2290.

MapR Data Science Refinery

The MapR Data Science Refinery is an easy-to-deploy and scalable data science toolkit with native access to all platform assets and superior out-of-the-box security.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.



The MapR Data Science Refinery offers:

Access to All Platform Assets	The MapR FUSE-based POSIX Client allows app servers, web servers, and other client nodes and apps to read and write data directly and securely to a MapR cluster, like a Linux filesystem. In addition, connectors are provided for interacting with both MapR Database and MapR Event Store For Apache Kafka via Apache Spark connectors.
Superior Security	The MapR Platform provides enhanced security. Apache Zeppelin on MapR leverages and integrates with this security layer using the built-in capabilities provided by the MapR Persistent Application Container (PACC).
Extensibility	Apache Zeppelin is paired with the Helium framework to offer pluggable visualization capabilities.
Simplified Deployment	A preconfigured Zeppelin Docker container provides the ability to leverage MapR as a persistent data store.

Getting Started Using the Data Science Refinery with Zeppelin

You can deploy the Apache Zeppelin Docker container included in the Data Science Refinery on any of the following, listed in order of recommendation for best practice, starting with the most preferable option:

- Container orchestration engines; for example: Docker Swarm, Kubernetes, OpenShift

- Cloud instances
- Shared edge node
- Personal computers

Note: Starting in version 1.2, you can deploy the Data Science Refinery on a MapR cluster node. Make sure you take into consideration the resource requirements of the Data Science Refinery, if you choose this deployment mode.

If you are already familiar with Apache Zeppelin on MapR and want to skip to the deployment instructions, see Running the Zeppelin Container on page 3034.

Related information

MapR Data Science Refinery Product Page

Zeppelin on MapR

The MapR Data Science Refinery includes a preconfigured Apache Zeppelin notebook, packaged as a Docker container. Apache Zeppelin is an open source web-based data science notebook. You can use it with MapR components to conduct data discovery, ETL, machine learning, and data visualization.

You can run the Zeppelin container either on your laptop or on MapR edge nodes. Out of box, the Zeppelin container image is integrated with open source data processing engines like Apache Spark, Apache Drill, and Apache Hive, as well as with native MapR engines (MapR File System, MapR Database, and MapR Event Store For Apache Kafka). Using the notebook simply requires running the Docker image and connecting to the container through your browser.

Zeppelin provides the following benefits for your data engineering and data science use cases:

- An interactive development environment for writing, testing, and sharing data processing code snippets
- The ability to run the notebooks in a local client environment, such as on a laptop
- Support for a variety of interpreters for integrating with different backend components
- Support for extensible visualization libraries

The Zeppelin notebook included with the Data Science Refinery provides additional benefits:

- A small footprint, pre-built, certified data science container that is easy to deploy and run
- An isolated environment where you can experiment with libraries and packages without affecting other users' work
- Secure authentication at the container level across a secure Web connection
- Preconfigured JDBC interpreters for accessing query engines like Apache Drill and Apache Hive
- The MapR FUSE-Based POSIX Client on page 1238, which you need to access File System on page 452 using shell commands

last-updated: Apr 11, 2022

- All client side services that you need to submit Apache Spark jobs, including jobs that access MapR Event Store For Apache Kafka on page 627
- MapR connectors, which you need to access MapR Database on page 496 (both binary and JSON tables)

See Zeppelin Release Notes on page 5641 for release specific information.

For additional information about Zeppelin, you can also refer to the open source documentation.

Running the Zeppelin Container

To run the Apache Zeppelin container, you must access the Zeppelin Docker image from MapR's public repository, run the Docker image, and access the deployed container from your web browser. From your browser, you can create Zeppelin notebooks.

To pull and run the Docker image, you must first install `Docker` on the host where you want to run the container. You can download the software from https://docs.docker.com/engine/installation/.

Docker is a tool that you use to package an application and its dependencies into a virtual container. You can deploy and run the container on any node. MapR uses this technology to package Apache Zeppelin with all software you need to integrate with various engines. After you deploy the container, all software and services used by Zeppelin are started and ready for you to use.

Important: After installing Docker, if you configure a memory limit, ensure that it is at least 3.5 GB of memory. Otherwise, you may not be able to start the Zeppelin container or may encounter log in problems.

See https://docs.docker.com/ for additional information about Docker and https://docs.docker.com/docker-hub/ for information about running Docker images from cloud-based repositories.

Accessing the Zeppelin Docker Image

To access a container image through Docker, you specify the image name and a tag. In the case of the MapR Data Science Refinery, the tag specifies the operating system of the running container. This operating system does not need to match your host operating system.

The image name and available tags for the latest release are:

Image Name	Operating System Version of Running Container	Tag
maprtech/data-science-refinery	CentOS 7	v1.4.1_6.1.0_6.3.0_centos7
	Ubuntu 16	v1.4.1_6.1.0_6.3.0_ubuntu16

For a complete list of all available tags, see https://hub.docker.com/r/maprtech/data-science-refinery/tags/.

1. Download the Docker image, referencing the image name and tag. The following downloads the image for CentOS:

   ```
   docker pull maprtech/data-science-refinery:v1.4.1_6.1.0_6.3.0_centos7
   ```

   Note: A best practice is to choose the image OS that matches the OS running in your MapR cluster.

2. Verify that you have downloaded the Docker image by listing the Docker images on your machine:

   ```
   docker images
   ```

Running the Zeppelin Docker Image

To run a Docker image, you use the `docker run` command. You must specify various parameters, including parameters indicating the MapR cluster you want to access from your notebook. You must also specify a user name and password. Only that user can access the running container.

If you are running the MapR Data Science Refinery as a Kubernetes service, follow the steps at Running MapR Data Science Refinery as a Kubernetes Service on page 3049.

Important: If you plan to use the FUSE-based POSIX client, make sure you have a MapR POSIX Client for Containers license on your MapR cluster before performing the steps described in this topic.

1. Determine what parameters you want to pass to Docker

   Note: You cannot access the container as the `root` user. Make sure to specify an alternative user name for the `MAPR_CONTAINER_USER` environment variable.

2. Pass the parameters you have selected to `docker run` by using one of the following options:

   Command Line

   Specify all your parameters in the command line:

   ```
   docker run -it <parameters> \
      maprtech/
   data-science-refinery:v1.4.1_6.1.0_6
   .3.0_centos7
   ```

   Env File

   Specify your environment variable (`-e`) parameters in a file and the remaining parameters in the command line.

   Pass the file to `docker run` using `--env-file`.

   In the following example, the file `env.list` contains your environment variable parameters:

   ```
   docker run -it --env-file ./
   env.list \
      -p 9995:9995 \
      -p 10000-10010:10000-10010 \
      -p 11000-11010:11000-11010 \
      -v /home/mapruser1/
   mapr_ticket:/tmp/mapr_ticket:ro \
      --cap-add SYS_ADMIN \
      --cap-add SYS_RESOURCE \
      --device /dev/fuse \
      --security-opt
   apparmor:unconfined \
      maprtech/
   data-science-refinery:v1.4.1_6.1.0_6
   .3.0_centos7
   ```

   The following shows an example of the contents of `env.list`:

   ```
   HOST_IP=172.24.9.151
   MAPR_CLUSTER=my.cluster.com
   MAPR_CLDB_HOSTS=172.24.11.84,172.24.
   8.72,172.24.9.248
   MAPR_CONTAINER_USER=mapuser1
   MAPR_CONTAINER_PASSWORD=SeCreTpAsSw0
   MAPR_CONTAINER_GROUP=mapr
   MAPR_CONTAINER_UID=5000
   ```


last-updated: Apr 11, 2022

```
MAPR_CONTAINER_GID=5000
MAPR_TICKETFILE_LOCATION=/tmp/
mapr_ticket
MAPR_MOUNT_PATH=/mapr
MAPR_HS_HOST=172.24.9.248
ZEPPELIN_NOTEBOOK_DIR=/mapr/
my.cluster.com/user/mapruser1/
notebook
MAPR_TZ=US/Pacific
```

3. Verify that Zeppelin is running:

```
docker ps
```

If you want to run a second Docker image on a host machine, follow the instructions at Running Multiple Zeppelin Containers on a Single Host on page 3045.

Understanding Zeppelin Docker Parameters

There are a set of key parameters to use when running Apache Zeppelin containers. This includes parameters related to the connection port, bridge networking, specifying your MapR cluster, enabling security through MapR ticketing, and enabling the FUSE-based POSIX client.

The general syntax for running the Apache Zeppelin Docker image is the following:

```
docker run -it -p 9995:9995 \
    -e HOST_IP=<docker-host-ip> \
      -p 10000-10010:10000-10010 \
      -p 11000-11010:11000-11010 \
    -e MAPR_CLUSTER=<cluster-name> \
      -e MAPR_CLDB_HOSTS=<cldb-ip-list> \
    -e MAPR_CONTAINER_USER=<user-name> \
      -e MAPR_CONTAINER_PASSWORD=<password> \
      -e MAPR_CONTAINER_GROUP=<group-name> \
      -e MAPR_CONTAINER_UID=<uid> \
      -e MAPR_CONTAINER_GID=<gid> \
      -e MAPR_TICKETFILE_LOCATION=<ticket-file-container-location> \
      -v <ticket-file-host-location>:<ticket-file-container-location>:ro \
    -e MAPR_MOUNT_PATH=<path-to-fuse-mount-point> \
      --cap-add SYS_ADMIN \
      --cap-add SYS_RESOURCE \
      --device /dev/fuse \
      --security-opt apparmor:unconfined \
    -e MAPR_HS_HOST=<historyserver-ip> \
    -e ZEPPELIN_NOTEBOOK_DIR=<path-for-notebook-storage> \
    -e MAPR_TZ=<time-zone> \
    maprtech/data-science-refinery:v1.4.1_6.1.0_6.3.0_centos7
```

Following is a sample command:

```
docker run -it -p 9995:9995 \
    -e HOST_IP=172.24.9.151 \
      -p 10000-10010:10000-10010 \
      -p 11000-11010:11000-11010 \
    -e MAPR_CLUSTER=my.cluster.com \
      -e MAPR_CLDB_HOSTS=172.24.11.84,172.24.8.72,172.24.9.248 \
    -e MAPR_CONTAINER_USER=mapuser1 \
      -e MAPR_CONTAINER_PASSWORD=SeCreTpAsSw0 \
      -e MAPR_CONTAINER_GROUP=mapr \
      -e MAPR_CONTAINER_UID=5000 \
      -e MAPR_CONTAINER_GID=5000 \
      -e MAPR_TICKETFILE_LOCATION=/tmp/mapr_ticket \
```

```
      -v /home/mapruser1/mapr_ticket:/tmp/mapr_ticket:ro \
    -e MAPR_MOUNT_PATH=/mapr \
      --cap-add SYS_ADMIN \
      --cap-add SYS_RESOURCE \
      --device /dev/fuse \
      --security-opt apparmor:unconfined \
    -e MAPR_HS_HOST=172.24.9.248 \
    -e ZEPPELIN_NOTEBOOK_DIR=/mapr/my.cluster.com/user/mapruser1/notebook \
    -e MAPR_TZ=US/Pacific \
    maprtech/data-science-refinery:v1.4.1_6.1.0_6.3.0_centos7
```

The following sections describe each category of parameters in more detail. Where appropriate, the descriptions reference the sample command.

For a list of all MapR-specific environment variables, refer to the MapR-Specific Environment Variables on page 3044 section at the end of this topic.

Connection Port

By default, the Zeppelin notebook runs on port 9995. To use a different port number, pass the `ZEPPELIN_SSL_PORT` environment variable in your `docker run` command and specify the `<port-number>`:

```
docker run -it \
    ...
    -e ZEPPELIN_SSL_PORT=<port-number> \
    -p <port-number>:<port-number> \
    ...
```

Important: If you are running on Mac, you must publish the container port by specifying `-p <port-number>:<port-number>` in your `docker run` command.

Bridge Networking

By default, Docker uses bridge networking. In general, bridge networking provides better isolation from the host machine and other containers.

You must set the `HOST_IP` environment variable, the `-p 10000-10010:10000-10010` and `-p 11000-11010:11000-11010` parameters when using bridge networking:

```
docker run -it \
    ...
    -e HOST_IP=<docker-host-ip> \
    -p 10000-10010:10000-10010 \
    -p 11000-11010:11000-11010 \
    ...
```

The `<docker-host-ip>` must be an actual IP address. If you are running the container on your laptop, you cannot specify `localhost` as the IP address.

Specifying the 10000-10010 port range reserves the range for the Livy launcher. If you are already using these ports for other reasons, use the `LIVY_RSC_PORT_RANGE` environment variable to specify a different range.

If you plan to use Spark interpreter, you must reserve the 11000-11010 port range for Spark. To reserve a different port range, use the `SPARK_PORT_RANGE` environment variable.

For example, the following command reserves two different sets of port ranges for Livy and Spark:

```
docker run -it \
    ...
```


last-updated: Apr 11, 2022

```
    -e HOST_IP=<docker-host-ip> \
    -p 10011-10021:10011-10021 \
    -e LIVY_RSC_PORT_RANGE="10011~10021" \
    -p 13011-13021:13011-13021 \
    -e SPARK_PORT_RANGE="13011~13021" \
    ...
```

Note: Use tilde (~) rather than dash (-) when specifying the range with the `LIVY_RSC_PORT_RANGE` and `SPARK_PORT_RANGE` environment variables.

If you prefer to use host networking, specify the following parameter in your `docker run` command instead:

```
docker run -it \
    ...
    --network=host \
    -e HOST_IP=<docker-host-ip> \
    ...
```

Note: You do not need to reserve port ranges when using host networking.

See https://docs.docker.com/engine/userguide/networking/ for more details about Docker networking.

MapR Cluster

Identify the MapR cluster you want your container to access by specifying the name of your MapR cluster and a comma separated list of the IP addresses of the cluster's CLDB nodes. The following specifies three CLDB nodes:

```
docker run -it \
    ...
    -e MAPR_CLUSTER=my.cluster.com \
    -e MAPR_CLDB_HOSTS=172.24.11.84,172.24.8.72,172.24.9.248 \
    ...
```

MapR Ticketing

If your MapR cluster is secure, you need a copy of the MapR ticket on your local host so you can specify a mount point in your `docker run` command. This makes the ticket visible to the Zeppelin container. The sample command shown earlier uses MapR tickets.

To determine whether your cluster is secure, view the contents of the file `/opt/mapr/conf/mapr-clusters.conf` on your MapR cluster. For example, the following shows a secure cluster:

```
my.cluster.com secure=true ip-172-24-11-84
```

If your cluster is secure, follow these steps to make the ticket visible to the Zeppelin container:

1. Generate a service ticket for the container user by following the instructions at Generating a Service Ticket on page 1428.
2. Copy the generated ticket file to your local host machine. This is your source ticket file.
3. Change the owner and group on your source ticket so it matches the UID and GID in the ticket file.
4. Specify the source ticket path in the Docker mount point, as described in the table below.

The table lists the parameters related to MapR tickets and their values in the sample command:

Parameter	Sample Parameter Value	Details
`MAPR_CONTAINER_USER`	`mapruser1`	The only user who can access the notebook
`MAPR_CONTAINER_PASSWORD`	`SeCreTpAsSw0`	The password you use to log in to your Zeppelin notebook. This password does not need to match the password in your MapR cluster. If not specified, it defaults to the value of `MAPR_CONTAINER_USER`.
`MAPR_CONTAINER_GROUP`	`mapr`	Name of the container user's group
`MAPR_CONTAINER_UID`	5000	UID of the container user; must be consistent with the value in the ticket file
`MAPR_CONTAINER_GID`	5000	GID of the container user; must be consistent with the value in the ticket file
`MAPR_TICKETFILE_LOCATION`	`/tmp/mapr_ticket`	Location of the ticket file in the container
`-v <ticket-file-host-location>:<ticket-file-container-location>:ro`	`-v /home/mapruser1/mapr_ticket:/tmp/mapr_ticket:ro`	Docker mount point for the source and destination of your ticket file **`<ticket-file-location>`**: • Source ticket file • Location of the ticket file on your local host **`<ticket-file-container-location>`**: • Destination ticket file • Location of the ticket file in the container • Must match the value of the `MAPR_TICKETFILE_LOCATION` parameter

See Security Considerations for the MapR PACC on page 406 for further information.

FUSE-Based POSIX Client

With the FUSE POSIX Client for File-Based Applications on page 403, you can access MapR filesystem using POSIX shell commands instead of Hadoop commands. To do so, you must specify the MapR filesystem mount point environment variable (`MAPR_MOUNT_PATH`) and other FUSE parameters in your `docker run` command. In the sample command shown earlier, the following are the relevant parameters and their settings:

Parameter	Sample Parameter Value
`MAPR_MOUNT_PATH`	`/mapr`
`--cap-add`	`SYS_ADMIN`
`--cap-add`	`SYS_RESOURCE`
`--device`	`/dev/fuse`
`--security-opt`	`apparmor:unconfined`

All of these parameters are required except `security-opt`. You must specify `security-opt` if you are running on an Ubuntu host.

last-updated: Apr 11, 2022

Important: You must have a MapR POSIX Client for Containers license on your MapR cluster to use the FUSE-based POSIX client. Without a license, the MapR filesystem mount point you specified will be empty. You can confirm a missing license by checking for errors in `/opt/mapr/logs/posix-client-container.log` inside your container.

Pig, Livy, and Spark Interpreters

If you plan to use the Pig, Livy, or Spark interpreters, you should set the `MAPR_HS_HOST` environment variable to the IP address of your MapR cluster's HistoryServer:

```
docker run -it ... -e MAPR_HS_HOST=172.24.9.248 ...
```

This enhances the performance of those interpreters. If your MapR cluster does not have a HistoryServer, your Pig and Spark jobs will run, but they may perform poorly.

Notebook Storage

The environment variable `ZEPPELIN_NOTEBOOK_DIR` specifies where to store your notebooks. If you do not specify `ZEPPELIN_NOTEBOOK_DIR`, Zeppelin stores your notebooks in the directory `/opt/mapr/zeppelin/zeppelin-0.8.0/notebook`.

Storage Options

You can store your notebooks either in MapR File System or your container's local filesystem:

MapR File System using the FUSE-based POSIX client

In the sample command shown earlier, MapR Data Science Refinery stores your notebooks in a directory named `/user/mapruser1/notebook` in MapR filesystem using the FUSE-Based POSIX Client on page 3039.

The example assumes your MapR filesystem mount point is `/mapr` and your cluster name is `my.cluster.com`:

```
docker run -it ... \
   -e ZEPPELIN_NOTEBOOK_DIR=/
mapr/my.cluster.com/user/mapruser1/
notebook ...
```

Important: You must specify the parameters used by the FUSE-Based POSIX Client on page 3039. If Docker is unable to start the FUSE-based client, you cannot open Zeppelin in your browser. Your browser will return the following error:

```
HTTP ERROR: 503
```

MapR File System without the FUSE-based POSIX client

Starting in MapR Data Science Refinery 1.3, you can store your notebooks in MapR filesystem without using the FUSE-based POSIX client. To use this option, specify the full MapR filesystem path in the `ZEPPLIN_NOTEBOOK_DIR` variable.

last-updated: Apr 11, 2022

The following example is equivalent to the previous:

```
docker run -it ... \
   -e ZEPPELIN_NOTEBOOK_DIR=maprfs:///
user/mapruser1/notebook ...
```

Local filesystem

To store your notebooks in your local filesystem, specify the container's local path in `ZEPPELIN_NOTEBOOK_DIR`:

```
docker run -it ... \
   -e ZEPPELIN_NOTEBOOK_DIR=/opt/mapr/
notebook ...
```

Zeppelin Tutorial Notebook

If you set `ZEPPELIN_NOTEBOOK_DIR`, perform the following steps to enable access to the tutorial:

1. Manually move the tutorial notebook from the default directory to your specified notebook directory.

   The following command to move the tutorial from the default location to MapR filesystem path should be run from inside the container running Zeppelin:

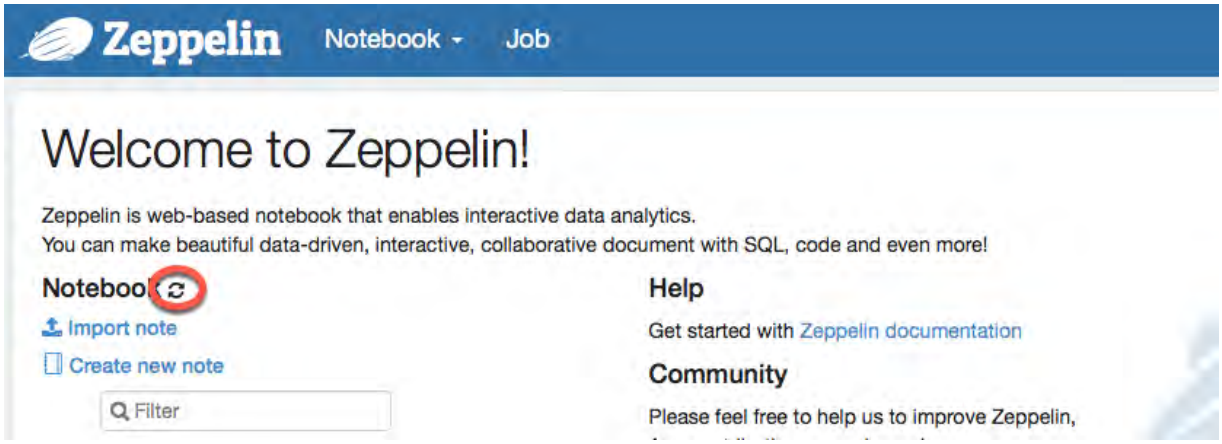
   POSIX

   ```
   cp -r /opt/mapr/zeppelin/
   zeppelin-0.8.0/notebook/* /mapr/
   my.cluster.com/user/mapruser1/
   notebook/
   ```

   Hadoop

   ```
   hadoop fs -put /opt/mapr/zeppelin/
   zeppelin-0.8.0/notebook/* maprfs:///
   user/mapruser1/notebook/
   ```

2. After moving the notebook, make sure you reload your notebooks from storage by clicking on the icon circled in red below:



Python Version

By default, when you use Python with either the Livy or Spark interpreters, the interpreters use Python 2. Although you can run only one version of Python with MapR Data Science Refinery, you can switch

last-updated: Apr 11, 2022

to Python 3, by setting the PYSPARK_PYTHON environment variable to the absolute path of the Python 3 executable on your MapR cluster:

```
docker run -it ... -e PYSPARK_PYTHON=/usr/local/bin/python3.6 ...
```

You can also install custom Python packages. See Installing Custom Packages for PySpark on page 3066

Idle Interpreter Timeout Threshold

Starting with the 1.3 release, by default, MapR Data Science Refinery terminates interpreters that have been idle for one hour. To modify this idle timeout threshold, specify the ZEPPELIN_INTERPRETER_LIFECYCLE_MANAGER_TIMEOUT_THRESHOLD environment variable. The parameter value is in milliseconds.

The following example sets the idle timeout to 10 minutes:

```
docker run -it ... \
   -e ZEPPELIN_INTERPRETER_LIFECYCLE_MANAGER_TIMEOUT_THRESHOLD=600000 ...
```

If a Spark job terminates due to the Spark interpreter reaching the timeout threshold, the job shows a status of FAILED in the YARN UI.

Configuration Storage

By default, the following Zeppelin configuration files are stored in /opt/mapr/zeppelin/zeppelin-0.8.0/conf/:

- interpreter.json
- notebook-authorization.json

Starting with MapR Data Science Refinery 1.3, you can store these files in MapR filesystem by specifying the ZEPPELIN_CONFIG_FS_DIR environment variable. You can also specify a local system for this variable.

The following shows sample commands for the three available options:

MapR File System using the FUSE-based POSIX client

This example assumes your MapR filesystem mount point is /mapr and your cluster name is my.cluster.com:

```
docker run -it ... \
   -e ZEPPELIN_CONFIG_FS_DIR=/mapr/
my.cluster.com/user/mapruser1/dsrconf
...
```

Important: You must specify the parameters used by the FUSE-Based POSIX Client on page 3039 to use this option.

MapR File System without the FUSE-based POSIX client

The following example is equivalent to the previous:

```
docker run -it ... \

-e ZEPPELIN_CONFIG_FS_DIR=maprfs:///
user/mapruser1/dsrconf ...
```


last-updated: Apr 11, 2022

Local filesystem

The following specifies a local filesystem path:

```
docker run -it ... \
   -e ZEPPELIN_CONFIG_FS_DIR=/opt/
mapr/dsrconf ...
```

If all of the following apply, then you must restart all containers to enable the new configuration settings:

- `ZEPPELIN_CONFIG_FS_DIR` is set to a MapR filesystem path
- Multiple containers share the two configuration files
- You make a change in either of the configuration files that requires a container restart

Default Drill JDBC Connection URL

The default Drill JDBC connection URL is `jdbc:drill:drillbit=drillbitnode:31010`. Starting with MapR Data Science Refinery 1.3, you can configure this default URL using one of the following two environment variables:

`MAPR_DRILLBITS_HOST`

Description

Comma separated list of Drillbit servers and optional port numbers.

Use if you want to connect to Drill through a Drillbit server. If you do not specify port numbers, they default to 31010.

Sample commands

```
docker run -it ... \
  -e
MAPR_DRILLBITS_HOSTS=node1.cluster.com
,node2.cluster.com ...

docker run -it ... \
  -e
MAPR_DRILLBITS_HOSTS=node1.cluster.com
:31010,node2.cluster.com:31010 ...
```

Resulting URL

```
jdbc:drill:drillbit=node1.cluster.com:
31010,node2.cluster.com:31010
```

`MAPR_ZK_QUORUM`

Description

Comma separated list of servers and optional port numbers in your Zookeeper cluster.

Use if you want to connect to Drill through a Zookeeper cluster. If you do not specify port numbers, they default to 5181.

Sample commands

```
docker run -it ... \
  -e
MAPR_ZK_QUORUM=node1.cluster.com,node2
.cluster.com,node3.cluster.com \
  ...

docker run -it ... \
  -e
```


last-updated: Apr 11, 2022

```
MAPR_ZK_QUORUM=node1.cluster.com:5181,
node2.cluster.com:5181,node3.cluster.c
om:5181 \
  ...
```

Resulting URL

```
jdbc:drill:zk=node1.cluster.com:5181,n
ode2.cluster.com:5181,node3.cluster.co
m:5181/drill/my.cluster.com-drillbits
```

You should specify only one of the two environment variables. If you mistakenly specify both, `MAPR_DRILLBITS_HOST` takes precedence.

See Start the Drill Shell (SQLLine) on page 3270 for more information about the Drill JDBC connection URL. See Drill JDBC on page 3071 for more information about setting other options in your Drill JDBC connection string for MapR Data Science Refinery.

MapR-Specific Environment Variables

The following table lists all MapR-specific environment variables you can use in your `docker run` command. The second column contains a short description of each variable. The third column provides links to detailed descriptions, including situations where you need to use each variable.

Environment Variable	Description	Documentation Link
`HOST_IP`	IP address of your Docker host machine	Bridge Networking on page 3037
`LIVY_RSC_PORT_RANGE`	Port range reserved for the Livy launcher	Bridge Networking on page 3037 Running Multiple Zeppelin Containers on a Single Host on page 3045
`MAPR_CLUSTER`	Name of your MapR cluster	MapR Cluster on page 3038
`MAPR_CLDB_HOSTS`	List of CLDB host IPs	MapR Cluster on page 3038
`MAPR_CONTAINER_GID`	GID of the container user	MapR Ticketing on page 3038
`MAPR_CONTAINER_GROUP`	Group name of the container user	MapR Ticketing on page 3038
`MAPR_CONTAINER_PASSWORD`	Password used to log in to the container UI	MapR Ticketing on page 3038
`MAPR_CONTAINER_UID`	UID of the container user	MapR Ticketing on page 3038
`MAPR_CONTAINER_USER`	Name of the container user	MapR Ticketing on page 3038
`MAPR_DRILLBITS_HOSTS`	Comma separated list of Drillbit servers for connecting to Drill	Default Drill JDBC Connection URL on page 3043
`MAPR_HS_HOST`	IP address of your MapR cluster's HistoryServer	Pig, Livy, and Spark Interpreters on page 3040
`MAPR_MOUNT_PATH`	Path of the mount point for MapR filesystem	FUSE-Based POSIX Client on page 3039
`MAPR_TICKETFILE_LOCATION`	Location of MapR ticket file in your container	MapR Ticketing on page 3038
`MAPR_TZ`	Time zone inside the container	Running the MapR PACC Using Docker on page 413

Environment Variable	Description	Documentation Link
`MAPR_ZK_QUORUM`	Comma separated list of servers in your Zookeeper cluster for connecting to Drill	Default Drill JDBC Connection URL on page 3043
`PYSPARK_PYTHON`	Location of Python 3 executable on your MapR cluster	Python Version on page 3041
`SPARK_PORT_RANGE`	Port range reserved for the Spark interpreter	Bridge Networking on page 3037 Running Multiple Zeppelin Containers on a Single Host on page 3045
`ZEPPELIN_ARCHIVE_PYTHON`	Path containing your archive with custom Python packages	Installing Custom Packages for PySpark on page 3066
`ZEPPELIN_CONFIG_FS_DIR`	Path containing the following Zeppelin configuration files: • `interpreter.json` • `notebook-authorization.json`	Configuration Storage on page 3042
`ZEPPELIN_DEPLOY_MODE`	Set to `kubernetes` if running Zeppelin as a Kubernetes service	Running MapR Data Science Refinery as a Kubernetes Service on page 3049
`ZEPPELIN_INTERPRETER_LIFECYCLE_MANAGER_TIMEOUT_THRESHOLD`	Timeout threshold that determines when to terminate idle interpreters	Idle Interpreter Timeout Threshold on page 3042
`ZEPPELIN_NOTEBOOK_DIR`	Location to store your Zeppelin notebooks	Notebook Storage on page 3040
`ZEPPELIN_SSL_PORT`	Port number for connecting to the Zeppelin UI	Connection Port on page 3037

Running Multiple Zeppelin Containers on a Single Host

To run multiple Apache Zeppelin containers on a single host, you must modify certain parameters in your `docker run` command. If you are using version 1.0 of the MapR Data Science Refinery, you also must modify the Livy interpreter's configuration and restart the Livy service.

Important: You cannot use host networking when running multiple Zeppelin containers on a single host. You must use the default Bridge Networking on page 3037.

1. Select different port numbers for your new container:
   a) Choose a different connection port number; for example, 9996
   b) Choose a different port range for the Livy launcher; for example, 10011-10021
   c) Choose a different port range for the Spark interpreter; for example, 13011-13021
2. Issue your `docker run` command with the port numbers you selected in Step 1 using one of the following two commands, depending on the version of the MapR Data Science Refinery you are using:

- Version 1.1 or later:

```
docker run -it ... \
   -e ZEPPELIN_SSL_PORT=9996 -p 9996:9996 \
   -p 10011-10021:10011-10021 -e LIVY_RSC_PORT_RANGE="10011~10021" \
   -p 13011-13021:13011-13021 -e SPARK_PORT_RANGE="13011~13021" \
   ... \
   maprtech/data-science-refinery:v1.4.1_6.1.0_6.3.0_centos7
```

  If you are not using the Spark interpreter, you can omit the following parameters:

```
-p 13011-13021:13011-13021 -e SPARK_PORT_RANGE="13011~13021"
```

  Use tilde (~) rather than dash (-) when specifying the range with the `LIVY_RSC_PORT_RANGE` and `SPARK_PORT_RANGE` environment variables.

- Version 1.0:

```
docker run -it ... \
   -e ZEPPELIN_SSL_PORT=9996 -p 9996:9996 \
   -p 10011-10021:10011-10021 \
   ... \
   maprtech/data-science-refinery:v1.4.1_6.1.0_6.3.0_centos7
```

3. If you are using Version 1.1 or later of the MapR Data Science Refinery, skip to Step 8. Otherwise, continue to Step 4.
4. Determine your `container-id` using the output from the following command:

```
docker ps
```

5. Log in to your container as the user running the container using the `container-id`:

```
docker exec -it --user <MAPR_CONTAINER_USER> <container-id> bash -l
```

6. Update the following property in `/opt/mapr/livy/livy-<version>/conf/livy-client.conf` to match the port range you chose in Step 1b:

```
livy.rsc.launcher.port.range = 10011~10021
```

   Note: Make sure to use tilde (~) rather than dash (-) when specifying the range.

7. Restart the Livy service:

```
/opt/mapr/livy/livy-<version>/bin/livy-server stop
/opt/mapr/livy/livy-<version>/bin/livy-server start
```

8. Connect to the new container using the port number you chose in Step 1a. The following URL assumes you are running the container on your local machine:

```
https://localhost:9996
```

Connecting MapR Data Science Refinery to MapR Sandbox
To connect MapR Data Science Refinery to MapR Sandbox, install MapR Sandbox, configure resource settings in your virtual machine (if needed), and then set the parameters corresponding to your virtual machine environment in your `docker run` command.

1. Install the sandbox using a virtual machine player
   a) You can use one of the following virtual machine players:
      - Installing the Sandbox on VirtualBox on page 106
      - Installing the Sandbox on VMware Player or VMware Fusion on page 104

   b) If you are installing VirtualBox, you must configure a second network adapter using either **Host-only Adapter** or **Bridged Adapter**. For the VMware players, you can use the default network configuration.

   c) You may also need to increase the default processor and memory configurations, depending on the workload you plan to run.

2. Determine the parameters you want to pass to Docker:
   a) See Understanding Zeppelin Docker Parameters on page 3036 to determine your initial parameters.

   b) Then determine the value of parameters that are specific to connecting to MapR Sandbox:

<table>
<tr><th>Parameter Name</th><th>Parameter Value</th></tr>
<tr><td>MAPR_CLDB_HOSTS</td><td>IP address of your virtual machine</td></tr>
<tr><td>MAPR_HS_HOST</td><td>IP address of your virtual machine</td></tr>
<tr><td>HOST_IP</td><td>Set this variable based on the network adapter and virtual machine player you are using:
<table>
<tr><th colspan="2">VirtualBox</th><th>VMware</th></tr>
<tr><td>Host-only Adapter</td><td>IP address of the gateway interface to the virtual machine's internal network</td><td rowspan="2">Output from the following command:
<pre>ip r get 8.8.8.8 | awk 'NR==1 {print $NF}'</pre></td></tr>
<tr><td>Bridged Adapter</td><td>Output from the following command:
<pre>ip r get 8.8.8.8 | awk 'NR==1 {print $NF}'</pre></td></tr>
</table></td></tr>
<tr><td>--add-host</td><td>"<hostname>":<IP address> of your virtual machine</td></tr>
</table>

3. Construct your `docker run` command based on Step 3.

   The following are examples of `docker run` commands that use VirtualBox with the different network adapters. Note the parameters highlighted in bold:

   Host-only Adapter

```
docker run -it -p 9995:9995 \
    -e HOST_IP=192.168.192.1  \
      -p 10000-10010:10000-10010 \
      -p 11000-11010:11000-11010 \
```


last-updated: Apr 11, 2022

```
    -e MAPR_CLUSTER=demo.mapr.com \
      -e
MAPR_CLDB_HOSTS=192.168.192.100 \
    -e MAPR_CONTAINER_USER=mapr \
      -e
MAPR_CONTAINER_PASSWORD=mapr \
      -e MAPR_CONTAINER_GROUP=mapr \
      -e MAPR_CONTAINER_UID=2000 \
      -e MAPR_CONTAINER_GID=2000 \
    -e MAPR_MOUNT_PATH=/mapr \
      --cap-add SYS_ADMIN \
      --cap-add SYS_RESOURCE \
      --device /dev/fuse \
    -e MAPR_HS_HOST=192.168.192.100
\
    --add-host="maprdemo:192.168.192
.100" \
    --add-host="maprdemo.local:192.1
68.192.100" \
    maprtech/
data-science-refinery:v1.4.1_6.1.0_6
.3.0_centos7
```

Bridged Adapter

```
docker run -it -p 9995:9995 \
    -e HOST_IP=$(ip r get 8.8.8.8 |
awk 'NR==1 {print $NF}')  \
      -p 10000-10010:10000-10010 \
      -p 11000-11010:11000-11010 \
    -e MAPR_CLUSTER=demo.mapr.com \
      -e
MAPR_CLDB_HOSTS=10.2.13.163 \
    -e MAPR_CONTAINER_USER=mapr \
      -e
MAPR_CONTAINER_PASSWORD=mapr \
      -e MAPR_CONTAINER_GROUP=mapr \
      -e MAPR_CONTAINER_UID=2000 \
      -e MAPR_CONTAINER_GID=2000 \
    -e MAPR_MOUNT_PATH=/mapr \
      --cap-add SYS_ADMIN \
      --cap-add SYS_RESOURCE \
      --device /dev/fuse \
    -e MAPR_HS_HOST=10.2.13.163 \
    --add-host="maprdemo:10.2.13.163
" \
    --add-host="maprdemo.local:10.2.
13.163" \
    maprtech/
data-science-refinery:v1.4.1_6.1.0_6
.3.0_centos7
```

After following these steps, you can refer to other MapR Data Science Refinery topics that describe how to use the resulting container.

If you encounter resource errors or hangs, you may need to increase the processor and memory configuration settings in your virtual machine player.

Important: Also, due to resource constraints, you cannot run both the Livy on page 3060 and Spark on page 3060 interpreters.

last-updated: Apr 11, 2022

Running MapR Data Science Refinery as a Kubernetes Service

This topic contains sample snippets from YAML files that enable you to run MapR Data Science Refinery as a Kubernetes service. The samples show you how to use various Kubernetes features. This includes specifying your MapR ticket as a Kubernetes secret, using a `ConfigMap` to define environment variables, mapping ports to route external traffic, passing the container password as a Kubernetes secret, exposing MapR Data Science Refinery as a service, and running MapR Data Science Refinery as a deployment for high availability.

The simplest way to run MapR Data Science Refinery as a Kubernetes service is to expose it to Kubernetes by including it in your Kubernetes pod manifest file. The following shows a sample file:

```
apiVersion: v1
kind: Pod
metadata:
  name: dsr-kube
  labels:
      app: dsr-svc
spec:
  containers:
  - name: dsr
    imagePullPolicy: Always
    image: maprtech/data-science-refinery:v1.4.1_6.1.0_6.3.0_centos7
    securityContext:
      capabilities:
        add: ["SYS_ADMIN" , "SYS_RESOURCE"]
    resources:
      requests:
        memory: "2Gi"
        cpu: "500m"
    env:
    - name: MAPR_MOUNT_PATH
      value: /mapr
    - name: MAPR_CLUSTER
      value: cluster1
    - name: MAPR_CLDB_HOSTS
      value: 10.10.102.95
    - name: MAPR_CONTAINER_USER
      value: mapr
    - name: MAPR_CONTAINER_GROUP
      value: mapr
    - name: MAPR_CONTAINER_PASSWORD
      value: mapr
    - name: HOST_IP
      valueFrom:
        fieldRef:
          fieldPath: status.hostIP
    - name: ZEPPELIN_DEPLOY_MODE
      value: kubernetes
    volumeMounts:
    - mountPath: /dev/fuse
      name: fuse
  volumes:
  - name: fuse
    hostPath:
      path: /dev/fuse
```

Make sure you include the following lines in your manifest file:

```
- name: ZEPPELIN_DEPLOY_MODE
      value: kubernetes
```

You can run MapR Data Science Refinery with Kubernetes 1.9 and later.

last-updated: Apr 11, 2022

Specifying your MapR Ticket as a Kubernetes Secret when Running MapR Data Science Refinery
To specify your MapR ticket as a Kubernetes Secret, you must create a secret and mount the secret in your MapR Data Science Refinery container. This topic describes the steps to do this.

1. Follow the instructions at Configuring a Secret on page 3167 to create a Kubernetes secret with your MapR ticket base64 encoded.

   The following is a sample YAML file named `dsr-ticket.yaml`:

```
apiVersion: v1
kind: Secret
metadata:
  name: dsr-ticket-secret
type: Opaque
data:
  CONTAINER_TICKET:
Y2x1c3RlcjMgMEpHTktQMWlBSU1td05wVWF4SXp5V0VLSjBvS080TndPSkU0SFVyazN6M3JVK
zVhcllDaWhqZ3Q2TDZNMmg4UnF4VGEzMU9IaDMvUmplYytTM0F3WVRlNTdjQ2kzL1dKYTJTbl
JEWkE4M2h2UEFqZkwyVUlzaVcydDJXdVZhT2xtSVIydmNwa3pMNVlqYUtIbm55RlpvbmJ4SUd
jQVRMTDJSaEdmL3JiY091Qk1wbzIxdlY3MnNYRHo5RXY5S1Fma0tKdElRTVNaK1JpTWwyZllC
MWtHa2hhdHJ3dE5qd2tqRlB6dUpQUEU2N2hDSnRLN1pRM0NuN01KM1JhRVliWkJlRjRFVVVxU
VVKSXV5T0VSV0RJbFJORCt3PT0K
```

2. In your Kubernetes pod manifest file, specify the following lines to mount the Kubernetes secret in your container.

   The example below mounts the ticket in `/tmp/mapr_ticket/CONTAINER_TICKET`:

```
env:
- name: MAPR_TICKETFILE_LOCATION
      value: "/tmp/mapr_ticket/CONTAINER_TICKET"

volumeMounts:
- mountPath: /tmp/mapr_ticket
      name: maprticket
      readOnly: true

volumes:
  - name: maprticket
    secret:
      secretName: dsr-ticket-secret
```

The following table summarizes the parameters that you must set in your YAML file and pod manifest. The values are highlighted in bold in the examples:

Parameter Description	Value in Example
Name of ticket file in your container	`CONTAINER_TICKET`
Mount path for the ticket file in your container	`/tmp/mapr_ticket`
Name of Kubernetes secret	`dsr-ticket-secret`
Name of volume for mounting the secret	`maprticket`

Related information
https://kubernetes.io/docs/concepts/configuration/secret/

last-updated: Apr 11, 2022

Using `ConfigMap` to Define MapR Data Science Refinery Variables in Kubernetes

You can use a `ConfigMap` to define your MapR Data Science Refinery environment variables. `ConfigMaps` enable you to decouple configuration artifacts from image content, providing more portable containerized applications. This topic describes the steps to define and deploy a `ConfigMap`.

1. Define your environment variables in a `ConfigMap` (`dsrconfigmap.yaml`):

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: dsr-configmap
data:
  MAPR_CLUSTER: "my.cluster.com"
  MAPR_CLDB_HOSTS: "1.1.1.1"
  MAPR_HS_HOST: "2.2.2.2"
  MAPR_CONTAINER_USER: "mapr"
  MAPR_CONTAINER_UID: "5000"
  MAPR_CONTAINER_GROUP: "mapr"
  MAPR_CONTAINER_GID: "5000"
  MAPR_MOUNT_PATH: /mapr
```

2. Create the `configMap` before deploying your MapR Data Science Refinery container by running the following command:

```
kubectl create -f dsrconfigmap.yaml
```

3. Modify your deployment YAML file and use `configMapKeyRef` to reference variables from the `ConfigMap` defined in step 1, as shown in the following sample:

```
env:
    - name: MAPR_MOUNT_PATH
      valueFrom:
        configMapKeyRef:
          name: dsr-configmap
          key: MAPR_MOUNT_PATH
    - name: MAPR_CLUSTER
      valueFrom:
        configMapKeyRef:
          name: dsr-configmap
          key: MAPR_CLUSTER
    - name: MAPR_CLDB_HOSTS
      valueFrom:
        configMapKeyRef:
          name: dsr-configmap
          key: MAPR_CLDB_HOSTS
    - name: MAPR_CONTAINER_USER
      valueFrom:
        configMapKeyRef:
          name: dsr-configmap
          key: MAPR_CONTAINER_USER
    - name: MAPR_CONTAINER_UID
      valueFrom:
        configMapKeyRef:
          name: dsr-configmap
          key: MAPR_CONTAINER_UID
    - name: MAPR_CONTAINER_GID
      valueFrom:
        configMapKeyRef:
          name: dsr-configmap
          key: MAPR_CONTAINER_GID
    - name: MAPR_CONTAINER_GROUP
      valueFrom:
        configMapKeyRef:
          name: dsr-configmap
          key: MAPR_CONTAINER_GROUP
    - name: MAPR_CONTAINER_PASSWORD
      valueFrom:
        secretKeyRef:
          name: dsr-container-secret
          key: password
    - name: HOST_IP
      valueFrom:
        fieldRef:
          fieldPath: status.hostIP
    - name: MAPR_HS_HOST
      valueFrom:
        configMapKeyRef:
          name: dsr-configmap
          key: MAPR_HS_HOST
```

Related information

https://kubernetes.io/docs/tasks/configure-pod-container/configure-pod-configmap/

Mapping MapR Data Science Refinery Ports for External Access

To access MapR Data Science Refinery through the internet, when it is running as a Kubernetes service, you must map the ports using `nodePort`. This topic provides a sample that shows you how to do this.

last-updated: Apr 11, 2022

Assume you have the following `configMap` settings for ports:

```
apiVersion: v1
  kind: ConfigMap
  metadata:
    name: dsr-confs
    namespace: dsr-ns
  data:
    ZEPPELIN_SSL_PORT: "9995"
    LIVY_RSC_PORT_RANGE: "30000~30009"
    SPARK_PORT_RANGE: "30010~30019"
```

The following defines mappings for these ports:

```
apiVersion: v1
kind: Service
metadata:
  name: dsr-svc
  namespace: dsr-ns
spec:
  type: LoadBalancer
  ports:
    - { name: dsr-ui, port: 9995, nodePort: 32000 }

  selector:
    app: dsr-app

---
apiVersion: v1
kind: Service
metadata:
  name: dsr-svc-ports
  namespace: dsr-ns
spec:
  type: NodePort
  selector:
    app: dsr-app
  ports:

    - { name: dsr-livy-0, port: 30000, nodePort: 30000 }
    - { name: dsr-livy-1, port: 30001, nodePort: 30001 }
    - { name: dsr-livy-2, port: 30002, nodePort: 30002 }
    - { name: dsr-livy-3, port: 30003, nodePort: 30003 }
    - { name: dsr-livy-4, port: 30004, nodePort: 30004 }
    - { name: dsr-livy-5, port: 30005, nodePort: 30005 }
    - { name: dsr-livy-6, port: 30006, nodePort: 30006 }
    - { name: dsr-livy-7, port: 30007, nodePort: 30007 }
    - { name: dsr-livy-8, port: 30008, nodePort: 30008 }
    - { name: dsr-livy-9, port: 30009, nodePort: 30009 }

    - { name: dsr-spark-0, port: 30010, nodePort: 30010 }
    - { name: dsr-spark-1, port: 30011, nodePort: 30011 }
    - { name: dsr-spark-2, port: 30012, nodePort: 30012 }
    - { name: dsr-spark-3, port: 30013, nodePort: 30013 }
    - { name: dsr-spark-4, port: 30014, nodePort: 30014 }
    - { name: dsr-spark-5, port: 30015, nodePort: 30015 }
    - { name: dsr-spark-6, port: 30016, nodePort: 30016 }
    - { name: dsr-spark-7, port: 30017, nodePort: 30017 }
    - { name: dsr-spark-8, port: 30018, nodePort: 30018 }
    - { name: dsr-spark-9, port: 30019, nodePort: 30019 }
```


last-updated: Apr 11, 2022

The example exposes the Livy and Spark ports through `NodePort`, using the same port numbers inside and outside the container.

It exposes the MapR Data Science Refinery UI port through a `LoadBalancer` service. To access the UI, use the following URL with port 32000, as defined in the mapping:

```
https://<k8s-worker-ip>:32000
```

`<k8s-worker-ip>` is the public IP address of any Kubernetes worker node. The worker node proxies the port into the MapR Data Science Refinery service.

Note: You must map port 9995 to a number in the range of 30000-32767 because `NodePort` supports only that port range.

Related tasks

Using ConfigMap to Define MapR Data Science Refinery Variables in Kubernetes on page 3051
You can use a `ConfigMap` to define your MapR Data Science Refinery environment variables. `ConfigMaps` enable you to decouple configuration artifacts from image content, providing more portable containerized applications. This topic describes the steps to define and deploy a `ConfigMap`.

Related information

https://kubernetes.io/docs/concepts/services-networking/service/#nodeport
https://kubernetes.io/docs/concepts/services-networking/service/#loadbalancer

Passing MapR Data Science Refinery Password as a Secret in Kubernetes
To avoid passing your `MAPR_CONTAINER_PASSWORD` in clear text, you can use a Kubernetes secret. This topic shows you how to do this.

1. Create a secret (`dsr-container-secret.yaml`) where you have base64 encoded `MAPR_CONTAINER_PASSWORD`:

   ```
   apiVersion: v1
   kind: Secret
   metadata:
     name: dsr-container-secret
   type: Opaque
   data:
     password: azhzUm9ja3M=
   ```

   `azhzUm9ja3M=` is the base64 encoding of `k8sRocks`.

2. Create the secret before deploying your MapR Data Science Refinery container by running the following command:

   ```
   kubectl create -f dsr-container-secret.yaml
   ```

3. Reference the secret in your deployment YAML file as follows:

   ```
   - name: MAPR_CONTAINER_PASSWORD
         valueFrom:
           secretKeyRef:
             name: dsr-container-secret
             key: password
   ```

Related information

https://kubernetes.io/docs/concepts/configuration/secret/

Exposing MapR Data Science Refinery as a Service in Kubernetes
You can expose MapR Data Science Refinery as a Kubernetes service. For example, if you run the service in a cloud Kubernetes deployment, you can expose it as a Kubernetes `LoadBalancer` service. This topic provides a sample that shows you how to do this. It also discusses the alternative where you are running in a non-cloud Kubernetes deployment.

The following sample YAML file exposes a `LoadBalancer` service:

```
apiVersion: v1
kind: Service
metadata:
  name: dsr-web-svc
  namespace: dsr-ns
spec:
  type: LoadBalancer
  ports:
    - { name: dsr-ui, port: 9995, nodePort: 32000 }

  selector:
    app: dsr-app
```

If you are running in a cloud Kubernetes deployment, run the following command to retrieve the external IP of the `LoadBalancer` service:

```
kubectl get service -n dsr-ns
```

Use the value in the `EXTERNAL-IP` column as the IP to connect to the MapR Data Science Refinery UI:

```
https://<EXTERNAL-IP>:9995
```

If you are running in a non-cloud Kubernetes deployment, the `EXTERNAL-IP` column returns `<pending>`, due to unavailability of a `LoadBalancer` service. You can still connect to the MapR Data Science Refinery UI by using the IP of any Kubernetes worker node and the `nodePort` specified in your YAML file:

```
https://<k8s-worker-ip>:32000
```

Related information
https://kubernetes.io/docs/concepts/services-networking/service/#loadbalancer

Running MapR Data Science Refinery as a Deployment in Kubernetes
This topic provides a sample YAML file that shows you how to run MapR Data Science Refinery as a Kubernetes `Deployment`. Doing so enables the Kubernetes deployment controller to spawn a new pod when a pod is killed or dies.

To run MapR Data Science Refinery as a Kubernetes `Deployment`, include the following configuration settings in your YAML file:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: dsr-app
spec:
  replicas: 1
  template:
    metadata:
          labels:
            app: dsr-app
    spec:
      containers:
      - name: dsr
```


last-updated: Apr 11, 2022

```
        imagePullPolicy: Always
        image:  maprtech/data-science-refinery:v1.4.1_6.1.0_6.3.0_centos7
```

The remainder of your YAML file is similar to a YAML file that specifies a Pod. The YAML file at Running MapR Data Science Refinery as a Kubernetes Service on page 3049 is an example.

Related information

https://kubernetes.io/docs/concepts/workloads/controllers/deployment/

Using MapR Data Fabric for Kubernetes with MapR Data Science Refinery

You can integrate MapR Data Fabric for Kubernetes with your MapR Data Science Refinery deployment. This provides persistent storage to MapR Filesystem, which you can use as an alternative to the MapR FUSE-Based POSIX Client. You have two integration options: use the Kubernetes Flexvolume driver, or use Kubernetes Persistent Volumes (PV) and Persistent Volume Claims (PVC).

Regardless of your integration choice, you must follow the instructions at Installing MapR and Kubernetes Software on Separate Nodes on page 241 to install MapR Data Fabric for Kubernetes.

Note: If you decide to use MapR Data Fabric for Kubernetes, do not set the `MAPR_MOUNT_PATH` variable. Otherwise, it enables the MapR FUSE-Based POSIX Client, which you do not need.

Using the Flexvolume Driver

The following sample shows you how to configure the Flexvolume driver:

```
            volumeMounts:
              - mountPath: /dev/fuse
                name: fuse
              - mountPath: /sys/fs/cgroup
                name: cgroup
              - mountPath: /tmp/maprticket_secret
                name: maprticket-secret
              - mountPath: /mapr-vol
                name: maprflex-volume

          volumes:
            - name: fuse
              hostPath:
                path: /dev/fuse
            - name: cgroup
              hostPath:
                path: /sys/fs/cgroup
            - name: maprticket-secret
              secret:
                secretName: mapr-ticket-secret
            - name: maprflex-volume
              flexVolume:
                driver: "mapr.com/maprfs"
                readOnly: true
                options:
                  cluster: "my.cluster.com"
                  cldbHosts: "cldb1 cldb2"
                  securityType: "secure"
                  ticketSecretName: "mapr-ticket-secret"
                  ticketSecretNamespace: "ns-mydsr"
```


last-updated: Apr 11, 2022

Using Persistent Volumes (PV) and Persistent Volume Claims (PVC)

The following defines the namespace used in subsequent configuration snippets:

```
apiVersion: v1
kind: Namespace
metadata:
  name: ns-mydsr
  labels:
    name: ns-mydsr
```

The following defines your PV and PVC:

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: dsr-pv
  namespace: ns-mydsr
spec:
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteOnce
  flexVolume:
    driver: "mapr.com/maprfs"
    readOnly: true
    options:
      cluster: "my.cluster.com"
      cldbHosts: "cldb1 cldb2"
      securityType: "secure"
      ticketSecretName: "mapr-ticket-secret"
      ticketSecretNamespace: "ns-mydsr"

---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: dsr-pvc
  namespace: ns-mydsr
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5G
```

The following references the previously defined PV and PVC:

```
            volumeMounts:
              - mountPath: /dev/fuse
                name: fuse
              - mountPath: /sys/fs/cgroup
                name: cgroup
              - mountPath: /tmp/maprticket_secret
                name: maprticket-secret
              - mountPath: /mapr-pvc
                name: maprflex-pv-pvc
        volumes:
          - name: fuse
            hostPath:
              path: /dev/fuse
          - name: cgroup
            hostPath:
```


last-updated: Apr 11, 2022

```
            path: /sys/fs/cgroup
        - name: maprticket-secret
          secret:
            secretName: mapr-ticket-secret
        - name: maprflex-pv-pvc
          persistentVolumeClaim:
            claimName: dsr-pvc
```

Related concepts

MapR Data Fabric for Kubernetes FlexVolume Driver Overview on page 671
Describes how the FlexVolume driver for MapR Data Fabric for Kubernetes integrates with Kubernetes to provide persistent data for containers.

Related information

https://kubernetes.io/docs/concepts/storage/volumes/#flexvolume

https://kubernetes.io/docs/concepts/storage/persistent-volumes/

Accessing and Creating Notebooks in Zeppelin

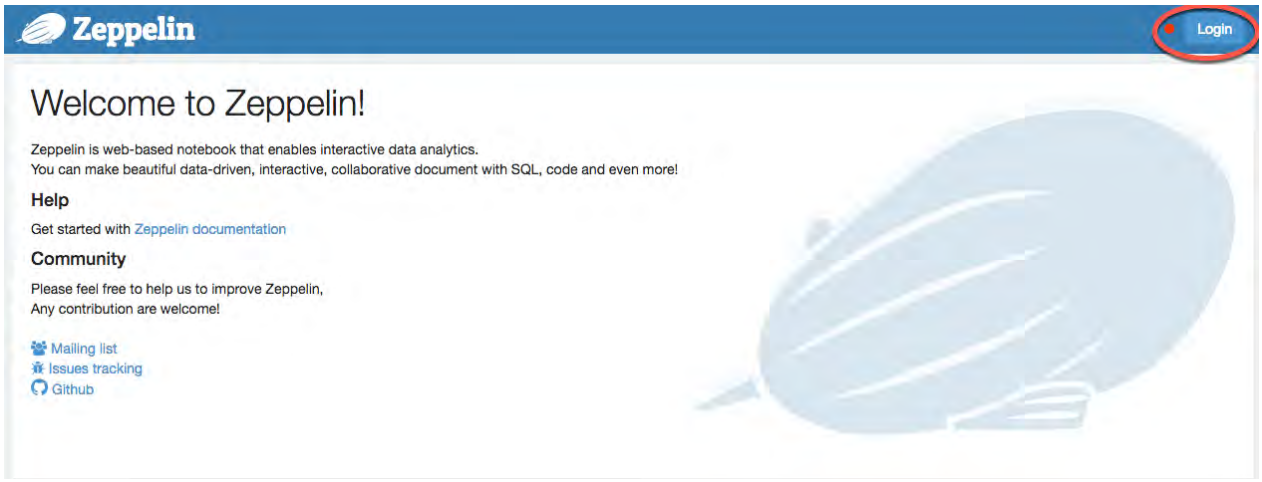
1. After the Apache Zeppelin Docker image is running, access the Zeppelin notebook in your browser by specifying the following URL:

   ```
   https://localhost:9995
   ```

   This URL loads the Zeppelin notebook's home page. You must specify a secure URL.

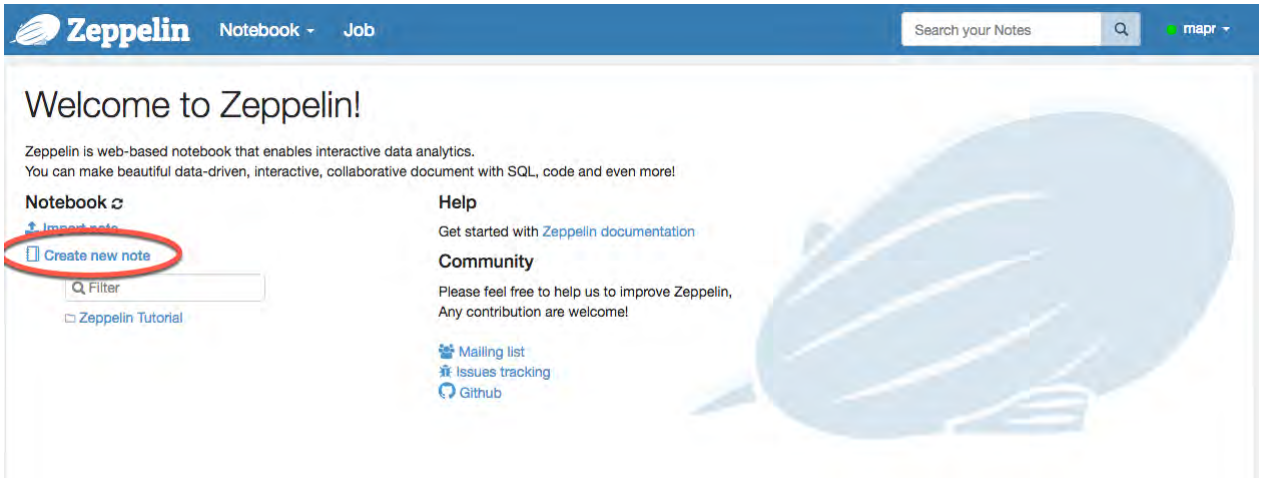
   If the Docker image is running on a remote node, such as a MapR edge node, replace `localhost` with the host name or IP address of the remote node. If you specified a different port number in your `docker run` command, replace 9995 with your port number.

2. Log in to Zeppelin using the user name and password you specified in your `docker run` command:



3. Create your notebook:

last-updated: Apr 11, 2022



See Understanding Zeppelin Interpreters on page 3059 for details on interpreters you can use.

Stopping Your Zeppelin Container

If you no longer need to use your Zeppelin container, you can stop the running image.

1. Determine the container id of your Docker instance by listing all running Docker containers:

   ```
   docker ps
   ```

   The output includes the following fields:

   ```
   CONTAINER ID
   IMAGE

   23b60c2720dd        maprtech/
   data-science-refinery:v1.4.1_6.1.0_6.3.0_centos7
   ```

2. Stop the running container specifying the revealed container id:

   ```
   docker stop 23b60c2720dd
   ```

Understanding Zeppelin Interpreters

Apache Zeppelin interpreters enable you to access specific languages and data processing backends. This section describes the interpreters you can use with MapR and the use cases they serve.

Supported Zeppelin Interpreters

Apache Zeppelin on MapR supports the following interpreters:

Shell	With the Shell interpreter, you can invoke system shell commands. If you have a MapR File System mount point, you can access MapR File System using shell commands like `ls` and `cat` by using the FUSE-Based POSIX Client on page 3039. See Running Shell Commands in Zeppelin on page 3085 for examples that use this interpreter.
Pig	The Apache Pig interpreter enables you to run Apache Pig scripts and queries. See Running Pig Scripts in Zeppelin on page 3086 for examples that use this interpreter.

JDBC - Drill and Hive

Apache Zeppelin on MapR provides preconfigured Apache Drill and Apache Hive JDBC interpreters. See Running Drill Queries in Zeppelin on page 3087 and Running Hive Queries in Zeppelin on page 3087 for examples that use these interpreters.

Livy

The Apache Livy interpreter is a RESTful interface for interacting with Apache Spark. With this interpreter, you can run interactive Scala, Python, and R shells, and submit Spark jobs.

The Spark jobs run in YARN cluster mode so they run inside an application master process managed by YARN. This has the following implications:

- Allows you to close your Zeppelin notebook without killing your Spark jobs.
- Supports Spark Dynamic Resource Allocation, which allows you to set idle timeouts in your Spark context to recapture wayward memory.

Starting in the 1.3 release, MapR Data Science Refinery uses a shared Livy session to run all Spark variations. In prior releases, it uses separate Livy sessions for Spark, PySpark, and SparkR jobs.

The Livy interpreter does not support ZeppelinContext and Angular Display System. See the description of the Spark interpreter for details about these features.

Although MapR Data Science Refinery includes Livy, you cannot run the Livy UI inside your Zeppelin container.

The following topics contain examples that use the Livy interpreter to access different backend engines:

- Running Spark Jobs in Zeppelin on page 3089
- Accessing MapR Database in Zeppelin Using the MapR Database Binary Connector on page 3093
- Accessing MapR Event Store For Apache Kafka in Zeppelin Using the Livy Interpreter on page 3097

Note: See MapR Data Science Refinery Support by MapR Core Version on page 5638 for limitations in version support when accessing MapR Event Store.

Spark

The Apache Spark interpreter is available starting in MapR Data Science Refinery 1.1. It provides an alternative to the Livy interpreter.

The Spark interpreter supports the following features not supported by the Livy interpreter:

- ZeppelinContext - Allows you to create dynamic forms and share objects between Spark Scala and PySpark code
- Angular Display System - Allows you to display charts using data returned from Spark and to pass variables from the Spark interpreter to the Angular interpreter

last-updated: Apr 11, 2022

Starting in MapR Data Science Refinery 1.3, Spark jobs run in YARN cluster mode. In prior releases, they run in YARN client mode. Running in YARN cluster mode avoids heavy resource consumption on your container host machine because the Spark driver process does not run on the container host. It also provides the advantages noted earlier for the Livy interpreter.

The following topics contain examples that use the Spark interpreter to access different backend engines:

- Running Spark Jobs in Zeppelin on page 3089
- Accessing MapR Database in Zeppelin Using the MapR Database Binary Connector on page 3093
- Accessing MapR Database in Zeppelin Using the MapR Database OJAI Connector on page 3095
- Accessing MapR Event Store For Apache Kafka in Zeppelin Using the Spark Interpreter on page 3100

  Note: See MapR Data Science Refinery Support by MapR Core Version on page 5638 for limitations in version support when accessing MapR Event Store.

MapR Database Shell

The MapR Database Shell interpreter allows you to run commands available in MapR Database Shell (JSON Tables) on page 5286 in the Zeppelin UI. Using dbshell commands, you can access MapR Database JSON tables without having to write Spark code. The interpreter supports all dbshell commands except `find` commands that specify an ordering.

The interpreter is available starting in MapR Data Science Refinery 1.2. You do not have to run any new additional configuration steps to use this interpreter.

Specify the following in the Zeppelin UI to invoke the interpreter:

```
%maprdb
```

See Running MapR Database Shell Commands in Zeppelin on page 3092 for examples that use this interpreter.

Livy vs Spark Interpreters

Starting in MapR Data Science Refinery 1.3, since both the Livy and Spark interpreters run in YARN cluster mode, the primary reason for choosing the Spark interpreter over the Livy interpreter is support for visualization features in the former.

Note: Neither interpreter supports Spark standalone mode.

Zeppelin Interpreter Use Cases

The table below summarizes which interpreters to use to access different backend engines for different data processing goals:

last-updated: Apr 11, 2022

Data Processing Goal	Zeppelin Interpreter	Backend Engine
Data discovery, exploratory querying	Livy, Spark	Spark SQL
	JDBC	Hive, Drill
	Shell	MapR File System
	MapR Database Shell	MapR Database JSON
ETL, preparation	Livy, Spark	Spark, PySpark, SparkSQL, SparkStreaming,
	Livy, Spark	MapR Database (through the MapR Database Connectors for Apache Spark on page 4050)
	Livy, Spark	MapR Event Store For Apache Kafka (through Spark jobs that query MapR Event Store For Apache Kafka) **Note:** See MapR Data Science Refinery Support by MapR Core Version on page 5638 for limitations in version support when accessing MapR Event Store.
	JDBC	Hive
	Pig	MapReduce
Machine and deep learning, data science	Livy, Spark	SparkML
Reporting, visualization	JDBC	Hive, Drill

The following are general guidelines for choosing between the Livy and Spark interpreters:

- Use Livy for jobs that are long running or resource intensive
- Use Spark if you use visualization features that Livy does not support

Unsupported Zeppelin Interpreters

Apache Zeppelin on MapR does not support the HBase interpreter. To access MapR Database binary tables, use the MapR Database Binary Connector for Apache Spark on page 4101 with either the Livy or Spark interpreter.

Sequential Execution of Notebook Paragraphs

Starting in the 1.3 release, MapR Data Science Refinery runs paragraphs in a notebook sequentially rather than in parallel. This allows paragraphs to run properly when they have dependencies on earlier paragraphs in the same notebook.

Related information

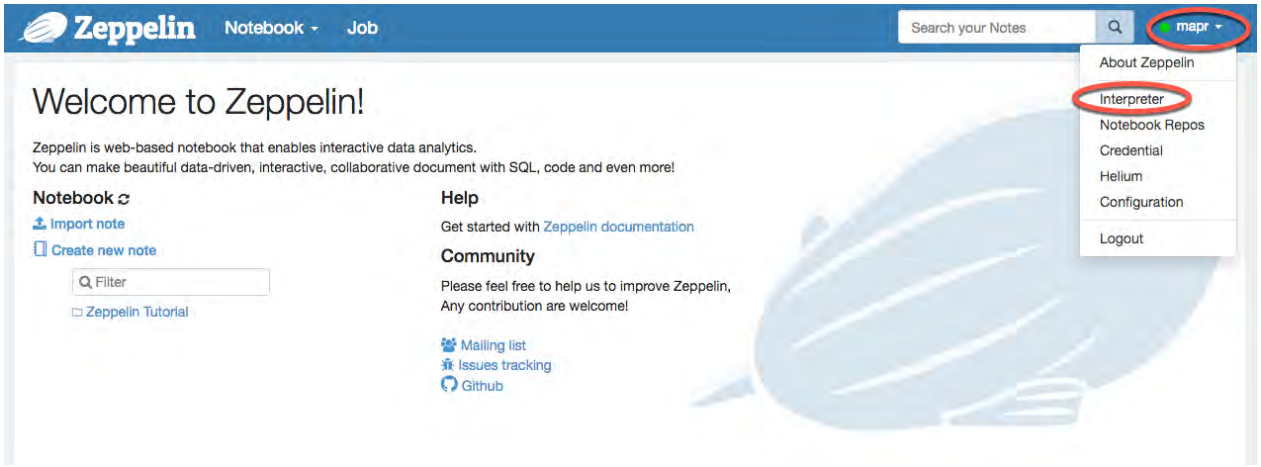
https://zeppelin.apache.org/docs/0.7.3/manual/interpreters.html

Configuring Zeppelin Interpreters

Out-of-box, the interpreters in Apache Zeppelin on MapR are preconfigured to run against different backend engines. You may need to perform manual steps to configure the Livy, Spark, and JDBC interpreters. No additional steps are needed to configure and run the Pig and Shell interpreters. You can configure the idle timeout threshold for interpreters.

last-updated: Apr 11, 2022

To configure an interpreter, click on your login name in the top right corner of the Zeppelin home screen and select **Interpreter**:



Configuring the Livy Interpreter

The Livy interpreter provides support for Spark Python, SparkR, Basic Spark, and Spark SQL jobs. To use the Livy interpreter for these variations of Spark, you must take certain actions, including configuring Zeppelin and installing software on your MapR cluster.

You must also issue your `docker run` command with the parameters the Livy interpreter requires. See the following material for details about these parameters:

- Bridge Networking on page 3037
- Pig, Livy, and Spark Interpreters on page 3040
- Python Version on page 3041

Spark Python

The Apache Livy interpreter supports both Python 2 and Python 3, but you can run only one or the other in your container. You must install the Python packages on your MapR cluster to use them through Livy. You do not need to install them in your container.

To use Python, specify the following in your notebook:

```
%livy.pyspark
```

By default, this invokes Python 2. To switch Python versions, see Python Version on page 3041.

To install custom Python packages, see Installing Custom Packages for PySpark on page 3066.

SparkR

You must install R on your MapR cluster to use Apache SparkR through Livy.

To install custom packages for SparkR, run the R interpreter and execute R commands to install the target package. You must install the packages on each node where SparkR will execute. This are the nodes that contain a YARN NodeManager. By default, the Livy interpreter submits Spark jobs in YARN cluster mode.

The following example installs the `data.table` and `googleVis` packages using the R interpreter:

```
sudo R
> install.packages("data.table")
> install.packages("googleVis")
```

To verify these installs, run the following code in your Zeppelin UI:

```
%livy.sparkr

print(packageVersion("data.table"))
print(packageVersion("googleVis"))
```

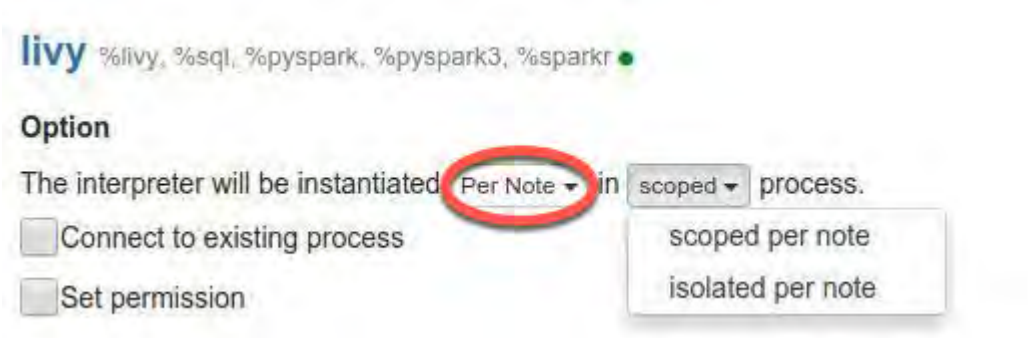
Your output should look similar to the following, depending on your package versions:

```
[1] '1.10.4.3'
[1] '0.6.2'
```

Spark Jobs

By default, the Livy interpreter is configured to submit Apache Spark jobs in YARN cluster mode.

To run Spark jobs in parallel, you must modify the Livy interpreter to instantiate **Per Note**:



You can set **scoped** to either of the two options.

Hive Tables

To use Apache Spark SQL with Apache Hive, follow the steps described at Integrate Spark-SQL (Spark 2.0.1 and later) with Hive on page 4115. As described on that page, to access Hive tables through Spark, you must make the `hive-site.xml` configuration file from your Hive cluster available to Spark running in your Zeppelin container.

One way to make the file available is through a volume mount when you start Docker:

1. Copy the file `/opt/mapr/hive/hive-<version>/conf/hive-site.xml` from your Hive cluster to the local host on which you are running the Docker container.
2. Add a volume mount argument to your `docker run` command.

   In the following example, the local `hive-site.xml` file is in `/tmp`:

   ```
   docker run -it -p 9995:9995 -e MAPR_CLUSTER=<cluster-name> … \
   -v /tmp/hive-site.xml:/opt/mapr/spark/spark-2.3.1/conf/hive-site.xml:ro \
   maprtech/data-science-refinery:v1.4.1_6.1.0_6.3.0_centos7
   ```

Another way to make the file available is to copy it into your running container:

1. Copy the file `/opt/mapr/hive/hive-<version>/conf/hive-site.xml` from your Hive cluster to the local host on which you are running the Docker container.

2. Determine your `container-id` using the output from the following command:

```
docker ps
```

3. Copy `hive-site.xml` from your local host into your Docker container:

```
docker cp /tmp/hive-site.xml <container-id>:/opt/mapr/spark/spark-2.3.1/
conf
```

4. Log in to your container as the user running the container using the `container-id`:

```
docker exec -it --user <MAPR_CONTAINER_USER> <container-id> bash -l
```

5. Restart the Livy service running in your container:

```
/opt/mapr/livy/livy-<version>/bin/livy-server stop
/opt/mapr/livy/livy-<version>/bin/livy-server start
```

Configuring the Spark Interpreter

The Spark interpreter is available starting in the 1.1 release of the MapR Data Science Refinery. It provides support for Spark Python, SparkR, Basic Spark, and Spark SQL jobs. To use the Spark interpreter for these variations of Spark, you must take certain actions, including configuring Zeppelin and installing software on your MapR cluster.

You must also issue your `docker run` command with the parameters the Spark interpreter requires. See the following material for details about these parameters:

- Bridge Networking on page 3037
- Pig, Livy, and Spark Interpreters on page 3040
- Python Version on page 3041

Spark Python

You must install Python in your MapR cluster to run Python code with the Spark interpreter. You do not need to install it in your container since Spark runs in YARN cluster mode.

To use Python in the Spark interpreter, specify the following in your notebook:

```
%spark.pyspark
```

By default, this invokes Python 2. To switch Python versions, see Python Version on page 3041.

To install custom Python packages, see Installing Custom Packages for PySpark on page 3066. This also describes how to use Python 3 with custom packages.

Note: Although the 1.3 release includes IPython with the Spark interpreter, MapR Data Science Refinery does not support this feature.

SparkR

The Zeppelin container includes R. Some Apache SparkR jobs require you to install R on your MapR cluster nodes to run these jobs in the Spark interpreter.

last-updated: Apr 11, 2022

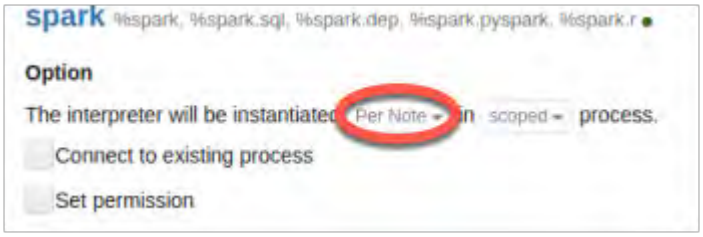
To use R in the Spark interpreter, specify the following in your notebook:

```
%spark.r
```

Spark Jobs

By default, the Spark interpreter is configured to submit Apache Spark jobs in YARN client mode. The interpreter does not support YARN cluster mode. Make sure you follow the steps described at Installing Spark on YARN on page 218 to install Spark on your MapR cluster.

To run Spark jobs in parallel, you must modify the Spark interpreter to instantiate **Per Note**:



You can set **scoped** to either of the two options.

Hive Tables

To access Apache Hive tables using the Spark interpreter, you must make the `hive-site.xml` configuration file from your Hive cluster available to Spark running in your Zeppelin container. Follow the same steps that describe how to access Hive tables with the Livy interpreter.

Installing Custom Packages for PySpark
You can install custom Python packages either by manually installing packages on each node in your MapR cluster or by using Conda. Using Conda allows you to perform the install from your Zeppelin host node without having to directly access your MapR cluster. The topics in this section describe the instructions for each method as well as instructions for Python 2 vs Python 3.

You can run only version of Python in your Zeppelin notebook.

Important: MapR supports the Python libraries included in the Zeppelin container, but does not support the libraries in custom Python packages. You should use Python versions that match the versions installed on your MapR cluster nodes. Choosing a Zeppelin Docker image OS that matches the OS running in your MapR cluster minimizes library version differences.

Manually Installing Custom Packages for PySpark
Use the Python package manager `pip` (or `pip3` for PySpark3) to manually install custom packages on each node in your MapR cluster. You need administrative access on your cluster nodes to install the packages.

1. Install the package manager using one of the following commands, depending on your operating system:
   a) RedHat:

Python 2

```
sudo yum install -y python-devel
python-setuptools
sudo easy_install pip
```

Python 3

```
sudo yum install -y python34-devel
python34-setuptools
sudo easy_install-3.4 pip
```

b) SLES:

Python 2

```
sudo zypper install -y
python-devel python-setuptools
sudo easy_install pip
```

Python 3

```
sudo zypper install -y
python3-devel python3-setuptools
sudo easy_install-3.4 pip
```

c) Ubuntu:

Python 2

```
sudo apt-get install -y python-dev
python-setuptools
sudo easy_install pip
```

Python 3

```
sudo apt-get install -y
python3-dev python3-setuptools
sudo easy_install3 pip
```

2. Install the custom package using the utility you downloaded in Step 1.

The following example installs the `matplotlib` package:

Python 2

```
sudo pip install matplotlib
```

Python 3

```
sudo pip3 install matplotlib
```

You must install the package on each node in your MapR cluster where PySpark jobs will run. These are the nodes that contain a YARN NodeManager.

3. To verify successful installs, run the following code snippet in your Zeppelin UI:

```
%livy.pyspark

import sys
print(sys.version)

import matplotlib
print(matplotlib.__version__)
```

The code snippet returns output similar to the following:

Python 2

```
2.7.5 (default, Nov 6 2016,
00:28:07)
[GCC 4.8.5 20150623 (Red Hat
4.8.5-11)]
2.1.0
```

Python 3

```
3.4.5 (default, May 29 2017,
15:17:55)
```

```
[GCC 4.8.5 20150623 (Red Hat
4.8.5-11)]
2.1.0
```

The minor versions of Python and `matplotlib` may differ depending on the versions you install.

Related information

https://mapr.com/blog/how-to-using-tensorflow-with-the-mapr-data-science-refinery/

Installing Custom Packages for PySpark Using Conda

To install custom packages for Python 2 (or Python 3) using Conda, you must create a custom Conda environment and pass the path of the custom environment in your `docker run` command.

To install Conda, follow the instructions at https://conda.io/docs/user-guide/install/index.html.

For each step of the following steps, select the tab corresponding to the Python version you want to install.

1. Create your custom Conda environment and archive it as a zip archive.

Python 2

The following example creates a custom Conda environment with Python 2 and three packages (`matplotlib`, `numpy`, and `pandas`):

```
mkdir custom_pyspark_env
conda create -p ./
custom_pyspark_env python=2 numpy
pandas matplotlib
cd custom_pyspark_env
zip -r custom_pyspark_env.zip ./
```

Python 3

The following example creates a custom Conda environment with Python 3 and three packages (`matplotlib`, `numpy`, and `pandas`):

```
mkdir custom_pyspark3_env
conda create -p ./
custom_pyspark3_env python=3 numpy
pandas matplotlib
cd custom_pyspark3_env
zip -r custom_pyspark3_env.zip ./
```

Important: Do not create an archive named `pyspark.zip`. This name is reserved for PySpark internals.

2. Launch the Zeppelin container, specifying the path of the Python archive in your `docker run` command.

Python 2

You can specify the archive in one of the following ways:

- **Option 1**: Specify the archive from MapR File System by uploading the archive to MapR File System
- **Option 2**: Specify the archive from your local filesystem using a Docker mount point

last-updated: Apr 11, 2022

Option 1

```
hadoop fs -put
custom_pyspark_e
nv.zip /
python_envs/
custom_pyspark_e
nv.zip
docker
run -it ... \
   -e
ZEPPELIN_ARCHIVE
_PYTHON=/
python_envs/
custom_pyspark_e
nv.zip \
   ... \
   maprtech/
data-science-ref
inery:v1.4.1_6.1
.0_6.3.0_centos7
```

Option 2

```
docker
run -it ... \
   -v /local/
path/
custom_pyspark_e
nv.zip:/tmp/
custom_pyspark_e
nv.zip:ro \
   -e
ZEPPELIN_ARCHIVE
_PYTHON=/tmp/
custom_pyspark_e
nv.zip \
   ... \
   maprtech/
data-science-ref
inery:v1.4.1_6.1
.0_6.3.0_centos7
```

The path parameters in the sample command correspond to the following:

Full Path to Archive from Step 1	Mount Point of the Archive in your Container
`/local/path/custom_pyspark_env.zip`	`/tmp/custom_pyspark_env.zip`

Python 3

If you want to use Python 3 instead of Python 2, set >>>>>>> Brought back DSR 1.3 content `ZEPPELIN_ARCHIVE_PYTHON` in one of the following ways:

last-updated: Apr 11, 2022

- **Option 1**: Specify the archive from MapR File System by uploading the archive to MapR File System
- **Option 2**: Specify the archive from your local file system using a Docker mount point

Option 1

```
hadoop fs -put
custom_pyspark3_
env.zip /
python_envs/
custom_pyspark3_
env.zip
docker
run -it ... \
   -e
ZEPPELIN_ARCHIVE
_PYTHON=/
python_envs/
custom_pyspark3_
env.zip \
   ... \
   maprtech/
data-science-ref
inery:v1.4.1_6.1
.0_6.3.0_centos7
```

Option 2

```
docker
run -it ... \
   -v /local/
path/
custom_pyspark3_
env.zip:/tmp/
custom_pyspark3_
env.zip:ro \
   -e
ZEPPELIN_ARCHIVE
_PYTHON=/tmp/
custom_pyspark3_
env.zip \
   ... \
   maprtech/
data-science-ref
inery:v1.4.1_6.1
.0_6.3.0_centos7
```

The path parameters in the sample command correspond to the following:

Full Path to Archive from Step 1	Mount Point of the Archive in your Container
`/local/path/custom_pyspark3_env.zip`	`/tmp/custom_pyspark3_env.zip`

3. To verify that you have successfully installed the `matplotlib` package, run the following code snippet in your Zeppelin UI:

Livy PySpark

```
%livy.pyspark

import sys
print(sys.version)

import matplotlib
print(matplotlib.__version__)
```

Spark PySpark

```
%spark.pyspark
import sys
print(sys.version)

import matplotlib
print(matplotlib.__version__)
```

The code snippet returns output similar to the following:

Python 2

```
2.7.14 |Anaconda, Inc.| (default,
Oct 27 2017, 18:21:12)
[GCC 7.2.0]
2.1.0
```

Python 3

```
3.6.3 |Anaconda, Inc.| (default,
Oct 27 2017, 19:41:01)
[GCC 7.2.0]
2.1.0
```

The minor versions of Python and `matplotlib` may differ depending on the versions you install.

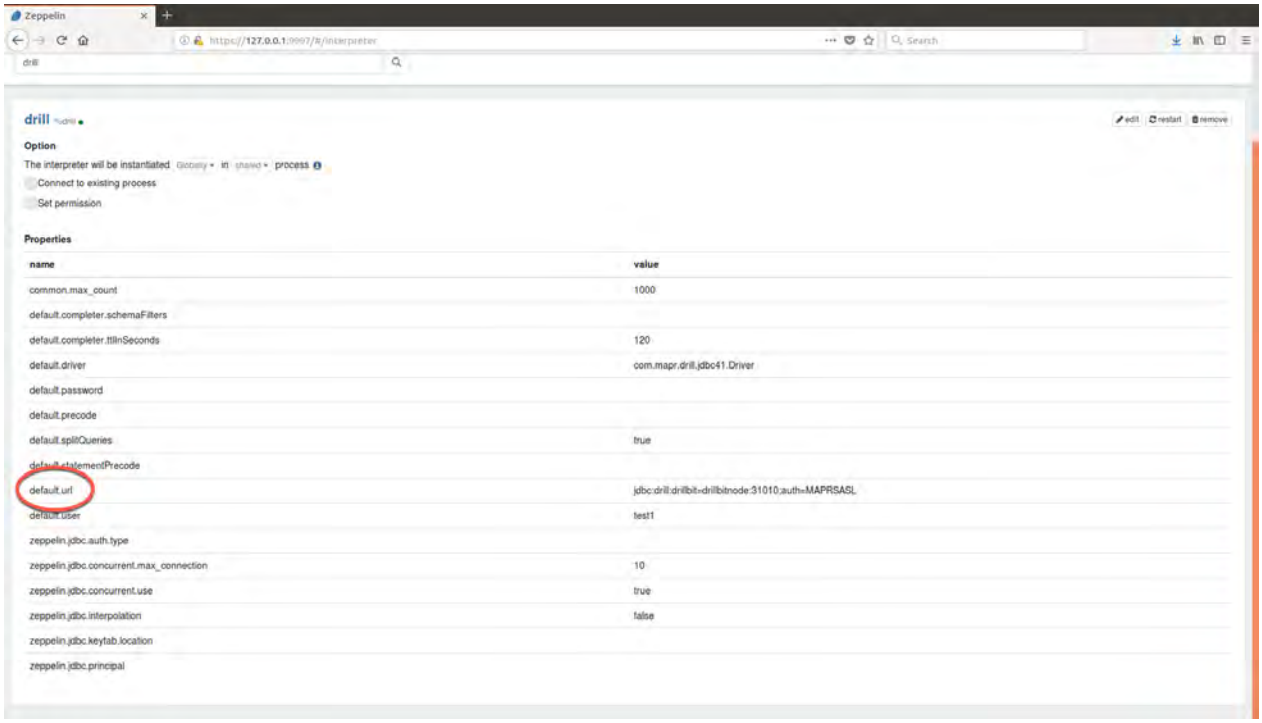
Configuring the JDBC Interpreter for Apache Drill and Apache Hive

Apache Zeppelin on MapR includes custom JDBC interpreters for Apache Drill and Apache Hive. Fields in each interpreter are prepopulated, but you need to customize them for your environment.

In particular, you must modify the JDBC URL, as described in the following sections.

Drill JDBC

You can set the default Apache Drill JDBC URL by specifying environment variables in your `docker run` command. See Default Drill JDBC Connection URL on page 3043 for details. To make additional changes to the URL, modify the `default.url` property:



The following is an example of a URL when MapR-SASL is enabled:

```
jdbc:drill:drillbit=drillbitnode:31010;auth=maprsasl
```

If MapR-SASL is not enabled, the URL is the following:

```
jdbc:drill:drillbit=node1:31010
```

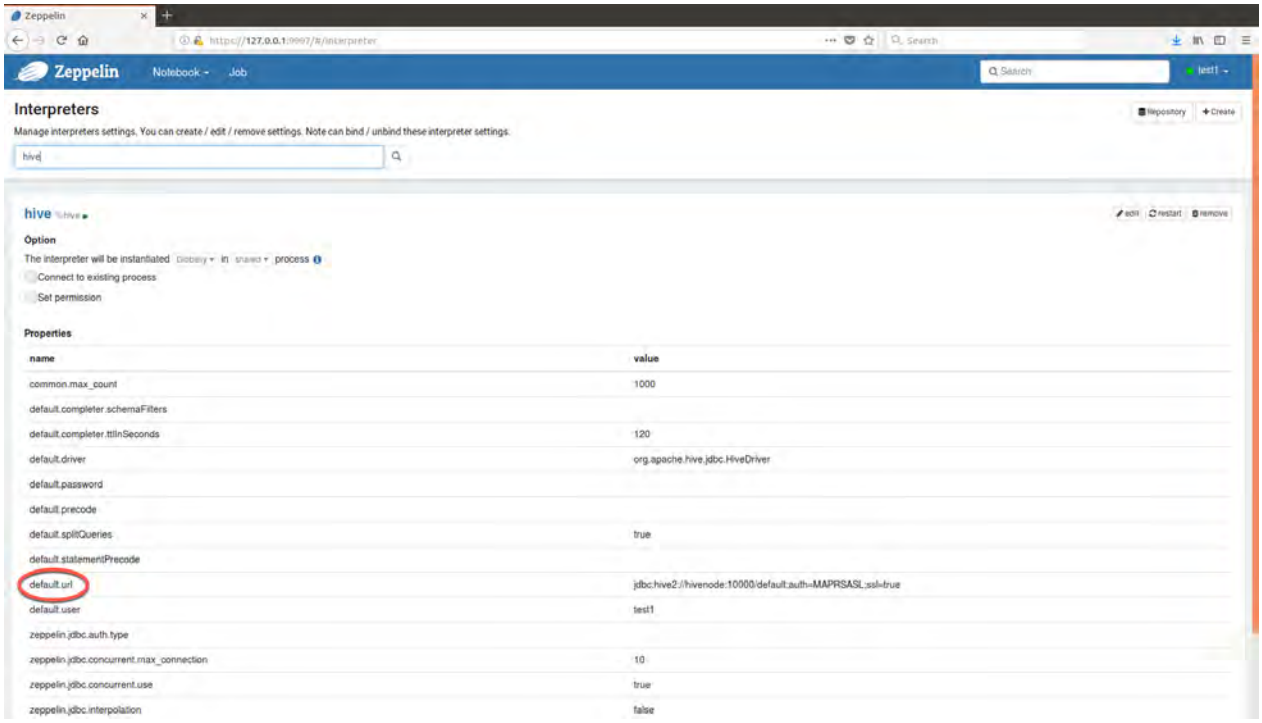
For non-secure clusters, `default.user` is prepopulated with the user running the container (`MAPR_CONTAINER_USER`). You can modify this property and `default.password`, as needed. Zeppelin submits your Drill queries using this user name and password (if specified).

For secure clusters, Zeppelin always submits Drill queries using the user name and password from your MapR ticket. You do not need to modify the `default.user` and `default.password` properties.

Hive JDBC

You must specify the Apache Hive JDBC URL in the `default.url` property:

last-updated: Apr 11, 2022



The following is an example of a URL when MapR-SASL is enabled:

```
jdbc:hive2://hivenode:10000/default;auth=maprsasl;ssl=true
```

Note: Starting with MapR Data Science Refinery 1.3, you must specify `ssl=true` when MapR-SASL is enabled.

If MapR-SASL is not enabled, the URL is the following:

```
jdbc:hive2://node2:10000/default
```

Troubleshooting Zeppelin

This section describes how to resolve common problems you may encounter when using Apache Zeppelin.

Problems Starting Zeppelin

Problem	Your `docker run` command fails with the following message: `Zeppelin start [FAILED]`
Possible Cause	You have configured Docker with too little memory
Resolution	Reconfigure Docker with at least 3.5 GB of memory

last-updated: Apr 11, 2022

Problems Connecting to Zeppelin UI

Problem

You specify storing your notebooks in MapR File System and encounter the following error when trying to connect to Zeppelin in your browser:

```
HTTP ERROR: 503
```

Possible Cause	Resolution
`ZEPPELIN_NOTEBOOK_DIR` is set to an incorrect MapR filesystem directory in your `docker run` command	Verify that you have set `ZEPPELIN_NOTEBOOK_DIR` to a valid MapR filesystem directory
Problems with the FUSE-Based POSIX Client on page 3039	Apply the MapR POSIX Client for Containers license if you are missing this license
Problems with your MapR ticket file	Make sure you have correctly set up your MapR ticket file, including setting the proper owner and group on your source ticket file. See MapR Ticketing on page 3038 for details.

See Notebook Storage on page 3040 for further details.

Problems Logging into the Zeppelin UI

Problem

Log in fails even though you have specified a correct username and password

Possible Cause

You have configured Docker with too little memory

Resolution

Reconfigure Docker with at least 3.5 GB of memory

Navigation Problems in Zeppelin UI

Problem

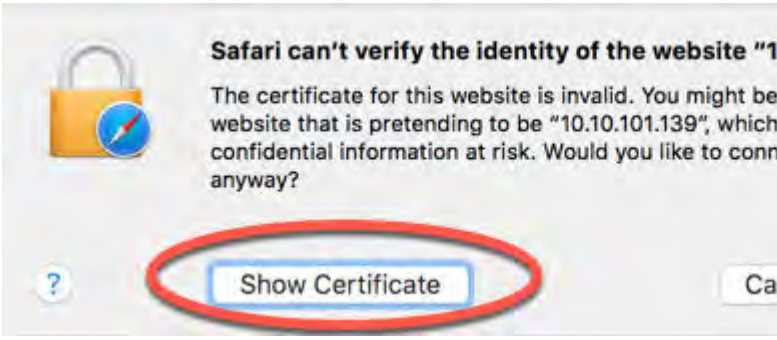
Zeppelin UI screens do not display correctly in your browser

Possible Cause

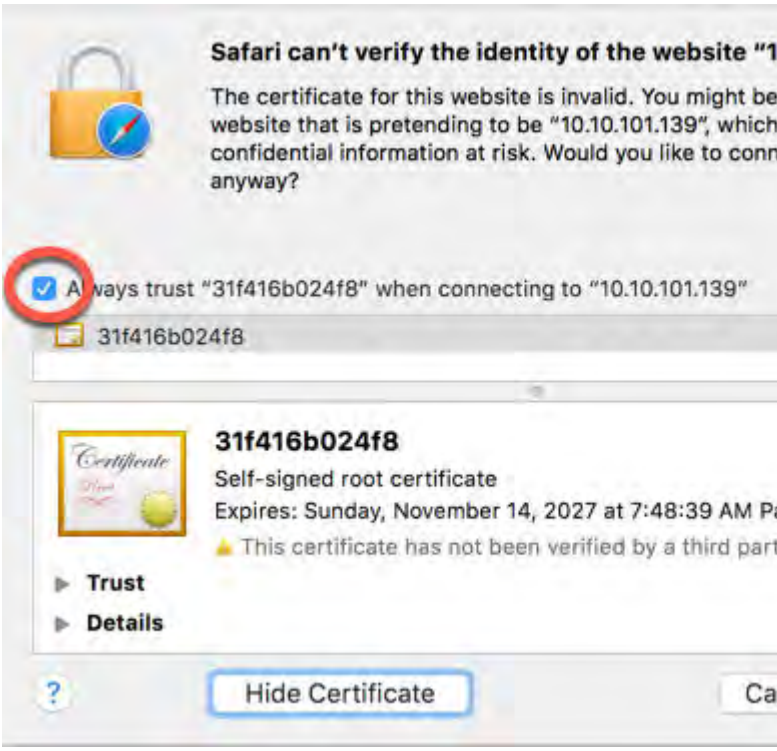
Due to security requirements, connecting to Zeppelin requires self-signed certificates

Resolution

Make sure you accept any certificates when connecting to Zeppelin. The following shows the prompts you need to click in the Safari browser:



last-updated: Apr 11, 2022



Unable to Access MapR File System

Problem	You are using the FUSE-based POSIX client to access MapR filesystem and your MapR filesystem mount point is empty.
Possible Cause	Missing MapR POSIX Client for Containers license
Resolution	Install the necessary license. See FUSE-Based POSIX Client on page 3039 for further details.

`FileClient` Errors

Problem	Running `ls` either in the Zeppelin shell interpreter or inside your container returns the following error: `ls: Could not create FileClient`
Possible Cause	Invalid MapR ticket
Resolution	Make sure you have a valid ticket with the proper permissions. You must generate a unique ticket for each MapR cluster. See MapR Ticketing on page 3038 for further details.

last-updated: Apr 11, 2022

Livy Errors when Using Bridge Networking

Problem

You are using bridge networking, and all commands you run in the Livy interpreter return errors. You see an error similar to the following in the Zeppelin UI:

```
org.apache.zeppelin.livy.LivyException
: Session 0 is finished, appId:
application_1511983726844_0005, log:
[ at
java.util.concurrent.ThreadPoolExecuto
r$Worker.run(ThreadPoolExecutor.java:6
24), at
java.lang.Thread.run(Thread.java:748),
 , Shell output: main : command
provided 1, main : user is mapr,
main : requested yarn user is
mapr, , , Container exited with a
non-zero exit code 15, Failing this
attempt. Failing the application.]
```

Possible Cause

You have not correctly specified the `HOST_IP` or Livy launcher's port range as described at bridge networking.

Solution

Make sure your `docker run` command includes the noted parameters. In particular, make sure you do not specify `localhost` as the `HOST_IP` if you are running the container and Zeppelin UI on the same machine. If you are running multiple Zeppelin containers on a single host, make sure you specify unique port ranges for the Livy launcher as described at Running Multiple Zeppelin Containers on a Single Host on page 3045.

Livy and Pig Errors when Using Bridge Networking

Problem

You are using bridge networking and encounter errors when using the Livy and Pig interpreters.

Possible Cause

The errors may be due to DNS issues. Bridge networking, which is the Docker default, may override your DNS settings. Examine your Zeppelin Log Files on page 3083 to see if there are errors related to unknown hosts.

Sample of errors in the case of Livy are the following:

- Livy interpreter output in the Zeppelin UI:

```
org.apache.zeppelin.livy.LivyExcept
ion: Session 0 is finished, appId:
null, log:
[java.lang.RuntimeException:
Unable to create proxy to the
ResourceManager null,
org.apache.hadoop.yarn.client.MapRZ
KBasedRMFailoverProxyProvider.getPr
oxy(MapRZKBasedRMFailoverProxyProvi
der.java:135)
...
```


last-updated: Apr 11, 2022

- Exceptions in the Livy log file (`/opt/mapr/livy/livy-0.5.0/logs/livy-mapruser1-server.out`):

```
17/11/08 14:41:02 INFO
ContextLauncher: 17/11/08 14:41:02
FATAL zookeeper.ZKDataRetrieval:
Could not create ZooKeeper
instance. Due to IOException. No
data from ZK with connect string:
node1:5181 will be returned.
17/11/08 14:41:02 INFO
ContextLauncher:
java.net.UnknownHostException:
node1: Name or service not known
17/11/08 14:41:02 INFO
ContextLauncher:         at
java.net.Inet4AddressImpl.lookupAll
HostAddr(Native Method)
...
17/11/08 14:41:02 INFO
ContextLauncher: 17/11/08 14:41:02
ERROR
client.MapRZKBasedRMFailoverProxyPr
ovider: Unable to create proxy to
the ResourceManager null
17/11/08 14:41:02 INFO
ContextLauncher: 17/11/08 14:41:02
INFO service.AbstractService:
Service
org.apache.hadoop.yarn.client.api.i
mpl.YarnClientImpl failed in state
STARTED; cause:
java.lang.RuntimeException: Unable
to create proxy to the
ResourceManager null
17/11/08 14:41:02 INFO
ContextLauncher:
java.lang.RuntimeException: Unable
to create proxy to the
ResourceManager null
17/11/08 14:41:02 INFO
ContextLauncher:         at
org.apache.hadoop.yarn.client.MapRZ
KBasedRMFailoverProxyProvider.getPr
oxy(MapRZKBasedRMFailoverProxyProvi
der.java:135)
...
```

Samples of errors in the case of Pig are the following:

last-updated: Apr 11, 2022

- Pig interpreter output in the Zeppelin UI:

```
org.apache.pig.impl.logicalLayer.Fr
ontendException: ERROR 1066:
Unable to open iterator for alias
paragraph_20161228_140730_190334287
7
    at
org.apache.pig.PigServer.openIterat
or(PigServer.java:1032)
...
```

- Exceptions in the Pig log file (`/opt/mapr/zeppelin/zeppelin-0.8.0/logs/zeppelin-interpreter-pig-mapruser1-a384f3320c46.log`):

```
INFO [2017-11-08 14:30:13,346]
({pool-2-thread-3}
ZooKeeper.java[<init>]:438) -
Initiating client connection,
connectString=node1:5181
sessionTimeout=30000
watcher=com.mapr.util.zookeeper.ZKD
ataRetrieval@5de4a6da
FATAL [2017-11-08 14:30:13,544]
({pool-2-thread-3}
ZKDataRetrieval.java[init]:105) -
Could not create ZooKeeper
instance. Due to IOException. No
data from ZK with connect string:
node1:5181 will be returned.
java.net.UnknownHostException:
node1: Name or service not known
        at
java.net.Inet4AddressImpl.lookupAll
HostAddr(Native Method)
```

Resolution

Add the following parameter to `docker run`:

```
--dns=<nameserver-that-resolves-cluste
r-hosts>
```

Problems Running PySpark or SparkR Code

Problem

You are unable to run PySpark or SparkR code in either the Livy or Spark interpreter.

Possible Cause

You have different versions of the Python or R libraries in your Zeppelin container and your MapR cluster.

Resolution

Install matching versions of the libraries in your MapR cluster.

Or, choose the Zeppelin Docker image OS that matches the OS running in your MapR cluster. This minimizes version differences between the libraries running in your container vs your MapR cluster.

NPE Using the Spark Interpreter

Problem

You encounter an NPE when you use the Spark interpreter.

Possible Cause

The directory `/apps/spark` is missing in your MapR cluster. If this is the case, you will see the following error in your Zeppelin Spark interpreter log file:

```
ERROR [2018-03-23 16:40:22,120]
({pool-2-thread-2}
Utils.java[invokeMethod]:40) -
java.lang.reflect.InvocationTargetExce
ption
...
Caused by:
java.io.FileNotFoundException:
Requested file maprfs:///apps/spark
does not exist.
        at
com.mapr.fs.MapRFileSystem.getMapRFile
Status(MapRFileSystem.java:1438)
        at
com.mapr.fs.MapRFileSystem.getFileStat
us(MapRFileSystem.java:1086)
        at
org.apache.spark.scheduler.EventLoggin
gListener.start(EventLoggingListener.s
cala:94)
        at
org.apache.spark.SparkContext.<init>(S
parkContext.scala:531)
        at
org.apache.spark.SparkContext$.getOrCr
eate(SparkContext.scala:2516)
        at
org.apache.spark.sql.SparkSession$Buil
der$
$anonfun$6.apply(SparkSession.scala:91
8)
        at
org.apache.spark.sql.SparkSession$Buil
der$
$anonfun$6.apply(SparkSession.scala:91
0)
        at
scala.Option.getOrElse(Option.scala:12
1)
        at
org.apache.spark.sql.SparkSession$Buil
der.getOrCreate(SparkSession.scala:910
)
        ... 20 more
 INFO [2018-03-23 16:40:22,121]
({pool-2-thread-2}
SparkInterpreter.java[createSparkSessi
on]:362) - Created Spark session with
Hive support
ERROR [2018-03-23 16:40:22,121]
({pool-2-thread-2}
Job.java[run]:181) - Job failed
java.lang.NullPointerException
```


last-updated: Apr 11, 2022

Resolution

Create the directory `/apps/spark` on your MapR cluster as described at Installing Spark on YARN on page 218.

Possible Cause

The credentials on your MapR ticket are incorrect. If this is the case, you will see the following errors in your Zeppelin Spark interpreter log file:

```
ERROR [2018-03-26 22:47:48,967]
({pool-2-thread-2}
Utils.java[invokeMethod]:40) -
java.lang.reflect.InvocationTargetExce
ption
...
Caused by: java.io.IOException:
failure to login:
javax.security.auth.login.LoginExcepti
on: Unable to obtain MapR credentials
        at
org.apache.hadoop.security.UserGroupIn
formation.loginUserFromSubject(UserGro
upInformation.java:751)
        at
org.apache.hadoop.security.UserGroupIn
formation.getLoginUser(UserGroupInform
ation.java:688)
        at
org.apache.hadoop.security.UserGroupIn
formation.getCurrentUser(UserGroupInfo
rmation.java:572)
        at
org.apache.spark.util.Utils$
$anonfun$getCurrentUserName$1.apply(Ut
ils.scala:2424)
        at
org.apache.spark.util.Utils$
$anonfun$getCurrentUserName$1.apply(Ut
ils.scala:2424)
        at
scala.Option.getOrElse(Option.scala:12
1)
        at
org.apache.spark.util.Utils$.getCurren
tUserName(Utils.scala:2424)
        at
org.apache.spark.SparkContext.<init>(S
parkContext.scala:295)
        at
org.apache.spark.SparkContext$.getOrCr
eate(SparkContext.scala:2516)
        at
org.apache.spark.sql.SparkSession$Buil
der$
$anonfun$6.apply(SparkSession.scala:91
8)
        at
org.apache.spark.sql.SparkSession$Buil
der$
$anonfun$6.apply(SparkSession.scala:91
0)
        at
scala.Option.getOrElse(Option.scala:12
```


last-updated: Apr 11, 2022

```
1)
        at
org.apache.spark.sql.SparkSession$Buil
der.getOrCreate(SparkSession.scala:910
)
        ... 20 more
Caused by:
javax.security.auth.login.LoginExcepti
on: Unable to obtain MapR credentials
        at
com.mapr.security.maprsasl.MaprSecurit
yLoginModule.login(MaprSecurityLoginMo
dule.java:228)
        at
sun.reflect.NativeMethodAccessorImpl.i
nvoke0(Native Method)
        at
sun.reflect.NativeMethodAccessorImpl.i
nvoke(NativeMethodAccessorImpl.java:62
)
        at
sun.reflect.DelegatingMethodAccessorIm
pl.invoke(DelegatingMethodAccessorImpl
.java:43)
        at
java.lang.reflect.Method.invoke(Method
.java:498)
        at
javax.security.auth.login.LoginContext
.invoke(LoginContext.java:755)
        at
javax.security.auth.login.LoginContext
.access$000(LoginContext.java:195)
        at
javax.security.auth.login.LoginContext
$4.run(LoginContext.java:682)
        at
javax.security.auth.login.LoginContext
$4.run(LoginContext.java:680)
        at
java.security.AccessController.doPrivi
leged(Native Method)
        at
javax.security.auth.login.LoginContext
.invokePriv(LoginContext.java:680)
        at
javax.security.auth.login.LoginContext
.login(LoginContext.java:587)
        at
org.apache.hadoop.security.UserGroupIn
formation.loginUserFromSubject(UserGro
upInformation.java:724)
        ... 32 more
Caused by:
com.mapr.login.MapRLoginException:
Unable to load ticket file '/tmp/
mapr_ticket', error = 13
        at
com.mapr.login.client.MapRLoginHttpsCl
ient.authenticateIfNeeded(MapRLoginHtt
psClient.java:149)
        at
```


last-updated: Apr 11, 2022

```
com.mapr.login.client.MapRLoginHttpsCl
ient.authenticateIfNeeded(MapRLoginHtt
psClient.java:115)
        at
com.mapr.security.maprsasl.MaprSecurit
yLoginModule.login(MaprSecurityLoginMo
dule.java:222)
        ... 44 more
```

Resolution

Make sure you have correctly set up your MapR ticket file, including setting the proper owner and group on your source ticket file. See MapR Ticketing on page 3038 for details.

Hangs when Using the Spark Interpreter

Problem

You encounter hangs when running code using the Spark interpreter.

Possible Cause

You are using bridge networking and have not reserved the ports used by the Spark driver.

Resolution

Add the following to your `docker run` command:

```
-p 11000-11010:11000-11010
```

See Bridge Networking on page 3037 for further details.

MapR Credential Errors

Problem

Your MapR cluster is secure and you encounter the following error while trying to access the cluster:

```
Unable to obtain MapR credentials
```

Possible Cause

This is likely a problem with your MapR ticket.

Resolution

As described in the MapR Ticketing on page 3038 section that discusses `docker run` parameters, make sure the following are true:

- All identity parameters (user name, group name, UID, and GID) are consistent with the values specified in the ticket file.
- The owner and group on the source ticket file match the UID and GID specified in the ticket file.

Note that it is not necessary for the user and group names to exist on the Docker host. If that is the case, to change the owner and group on the source ticket file, use the UID and GID in the `chown` command:

```
sudo chown
<MAPR_CONTAINER_UID>:<MAPR_CONTAINER_G
ID> <path on Docker host to MapR
ticket>
```

Access Control Errors when Creating Hive Tables

Problem	When creating Hive tables, you encounter an `AccessControlException`.
Possible Cause	This is likely a permission issue. Check the permissions on the Hive Warehouse Directory on page 3410 whose default location is `/user/hive/warehouse`. The permissions on the directory must be 777.
Resolution	Update the permissions if not already set to 777: `hadoop fs -chown 777 /user/hive/warehouse`

Zeppelin Log Files

Apache Zeppelin generates log files for each interpreter you use. Each interpreter also generates its own log files. This section describes the location and naming conventions of these files.

Zeppelin Interpreter Logs

Zeppelin stores these log files in the `/opt/mapr/zeppelin/zeppelin-0.8.0/logs` directory. The names of the files have the following pattern:

```
zeppelin-interpreter-<INTERPRETER_ID>-<MAPR_CONTAINER_USER>-<CONTAINER_ID>.log
```

This is an example of a Pig log file name:

```
/opt/mapr/zeppelin/zeppelin-0.8.0/logs/
zeppelin-interpreter-pig-mapruser1-a384f3320c46.log
```

The following table summarizes the tags used in the name:

Tag Name	Description	Value in Example
<INTERPRETER_ID>	The interpreter family name, e.g., livy, jdbc	pig
<MAPR_CONTAINER_USER>	The user running the container	mapruser1
<CONTAINER_ID>	The container ID generated by Docker for your Zeppelin container	a384f3320c46

Interpreter Log Files

These log files are specific to each interpreter. You can find them in the directories corresponding to each interpreter. For example, in the case of Livy, the log files are in the `/opt/mapr/livy/livy-0.5.0/logs` directory. The names of the Livy log files have the following pattern:

```
livy-<MAPR_CONTAINER_USER>-server.out
```

where `<MAPR_CONTAINER_USER>` is the name of the user running the container.

This is an example of a Livy log file name:

```
/opt/mapr/livy/livy-0.5.0/logs/livy-mapruser1-server.out
```


last-updated: Apr 11, 2022

Using Visualization Packages in Zeppelin

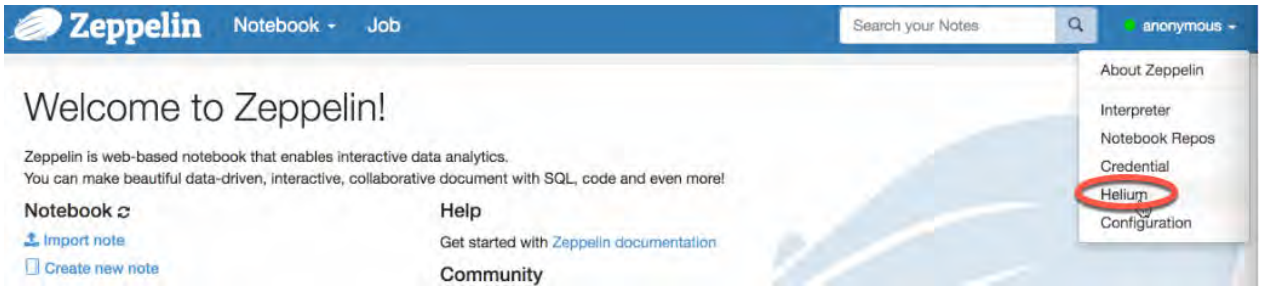
Apache Zeppelin supports the Helium framework. Using visualization packages, you can view your data through area charts, bar charts, scatter charts, and other displays. To use a visualization package, you must enable it through the Helium repository browser in the Zeppelin UI. Like Zeppelin interpreters, Helium is automatically installed in your Zeppelin container.

The example in this section assumes that you are using Zeppelin 0.8.0 and are enabling the `ultimate-pie-chart` package. Zeppelin 0.8.0 is supported starting in MapR Data Science Refinery 1.3. If you are using an earlier release of MapR Data Science Refinery, see earlier versions of the documentation for instructions.

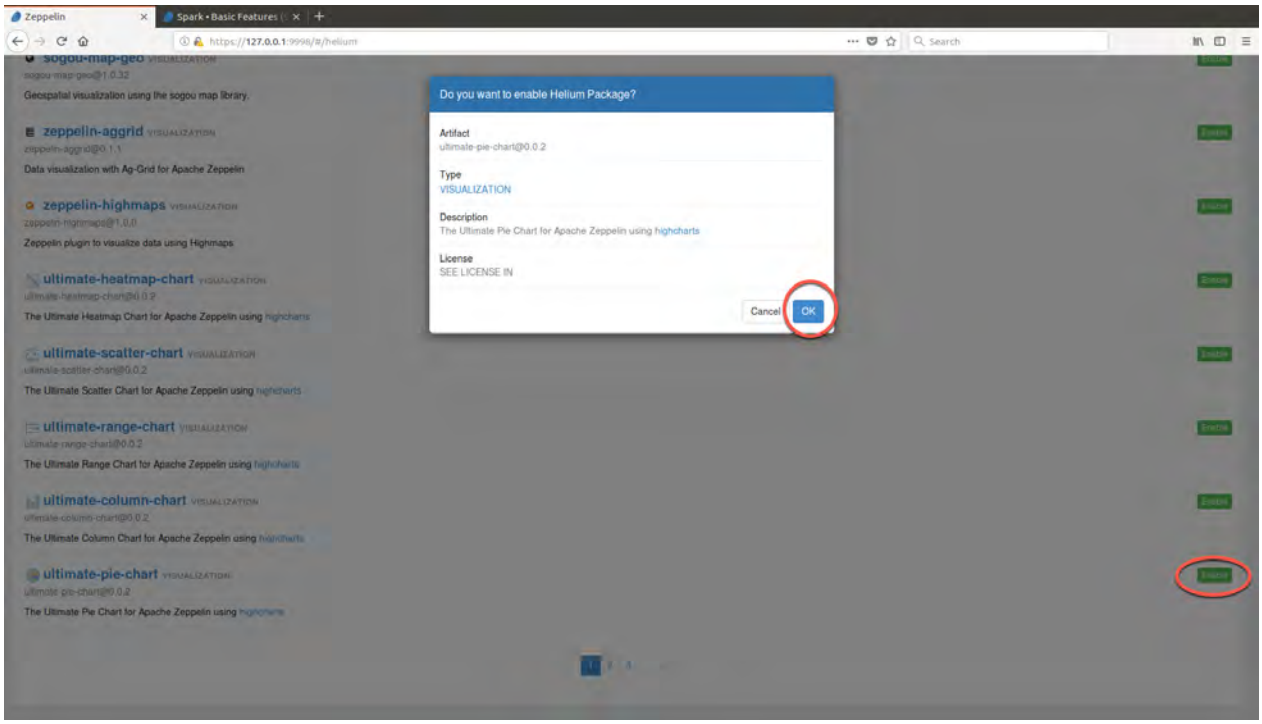
Important: The Apache Community provides and supports the visualization packages available through the Helium repository browser. MapR does not provide support for these packages.

Follow these steps to enable a package:

1. Open the Helium repository browser by selecting the **Helium** tab in the main menu of the Zeppelin UI:



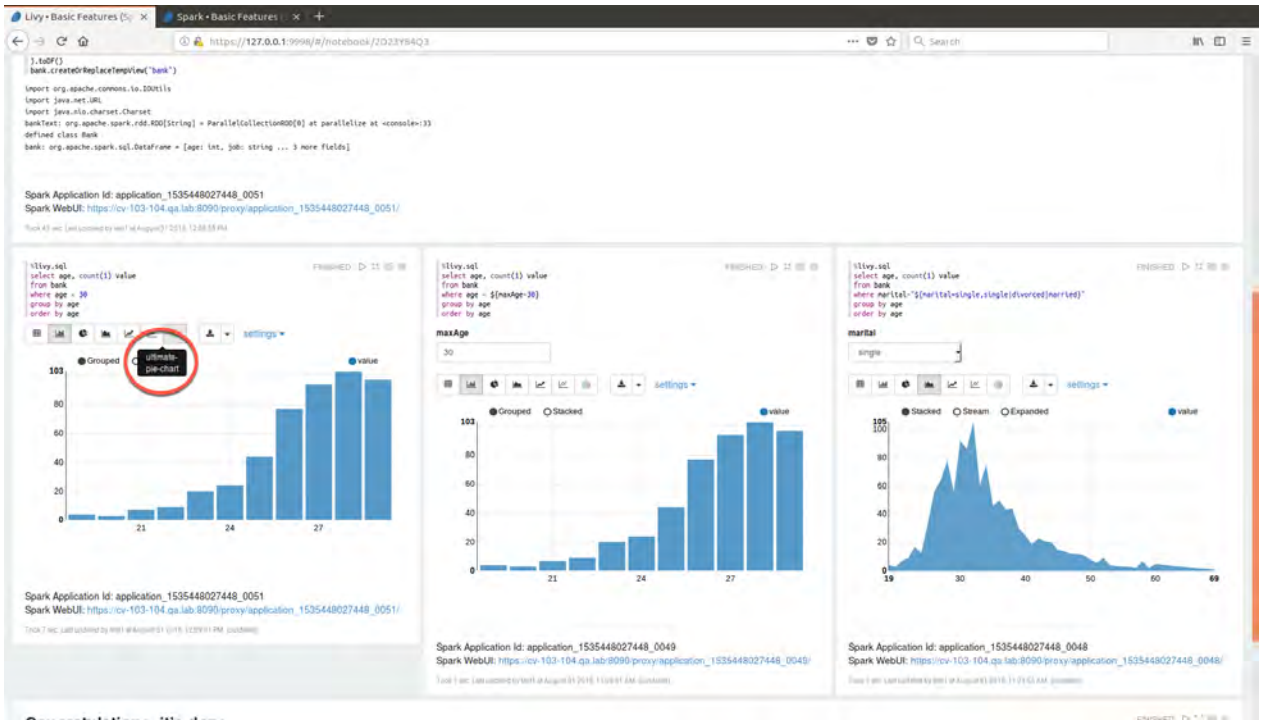
2. Locate your package, click **Enable**, and then click **OK** in the popup window:



The time it takes to enable a package depends on your internet connection speed.

After enabling the package and refreshing your browser to reload notebook content, you can use the package. The following shows output that uses the `ultimate-pie-chart` package:

last-updated: Apr 11, 2022



Using Zeppelin to Access Different Backend Engines

This section contains examples of how to use Apache Zeppelin interpreters to access the different backend engines. This includes running Apache Pig scripts, Apache Drill queries, Apache Hive queries, and Apache Spark jobs, as well as accessing MapR Database and MapR Event Store For Apache Kafka.

Note:

The Data Science Refinery includes the libraries that allow you to access MapR Event Store For Apache Kafka through the MapR Event Store For Apache Kafka Java and C APIs. You must run these applications inside your container. To log in to your container, follow these steps:

1. Determine your `container-id` using the output from the following command:

   ```
   docker ps
   ```

2. Log in to your container as the user running the container using the `container-id`:

   ```
   docker exec -it --user <MAPR_CONTAINER_USER> <container-id> bash -l
   ```

See MapR Event Store For Apache Kafka Java Applications on page 2754 and MapR Event Store For Apache Kafka C Applications on page 2795 for details about how to use these APIs.

Running Shell Commands in Zeppelin

This section shows you how to access files in your local filesystem and MapR File System by using shell commands in your Apache Zeppelin notebook.

To use POSIX shell commands to access MapR File System, you must have a MapR File System mount point in your container. The FUSE-Based POSIX Client on page 3039 provides this functionality.

In the following example, your MapR File System mount point is in `/mapr` and your cluster name is `my.cluster.com`. You can find the name of your cluster in the file `/opt/mapr/conf/mapr-clusters.conf` on your MapR cluster.

1. Create a file of test data in `/tmp`:

```
%sh
cat > /tmp/test.data << EOF
John,Smith
Brian,May
Rodger,Taylor
John,Deacon
Max,Plank
Freddie,Mercury
Albert,Einstein
Fedor,Dostoevsky
Lev,Tolstoy
Niccolo,Paganini
EOF
```

2. Copy the file to your home directory in MapR File System (`/user/mapruser1`) and display the contents of the file:

POSIX

```
%sh
cp /tmp/test.data /mapr/
my.cluster.com/user/mapruser1
cat /mapr/my.cluster.com/user/
mapruser1/test.data
```

Hadoop

```
%sh
hadoop fs -put /tmp/test.data /user/
mapruser1
hadoop fs -cat /user/mapruser1/
test.data
```

Running Pig Scripts in Zeppelin

This section contains a sample of an Apache Pig script that you can run in your Apache Zeppelin notebook.

This example reads the contents of your password file and outputs the first field in the file: the user name.

1. Using the shell interpreter, copy the password file to MapR File System:

POSIX

To use POSIX shell commands like `cp`, you must have a MapR File System mount point in your container. The example below assumes your mount point is `/mapr` and your cluster name is `my.cluster.com`:

```
%sh
cp /etc/passwd /mapr/my.cluster.com/
user/mapruser1
```

Hadoop

```
%sh
hadoop fs -put /etc/passwd /user/
mapruser1/
```

2. Load the password file in Pig and output the first field in each line in the file:

```
%pig
A = load 'passwd' using PigStorage(':');
B = foreach A generate $0 as id;
dump B;
```

There are other examples of using Pig in the Zeppelin tutorial, including running Pig queries.

Running Drill Queries in Zeppelin

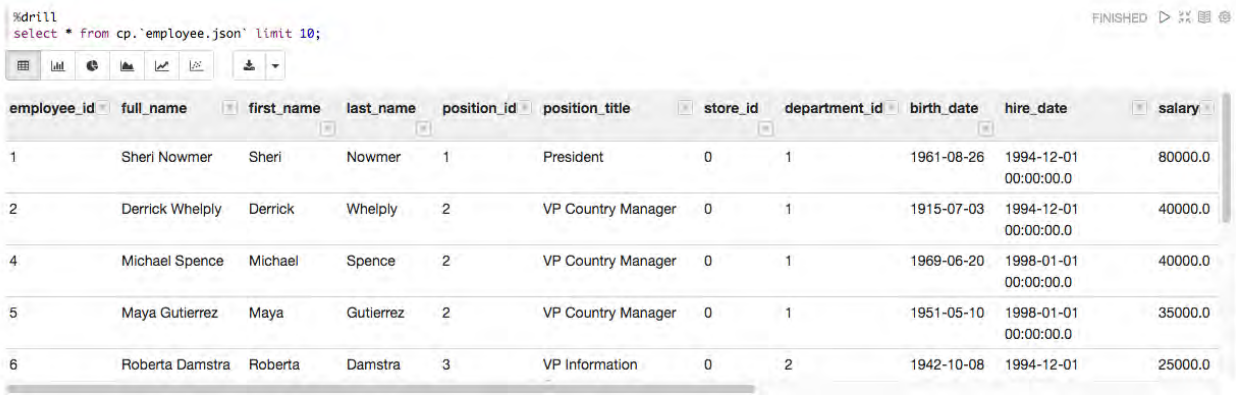
This section contains samples of Apache Drill queries that you can run in your Apache Zeppelin notebook.

Before running Drill queries, make sure you have configured the Drill JDBC interpreter.

The following example queries a JSON file:

```
%drill
select * from cp.`employee.json` limit 10;
```

The output looks like the following:



employee_id	full_name	first_name	last_name	position_id	position_title	store_id	department_id	birth_date	hire_date	salary
1	Sheri Nowmer	Sheri	Nowmer	1	President	0	1	1961-08-26	1994-12-01 00:00:00.0	80000.0
2	Derrick Whelply	Derrick	Whelply	2	VP Country Manager	0	1	1915-07-03	1994-12-01 00:00:00.0	40000.0
4	Michael Spence	Michael	Spence	2	VP Country Manager	0	1	1969-06-20	1998-01-01 00:00:00.0	40000.0
5	Maya Gutierrez	Maya	Gutierrez	2	VP Country Manager	0	1	1951-05-10	1998-01-01 00:00:00.0	35000.0
6	Roberta Damstra	Roberta	Damstra	3	VP Information	0	2	1942-10-08	1994-12-01	25000.0

This example creates a table from a JSON source file and queries it:

```
%drill
use dfs.tmp;
create table drill_one as select * from cp.`employee.json`;
select * from dfs.tmp.drill_one limit 10;
```

Running Hive Queries in Zeppelin

This section contains samples of Apache Hive queries that you can run in your Apache Zeppelin notebook.

last-updated: Apr 11, 2022

Before running Hive queries, make sure you have configured the Hive JDBC interpreter. Also, see MapR Data Science Refinery Support by MapR Core Version on page 5638 for limitations when connecting to a secure MapR 6.1 cluster.

1. Using the shell interpreter, create a source data file:

```
%sh
cat > /tmp/test.data << EOF
John,Smith
Brian,May
Rodger,Taylor
John,Deacon
Max,Plank
Freddie,Mercury
Albert,Einstein
Fedor,Dostoevsky
Lev,Tolstoy
Niccolo,Paganini
EOF
```

2. Copy the file to MapR File System:

POSIX

To use POSIX shell commands like `cp`, you must have a MapR filesystem mount point in your container. The example below assumes your mount point is `/mapr` and your cluster name is `my.cluster.com`:

```
%sh
cp /tmp/test.data /mapr/
my.cluster.com/user/mapruser1
```

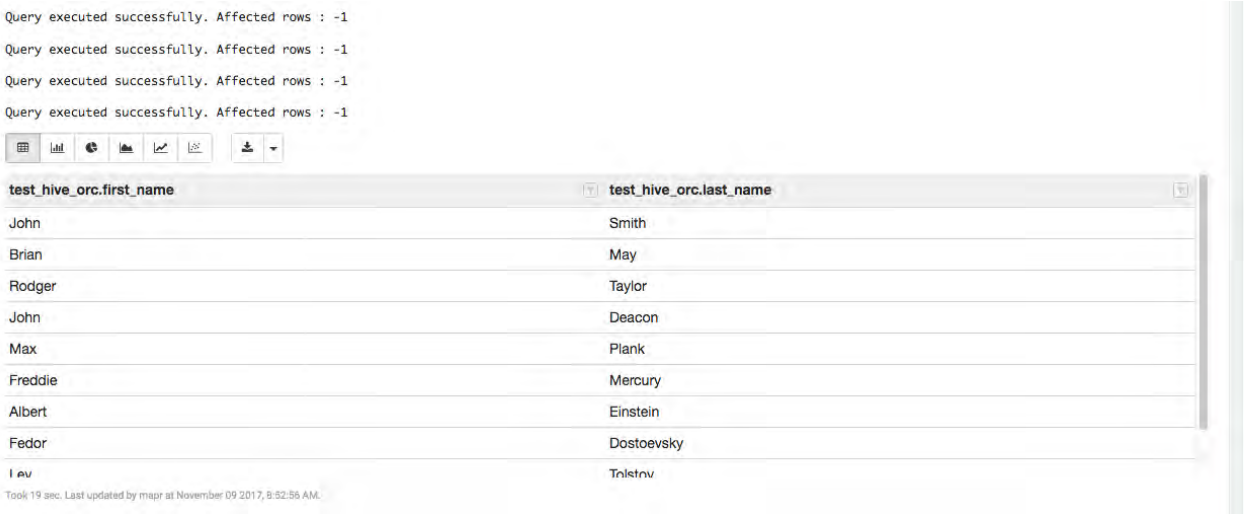
Hadoop

```
%sh
hadoop fs -put /tmp/test.data /user/
mapruser1
```

3. Run the Hive code using the Hive JDBC interpreter:

```
%hive
-- create and load Hive table
create table test_hive(first_name string, last_name string) ROW FORMAT
DELIMITED FIELDS TERMINATED BY ',';
load data inpath '/user/mapruser1/test.data' overwrite into table
test_hive;
-- create and load Hive ORC table
create table test_hive_orc(first_name string, last_name string) stored
as orc tblproperties ("orc.compress"="NONE");
insert overwrite table test_hive_orc  select * from test_hive;
-- query the Hive ORC table
select * from test_hive_orc;
```

The output looks like the following:



4. Drop the Hive tables created in the example:

```
%hive
drop table test_hive;
drop table test_hive_orc;
```

Running Spark Jobs in Zeppelin

This section contains code samples for different types of Apache Spark jobs that you can run in your Apache Zeppelin notebook. You can run these examples using either the Livy or Spark interpreter. The Spark interpreter is available starting in the 1.1 release of the MapR Data Science Refinery.

Before running these examples, depending on whether you are using the Livy or Spark interpreter, make sure you have configured the interpreter.

Note: The examples in this section use Hadoop commands to access files in MapR File System. If you have a MapR File System mount point in your container, you can replace the Hadoop commands with standard shell commands. Refer to Running Shell Commands in Zeppelin on page 3085 for an example of how to do this.

Running a Spark Job Using PySpark

The following example shows how to run a Spark job using Python. Make sure you have installed Python on your MapR cluster.

1. Before running the sample PySpark code, copy the files that the code references to MapR File System:

```
%sh
hadoop fs -mkdir -p /user/mapruser1/examples/src/main/resources/
hadoop fs -put /opt/mapr/spark/spark-2.3.1/examples/src/main/resources/
people.txt /user/mapruser1/examples/src/main/resources/
hadoop fs -put /opt/mapr/spark/spark-2.3.1/examples/src/main/resources/
people.json /user/mapruser1/examples/src/main/resources/
```

2. Run the PySpark code specifying either `%livy.pyspark` or `%spark.pyspark` as your interpreter:

```
from pyspark.sql.types import *
sc = spark.sparkContext
lines = sc.textFile("examples/src/main/resources/people.txt")
parts = lines.map(lambda l: l.split(","))
people = parts.map(lambda p: (p[0], p[1].strip()))
schemaString = "name age"
fields = [StructField(field_name, StringType(), True) for field_name in
schemaString.split()]
schema = StructType(fields)
schemaPeople = spark.createDataFrame(people, schema)
schemaPeople.createOrReplaceTempView("people")
schemaPeople.createOrReplaceTempView("people")
results = spark.sql("SELECT name FROM people")
results.show()
```

Running a Spark Job Using SparkR

The following SparkR code example creates a table and queries it using HiveQL. Make sure you have installed SparkR on your MapR cluster. Set your interpreter to either `%livy.sparkr` or `%spark.r`, depending on whether you are using Livy or Spark.

```
sqlContext <- sparkR.session(sc)
sql("CREATE TABLE IF NOT EXISTS src (key INT, value STRING)")
sql("LOAD DATA LOCAL INPATH '/opt/mapr/spark/spark-2.3.1/examples/src/main/
resources/kv1.txt' INTO TABLE src")

# Queries can be expressed in HiveQL.
results <- collect(sql("FROM src SELECT key, value"))
print(results)
```

Reading a JSON File Using Spark

Set your interpreter to either `%livy.spark` or `%spark`, depending on whether you are using Livy or Spark.

The following example loads a JSON file into a Spark DataFrame and then displays it:

```
val pathJSON = "file:/opt/mapr/spark/spark-2.3.1/examples/src/main/
resources/people.json"
val df = spark.read.json(pathJSON)
df.show()
```

Querying Using Spark

Set your interpreter to either `%livy.spark` or `%spark`, depending on whether you are using Livy or Spark.

The following example extends the JSON file reading example by issuing various queries on the data read into the DataFrame:

```
import spark.implicits._
val pathJSON = "file:/opt/mapr/spark/spark-2.3.1/examples/src/main/
resources/people.json"
val df = spark.read.json(pathJSON)
df.printSchema()
df.select("name").show()
df.select($"name", $"age" + 1).show()
```

```
df.filter($"age" > 21).show()
df.groupBy("age").count().show()
```

Querying Hive Tables Using Spark SQL

The following two examples query Hive tables using Spark SQL queries. Make sure the `hive-site.xml` configuration file from your Hive cluster is available in your Zeppelin container. Hive Tables on page 3064 describes the detailed steps.

If the code snippets in these examples do not specify an interpreter, specify `%livy.spark` to use the Livy interpreter or `%spark` to use the Spark interpreter.

Example 1

1. Run the following code to create Hive tables and issue various select statements against them:

```
import org.apache.spark.sql.Row
import org.apache.spark.sql.SparkSession
case class Record(key: Int, value: String)
import spark.implicits._
import spark.sql

// Create table, loading data from text file
sql("CREATE TABLE IF NOT EXISTS src (key INT, value STRING)")
sql("LOAD DATA LOCAL INPATH '/opt/mapr/spark/spark-2.3.1/examples/src/
main/resources/kv1.txt' INTO TABLE src")
sql("SELECT * FROM src").show()
sql("SELECT COUNT(*) FROM src").show()

// Find records where key < 10, ordering the result
val sqlDF = sql("SELECT key, value FROM src WHERE key < 10 ORDER BY key")

// Create a second table and join with the first
val stringsDS = sqlDF.map {case Row(key: Int, value: String) => s"Key:
$key, Value: $value"}
stringsDS.show()
val recordsDF = spark.createDataFrame((1 to 100).map(i => Record(i,
s"val_$i")))
recordsDF.createOrReplaceTempView("records")
sql("SELECT * FROM records r JOIN src s ON r.key = s.key").show()
```

2. Drop the table created in the previous code snippet:

```
sql("DROP TABLE src").show()
```

Example 2

1. Using the shell interpreter, create the source data file that you will use to load your Hive table:

```
%sh
cat > /tmp/test.data << EOF
John,Smith
Brian,May
Rodger,Taylor
John,Deacon
Max,Plank
Freddie,Mercury
Albert,Einstein
Fedor,Dostoevsky
Lev,Tolstoy
Niccolo,Paganini
EOF
hadoop fs -put /tmp/test.data /user/mapruser1/
```

2. Create, load, and query the Hive table:

```
sql ("create table test_hive(first_name string, last_name string) ROW
FORMAT DELIMITED FIELDS TERMINATED BY ','")
sql("load data inpath '/user/mapruser1/test.data' overwrite into table
test_hive")
sql("CREATE TABLE test_hive_parquet (first_name string, last_name
string) STORED AS PARQUET")
sql("INSERT OVERWRITE TABLE test_hive_parquet SELECT first_name,
last_name FROM test_hive")
sql("SELECT * from test_hive_parquet").show()
```

Running MapR Database Shell Commands in Zeppelin
This section contains a sample of MapR Database shell commands that you can run in your Apache Zeppelin notebook.

1. Invoke the MapR Database shell:

```
%maprdb.shell
```

2. Create a MapR Database JSON table in your home directory on MapR filesystem:

```
create ./sample_table
```

3. Insert some documents in to the table:

```
insert ./sample_table --value '{"_id": "FYWN1w","name":"Dental by
Design","city":"Ahwatukee","stars":4.0}'
insert ./sample_table --value '{"_id": "He-G7v","name":"Stephen Szabo
Salon","city":"McMurray","stars":3.0}'
insert ./sample_table --value '{"_id": "KQPW8l","name":"Western Motor
Vehicle","city":"Phoenix","stars":1.5}'
insert ./sample_table --value '{"_id": "8DShNS","name":"Sports
Authority","city":"Tempe","stars":3.0}'
insert ./sample_table --value '{"_id": "PfOCPj","name":"Brick House
Tavern + Tap","city":"Cuyahoga Falls","stars":3.5}'
insert ./sample_table --value '{"_id":
"o9eMRC","name":"Messina","city":"Stuttgart","stars":4.0}'
insert ./sample_table --value '{"_id": "kCoE3j","name":"BDJ
Realty","city":"Las Vegas","stars":4.0}'
insert ./sample_table --value '{"_id": "OD2hnu","name":"Soccer
Zone","city":"Las Vegas","stars":1.5}'
insert ./sample_table --value '{"_id": "EsMcGi","name":"Any Given
Sundae","city":"Wexford","stars":5.0}'
insert ./sample_table --value '{"_id": "TGWhGN","name":"Detailing Gone
Mobile","city":"Henderson","stars":5.0}'
```

4. Retrieve the documents with at least a 4 star rating:

```
find ./sample_table  --where '{"$ge":{"stars":4}}'
```

   The query returns the following:

```
{"_id": "EsMcGi","name":"Any Given Sundae","city":"Wexford","stars":5.0}
{"_id": "FYWN1w","name":"Dental by
Design","city":"Ahwatukee","stars":4.0}
{"_id": "TGWhGN","name":"Detailing Gone
Mobile","city":"Henderson","stars":5.0}
{"_id": "kCoE3j","name":"BDJ Realty","city":"Las Vegas","stars":4.0}
{"_id": "o9eMRC","name":"Messina","city":"Stuttgart","stars":4.0}
5 document(s) found.
```

Accessing MapR Database in Zeppelin Using the MapR Database Binary Connector

This section contains an example of an Apache Spark job that uses the MapR Database Binary Connector for Apache Spark to write and read a MapR Database Binary table. You can run this example using either the Livy or Spark interpreter. The Spark interpreter is available starting in the 1.1 release of the MapR Data Science Refinery.

Before running the example, make sure you have configured either your Livy or Spark interpreter to run Spark jobs.

This example is derived from the example at SparkSQL and DataFrames on page 4108. That page provides a more detailed explanation of the code.

The Zeppelin on MapR Tutorial also includes a notebook with Scala code examples using the MapR Database Binary Connector.

1. Set your interpreter to either `%livy.spark` or `%spark`, depending on whether you are using Livy or Spark.

2. Run the following code in your notebook:

```
import org.apache.spark.sql.{DataFrame, SQLContext}
import org.apache.spark.{SparkContext, SparkConf}
import org.apache.spark.sql.datasources.hbase.HBaseTableCatalog
case class HBaseRecordClass(
  col0: String,
  col1: Boolean,
  col2: Double,
  col3: Float,
  col4: Int,
  col5: Long,
  col6: Short,
  col7: String,
  col8: Byte)
object HBaseRecord {
  def apply(i:Int): HBaseRecordClass = {
    val s = "row" + "%03d".format(i)
    new HBaseRecordClass(s,
      i % 2 == 0,
      i.toDouble,
      i.toFloat,
      i,
      i.toLong,
      i.toShort,
      s"String$i extra",
      i.toByte)
  }
}
val tableName = "/user/mapruser1/test1"
val cat = s"""{
          |"table":{"namespace":"default", "name":"$tableName"},
          |"rowkey":"key",
          |"columns":{
            |"col0":{"cf":"rowkey", "col":"key", "type":"string"},
            |"col1":{"cf":"cf1", "col":"col1", "type":"boolean"},
            |"col2":{"cf":"cf2", "col":"col2", "type":"double"},
            |"col3":{"cf":"cf3", "col":"col3", "type":"float"},
            |"col4":{"cf":"cf4", "col":"col4", "type":"int"},
            |"col5":{"cf":"cf5", "col":"col5", "type":"bigint"},
            |"col6":{"cf":"cf6", "col":"col6", "type":"smallint"},
            |"col7":{"cf":"cf7", "col":"col7", "type":"string"},
            |"col8":{"cf":"cf8", "col":"col8", "type":"tinyint"}
            |}
          |}""".stripMargin
val sqlContext = new SQLContext(sc)
import sqlContext.implicits._

def withCatalog(cat: String): DataFrame = {
  sqlContext
    .read
    .options(Map(HBaseTableCatalog.tableCatalog->cat))
    .format("org.apache.hadoop.hbase.spark")
    .load()
}
val data = (0 to 255).map { i =>
  HBaseRecord(i)
}
sc.parallelize(data).toDF.write.options(
  Map(HBaseTableCatalog.tableCatalog ->
cat, HBaseTableCatalog.newTable ->
"5")).format("org.apache.hadoop.hbase.spark").save()
```

```
val df = withCatalog(cat)
df.show
```

The output looks like the following:

```
+----+--------------+-----+----+----+------+----+----+----+
|col4|          col7| col1|col3|col6|  col0|col8|col2|col5|
+----+--------------+-----+----+----+------+----+----+----+
|   0| String0 extra| true| 0.0|   0|row000|   0| 0.0|   0|
|   1| String1 extra|false| 1.0|   1|row001|   1| 1.0|   1|
|   2| String2 extra| true| 2.0|   2|row002|   2| 2.0|   2|
|   3| String3 extra|false| 3.0|   3|row003|   3| 3.0|   3|
|   4| String4 extra| true| 4.0|   4|row004|   4| 4.0|   4|
|   5| String5 extra|false| 5.0|   5|row005|   5| 5.0|   5|
|   6| String6 extra| true| 6.0|   6|row006|   6| 6.0|   6|
|   7| String7 extra|false| 7.0|   7|row007|   7| 7.0|   7|
|   8| String8 extra| true| 8.0|   8|row008|   8| 8.0|   8|
|   9| String9 extra|false| 9.0|   9|row009|   9| 9.0|   9|
|  10|String10 extra| true|10.0|  10|row010|  10|10.0|  10|
|  11|String11 extra|false|11.0|  11|row011|  11|11.0|  11|
|  12|String12 extra| true|12.0|  12|row012|  12|12.0|  12|
|  13|String13 extra|false|13.0|  13|row013|  13|13.0|  13|
|  14|String14 extra| true|14.0|  14|row014|  14|14.0|  14|
|  15|String15 extra|false|15.0|  15|row015|  15|15.0|  15|
|  16|String16 extra| true|16.0|  16|row016|  16|16.0|  16|
|  17|String17 extra|false|17.0|  17|row017|  17|17.0|  17|
|  18|String18 extra| true|18.0|  18|row018|  18|18.0|  18|
|  19|String19 extra|false|19.0|  19|row019|  19|19.0|  19|
+----+--------------+-----+----+----+------+----+----+----+
only showing top 20 rows
```

Note: Zeppelin displays only the first 20 rows in the output.

See MapR Database Binary Connector for Apache Spark on page 4101 for additional information about this connector.

Accessing MapR Database in Zeppelin Using the MapR Database OJAI Connector

This section contains examples of Apache Spark jobs that use the MapR Database OJAI Connector for Apache Spark to read and write MapR Database JSON tables. The examples use the Spark Python interpreter. The Spark interpreter is available starting in the 1.1 release of the MapR Data Science Refinery. The Python API in the MapR Database OJAI Connector is available starting in the EEP 4.1 release.

Before running the examples, make sure you have configured your Spark interpreter to run Spark jobs.

Inserting a Spark DataFrame into a MapR Database JSON Table

The following code sample creates a Spark DataFrame, inserts it into a MapR Database JSON table that is created as part of the insert, and then loads it into another DataFrame:

```
%spark.pyspark
df = sc.parallelize([ { "_id": "rsmith", "address": { "city": "San
Francisco", "line": "100 Main Street", "zip": 94105 }, "dob": "1982-02-03",
"first_name": "Robert", "interests": [ "electronics", "music", "sports" ],
"last_name": "Smith" }, { "_id": "mdupont", "address": { "city": "San
Jose", "line": "1223 Broadway", "zip": 95109 }, "dob": "1982-02-03",
"first_name": "Maxime", "interests": [ "sports", "movies", "electronics" ],
"last_name": "Dupont" }, { "_id": "jdoe", "address": None, "dob":
"1970-06-23", "first_name": "John", "interests": None, "last_name":
"Doe" }, { "_id": "dsimon", "address": None, "dob": "1980-10-13",
"first_name": "David", "interests": None, "last_name": "Simon" }, { "_id":
```

```
"alehmann", "address": None, "dob": "1980-10-13", "first_name": "Andrew",
"interests": [ "html", "css", "js" ], "last_name": "Lehmann" } ]).toDF()

# Insert into MapR-DB table
spark.insertToMapRDB(df, "/user/mapruser1/table1", create_table=True)

# Load previously inserted data from MapR-DB table
df_loaded = spark.loadFromMapRDB("/user/mapruser1/table1").show()
```

Inserting a Spark DataFrame into a MapR Database JSON Table Using Bulk Insert

To bulk insert into a MapR Database JSON table that is created as part of the insert operation, you must order the records in the DataFrame as shown in the following example:

```
%spark.pyspark
df = sc.parallelize([ { "_id": "rsmith", "address":{ "city": "San
Francisco", "line": "100 Main Street", "zip": 94105 }, "dob": "1982-02-03",
"first_name": "Robert", "interests": [ "electronics", "music", "sports" ],
"last_name": "Smith" }, { "_id": "mdupont", "address":{ "city": "San Jose",
"line": "1223 Broadway", "zip": 95109 }, "dob": "1982-02-03", "first_name":
"Maxime", "interests": [ "sports", "movies", "electronics" ], "last_name":
"Dupont" },{ "_id": "jdoe", "address": None, "dob": "1970-06-23",
"first_name": "John", "interests": None, "last_name": "Doe" },{ "_id":
"dsimon", "address": None, "dob": "1980-10-13", "first_name": "David",
"interests": None, "last_name": "Simon" },{ "_id": "alehmann", "address":
None, "dob": "1980-10-13", "first_name": "Andrew", "interests": [ "html",
"css", "js" ], "last_name": "Lehmann" }]).toDF().orderBy("_id")

# Bulk insert into MapR-DB table
spark.insertToMapRDB(df, "/user/mapruser1/table2", create_table=True,
bulk_insert=True)

# Load previously inserted data from MapR-DB table
df_loaded = spark.loadFromMapRDB("/user/mapruser1/table2").show()
```

Selecting and Filtering Data when Loading a Spark DataFrame

The following code sample uses projection and filtering when loading a Spark DataFrame from a MapR Database JSON table:

```
%spark.pyspark

from pyspark.sql.functions import col, asc

df = sc.parallelize([{ "_id": "rsmith", "address": { "city": "San
Francisco", "line": "100 Main Street", "zip": 94105 }, "dob": "1982-02-03",
"first_name": "Robert", "interests": [ "electronics", "music", "sports" ],
"last_name": "Smith" }, { "_id": "mdupont", "address": { "city": "San
Jose", "line": "1223 Broadway", "zip": 95109 }, "dob": "1982-02-03",
"first_name": "Maxime", "interests": [ "sports", "movies", "electronics" ],
"last_name": "Dupont" }]).toDF()

spark.saveToMapRDB(df, "/user/mapruser1/table3", create_table=True)

# Load previously saved data from the MapR-DB table
df_loaded_select = spark.loadFromMapRDB("/user/mapruser1/table3")\
    .select("_id","first_name","address")\
    .filter(col("first_name") == "Maxime").show()
```

Joining DataFrames when Loading a Spark DataFrame

The following code sample loads a Spark DataFrame from a MapR Database JSON table and joins the DataFrame with a second DataFrame:

```
%spark.pyspark
df = sc.parallelize([ { "_id": "rsmith", "address": { "city": "San
Francisco", "line": "100 Main Street", "zip": 94105 }, "dob": "1982-02-03",
"first_name": "Robert", "interests": [ "electronics", "music", "sports" ],
"last_name": "Smith" }, { "_id": "mdupont", "address": { "city": "San
Jose", "line": "1223 Broadway", "zip": 95109 }, "dob": "1982-02-03",
"first_name": "Maxime", "interests": [ "sports", "movies", "electronics" ],
"last_name": "Dupont" }, { "_id": "jdoe", "address": None, "dob":
"1970-06-23", "first_name": "John", "interests": None, "last_name":
"Doe" }, { "_id": "dsimon", "address": None, "dob": "1980-10-13",
"first_name": "David", "interests": None, "last_name": "Simon" }, { "_id":
"alehmann", "address": None, "dob": "1980-10-13", "first_name": "Andrew",
"interests": [ "html", "css", "js" ], "last_name": "Lehmann" } ]).toDF()
dfProfessions = sc.parallelize([ { "_id": "rsmith", "profession":
"Engineer" }, { "_id": "alehmann", "profession": "Doctor" }, { "_id":
"alehmann", "profession": "Accountant" }, { "_id": "fake", "profession":
"Software developer" } ]).toDF()

# Save to MapR-DB table
spark.saveToMapRDB(df, "/user/mapruser1/table4", create_table=True)

# Load previously saved data from MapR-DB Table and join with another
DataFrame
df_loaded_select = spark.loadFromMapRDB("/user/mapruser1/table4")\
    .join(dfProfessions, "_id").show()
```

Tutorial Examples

The Zeppelin on MapR Tutorial includes notebooks with Python and Scala code examples using the MapR Database OJAI Connector.

Related concepts

Understanding the MapR Database OJAI Connector for Spark on page 4050
Using the MapR Database OJAI connector for Spark enables you build real-time and batch pipelines between your data and MapR Database JSON. Before getting started, it is important that you understand Spark terminology and workflow, system requirements and support, and OJAI connector and API features.

Accessing MapR Event Store For Apache Kafka in Zeppelin Using the Livy Interpreter

This section contains a MapR Event Store For Apache Kafka streaming example that you can run in your Apache Zeppelin notebook using the Livy interpreter.

Note: See MapR Data Science Refinery Support by MapR Core Version on page 5638 for limitations in version support when accessing MapR Event Store.

The example references a stream named `test_stream` created in the path `/streaming_test/test_stream`. The stream contains a topic called `test_topic`. You can use the following commands to create this stream and topic, but you cannot run them in Zeppelin; they must be run in a MapR cluster.

```
hadoop fs -mkdir /streaming_test
hadoop fs -chown <user>:<group> /streaming_test
sudo su - <user>
maprcli stream create -path /streaming_test/test_stream
maprcli stream topic create -path /streaming_test/test_stream -topic
test_topic
```


last-updated: Apr 11, 2022

When the stream and topic are available, perform the following actions in your notebook:

1. Configure the Livy interpreter. Make sure to follow the steps described in the Spark Jobs on page 3064 section to allow Spark jobs to run in parallel.

2. Create a streaming consumer in your notebook using the `%livy.spark` interpreter:

```
import org.apache.kafka.clients.consumer.ConsumerConfig

import org.apache.spark.SparkConf
import org.apache.spark.streaming.{Seconds, StreamingContext}
import org.apache.spark.streaming.kafka09.{ConsumerStrategies,
KafkaUtils, LocationStrategies}

val ssc = new StreamingContext(sc, Seconds(1))

val topicsSet = Set("/streaming_test/test_stream:test_topic")
val kafkaParams = Map[String, String](
  ConsumerConfig.BOOTSTRAP_SERVERS_CONFIG -> "localhost:9092",
  ConsumerConfig.GROUP_ID_CONFIG -> "none",
  ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG ->
    "org.apache.kafka.common.serialization.StringDeserializer",
  ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG ->
    "org.apache.kafka.common.serialization.StringDeserializer",
  ConsumerConfig.AUTO_OFFSET_RESET_CONFIG -> "earliest",
  ConsumerConfig.ENABLE_AUTO_COMMIT_CONFIG -> "false"
)

val consumerStrategy =
      ConsumerStrategies.Subscribe[String, String](topicsSet,
kafkaParams)
val messages = KafkaUtils.createDirectStream[String, String](
      ssc,
      LocationStrategies.PreferConsistent,
      consumerStrategy)

val lines = messages.map(_.value())
val words = lines.flatMap(_.split(" "))
val wordCounts = words.map(x => (x, 1L)).reduceByKey(_ + _)
wordCounts.print()

ssc.start()
ssc.awaitTerminationOrTimeout(3 * 60 * 1000)
```

3. Create a streaming producer in another notebook, also with the `%livy.spark` interpreter:

```
import java.util.Properties
import org.apache.kafka.clients.producer.{KafkaProducer, ProducerRecord}

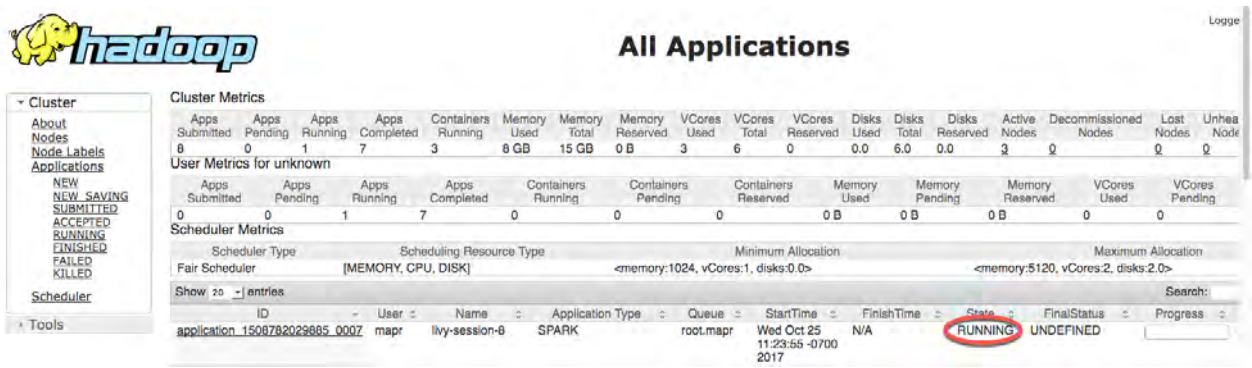
val props = new Properties()
props.put("bootstrap.servers", "localhost:9092")
props.put("acks", "all")
props.put("retries", "0")
props.put("batch.size", "16384")
props.put("linger.ms", "1")
props.put("buffer.memory", "33554432")
props.put("key.serializer",
"org.apache.kafka.common.serialization.StringSerializer")
props.put("value.serializer",
"org.apache.kafka.common.serialization.StringSerializer")

val producer = new KafkaProducer[String, String](props)

for (i <- 1 to 1000) {
    val message = new ProducerRecord[String, String]("/streaming_test/
test_stream:test_topic", i.toString(), i.toString())
    producer.send(message)
}
```

4. Start running the consumer notebook from Step 2.

   Wait until the Livy session in YARN for this consumer is initialized and running. You can determine this by locating the session in the YARN resource manager UI:



   The URL for the YARN resource manager is one of the following:

   - Secure cluster:

     ```
     https://<resource-manager-host>:8090
     ```

   - Non-secure cluster:

     ```
     http://<resource-manager-host>:8088
     ```

5. Run the producer notebook from Step 3 several times.

   The consumer notebook has a three-minute timeout that was set by the following line:

   ```
   ssc.awaitTerminationOrTimeout(3 * 60 * 1000)
   ```

   You will see output in the consumer notebook after the timeout has expired.

Accessing MapR Event Store For Apache Kafka in Zeppelin Using the Spark Interpreter
This section contains a MapR Event Store For Apache Kafka streaming example that you can run in your Apache Zeppelin notebook using the Spark interpreter. The Spark interpreter is available starting in the 1.1 release of the MapR Data Science Refinery.

Note: See MapR Data Science Refinery Support by MapR Core Version on page 5638 for limitations in version support when accessing MapR Event Store.

The example references a stream named `test_stream` created in the path `/streaming_test/test_stream`. The stream contains a topic called `test_topic`. You can use the following commands to create this stream and topic, but you cannot run them in Zeppelin; they must be run in a MapR cluster.

```
hadoop fs -mkdir /streaming_test
hadoop fs -chown <user>:<group> /streaming_test
sudo su - <user>
maprcli stream create -path /streaming_test/test_stream
maprcli stream topic create -path /streaming_test/test_stream -topic
test_topic
```

When the stream and topic are available, perform the following actions in your notebook:

1. Configure the Spark interpreter. Make sure to follow the steps described in the Spark Jobs on page 3066 section to allow Spark jobs to run in parallel.

2. Create a streaming consumer in your notebook using the `%spark` interpreter:

```
import org.apache.kafka.clients.consumer.ConsumerConfig

import org.apache.spark.SparkConf
import org.apache.spark.streaming.{Seconds, StreamingContext}
import org.apache.spark.streaming.kafka09.{ConsumerStrategies,
KafkaUtils, LocationStrategies}

val ssc = new StreamingContext(sc, Seconds(1))

val topicsSet = Set("/streaming_test/test_stream:test_topic")
val kafkaParams = Map[String, String](
  ConsumerConfig.BOOTSTRAP_SERVERS_CONFIG -> "localhost:9092",
  ConsumerConfig.GROUP_ID_CONFIG -> "none",
  ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG ->
    "org.apache.kafka.common.serialization.StringDeserializer",
  ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG ->
    "org.apache.kafka.common.serialization.StringDeserializer",
  ConsumerConfig.AUTO_OFFSET_RESET_CONFIG -> "latest",
  ConsumerConfig.ENABLE_AUTO_COMMIT_CONFIG -> "false"
)

val consumerStrategy =
      ConsumerStrategies.Subscribe[String, String](topicsSet,
kafkaParams)
val messages = KafkaUtils.createDirectStream[String, String](
      ssc,
      LocationStrategies.PreferConsistent,
      consumerStrategy)

val lines = messages.map(_.value())
val words = lines.flatMap(_.split(" "))
val wordCounts = words.map(x => (x, 1L)).reduceByKey(_ + _)
wordCounts.print()

ssc.start()
ssc.awaitTermination()
```

3. Create a streaming producer in another notebook, also with the `%spark` interpreter:

```
import java.util.Properties
import org.apache.kafka.clients.producer.{KafkaProducer, ProducerRecord}

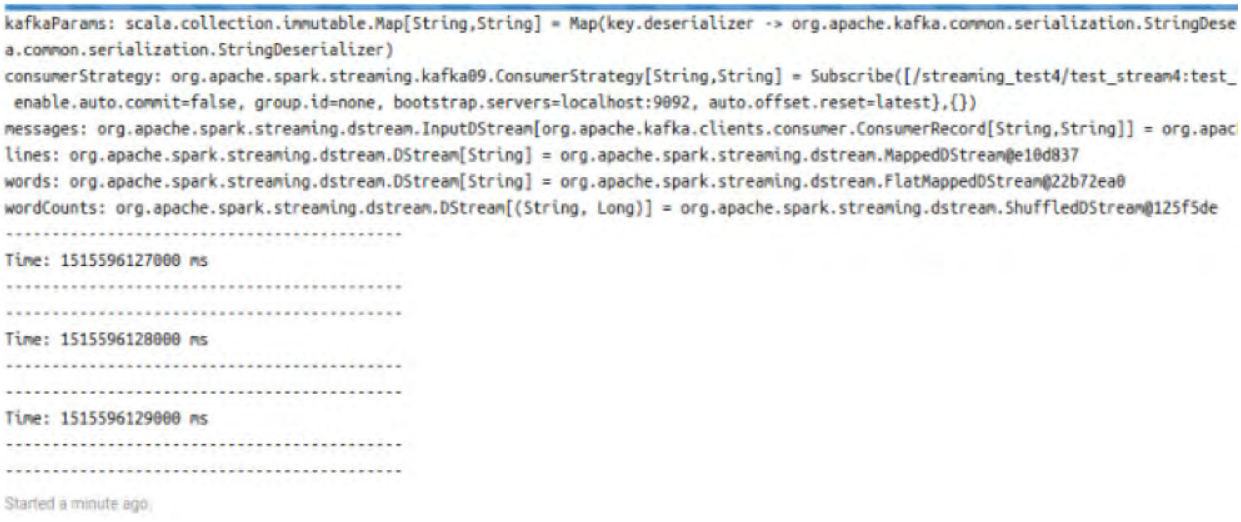
val props = new Properties()
props.put("bootstrap.servers", "localhost:9092")
props.put("acks", "all")
props.put("retries", "0")
props.put("batch.size", "16384")
props.put("linger.ms", "1")
props.put("buffer.memory", "33554432")
props.put("key.serializer",
"org.apache.kafka.common.serialization.StringSerializer")
props.put("value.serializer",
"org.apache.kafka.common.serialization.StringSerializer")

val producer = new KafkaProducer[String, String](props)

for (i <- 1 to 1000) {
    val message = new ProducerRecord[String, String]("/streaming_test/
test_stream:test_topic", i.toString(), i.toString())
    producer.send(message)
}
```

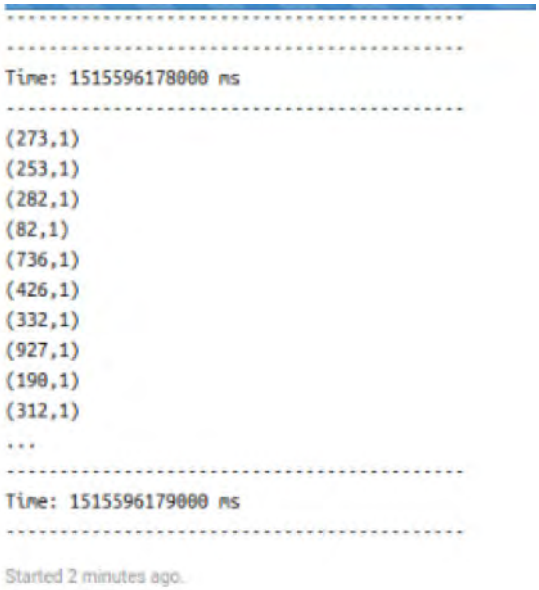
4. Start running the consumer notebook from Step 2.

   Wait until this consumer session is initialized and running. The following sample output in your notebook indicates the session is running:



5. Run the producer notebook from Step 3.

   The consumer notebook displays the following sample output after you run the producer:



Sharing Zeppelin Notebook Content
By default, Zeppelin stores notebooks in the local filesystem in your container. An alternative is to store them in MapR File System. This allows you to share the notebooks with other users.

To store notebooks in MapR File System, see Notebook Storage on page 3040.

For information about other available storage options, see https://zeppelin.apache.org/docs/0.8.0/.

Building your own MapR Data Science Refinery Docker Image

MapR provides a preconfigured and prepackaged Docker image for the MapR Data Science Refinery. Starting with the1.3 release, you can build your own custom Docker image.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

1. Determine the OS of the Docker image you want to build:

Operating System	URL
CentOS	https://package.mapr.com/labs/data-science-refinery/v1.4.0/redhat/
Ubuntu	https://package.mapr.com/labs/data-science-refinery/v1.4.0/ubuntu/

2. Download the Dockerfile corresponding to your OS:

```
wget https://package.mapr.com/labs/data-science-refinery/v1.4.0/redhat/Dockerfile
```

3. Build the Docker image:

```
docker build -t my_custom_dsr .
```

Note: The sample command specifies the current working directory (`.`) as the build path.

4. Confirm that the image appears in the following command's output:

```
docker image list
```


last-updated: Apr 11, 2022

To run and configure the image, see Zeppelin on MapR on page 3033.

MapR Data Fabric for Kubernetes

This section describes how to leverage the capabilities of the MapR Data Fabric for Kubernetes.

MapR Container Storage Interface (CSI) Storage Plugin Configuration

This section describes how to use and troubleshoot the MapR Container Storage Interface (CSI) Storage Plugin.

See MapR Container Storage Interface (CSI) Storage Plugin Overview on page 666 for more information.

Using the MapR Container Storage Interface (CSI) Storage Plugin

This section describes how to configure for static and dynamic provisioning and mounting using example configuration files.

For an overview of the MapR Container Storage Interface (CSI) Storage Plugin, see MapR Container Storage Interface (CSI) Storage Plugin Overview on page 666.

Before You Begin CSI Configuration

Before configuring the MapR Container Storage Interface (CSI) Storage Plugin, be sure to review the following notes about supported and unsupported features and parameters. For an overview of the MapR Container Storage Interface (CSI) Storage Plugin, see MapR Container Storage Interface (CSI) Storage Plugin Overview on page 666.

MapR Parameters for Static and Dynamic Provisioning

In dynamic provisioning, you can specify parameters for the MapR volume to be created. For a list of the parameters that you can use, see `volume create` on page 1931. Note these considerations for using the parameters:

- Volume attributes must be represented as a string (enclosed within quotations). Using an integer or boolean is not supported. In the following example, the `aetype` attribute will generate an error because the value (`1`) is not enclosed in quotations.

```
namePrefix: "pv"
mountPrefix: "/pv"
type: "rw"
advisoryquota: "100M"
aetype: 1
```

- The following parameters are ignored because they are redundant, and the CSI Driver configures these parameters automatically during volume creation:
  - `mount`
  - `quota*`
  - `createparent`
  - `path`
  - `name`

*Specifying `resources: requests: storage` in a PersistentVolumeClaim (PVC) makes it unnecessary to set the `quota` parameter. For an example, see Example: Statically Provisioning a Volume Using the MapR Container Storage Interface (CSI) Storage Plugin on page 3111.

last-updated: Apr 11, 2022

Kubernetes Access Modes

Kubernetes access modes control how a PersistentVolume (PV) is mounted on the host. Access modes can be specified on both PVs and PVCs. Only Volumes with a matching Access Mode will be bound to a PVC. MapR Container Storage Interface (CSI) Storage Plugin supports ROX (ReadOnlyMany), RWO (ReadWriteOnce) and RWX (ReadWriteMany) access modes for the PV and PVC spec. See Kubernetes CSI documentation for more information.

Reclaim Policy

The Kubernetes `reclaimPolicy` parameter controls what happens to a PersistentVolume if the corresponding PersistentVolumeClaim is deleted. The `Recycle` Reclaim Policy is not supported by Kubernetes CSI Drivers, so it cannot be used with the Kubernetes Interfaces for Data Fabric. You can specify the reclaim policy normally when you configure a persistent volume.

The following table shows the supported values for the reclaim policy:

Reclaim Policy Value	Description	Support
Delete (default value)	The PersistentVolume and the MapR volume are deleted when the user deletes the corresponding PersistentVolumeClaim.	Supported
Retain	The PersistentVolume and the MapR volume are not deleted when the user deletes the corresponding PersistentVolumeClaim.	Supported

For more information about the reclaim policy, see Change the Reclaim Policy of a PersistentVolume.

Kubernetes Mount Options

The Kubernetes `mountOptions` parameter is not supported for use with the MapR Container Storage Interface (CSI) Storage Plugin.

Configuring Static and Dynamic Provisioning Using MapR Container Storage Interface (CSI) Storage Plugin

This page summarizes the high-level steps for configuring the MapR Container Storage Interface (CSI) Storage Plugin after installation to provide static or dynamic provisioning. To learn more about static and dynamic provisioning, see Static and Dynamic Volume Provisioning Using MapR Container Storage Interface (CSI) Storage Plugin on page 667.

Static Provisioning

1. Create the ticket secret and deploy the secret in the Pod only if the volume is on a secure MapR cluster.

   See Configuring a Secret on page 3167 for more information.

2. Configure a PersistentVolume in your Pod spec or as part of a separate configuration file and provide information about the MapR volume.

   See Example: Statically Provisioning a Volume Using the MapR Container Storage Interface (CSI) Storage Plugin on page 3111, Persistent Volumes, and Example: Mounting a PersistentVolume for Static Provisioning on page 3113.

3. Configure a PersistentVolumeClaim in your Pod spec or as part of a separate configuration file.

   See PersistentVolumeClaims.

4. Run the Pod spec using `kubectl` commands.

   See Overview of kubectl.

last-updated: Apr 11, 2022

Dynamic Provisioning

1. Create the REST and ticket secrets and deploy the secrets in the Pod only if the volume is on a secure MapR cluster.
   See Configuring a Secret on page 3167 for more information.
2. Create a storage class in your Pod spec or in a separate configuration file.
   See Storage Classes and Example: Mounting a PersistentVolume for Dynamic Provisioning Using MapR Container Storage Interface (CSI) Storage Plugin on page 3120.
3. Configure a PersistentVolumeClaim in your Pod spec or in a separate configuration file.
   See Example: Mounting a PersistentVolume for Dynamic Provisioning Using MapR Container Storage Interface (CSI) Storage Plugin on page 3120.
4. Run the Pod spec using `kubectl` commands.
   See Overview of kubectl.

Configuring Static and Dynamic Provisioning for a Raw Block Volume

Raw block volumes are supported for both static and dynamic provisioning. To request a raw-block PersistentVolumeClaim, set `volumeMode: Block`. If not specified, `volumeMode` defaults to `Filesystem` in the PersistentVolumeClaimSpec. PersistentVolumes also have a `volumeMode` field in the PersistentVolumeSpec that is used for static provisioning. `Block`-type PVCs can only bind to `Block`-type PVs.

All the features supported on `Filesystem`-persistent volumes are supported on `Block` volumes. For example:

- Create and Delete Volumes
- Expand Volumes
- Clone Volumes
- Create and Delete Snapshot
- Snapshot Restore

Block volumes are supported only in single-node-writer access modes. At any given time, they can only be published once as read/write on a single node. For `Block` volumes, the CSI driver does not format the block device; it just binds the block device to the target path. The application pod can choose to format the block device to any required Linux file system, such as ext4, xfs, btrfs, and others.

Each block volume is stored in an HPE Ezmeral Data Fabric file. Statically provisioned block files are located at the path designated in the `volumePath` specified in the persistent volume definition. Dynamically provisioned block files are located at the path designated by the `mountPrefix` in the storage class. For fast, random block-write performance, these files should not be erasure coded (warm tiering) or tiered off to an objectstore (cold tiering).

Static Provisioning Example

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: test-blockpv
  namespace: test-csi
spec:
  accessModes:
  - ReadWriteOnce
  volumeMode: Block
```


last-updated: Apr 11, 2022

```
  persistentVolumeReclaimPolicy: Delete
  capacity:
    storage: 5G
  csi:
    driver: com.mapr.csi-kdf
    volumeHandle: test-simplepv
    volumeAttributes:
      volumePath: "/user/guest/myblockvolume"
      cluster: "clusterA"
      cldbHosts: "10.10.10.210"
      securityType: "secure"
      platinum: "true"
      capacityBytes: "5000000000"
```

Note: For the Loopback NFS CSI driver, change `driver` to `com.mapr.csi-nfskdf`.

Dynamic Provisioning Example

Note that no change in the StorageClass is required:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: test-secure-block-pvc
  namespace: test-csi
spec:
  storageClassName: test-secure-sc
  accessModes:
    - ReadWriteOnce
  volumeMode: Block
  resources:
    requests:
      storage: 5G
```

Pod Specification

In the pod specification, you must specify `volumeDevices` and `devicePath` for the block volume instead of `volumeMounts` and `mountPath`.

```
apiVersion: v1
kind: Pod
metadata:
  name: test-secure-block-pod
spec:
  containers:
    - name: fc-container
      image: fedora:26
      command: ["/bin/sh", "-c"]
      args: [ "tail -f /dev/null" ]
      volumeDevices:
        - name: data
          devicePath: /dev/xvda
  volumes:
    - name: data
      persistentVolumeClaim:
        claimName: test-secure-block-pvc
```

Configuring a Secret

Kubernetes Secrets enable you to inject sensitive data into a pod. For more information about Secrets, see Secrets.

last-updated: Apr 11, 2022

The examples in this section show how Secrets can be used in static and dynamic provisioning. Secrets are not by themselves secure. For more information about security and Secrets, see Security Properties. Specifically, it is important to turn on encryption at rest for Secrets. See Encrypting Secret Data at Rest.

During installation of the Driver, the Kubernetes token that was moved into the pod is written to the host node so that the plugin can query a Secret to pull the ticket for mounting. This Kubernetes token is sensitive and should be protected. The token is placed in `/var/run/secrets/kubernetes.io/serviceaccount`.

Here is an example of a configuration file for a Kubernetes Secret:

```
apiVersion: v1
kind: Secret
metadata:
  name: mapr-provisioner-secrets
  namespace: test-driver
type: Opaque
data:
  ...
```

The following table describes the fields in the sample Secret file. For more information, see Secrets in the Kubernetes documentation.

Parameter	Notes
`apiVersion`	The Kubernetes API version.
`kind`	The type of object being created.
`name`	A string to identify the Secret.
`type`	The type of Secret being created. For type `Opaque`, clients must treat these values as opaque and pass them unmodified back to the server.

REST Secrets

For dynamic provisioning, you must use a Secret to pass the user name and password of a MapR user to the provisioner. This user must have privileges to create and delete a MapR volume. The credentials allow the provisioner to make REST calls to the MapR webserver. Secrets are protected by the Kubernetes RBAC.

The following example shows a REST secret in the Secret file:

```
apiVersion: v1
kind: Secret
metadata:
  name: mapr-provisioner-secrets
  namespace: test-driver
type: Opaque
data:
 MAPR_CLUSTER_USER: cm9vdA==
  MAPR_CLUSTER_PASSWORD: bWFwcg==
```

The following table describes the REST secret fields in the REST Secret example.

Parameter	Notes
`MAPR_CLUSTER_USER`	The base64 representation of a MapR user that has the ability to create and delete MapR volumes. See Converting a String to Base64 on page 3169.

MAPR_CLUSTER_PASSWORD	The base64 representation of the password for the user defined by the MAPR_CLUSTER_USER parameter. See .Converting a String to Base64 on page 3169
MAPR_CLUSTER_TICKET	The base64 representation of the ticket contents generated on the data-fabric cluster using the maprlogin utility. For dynamic provisioning, with the latest CSI drivers, you can configure a data-fabric ticket to authenticate to the data-fabric webserver to make REST calls. This parameter is provided as a Beta feature.

Ticket Secrets

For static and dynamic provisioning, you must specify a Secret, which is the base64 representation of the ticket, to enable the POSIX client to communicate with a secure MapR cluster. The ticket for the POSIX client can be generated on the MapR cluster using the maprlogin on page 2130 utility.

The following example shows a ticket Secret:

```
apiVersion: v1
kind: Secret
metadata:
  name: mapr-ticket-secret
  namespace: mapr-examples
type: Opaque
data:
  CONTAINER_TICKET: CHANGETHIS!
```

The following table describes the CONTAINER_TICKET field in the ticket Secret example.

Parameter	Notes
CONTAINER_TICKET	Base64-encoded ticket value. See Converting a String to Base64 on page 3169.

To create the secret:

1. Run the following command to create the Secret file:

   ```
   kubectl create -f <secret-file-name>.yaml
   ```

2. Convert sensitive data, such as a user name and password, to a base64 representation.

   See Converting a String to Base64 on page 3169.

3. Add the base64 representation of sensitive data in the Secret file.

   For more information about the format of the Secret files, see REST Secrets on page 3108 and Ticket Secrets on page 3109 earlier in this section.

4. Deploy the secret on the pod by running the following command:

   ```
   kubectl apply -f <secret-file-name>.yaml
   ```

Converting a String to Base64

Sensitive data contained in a Secret must be represented in base64. Use these steps to convert such information to the base64 representation:

last-updated: Apr 11, 2022

For example, in Linux:

```
echo -n 'mapr' | base64
```

The output shows the base64 representation of the user name `mapr` is `bWFwcg==`.

MapR tickets include a cluster name followed by a base64-encoded string. It is not sufficient to insert the base64-encoded string into a Kubernetes Secret. You must convert *both* the cluster name and string into base64 representation and then insert the result into the Secret.

The following command shows how to convert a MapR ticket to base64 representation:

```
echo -n "cluster-name <base64-encoded ticket-value>" | base64
```

For example:

```
echo -n "cluster2 PuG0lpuPXuDxj9ERgKCTXOqsXYPTnqRJl6/
mlWJjdVKvE5r46QS2Bh9nC+I4Rcu0GtnWRUOtKBG9gp65bsZN9Kphnr/
Wp15z8D3O2go951CANes/
7QQllYVP7l2BOpGR6I1zIrC3XGwI8OQWT61qpsjSVZv8zO5oQ5GDYQTkPttI/yAk/
uJBES1ohCz38n9HgYALLvMALVsBPtUtG+cNGc1ktUDDMR2q1EgVzdJbuYsOuHnZX3LO3euKDGl4C
4MCmrv9DWiWJxwiZ1yZu69GbZJlXxqLOQBlkdMoTXk=" | base64

Y2x1c3RlcjIgUHVHMGxwdVBYdUR4ajlFUmdLQ1RYT3FzWFlQVG5xUkpsNi9tbFdKamRWS3ZFNXI0
NlFTMkJoOW5DK0k0UmN1MEd0bldSVU90S0JHOWdwNjVic1pOOUtwaG5yL1dwMTV6OEQzTzJnbzk1
MUNBTmVzLzdRUWxsWVZQN2wyQk9wR1I2STF6SXJDM1hHd0k4T1FXVDYxcXBzalNWWnY4ek81b1E1
R0RZUVRrUHR0SS95QWsvdUpCRVMxb2hDejM4bjlIZ1lBTEx2TUFMVnNCUHRVdEcrY05HYzFrdFVE
RE1SMnExRWdWemRKYnVZc091SG5aWDNMTzNldUtER2w0QzRNQ21ydjlEV2lXSnh3aVoxeVp1NjlH
YlpKbFh4cUxPUUJsa2RNb1RYaz0K
```

Note: Another method for converting values to base64 is to use an Internet tool such as https://www.base64encode.org to encode or decode data.

Best Practices for Using Tickets

When using secure MapR clusters with the Kubernetes Interfaces for Data Fabric, you must generate tickets for your containers. Here are some best practices:

- Create a different user for each container.
- To avoid frequent renewals, use long-lived user tickets or servicewithimpersonation tickets. If you refresh or update a ticket, you must restart your containers.
- If you use an impersonation ticket, it is CRITICAL that you use security contexts in the pod definitions to avoid a misbehaving container impersonating all user IDs. For restrictions that apply to the use of impersonation tickets, see How Impersonation Works on page 1478 and maprlogin on page 2130.
- Match the security context `runAsUser:` ID and `fsGroup:` group to the ID or group used to create the ticket.

Here is an example of a pod spec that specifies a security context:

```
apiVersion: v1
kind: Pod
metadata:
  name: test-secure
  namespace: mapr-examples
spec:
  securityContext:
    runAsUser: 1000
    fsGroup: 2000
```


last-updated: Apr 11, 2022

Example: Statically Provisioning a Volume Using the MapR Container Storage Interface (CSI) Storage Plugin

You can designate a volume for use with Kubernetes by specifying the volume parameters directly inside the PersistentVolume spec.

Suppose you want to get an application container up and running quickly in the MapR Data Platform. You already have a file-system path that you want to use for the application. You only need the data accessible to read. To make this work, you must do the following:

1. Generate a service ticket and set the `securityType` parameter in the PersistentVolume spec to `secure` if the volume to mount is on a secure cluster.

   See Generating a Service Ticket on page 1428 for more information. For example:

```
kind: PersistentVolume
metadata:
  name: pv-securepv-test
  namespace: test-csi
  spec:
    accessModes:
    …
    csi:
      …
      volumeAttributes:
        …
        securityType: "secure"
```

2. If the volume to mount is on a secure cluster, configure a Ticket Secret, and include the base64-encoded contents of the ticket file in the Ticket Secret.

   For more information, see Configuring a Secret on page 3167. The following table describes the properties of the Secret file:

Property	Notes
`apiVersion`	The Kubernetes API version.
`kind`	The type of object being created.
`name`	A string to identify the Secret.
`namespace`	The namespace in which the Secret runs.
`type`	The type of Secret being created. For type `Opaque`, clients must treat these values as opaque and pass them unmodified back to the server.
`CONTAINER_TICKET`	The contents of the ticket encoded in base64. If you specified `secure` for the `securityType`, you must provide the ticket. To encode the ticket, see Converting a String to Base64 on page 3169. You may remove the ticket if the cluster is not secure.

3. Set the `runAsUser` and the `fsGroup` parameters in the pod spec to the UID and GID of the user that created the ticket.

   For example:

```
apiVersion: v1
kind: Pod
metadata:
  name: test-pv1
  namespace: test-csi
spec:
  ...
  securityContext:
    runAsUser: 1000
```


last-updated: Apr 11, 2022

```
    fsGroup:2000
  ...
```

The following table lists the properties specified in the sample pod spec:

Parameter	Notes
`apiVersion`	The Kubernetes API version for the pod spec.
`kind`	The kind of object being created. For clarity, the example uses a naked pod. Generally, it is better to use a Deployment, DaemonSet, or StatefulSet for high availability (HA) and ease of upgrade.
`metadata: name`	The pod name.
`metadata: namespace`	The namespace in which the pod runs.
`numrpcthreads`	Sets the number of RPC threads for the data-fabric client. The default value is 1. The maximum value is 4. Use this option to increase throughput with FUSE basic or container licenses or with the Loopback NFS driver.
`securityContext: runAsUser`	The user ID to run the container under. This user ID must be the same as the user ID for which the ticket was generated.
`securityContext: fsGroup`	The group ID to run the container under. This group ID must be the same as the group ID of the user for which the ticket was generated.

4. Point the `volumePath` in the CSI driver setting to the desired path, and fill in the `cldbHosts` and `cluster` information.

   For the complete list of volume attributes, see `volume create` on page 1931; however, note that volume attributes like `mount`, `quota`, `createparent`, `path`, and `name` are ignored when provisioning a volume. For more information, see MapR Parameters for Static and Dynamic Provisioning on page 3104.

   For example:

   FUSE

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: test-simplepv
  namespace: test-csi
spec:
  accessModes:
  - ReadWriteOnce
  persistentVolumeReclaimPolicy: Delete
  capacity:
    storage: 5Gi
  csi:
    driver: com.mapr.csi-kdf
    volumeHandle: test-simplepv
    volumeAttributes:
      volumePath: "/"
      cluster: "clusterA"
      cldbHosts: "10.10.10.210"
      securityType: "secure"
      platinum: "true"
```

   Loopback NFS

```
apiVersion: v1
kind: PersistentVolume
metadata:
```

```
  name: test-simplepv
  namespace: test-csi
spec:
  accessModes:
  - ReadWriteOnce
  persistentVolumeReclaimPolicy: Delete
  capacity:
    storage: 5Gi
  csi:
    driver: com.mapr.csi-nfskdf
    volumeHandle: test-simplepv
    volumeAttributes:
      volumePath: "/"
      cluster: "clusterA"
      cldbHosts: "10.10.10.210"
      securityType: "secure"
```

The following table lists the properties shown in the sample PersistentVolume spec:

Parameter	Notes
`apiVersion`	The Kubernetes API version for the Pod spec.
`kind`	The kind of object being created.
`metadata: name`	The Pod name.
`metadata: namespace`	The namespace in which the Pod runs.
`accessModes`	How the PersistentVolume is mounted on the host. All modes work the same.  **Note:** The PV and PVC modes must be the same so that they can bind. For more information, see Access Modes.
`csi: driver`	The CSI Driver being used. Call it using one of these drivers: • FUSE driver: `com.mapr.csi-kdf` • Loopback NFS driver: `com.mapr.csi-nfskdf`
`csi: volumeHandle`	The existing volume name or unique volume name for static provisioning.
`volumePath`	The mount point within the filesystem. This parameter specifies an existing MapR path.
`cluster`	The cluster name.
`cldbHosts`	The DNS names or IP addresses of the CLDB hosts for the cluster. You must provide at least one CLDB host. For fault-tolerance, providing multiple CLDB hosts is recommended. To specify multiple hosts, separate each name or IP address by a space.
`securityType`	A parameter that indicates whether tickets are used or not used. If tickets are used, specify `secure`. Otherwise, specify `unsecure`.

Example: Mounting a PersistentVolume for Static Provisioning

The information on this page is valid for both FUSE POSIX and Loopback NFS plugins. Examples or tables that mention the FUSE POSIX driver (`com.mapr.csi-kdf`) are equally valid for the Loopback NFS driver (`com.mapr.csi-nfskdf`).

For static provisioning, configuring a PersistentVolume has some advantages over annotating Kubernetes volume information in a pod spec:

- The configuration file can be shared for use by multiple pod specs.

last-updated: Apr 11, 2022

- The configuration file enables the PersistentVolume to be mounted and available even when the pod spec that references it is removed.

For example, suppose a marketing volume exists in the secure MapR File System under the path `/Departments/Marketing`. An administrator wants to statically provision this volume and make it available to multiple users. It is critical that data access is as fast as possible. To make this work, the administrator must do the following:

1. Create a PersistentVolume (PV) (if you have already not statically provisioned a MapR volume as described in this example) and set the following volumeAttributes:
   - `accessMode` of the PV to `ReadWriteOnce`
   - `securityType` parameter to `secure` because the volume is on a secure MapR cluster
   - `volumePath` in the CSI driver setting to the desired path, and fill in the `cldbHosts` and `cluster` information
   - `platinum` parameter to use the POSIX platinum client or the `license` parameter to select from three POSIX clients

   For example:

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: test-simplepv
  namespace: test-csi
  labels:
    name: pv-simplepv-test
spec:
  accessModes:
  - ReadWriteOnce
  persistentVolumeReclaimPolicy: Delete
  capacity:
    storage: 5Gi
  csi:
    nodePublishSecretRef:
      name: "mapr-ticket-secret"
      namespace: "test-csi"
    driver: com.mapr.csi-kdf
    volumeHandle: test-simplepv
    volumeAttributes:
      volumePath: "/"
      cluster: "clusterA"
      cldbHosts: "10.10.102.96"
      securityType: "secure"
      platinum: "true"
```

   The preceding example specifies the high-performance Platinum POSIX license by including `platinum: "true"` in the `volumeAttributes`.

   If you have a Platinum FUSE POSIX license, Release 1.0.2 and later provide another way to control the POSIX client. Instead of specifying `platinum: "true"`, you can specify `license: "<license-name>"` and select one of three POSIX licenses (Basic, Container, or Platinum). Release 1.0.2 also adds support for a `startupConfig` line that lets you pass custom startup parameters to the FUSE process. The following example shows these options:

```
apiVersion: v1
kind: PersistentVolume
metadata:
```


last-updated: Apr 11, 2022

```
  name: test-simplepv
  namespace: test-csi
  labels:
    name: pv-simplepv-test
spec:
  accessModes:
  - ReadWriteMany
  persistentVolumeReclaimPolicy: Delete
  capacity:
    storage: 5Gi
  csi:
    nodePublishSecretRef:
      name: "mapr-ticket-secret"
      namespace: "test-csi"
    driver: com.mapr.csi-kdf
    volumeHandle: test-simplepv
    volumeAttributes:
      volumePath: "/"
      cluster: "clusterB"
      cldbHosts: "10.10.10.210"
      securityType: "secure"
      license: "container"
      startupConfig: "-o allow_other -o big_writes -o auto_unmount -o
async_dio -o max_background=24 -o auto_inval_data --disable_writeback"
```

The following table shows the properties defined in the sample PersistentVolume:

Parameter	Notes
`metadata: name`	The PersistentVolume name.
`metadata: namespace`	The namespace in which the PersistentVolume is stored.
`accessModes`	How the PersistentVolume is mounted on the host. All modes work the same. The example uses `ReadWriteOnce`. Note that `ReadOnlyMany` does not mount read-only. **Note:** The accessMode is not used to set the access mode bit on the volume. The accessMode of the PV and PVC should be the same so that they can bind. For more information, see Access Modes.
`persistentVolumeReclaimPolicy`	Specifies what happens to the volume when it is released by its claim. The `Retain` value keeps the PVC around for manual cleanup. `Delete` deletes the PV from Kubernetes. **Note:** If this volume was created using dynamic provisioning, `Delete` causes the underlying volume to be deleted. For more information, see Reclaiming.
`capacity`	Specifies how big the allocated storage should be. This value is not validated against the MapR quota or advisory quota. It is up to the person creating the PV to specify this value accurately.
`csi: nodePublishSecretRef`	The Ticket Secret for the CSI driver.
`nodePublishSecretRef:name`	The name of the Ticket Secret that contains the ticket to use when mounting to the MapR cluster. See Configuring a Secret on page 3167.
`nodePublishSecretRef:namespace`	The namespace that contains the Ticket Secret. Use the same namespace as the namespace used by the PersistentVolume.
`csi: driver`	The MapR CSI driver being used. Call it by specifying driver: `mapr.com/maprfs`.

last-updated: Apr 11, 2022

`volumeHandle`	The existing volume name or unique volume name for static provisioning.
`volumePath`	The mount point within the MapR File System. This parameter specifies an existing MapR path. For example, you can specify the root volume as `"/"`, providing access to the entire filesystem.
`cluster`	The MapR cluster name.
`cldbHosts`	The hostname or IP addresses of the CLDB hosts for the cluster. You must provide at least one CLDB host. For fault-tolerance, providing multiple CLDB hosts is recommended. To specify multiple hosts, separate each name or IP address by a space.
`securityType`	A parameter that indicates whether MapR tickets are used or not used. If MapR tickets are used, specify `secure`. Otherwise, specify `unsecure`.
`platinum`	If set to `platinum: "true"`, the POSIX client uses the platinum driver for better performance. Note that the platinum driver consumes more host resources and MapR Platinum licenses.
`license` (FUSE POSIX)	Release 1.0.2 and later support the `license: "<license-name>"` parameter in addition to the `platinum: "<true | false>"` parameter for the FUSE POSIX plugin. The `license: "<license-name>"` parameter can have one of three values that control the number of host resources that are consumed: • `"container"` for the Container driver (one binary, 8 threads) • `"basic"` for the Basic driver (one binary, 64 threads) • `"platinum"` for the Platinum driver (multiple binaries, each running 64 threads) Note the following considerations for using the `license: "<license-name>"` parameter: • To use the `license: "<license-name>"` parameter, you must have a Platinum license. • If you specify both the `platinum: "<true | false>"` parameter and the `license: "<license-name>"` parameter, the `platinum: "<true | false>"` parameter overrides the `license: "<license-name>"` parameter. • If neither the `platinum: "<true | false>"` nor the `license:` parameter is specified, the `container` driver is implemented.

`startupConfig` (FUSE POSIX)	Release 1.0.2 and later support specifying the `startupConfig` line. The `startupConfig` line allows you to specify FUSE configuration parameters that are passed to the `fuse.conf` file. For the parameters that can be passed, see Configuring the MapR FUSE-Based POSIX Client on page 1240.
	If no `startupConfig` line is specified, these default startup settings are used:
	`"-o allow_other -o big_writes -o auto_unmount"`
	The default settings allow other users to access the mount point, enable writes larger than 4 KB, and automatically unmount the filesystem when the process is terminated.
	The following example includes the three default settings and adds some additional settings (shown in **bold**):
	`startupConfig: "-o allow_other -o big_writes -o auto_unmount` **`-o async_dio -o max_background=24 -o auto_inval_data --disable_writeback`**`"`
	The additional settings enable asynchronous direct I/O, set the maximum number of asynchronous requests to 24, automatically invalidate the kernel FUSE cache for any data change that causes a change in the files, and disable the writeback cache.

last-updated: Apr 11, 2022

`startupConfig` (Loopback NFS)	The `startupConfig` line allows you to specify configuration parameters that are passed to the `nfsserver.conf` file. For the parameters that can be passed, see nfsserver.conf on page 2207. The `startupConfig` values supported for Loopback NFS are all the configs supported in the `nfsserver.conf` file. Values must be separated by a space. If no `startupConfig` line is specified, these default startup settings are used: `startupConfig: "NFS_HEAPSIZE=1024 DrCacheSize=1024000"`
`trackMemory`	Enables memory profiling to debug memory leaks in the FUSE or Loopback NFS process. To be enabled after direction from the DF support team. The default value is `false`.
`logLevel`	Sets the log level to one of the following values: `error`, `warn`, `info`, or `debug`. For the FUSE POSIX driver (`com.mapr.csi-kdf`), the default value is `error`. For the Loopback NFS driver (`com.mapr.csi-nfskdf`), the default value is `info`.
`retainLogs`	Retains the logs for the pod on the host machine. The default value is `false`.

See Example: Statically Provisioning a Volume Using the MapR Container Storage Interface (CSI) Storage Plugin on page 3111 for more information.

2. Create a PersistentVolumeClaim (PVC) spec and set the `accessMode` of the PVC to `ReadWriteOnce`.

   For example:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: test-simplepvc
  namespace: test-csi
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5G
```

   The following table shows the properties used in the sample PersistentVolumeClaim:

Parameter	Notes
`metadata: name`	The PersistentVolumeClaim name.
`metadata: namespace`	The namespace in which the PersistentVolumeClaim is configured.
`accessMode`	How the requested PersistentVolume is mounted on the host. All modes work the same. The example uses `ReadWriteOnce`. Note that `ReadOnlyMany` does not mount read-only.  **Note:** The PV and PVC modes should be the same so that they can bind. For more information, see Access Modes.

3. Generate a service ticket, and create and deploy a ticket secret on the pod (if you have already not done it as described in steps 1 and 2 of this example).

   See `maprlogin on page 2130` for information on generating a ticket and Configuring a Secret on page 3167 for information on creating and deploying a ticket secret.

4. Create the pod spec and set the `runAsUser` and the `fsGroup` parameters to the UID and GID of the user that created the ticket.

last-updated: Apr 11, 2022

For example:

```
apiVersion: v1
kind: Pod
metadata:
  name: test-pv
  namespace: test-csi
spec:
  securityContext:
    runAsUser: 1000
    fsGroup: 2000
  containers:
  - name: busybox
    image: busybox
    args:
    - sleep
    - "1000000"
    resources:
      requests:
        memory: "2Gi"
        cpu: "500m"
    volumeMounts:
    - mountPath: /mapr
      name: maprflex
  volumes:
    - name: maprflex
      persistentVolumeClaim:
        claimName: test-simplepvc
```

The following table shows the properties defined in the sample pod spec:

Parameter	Notes
`apiVersion`	The Kubernetes API version for the pod spec.
`kind`	The kind of object being created. The example uses a naked pod for clarity. Generally, it is better to use a Deployment, DaemonSet, or StatefulSet for high availability and ease of upgrade.
`metadata: name`	The pod name.
`metadata: namespace`	The namespace in which the pod runs.
`numrpcthreads`	Sets the number of RPC threads for the data-fabric client. The default value is 1. The maximum value is 4. Use this option to increase throughput with FUSE basic or container licenses or with the Loopback NFS driver.
`securityContext: runAsUser`	The user ID to run the container under. This user ID must be the same as the user ID for which the ticket was generated.
`securityContext: fsGroup`	The group ID to run the container under. This group ID must be the same as the group ID of the user for which the ticket was generated.
`volumeMounts: mountPath`	A directory inside the container that is designated as the mount path.
`volumeMounts: name`	A name that you assign to the Kubernetes `volumeMounts` resource. Matches with `Volumes: name`.
`Volumes: name`	A string to identify the name of the Kubernetes `volumes` resource. The value should match `volumeMounts: name`.
`persistentVolumeClaim: claimName`	The name of the PersistentVolumeClaim (PVC). For more information, see PersistentVolumeClaims.

last-updated: Apr 11, 2022

5. Deploy the `.yaml` file on the pod by running the following command:

```
kubectl apply -f <filename>.yaml
```

For each Pod mount request, the POSIX client starts with the pod's hostname and new generated hostid, which is tracked on the MapR cluster. You can run the `node list` on page 1705 command on the MapR cluster to determine the number of POSIX clients. For example:

FUSE POSIX

```
# maprcli node list -clientsonly true
clienttype       clienthealth hostname
ip                                        lasthb  id
posixclientbasic Inactive     4f3d34fe-2007-11e9-8980-0cc47ab39644
10.10.102.94,172.17.0.1,192.168.28.0     11225   7407394893618656436
posixclientbasic Inactive     7906d011-200f-11e9-84c0-0cc47ab39644
10.10.102.94,172.17.0.1,192.168.28.0     8174    7544602061076655421
posixclientbasic Inactive     9ed61912-2004-11e9-8980-0cc47ab39644
10.10.102.92,172.17.0.1,192.168.184.128  11224   2540810767207593086
posixclientbasic Inactive     c35ab639-2010-11e9-84c0-0cc47ab39644
10.10.102.94,172.17.0.1,192.168.28.0     7568    7947067275504513691
posixclientbasic Active       e5dc10e8-2012-11e9-84c0-0cc47ab39644
10.10.102.94,172.17.0.1,192.168.28.0     18      5849529086453778130
```

Loopback NFS

```
# maprcli node list -clientsonly true
clienttype    clienthealth  hostname
ip                              lasthb  id
LOOPBACK_NFS  Active        3ae5bb79-0aa1-431d-a17b-2cf0ef692060
10.163.160.104,192.168.252.65   1       3740102597316282880
LOOPBACK_NFS  Active        8c096a3c-0424-466a-8eda-6a61999ac3e4
10.163.160.103,192.168.19.192   1       6892565781040807680
LOOPBACK_NFS  Active        ae92fe4b-a3c9-4cb3-8858-c688dd6e0bdc
10.163.160.103,192.168.19.192   1       1038944668644089888
LOOPBACK_NFS  Active        fe855a47-bf66-4b72-8f28-c713b5ec4004
10.163.160.105,192.168.153.128  1       5958455784535826944
```

Full example, which includes PV, PVC, and pod configuration

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: test-simplepv
  namespace: test-csi
  labels:
    name: pv-simplepv-test
spec:
  accessModes:
  - ReadWriteOnce
  persistentVolumeReclaimPolicy: Delete
  capacity:
    storage: 5Gi
  csi:
    nodePublishSecretRef:
      name: "mapr-ticket-secret"
      namespace: "test-csi"
    driver: com.mapr.csi-kdf
    volumeHandle: test-simplepv
    volumeAttributes:
```


last-updated: Apr 11, 2022

```
      volumePath: "/"
      cluster: "clusterA"
      cldbHosts: "10.10.102.96"
      securityType: "secure"
      platinum: "true"
---
apiVersion: v1
kind: Pod
metadata:
  name: test-pv
  namespace: test-csi
spec:
  securityContext:
    runAsUser: 1000
    fsGroup: 2000
  containers:
  - name: busybox
    image: busybox
    args:
    - sleep
    - "1000000"
    resources:
      requests:
        memory: "2Gi"
        cpu: "500m"
    volumeMounts:
    - mountPath: /mapr
      name: maprflex
  volumes:
    - name: maprflex
      persistentVolumeClaim:
        claimName: test-simplepvc
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: test-simplepvc
  namespace: test-csi
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5G
```

Example: Mounting a PersistentVolume for Dynamic Provisioning Using MapR Container Storage Interface (CSI) Storage Plugin

This example also uses a PersistentVolume. However, unlike the previous example, when you use the dynamic provisioner, you do not need to create a PersistentVolume manually. The PersistentVolume is created automatically based on the parameters specified in the referenced StorageClass.

Dynamic provisioning is useful in cases where you do not want MapR and Kubernetes cluster administrators to create storage manually to store the pod storage state.

The following example uses a PersistentVolumeClaim that references a Storage Class. In this example, a Kubernetes administrator has created a storage class called `test-secure-sc` for pod creators to use when they want to create persistent storage for their pods. In this example, it is important for the created pod storage to survive the deletion of a pod.

The information on this page is valid for both FUSE POSIX and Loopback NFS plugins. Examples or tables that mention the FUSE POSIX provisioner (`com.mapr.csi-kdf`) are equally valid for the Loopback NFS provisioner (`com.mapr.csi-nfskdf`).

last-updated: Apr 11, 2022

To dynamically provision a volume, you must do the following:

1. Generate a user ticket, and create and deploy a ticket secret on the pod. See:
   - Best Practices for Using Tickets on page 3170 to select the right ticket
   - maprlogin on page 2130 for information about generating a ticket
   - Configuring a Secret on page 3167 for information about creating and deploying a ticket secret
2. Create the REST secret and deploy the secret on the pod.
   See Configuring a Secret on page 3167 for information about creating and deploying a ticket secret.
3. Create a StorageClass similar to the following:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: test-secure-sc
  namespace: test-csi
provisioner: com.mapr.csi-kdf
allowVolumeExpansion: true
reclaimPolicy: Delete
parameters:
    csiProvisionerSecretName: "mapr-provisioner-secrets"
    csiProvisionerSecretNamespace: "test-csi"
    csiNodePublishSecretName: "mapr-ticket-secret"
    csiNodePublishSecretNamespace: "test-csi"
    restServers: "10.10.10.210:8443"
    cldbHosts: "10.10.10.210:7222"
    cluster: "clusterA"
    securityType: "secure"
    namePrefix: "csi-pv"
    mountPrefix: "/csi"
    advisoryquota: "100M"
    trackMemory: "false"
    logLevel: "error"
    retainLogs: "false"
```

   For more information, see Storage Classes. The following table shows the properties defined in the sample StorageClass:

Property	Description
`apiVersion`	The Kubernetes API version for the StorageClass spec.
`kind`	The kind of object being created. This is a StorageClass.
`metadata: name`	The name of the StorageClass. Administrators should specify the name carefully because it will be used by pod authors to help select the right StorageClass for their needs.
`metadata: namespace`	The namespace in which the StorageClass runs. This namespace can be different from the namespace used by the PVC and pod, since the StorageClass namespace can be a cross-namespace resource.
`provisioner`	The provisioner being used. For the FUSE POSIX provisioner, specify `com.mapr.csi-kdf`. For the Loopback NFS provisioner, specify `com.mapr.csi-nfskdf`.
`csiNodePublishSecretName`	The name of the Secret that contains the ticket to use when mounting to the MapR cluster. See Configuring a Secret on page 3167.

last-updated: Apr 11, 2022

Property	Description
`csiNodePublishSecretNamespace`	The namespace that contains the Secret. Use the same namespace as the namespace used by the pod.
`csiProvisionerSecretName` (deprecated) `csi.storage.k8s.io/provisioner-secret-name`	The name of the Kubernetes Secret that is used to store MapR administrative credentials (user, password, and ticket information for the MapR webserver). To use the provisioner, you must configure a Secret. See Configuring a Secret on page 3167.
`csiProvisionerSecretNamespace` (deprecated) `csi.storage.k8s.io/provisioner-secret-namespace`	The namespace for the Secret containing the MapR administrative credentials (user name and password information for a MapR user that has the privileges to create volumes). This namespace can be different from the namespace used by the pod, since a pod author or namespace admin might not be trusted to create administration Secrets for the MapR cluster.
`restServers`	A space-separated list of MapR webservers. Specify the hostname or IP address and port number of each REST server for the cluster. For fault tolerance, providing multiple REST server hosts is recommended.
`cldbHosts`	The hostname or IP addresses of the CLDB hosts for the MapR cluster. You must provide at least one CLDB host. For fault-tolerance, providing multiple CLDB hosts is recommended. To specify multiple hosts, separate each name or IP address by a space.
`cluster`	The MapR cluster name.
`securityType`	A parameter that indicates whether MapR tickets are used or not used. If MapR tickets are used, specify `secure`. Otherwise, specify `unsecure`.
`namePrefix`	A prefix for the MapR volume to be created. For example, if you specify PV as the `namePrefix`, the first dynamically created volume might be named `PV.bevefsescr`. The provisioner generates random names using lower-case letters. If you do not specify a prefix, the provisioner uses `maprprovisioner` as a prefix.
`mountPrefix`	The parent path of the mount in the MapR filesystem. If you do not specify a mount prefix, the provisioner mounts your volume under the MapR root. **Note:** User provisioning a volume under this `mountPrefix` requires read-write permissions to mount the newly created volume; otherwise, the volume provision will fail.
`advisoryquota`	The advisory storage quota for the volume. The `advisoryquota` is one of the MapR parameters that you can specify for dynamic provisioning. For more information, see Before You Begin on page 3151.
`trackMemory`	Enables memory profiling to debug memory leaks in the FUSE or Loopback NFS process. To be enabled after direction from the DF support team. The default value is `false`.
`logLevel`	Sets the log level to one of the following values: `error`, `warn`, `info`, or `debug`. For the FUSE POSIX driver (`com.mapr.csi-kdf`), the default value is `error`. For the Loopback NFS driver (`com.mapr.csi-nfskdf`), the default value is `info`.
`retainLogs`	Retains the logs for the pod on the host machine. The default value is `false`.

4. Configure a PersistentVolumeClaim similar to the following:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: test-secure-pvc
  namespace: test-csi
```

```
spec:
  storageClassName: test-secure-sc
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5G
```

The following table shows the properties defined in the sample PersistentVolumeClaim:

Property	Description
`apiVersion`	The Kubernetes API version for the pod spec.
`kind`	The kind of object being created. This is a PersistentVolumeClaim (PVC).
`metadata: name`	The PVC name.
`metadata: namespace`	The namespace in which the PVC runs. This should be the same namespace used by the pod.
`storageClassName`	The name of the storage class requested by the PersistentVolumeClaim. For more information, see Dynamic Provisioning and Storage Classes.
`accessModes`	How the PersistentVolume is mounted on the host. For more information, see Access Modes.
`requests: storage`	The storage resources being requested, or that were requested and have been allocated. The pod author can use this parameter to specify how much quota is needed for the MapR volume. For the units, see Resource Model.

5. Create the pod spec similar to the following:

```
apiVersion: v1
kind: Pod
metadata:
  name: test-secure-pod
  namespace: test-csi
spec:
  containers:
  - name: busybox
    image: busybox
    args:
    - sleep
    - "1000000"
    resources:
      requests:
        memory: "2Gi"
        cpu: "500m"
    volumeMounts:
    - mountPath: /mapr
      name: maprflex
  volumes:
    - name: maprflex
      persistentVolumeClaim:
        claimName: test-secure-pvc
```

The following table shows the properties defined in the sample pod spec:

Property	Description
`apiVersion`	The Kubernetes API version for the pod spec.

last-updated: Apr 11, 2022

Property	Description
`kind`	The kind of object being created. For clarity, this example uses a naked Pod. Generally, it is better to use a Deployment, DaemonSet, or StatefulSet for high availability and ease of upgrade.
`metadata: name`	The pod name.
`metadata: namespace`	The namespace in which the pod runs. It should be the same namespace in which the PVC runs.
`numrpcthreads`	Sets the number of RPC threads for the data-fabric client. The default value is 1. The maximum value is 4. Use this option to increase throughput with FUSE basic or container licenses or with the Loopback NFS driver.
`volumeMounts: mountPath`	A directory inside the container that is designated as the mount path.
`volumeMounts: name`	A name that you assign to the Kubernetes `volumeMounts` resource. The value should match `Volumes: name`.
`Volumes: name`	A string to identify the name of the Kubernetes `volumes` resource. The value should match `volumeMounts: name`.
`persistentVolumeClaim: claimName`	The name of the PersistentVolumeClaim (PVC). For more information, see PersistentVolumeClaims.

6. Deploy the `.yaml` file on the pod by running the following command:

```
kubectl apply -f <filename>.yaml
```

For each pod mount request, the POSIX client starts with the pod's hostname and new generated hostid, which is tracked on the MapR cluster. You can run the `node list on page 1705` command on the cluster to determine the number of POSIX clients. For example:

FUSE POSIX

```
# maprcli node list -clientsonly true
clienttype clienthealth hostname ip lasthb id
posixclientbasic Inactive 4f3d34fe-2007-11e9-8980-0cc47ab39644
10.10.102.94,172.17.0.1,192.168.28.0 11225 7407394893618656436
posixclientbasic Inactive 7906d011-200f-11e9-84c0-0cc47ab39644
10.10.102.94,172.17.0.1,192.168.28.0 8174 7544602061076655421
posixclientbasic Inactive 9ed61912-2004-11e9-8980-0cc47ab39644
10.10.102.92,172.17.0.1,192.168.184.128 11224 2540810767207593086
posixclientbasic Inactive c35ab639-2010-11e9-84c0-0cc47ab39644
10.10.102.94,172.17.0.1,192.168.28.0 7568 7947067275504513691
posixclientbasic Active e5dc10e8-2012-11e9-84c0-0cc47ab39644
10.10.102.94,172.17.0.1,192.168.28.0 18 5849529086453778130
```

Loopback NFS

```
# maprcli node list -clientsonly true
clienttype    clienthealth  hostname
ip                              lasthb  id
LOOPBACK_NFS  Active        3ae5bb79-0aa1-431d-a17b-2cf0ef692060
10.163.160.104,192.168.252.65   1       3740102597316282880
LOOPBACK_NFS  Active        8c096a3c-0424-466a-8eda-6a61999ac3e4
10.163.160.103,192.168.19.192   1       6892565781040807680
LOOPBACK_NFS  Active        ae92fe4b-a3c9-4cb3-8858-c688dd6e0bdc
10.163.160.103,192.168.19.192   1       1038944668644089888
LOOPBACK_NFS  Active        fe855a47-bf66-4b72-8f28-c713b5ec4004
10.163.160.105,192.168.153.128  1       5958455784535826944
```


last-updated: Apr 11, 2022

Full example, which includes PV, PVC, and Pod configuration

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: test-secure-sc
  namespace: test-csi
provisioner: com.mapr.csi-kdf
parameters:
    csiProvisionerSecretName: "mapr-provisioner-secrets"
    csiProvisionerSecretNamespace: "test-csi"
    csiNodePublishSecretName: "mapr-ticket-secret"
    csiNodePublishSecretNamespace: "test-csi"
    restServers: "10.10.10.210"
    cldbHosts: "10.10.10.210"
    cluster: "clusterA"
    securityType: "secure"
    namePrefix: "csi-pv"
    mountPrefix: "/csi"
    advisoryquota: "100M"
--
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: test-secure-pvc
  namespace: test-csi
spec:
  storageClassName: test-secure-sc
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5G
--
apiVersion: v1
kind: Pod
metadata:
  name: test-secure-pod
  namespace: test-csi
spec:
  containers:
  - name: busybox
    image: busybox
    args:
    - sleep
    - "1000000"
    resources:
      requests:
        memory: "2Gi"
        cpu: "500m"
    volumeMounts:
    - mountPath: /mapr
      name: maprflex
  volumes:
    - name: maprflex
      persistentVolumeClaim:
        claimName: test-secure-pvc
```

Example: Volume Cloning for Dynamic Provisioning

last-updated: Apr 11, 2022

You can clone a volume from an existing volume by configuring a PersistentVolumeClaim that specifies the volume PVC as the data source. In the following example, the PVC named `testcsi-secure-pvc` serves as the data source for creating a new volume named `testcsi-secure-pvc-clone`:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
 name: testcsi-secure-pvc-clone
 namespace: test-csi
spec:
 storageClassName: testcsi-secure-sc
 accessModes:
    - ReadWriteOnce
 resources:
 requests:
 storage: 10G
 dataSource:
 kind: PersistentVolumeClaim
 name: testcsi-secure-pvc
```

When cloning extra large volumes (volumes measuring hundreds of GB), you might experience timeouts or a failure to clone the volume. To prevent timeouts with extra large volumes, increase the retry timeout setting for the `csi-provisioner` sidecar container. See the `--timeout` argument in the latest `csi-maprkdf-<version>.yaml`.

For more information, see CSI Volume Cloning.

Example: Volume Expansion for Dynamic Provisioning Using MapR Container Storage Interface (CSI) Storage Plugin

The following versions of the MapR Container Storage Interface (CSI) Storage Plugin support a volume expansion feature for dynamically provisioned volumes:

- FUSE POSIX plugin versions 1.1.0 and later
- Loopback NFS plugin versions 1.0.0 and later

Volume expansion means you can increase the storage quota of volumes created by the CSI driver. Note that the `StorageClass` must have `allowVolumeExpansion` set to `true` for volume expansion to succeed. For more information about volume expanssion, see Expanding Persistent Volume Claims.

To use volume expansion, increase the storage value and reapply the PersistentVolumeClaim configuration. For example, in the following PVC configuration, to increase the storage quota for the volume from 5G to 10G, change `storage: 5G` to `storage: 10G`, and run the `kubectl apply -f <path_to_pvc>.yaml` command:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: test-secure-pvc
  namespace: test-csi
spec:
  storageClassName: test-secure-sc
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5G
```

Verifying Creation of a Kubernetes PersistentVolumeClaim and Persistent Volume

Once the pod spec is installed, you can verify the status of a PersistentVolumeClaim and/or a PersistentVolume by using the `kubectl` command. For example:

1. Run the Kubernetes `get` command to verify the status of the PersistentVolumeClaim:

   Note: The information on this page is valid for both FUSE POSIX and Loopback NFS plugins. Examples or tables that mention the FUSE POSIX driver (`com.mapr.csi-kdf`) are equally valid for the Loopback NFS driver (`com.mapr.csi-nfskdf`).

Static Provisioning

```
# kubectl describe pvc -n test-csi
Name:          mapr-secure-claim
Namespace:     test-csi
StorageClass:
Status:        Bound
Volume:        pv-securepv-test
Labels:        <none>
Annotations:
kubectl.kubernetes.io/
last-applied-configuration:

{"apiVersion":"v1","kind":"Persisten
tVolumeClaim","metadata":
{"annotations":
{},"name":"mapr-secure-claim","names
pace":"test-csi"},"spec":{...
               pv.kubernetes.io/
bind-completed: yes
               pv.kubernetes.io/
bound-by-controller: yes
Finalizers:    [kubernetes.io/
pvc-protection]
Capacity:      5Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Events:        <none>
Mounted By:    test-secure-pv
```

```
# kubectl get pvc -n test-csi -o
yaml
apiVersion: v1
items:
- apiVersion: v1
  kind: PersistentVolumeClaim
  metadata:
    annotations:
      kubectl.kubernetes.io/
last-applied-configuration: |

{"apiVersion":"v1","kind":"Persisten
tVolumeClaim","metadata":
{"annotations":
{},"name":"mapr-secure-claim","names
pace":"test-csi"},"spec":
{"accessModes":
["ReadWriteOnce"],"resources":
{"requests":{"storage":"5G"}}}}
      pv.kubernetes.io/
bind-completed: "yes"
      pv.kubernetes.io/
bound-by-controller: "yes"
```

```
    creationTimestamp:
"2019-01-24T18:19:42Z"
    finalizers:
    - kubernetes.io/pvc-protection
    name: mapr-secure-claim
    namespace: test-csi
    resourceVersion: "1024139"
    selfLink: /api/v1/namespaces/
test-csi/persistentvolumeclaims/
mapr-secure-claim
    uid:
9eddbddb-2004-11e9-8980-0cc47ab39644
  spec:
    accessModes:
    - ReadWriteOnce
    dataSource: null
    resources:
      requests:
        storage: 5G
    volumeMode: Filesystem
    volumeName: pv-securepv-test
  status:
    accessModes:
    - ReadWriteOnce
    capacity:
      storage: 5Gi
    phase: Bound
kind: List
metadata:
  resourceVersion: ""
  selfLink: ""
```

Dynamic Provisioning

```
# kubectl describe pvc
test-secure-pvc -n test-csi
Name:          test-secure-pvc
Namespace:     test-csi
StorageClass:  test-secure-sc
Status:        Bound
Volume:
mapr-pv-4f494906-2007-11e9-8980-0cc4
7ab39644
Labels:        <none>
Annotations:
kubectl.kubernetes.io/
last-applied-configuration:

{"apiVersion":"v1","kind":"Persisten
tVolumeClaim","metadata":
{"annotations":
{},"name":"test-secure-pvc","namespa
ce":"test-csi"},"spec":{"a...
               pv.kubernetes.io/
bind-completed: yes
               pv.kubernetes.io/
bound-by-controller: yes

volume.beta.kubernetes.io/
storage-provisioner:
com.mapr.csi-kdf
Finalizers:    [kubernetes.io/
pvc-protection]
```

```
Capacity:      5Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Events:
  Type       Reason
Age
From

    Message
  ----       ------
 ----   ----

           -------
  Normal     ExternalProvisioning
4m43s
persistentvolume-controller

    waiting for a volume to be
created, either by external
provisioner "com.mapr.csi-kdf" or
manually created by system
administrator
  Normal     Provisioning
4m43s
com.mapr.csi-kdf_csi-controller-kd
f-0_087074d9-2004-11e9-be6e-32d95d1d
c62d  External provisioner is
provisioning volume for claim
"test-csi/test-secure-pvc"
  Normal     ProvisioningSucceeded
4m40s
com.mapr.csi-kdf_csi-controller-kd
f-0_087074d9-2004-11e9-be6e-32d95d1d
c62d  Successfully provisioned
volume
mapr-pv-4f494906-2007-11e9-8980-0cc4
7ab39644
Mounted By:  test-secure-pod
```

```
# kubectl get pvc
test-secure-pvc -n test-csi -o yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    kubectl.kubernetes.io/
last-applied-configuration: |

{"apiVersion":"v1","kind":"Persisten
tVolumeClaim","metadata":
{"annotations":
{},"name":"test-secure-pvc","namespa
ce":"test-csi"},"spec":
{"accessModes":
["ReadWriteOnce"],"resources":
{"requests":
{"storage":"5G"}},"storageClassName"
:"test-secure-sc"}}
    pv.kubernetes.io/
bind-completed: "yes"
    pv.kubernetes.io/
bound-by-controller: "yes"
```

```
    volume.beta.kubernetes.io/
storage-provisioner:
com.mapr.csi-kdf
  creationTimestamp:
"2019-01-24T18:38:57Z"
  finalizers:
  - kubernetes.io/pvc-protection
  name: test-secure-pvc
  namespace: test-csi
  resourceVersion: "1025704"
  selfLink: /api/v1/namespaces/
test-csi/persistentvolumeclaims/
test-secure-pvc
  uid:
4f494906-2007-11e9-8980-0cc47ab39644
spec:
  accessModes:
  - ReadWriteOnce
  dataSource: null
  resources:
    requests:
      storage: 5G
  storageClassName: test-secure-sc
  volumeMode: Filesystem
  volumeName:
mapr-pv-4f494906-2007-11e9-8980-0cc4
7ab39644
status:
  accessModes:
  - ReadWriteOnce
  capacity:
    storage: 5Gi
  phase: Bound
```

2. Run the Kubernetes `describe` command to determine the status of the PersistentVolume:

Static Provisioning

```
# kubectl describe pv
pv-securepv-test -n test-csi
Name:            pv-securepv-test
Labels:
name=pv-securepv-test
Annotations:
kubectl.kubernetes.io/
last-applied-configuration:

{"apiVersion":"v1","kind":"Persisten
tVolume","metadata":{"annotations":
{},"labels":
{"name":"pv-securepv-test"},"name":"
pv-securepv-test"},...
                 pv.kubernetes.io/
bound-by-controller: yes
Finalizers:      [kubernetes.io/
pv-protection]
StorageClass:
Status:          Bound
Claim:           test-csi/
mapr-secure-claim
Reclaim Policy:  Delete
Access Modes:    RWO
```

```
VolumeMode:      Filesystem
Capacity:        5Gi
Node Affinity:   <none>
Message:
Source:
    Type:              CSI (a
Container Storage Interface (CSI)
volume source)
    Driver:
com.mapr.csi-kdf
    VolumeHandle:      test-id
    ReadOnly:          false
    VolumeAttributes:
cldbHosts=10.10.10.210

cluster=clusterA

securityType=secure

volumePath=/volume1
Events:                <none>
```

```
# kubectl get pv
pv-securepv-test -n test-csi -o yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  annotations:
    kubectl.kubernetes.io/
last-applied-configuration: |

{"apiVersion":"v1","kind":"Persisten
tVolume","metadata":{"annotations":
{},"labels":
{"name":"pv-securepv-test"},"name":"
pv-securepv-test"},"spec":
{"accessModes":
["ReadWriteOnce"],"capacity":
{"storage":"5Gi"},"csi":
{"driver":"com.mapr.csi-kdf","nodePu
blishSecretRef":
{"name":"mapr-ticket-secret","namesp
ace":"test-csi"},"volumeAttributes":
{"cldbHosts":"10.10.10.210","cluster
":"clusterA","securityType":"secure"
,"volumePath":"/
volume1"},"volumeHandle":"test-id"},
"persistentVolumeReclaimPolicy":"Del
ete"}}
    pv.kubernetes.io/
bound-by-controller: "yes"
  creationTimestamp:
"2019-01-24T18:19:42Z"
  finalizers:
  - kubernetes.io/pv-protection
  labels:
    name: pv-securepv-test
  name: pv-securepv-test
  resourceVersion: "1024135"
  selfLink: /api/v1/
persistentvolumes/pv-securepv-test
```

```
  uid:
9ed086b3-2004-11e9-8980-0cc47ab39644
spec:
  accessModes:
  - ReadWriteOnce
  capacity:
    storage: 5Gi
  claimRef:
    apiVersion: v1
    kind: PersistentVolumeClaim
    name: mapr-secure-claim
    namespace: test-csi
    resourceVersion: "1024131"
    uid:
9eddbddb-2004-11e9-8980-0cc47ab39644
  csi:
    driver: com.mapr.csi-kdf
    nodePublishSecretRef:
      name: mapr-ticket-secret
      namespace: test-csi
    volumeAttributes:
      cldbHosts: 10.10.10.210
      cluster: clusterA
      securityType: secure
      volumePath: /volume1
    volumeHandle: test-id
  persistentVolumeReclaimPolicy:
Delete
  volumeMode: Filesystem
status:
  phase: Bound
```

Dynamic Provisioning

```
# kubectl describe pv
mapr-pv-4f494906-2007-11e9-8980-0cc4
7ab39644 -n test-csi
Name:
mapr-pv-4f494906-2007-11e9-8980-0cc4
7ab39644
Labels:          <none>
Annotations:     pv.kubernetes.io/
provisioned-by: com.mapr.csi-kdf
Finalizers:      [kubernetes.io/
pv-protection]
StorageClass:    test-secure-sc
Status:          Bound
Claim:           test-csi/
test-secure-pvc
Reclaim Policy:  Delete
Access Modes:    RWO
VolumeMode:      Filesystem
Capacity:        5Gi
Node Affinity:   <none>
Message:
Source:
    Type:              CSI (a
Container Storage Interface (CSI)
volume source)
    Driver:
com.mapr.csi-kdf
    VolumeHandle:
csidynamic-securepv.admnqeepfu
```

```
    ReadOnly:          false
    VolumeAttributes:
cldbHosts=10.10.10.210

cluster=clusterA

mountOptions=

platinum=false

readOnly=false

securityType=secure

storage.kubernetes.io/
csiProvisionerIdentity=154835372470
2-8081-com.mapr.csi-kdf

volumePath=/csidynamic/
csidynamic-securepv-admnqeepfu
Events:                <none>
```

```
# kubectl get pv
mapr-pv-4f494906-2007-11e9-8980-0cc4
7ab39644 -n test-csi -o yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  annotations:
    pv.kubernetes.io/
provisioned-by: com.mapr.csi-kdf
  creationTimestamp:
"2019-01-24T18:39:03Z"
  finalizers:
  - kubernetes.io/pv-protection
  name:
mapr-pv-4f494906-2007-11e9-8980-0cc4
7ab39644
  resourceVersion: "1025707"
  selfLink: /api/v1/
persistentvolumes/
mapr-pv-4f494906-2007-11e9-8980-0cc4
7ab39644
  uid:
527271b6-2007-11e9-8980-0cc47ab39644
spec:
  accessModes:
  - ReadWriteOnce
  capacity:
    storage: 5Gi
  claimRef:
    apiVersion: v1
    kind: PersistentVolumeClaim
    name: test-secure-pvc
    namespace: test-csi
    resourceVersion: "1025691"
    uid:
4f494906-2007-11e9-8980-0cc47ab39644
  csi:
    driver: com.mapr.csi-kdf
    fsType: ext4
```

```
    nodePublishSecretRef:
      name: mapr-ticket-secret
      namespace: test-csi
    volumeAttributes:
      cldbHosts: 10.10.10.210
      cluster: clusterA
      mountOptions: ""
      platinum: "false"
      readOnly: "false"
      securityType: secure
      storage.kubernetes.io/
csiProvisionerIdentity:
1548353724702-8081-com.mapr.csi-kdf
      volumePath: /csidynamic/
csidynamic-securepv-admnqeepfu
    volumeHandle:
csidynamic-securepv.admnqeepfu
  persistentVolumeReclaimPolicy:
Delete
  storageClassName: test-secure-sc
  volumeMode: Filesystem
status:
  phase: Bound
```

3. Run the `node list` on page 1705 command on the MapR cluster to determine the number of POSIX clients.

   For each pod mount request, the POSIX client starts with the pod's hostname and new generated hostid, which is tracked on the MapR cluster. For example:

```
# maprcli node list -clientsonly true
clienttype        clienthealth  hostname
ip                                        lasthb  id
posixclientbasic  Inactive      4f3d34fe-2007-11e9-8980-0cc47ab39644
10.10.102.94,172.17.0.1,192.168.28.0    11225   7407394893618656436
posixclientbasic  Inactive      7906d011-200f-11e9-84c0-0cc47ab39644
10.10.102.94,172.17.0.1,192.168.28.0    8174    7544602061076655421
posixclientbasic  Inactive      9ed61912-2004-11e9-8980-0cc47ab39644
10.10.102.92,172.17.0.1,192.168.184.128 11224   2540810767207593086
posixclientbasic  Inactive      c35ab639-2010-11e9-84c0-0cc47ab39644
10.10.102.94,172.17.0.1,192.168.28.0    7568    7947067275504513691
posixclientbasic  Active        e5dc10e8-2012-11e9-84c0-0cc47ab39644
10.10.102.94,172.17.0.1,192.168.28.0    18      5849529086453778130
```

Managing Snapshots Using the MapR Container Storage Interface (CSI) Storage Plugin

This section describes how to create and delete one or more snapshots of volumes dynamically provisioned by the MapR Container Storage Interface (CSI) Storage Plugin on the MapR cluster.

The MapR Container Storage Interface (CSI) Storage Plugin v1.x.x uses the `csi-snapshotter` to support snapshot provisioning.

CSI now supports snapshot restore, which allows you to create a new volume from the snapshot data of another volume. See Creating a Volume from a Snapshot on page 3146. To manually restore a volume, see Copying From a Snapshot Using the CLI on page 951.

Creating a Snapshot Using the MapR Container Storage Interface (CSI) Storage Plugin

You can create one or more snapshots of a dynamically provisioned volume using the MapR Container Storage Interface (CSI) Storage Plugin.

Creating a Snapshot of a Dynamically Provisioned Volume on the Cluster

1. Verify that the volume was successfully provisioned by checking the PersistentVolume (PV) and PersistentVolumeClaim (PVC) for the volume.

   For example, run the describe `kubectl command` to verify the PV and then the PVC.

```
# kubectl describe pv -n test-csi
Name:            mapr-pv-e46a50cd-2012-11e9-84c0-0cc47ab39644
Labels:          <none>
Annotations:     pv.kubernetes.io/provisioned-by: com.mapr.csi-kdf
Finalizers:      [kubernetes.io/pv-protection]
StorageClass:    test-secure-sc
Status:          Bound
Claim:           test-csi/test-secure-pvc
Reclaim Policy:  Delete
Access Modes:    RWO
VolumeMode:      Filesystem
Capacity:        5Gi
Node Affinity:   <none>
Message:
Source:
    Type:              CSI (a Container Storage Interface (CSI) volume
source)
    Driver:            com.mapr.csi-kdf
    VolumeHandle:      csisc-securesc.txiqvsdxwu
    ReadOnly:          false
    VolumeAttributes:      cldbHosts=10.10.10.210
                           cluster=clusterA
                           mountOptions=
                           platinum=false
                           readOnly=false
                           securityType=secure
                           storage.kubernetes.io/
csiProvisionerIdentity=1548359007307-8081-com.mapr.csi-kdf
                           volumePath=/csisc/csisc-securesc-txiqvsdxwu
Events:                <none>
```

```
# kubectl describe pvc -n test-csi
Name:          test-secure-pvc
Namespace:     test-csi
StorageClass:  test-secure-sc
Status:        Bound
Volume:        mapr-pv-e46a50cd-2012-11e9-84c0-0cc47ab39644
Labels:        <none>
Annotations:   kubectl.kubernetes.io/last-applied-configuration:

{"apiVersion":"v1","kind":"PersistentVolumeClaim","metadata":
{"annotations":
{},"name":"test-secure-pvc","namespace":"test-csi"},"spec":{"a...
               pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
               volume.beta.kubernetes.io/storage-provisioner:
com.mapr.csi-kdf
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      5Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Events:
  Type       Reason                Age
From
   Message
  ----       ------                ----   ----
```

```
                                                    -------
  Normal     ExternalProvisioning   3m43s
persistentvolume-controller
   waiting for a volume to be created, either by external provisioner
"com.mapr.csi-kdf" or manually created by system administrator
  Normal     Provisioning           3m43s
com.mapr.csi-kdf_csi-controller-kdf-0_69805ad1-2010-11e9-88dc-d610076b9fb
3  External provisioner is provisioning volume for claim "test-csi/
test-secure-pvc"
  Normal     ProvisioningSucceeded  3m40s
com.mapr.csi-kdf_csi-controller-kdf-0_69805ad1-2010-11e9-88dc-d610076b9fb
3  Successfully provisioned volume
mapr-pv-e46a50cd-2012-11e9-84c0-0cc47ab39644
Mounted By:  test-secure-pod
```

2. Deploy the REST secret `.yaml` file by running the following command:

```
kubectl apply -f <secret filename>.yaml
```

   The Secret file should look similar to the following:

```
# Copyright (c) 2009 & onwards. MapR Tech, Inc., All rights reserved
apiVersion: v1
kind: Secret
metadata:
  name: mapr-snapshot-secrets
  namespace: test-csi
type: Opaque
data:
  MAPR_CLUSTER_USER: cm9vdA==
  MAPR_CLUSTER_PASSWORD: bWFwcg==
```

   For more information, see Configuring a Secret on page 3167.

3. Create a snapshot class for provisioning a snapshot of the volume. For example, the snapshot class file should look similar to the following:

   FUSE

```
apiVersion: snapshot.storage.k8s.io/v1beta1
kind: VolumeSnapshotClass
metadata:
 name: testcsi-snapshotclass
 namespace: test-csi
driver: com.mapr.csi-kdf
deletionPolicy: Delete
parameters:
 restServers: "10.10.102.95:8443"
 cluster: "mycluster"
 csi.storage.k8s.io/snapshotter-secret-name: mapr-snapshot-secrets
 csi.storage.k8s.io/snapshotter-secret-namespace: test-csi
```

   Loopback NFS

```
apiVersion: snapshot.storage.k8s.io/v1beta1
kind: VolumeSnapshotClass
metadata:
 name: testcsi-snapshotclass
 namespace: test-csi
driver: com.mapr.csi-nfskdf
```

```
deletionPolicy: Delete
parameters:
 restServers: "10.10.102.95:8443"
 cluster: "mycluster"
 csi.storage.k8s.io/snapshotter-secret-name: mapr-snapshot-secrets
 csi.storage.k8s.io/snapshotter-secret-namespace: test-csi
```

The sample snapshot class file shown above contains the following properties:

Property	Description
`apiVersion`	The Kubernetes APi version for the StorageClass spec.
`kind`	The kind of object being created. This is a StorageClass.
`metadata: name`	The name of the snapshot calss. Administrators should specify the name carefully because it will be used by Pod authors to help select the right snapshot class for their needs.
`metadata: namespace`	The namespace in which the snapshot class runs.
`driver`	The CSI volume plugin to use for provisioning the volume snapshots. For example: `com.mapr.csi-kdf`.
`restServers`	A space-separated list of webservers. Specify the hostname or IP address and port number of each REST server for the cluster. For fault tolerance, providing multiple REST server hosts is recommended.
`cluster`	The cluster name.
`csiSnapshotterSecretName` (deprecated) `csi.storage.k8s.io/snapshotter-secret-name`	The name of the Kubernetes Secret that is used to store administrative credentials (user, password, and ticket information for the webserver). To use the provisioner, you must configure a Secret. See Configuring a Secret on page 3167.
`csiSnapshotterSecretNamespace` (deprecated) `csi.storage.k8s.io/snapshotter-secret-namespace`	The namespace for the Secret containing the administrative credentials (user name and password information for a user that has the privileges to create volumes). This namespace can be different from the namespace used by the Pod, since a Pod author or namespace admin might not be trusted to create administration Secrets for the cluster.

4. Deploy the snapshot class by running the following command:

```
kubectl apply -f <snapshot class>.yaml
```

5. Verify whether the snapshot class was successfully deployed by running one of the following commands:

```
# kubectl get volumesnapshotclass -n test-csi
NAME                 AGE
test-snapshotclass   41s
root@qa102-92:~/csi-kdf-3/csi-kdf/examples/snapshot# kubectl describe
volumesnapshotclass -n test-csi
Name:         test-snapshotclass
Namespace:
Labels:       <none>
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"snapshot.storage.k8s.io/
v1alpha1","kind":"VolumeSnapshotClass","metadata":{"annotations":
{},"name":"test-snapshotclass"},"p...
API Version:  snapshot.storage.k8s.io/v1alpha1
Kind:         VolumeSnapshotClass
Metadata:
  Creation Timestamp:  2019-01-24T21:13:35Z
  Generation:          1
  Resource Version:    1039219
  Self Link:           /apis/snapshot.storage.k8s.io/v1alpha1/
volumesnapshotclasses/test-snapshotclass
  UID:                 e94a1fc8-201c-11e9-84c0-0cc47ab39644
Parameters:
  Cluster:                              clusterA
  Csi Snapshotter Secret Name:          mapr-snapshot-secrets
  Csi Snapshotter Secret Namespace:     test-csi
  Name Prefix:                          test-snapshot
  Rest Servers:                         10.10.10.210:8443
Snapshotter:                            com.mapr.csi-kdf
Events:                                 <none>
```

```
# kubectl get volumesnapshotclass -n test-csi -o yaml
apiVersion: v1
items:
- apiVersion: snapshot.storage.k8s.io/v1alpha1
  kind: VolumeSnapshotClass
  metadata:
    annotations:
      kubectl.kubernetes.io/last-applied-configuration: |
        {"apiVersion":"snapshot.storage.k8s.io/
v1alpha1","kind":"VolumeSnapshotClass","metadata":{"annotations":
{},"name":"test-snapshotclass"},"parameters":
{"cluster":"clusterA","csiSnapshotterSecretName":"mapr-snapshot-secrets",
"csiSnapshotterSecretNamespace":"test-csi","namePrefix":"test-snapshot","
restServers":"10.10.10.210:8443"},"snapshotter":"com.mapr.csi-kdf"}
    creationTimestamp: "2019-01-24T21:13:35Z"
    generation: 1
    name: test-snapshotclass
    resourceVersion: "1039219"
    selfLink: /apis/snapshot.storage.k8s.io/v1alpha1/
volumesnapshotclasses/test-snapshotclass
    uid: e94a1fc8-201c-11e9-84c0-0cc47ab39644
  parameters:
    cluster: clusterA
    csiSnapshotterSecretName: mapr-snapshot-secrets
    csiSnapshotterSecretNamespace: test-csi
    namePrefix: test-snapshot
    restServers: 10.10.10.210:8443
  snapshotter: com.mapr.csi-kdf
kind: List
```


last-updated: Apr 11, 2022

```
metadata:
  resourceVersion: ""
  selfLink: ""
```

6. Associate the snapshot class with the PersistentVolumeClaim (for the volume to take a snapshot of) by creating a VolumeSnapshot. For example, the VolumeSnapshot file should look similar to the following:

```
apiVersion: snapshot.storage.k8s.io/v1beta1
kind: VolumeSnapshot
metadata:
 name: testcsi-secure-snapshot
 namespace: test-csi
spec:
 volumeSnapshotClassName: testcsi-snapshotclass
 source:
    persistentVolumeClaimName: testcsi-secure-pvc
```

   The sample VolumeSnapshot file shown above contains the following properties:

Property	Description
`metadata: name`	The VolumeSnapshot name.
`metadata: namespace`	The namespace in which the VolumeSnapshot runs.
`snapshotClassName`	The volumeSnapshotClassName.
`source: name`	The persistentVolumeClaimName.

7. Deploy the VolumeSnapshot by running the following command:

```
kubectl apply -f <volume snapshot>.yaml
```

8. Verify whether VolumeSnapshot was successfully deployed by doing the following:

last-updated: Apr 11, 2022

a) Run one of the following commands to retrieve the VolumeSnapshot:

```
# kubectl get volumesnapshot -n test-csi -o yaml
apiVersion: v1
items:
- apiVersion: snapshot.storage.k8s.io/v1alpha1
  kind: VolumeSnapshot
  metadata:
    annotations:
      kubectl.kubernetes.io/last-applied-configuration: |
        {"apiVersion":"snapshot.storage.k8s.io/
v1alpha1","kind":"VolumeSnapshot","metadata":{"annotations":
{},"name":"test-snapshot","namespace":"test-csi"},"spec":
{"snapshotClassName":"test-snapshotclass","source":
{"kind":"PersistentVolumeClaim","name":"test-secure-pvc"}}}
    creationTimestamp: "2019-01-24T21:16:21Z"
    finalizers:
    - snapshot.storage.kubernetes.io/volumesnapshot-protection
    generation: 5
    name: test-snapshot
    namespace: test-csi
    resourceVersion: "1039445"
    selfLink: /apis/snapshot.storage.k8s.io/v1alpha1/namespaces/
test-csi/volumesnapshots/test-snapshot
    uid: 4c5293bc-201d-11e9-84c0-0cc47ab39644
  spec:
    snapshotClassName: test-snapshotclass
    snapshotContentName:
snapcontent-4c5293bc-201d-11e9-84c0-0cc47ab39644
    source:
      apiGroup: null
      kind: PersistentVolumeClaim
      name: test-secure-pvc
  status:
    creationTime: "2019-01-24T21:16:22Z"
    readyToUse: true
    restoreSize: null
kind: List
metadata:
  resourceVersion: ""
  selfLink: ""
```

```
# kubectl describe volumesnapshot -n test-csi
Name:         test-snapshot
Namespace:    test-csi
Labels:       <none>
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"snapshot.storage.k8s.io/
v1alpha1","kind":"VolumeSnapshot","metadata":{"annotations":
{},"name":"test-snapshot","namespace":"...
API Version:  snapshot.storage.k8s.io/v1alpha1
Kind:         VolumeSnapshot
Metadata:
  Creation Timestamp:  2019-01-24T21:16:21Z
  Finalizers:
    snapshot.storage.kubernetes.io/volumesnapshot-protection
  Generation:        5
  Resource Version:  1039445
  Self Link:         /apis/snapshot.storage.k8s.io/v1alpha1/
namespaces/test-csi/volumesnapshots/test-snapshot
  UID:               4c5293bc-201d-11e9-84c0-0cc47ab39644
Spec:
```

```
  Snapshot Class Name:    test-snapshotclass
  Snapshot Content Name:
snapcontent-4c5293bc-201d-11e9-84c0-0cc47ab39644
  Source:
    API Group:  <nil>
    Kind:       PersistentVolumeClaim
    Name:       test-secure-pvc
Status:
  Creation Time:  2019-01-24T21:16:22Z
  Ready To Use:   true
  Restore Size:   <nil>
Events:           <none>
```

b) Retrieve the VolumeSnapshot contents, which shows the associated PersistentVolume, by running one of the following commands:

```
# kubectl get volumesnapshotcontents -n test-csi -o yaml
apiVersion: v1
items:
- apiVersion: snapshot.storage.k8s.io/v1alpha1
  kind: VolumeSnapshotContent
  metadata:
    creationTimestamp: "2019-01-24T21:16:22Z"
    finalizers:
    - snapshot.storage.kubernetes.io/volumesnapshotcontent-protection
    generation: 1
    name: snapcontent-4c5293bc-201d-11e9-84c0-0cc47ab39644
    resourceVersion: "1039443"
    selfLink: /
apis/snapshot.storage.k8s.io/v1alpha1/volumesnapshotcontents/
snapcontent-4c5293bc-201d-11e9-84c0-0cc47ab39644
    uid: 4cab5cb5-201d-11e9-84c0-0cc47ab39644
  spec:
    csiVolumeSnapshotSource:
      creationTime: 1548364582387786034
      driver: com.mapr.csi-kdf
      restoreSize: 0
      snapshotHandle:
mapr-snapshot-4c5293bc-201d-11e9-84c0-0cc47ab39644
    deletionPolicy: Delete
    persistentVolumeRef:
      apiVersion: v1
      kind: PersistentVolume
      name: mapr-pv-e46a50cd-2012-11e9-84c0-0cc47ab39644
      resourceVersion: "1033559"
      uid: ea32c304-2012-11e9-84c0-0cc47ab39644
    snapshotClassName: test-snapshotclass
    volumeSnapshotRef:
      apiVersion: snapshot.storage.k8s.io/v1alpha1
      kind: VolumeSnapshot
      name: test-snapshot
      namespace: test-csi
      resourceVersion: "1039439"
      uid: 4c5293bc-201d-11e9-84c0-0cc47ab39644
kind: List
metadata:
  resourceVersion: ""
  selfLink: ""
```

```
# kubectl describe volumesnapshotcontents -n test-csi
Name:         snapcontent-4c5293bc-201d-11e9-84c0-0cc47ab39644
Namespace:
Labels:       <none>
Annotations:  <none>
API Version:  snapshot.storage.k8s.io/v1alpha1
Kind:         VolumeSnapshotContent
Metadata:
  Creation Timestamp:  2019-01-24T21:16:22Z
  Finalizers:
    snapshot.storage.kubernetes.io/volumesnapshotcontent-protection
  Generation:        1
  Resource Version:  1039443
  Self Link:         /
apis/snapshot.storage.k8s.io/v1alpha1/volumesnapshotcontents/
snapcontent-4c5293bc-201d-11e9-84c0-0cc47ab39644
```

```
  UID:               4cab5cb5-201d-11e9-84c0-0cc47ab39644
Spec:
  Csi Volume Snapshot Source:
    Creation Time:    1548364582387786034
    Driver:           com.mapr.csi-kdf
    Restore Size:     0
    Snapshot Handle:
mapr-snapshot-4c5293bc-201d-11e9-84c0-0cc47ab39644
  Deletion Policy:    Delete
  Persistent Volume Ref:
    API Version:        v1
    Kind:               PersistentVolume
    Name:               mapr-pv-e46a50cd-2012-11e9-84c0-0cc47ab39644
    Resource Version:   1033559
    UID:                ea32c304-2012-11e9-84c0-0cc47ab39644
  Snapshot Class Name:  test-snapshotclass
  Volume Snapshot Ref:
    API Version:       snapshot.storage.k8s.io/v1alpha1
    Kind:              VolumeSnapshot
    Name:              test-snapshot
    Namespace:         test-csi
    Resource Version:  1039439
    UID:               4c5293bc-201d-11e9-84c0-0cc47ab39644
Events:                <none>
```

9. Log in to the cluster and verify by running the `volume snapshot list` on page 2030 command. For example:

```
# maprcli volume snapshot list -path /csisc/
csisc-securesc-txiqvsdxwu -cluster clusterA -json
{
    "timestamp":1548365090744,
    "timeofday":"2019-01-24 01:24:50.744 GMT-0800 PM",
    "status":"OK",
    "total":1,
    "data":[
        {
            "ownername":"root",
            "ownertype":"1",
            "volumeid":"234021649",
            "volumename":"csisc-securesc.txiqvsdxwu",
            "volumepath":"/csisc/csisc-securesc-txiqvsdxwu",
            "snapshotid":"256000051",

"snapshotname":"mapr-snapshot-4c5293bc-201d-11e9-84c0-0cc47ab39644",
            "creationtime":"Thu Jan 24 13:16:22 PST 2019",
            "cumulativeReclaimSizeMB":"0",
            "ownedsize":"0",
            "sharedSize":"0",
            "volumeSnapshotAces":{
                "readAce":"p",
                "writeAce":"p"
            }
        }
    ]
}
```

Creating Multiple Snapshots of a Dynamically Provisioned Volume

1. Perform steps 1 - 5 described in the Creating a Snapshot of a Dynamically Provisioned Volume on the Cluster on page 3135 section.

last-updated: Apr 11, 2022

2. Create a VolumeSnapshot similar to the one shown in step 6 of the Creating a Snapshot of a Dynamically Provisioned Volume on the Cluster on page 3135 section for each additional snapshot to create for the volume.

   For example:

   ```
   apiVersion: snapshot.storage.k8s.io/v1alpha1
   kind: VolumeSnapshot
   metadata:
     name: test-snapshot1
     namespace: test-csi
   spec:
     snapshotClassName: test-snapshotclass
     source:
       name: test-secure-pvc
       kind: PersistentVolumeClaim
   ```

3. Repeat step 7 in Creating a Snapshot of a Dynamically Provisioned Volume on the Cluster on page 3135 for each additional volume snaphots you have created.

4. Log in to the cluster and verify by running the `volume snapshot list` on page 2030 command. For example:

```
# maprcli volume snapshot list -path /csisc/
csisc-securesc-txiqvsdxwu -cluster clusterA -json
{
    "timestamp":1548365359138,
    "timeofday":"2019-01-24 01:29:19.138 GMT-0800 PM",
    "status":"OK",
    "total":2,
    "data":[
        {
            "ownername":"root",
            "ownertype":"1",
            "volumeid":"234021649",
            "volumename":"csisc-securesc.txiqvsdxwu",
            "volumepath":"/csisc/csisc-securesc-txiqvsdxwu",
            "snapshotid":"256000051",

"snapshotname":"mapr-snapshot-4c5293bc-201d-11e9-84c0-0cc47ab39644",
            "creationtime":"Thu Jan 24 13:16:22 PST 2019",
            "cumulativeReclaimSizeMB":"0",
            "ownedsize":"0",
            "sharedSize":"0",
            "volumeSnapshotAces":{
                "readAce":"p",
                "writeAce":"p"
            }
        },
        {
            "ownername":"root",
            "ownertype":"1",
            "volumeid":"234021649",
            "volumename":"csisc-securesc.txiqvsdxwu",
            "volumepath":"/csisc/csisc-securesc-txiqvsdxwu",
            "snapshotid":"256000052",

"snapshotname":"mapr-snapshot-19282d27-201f-11e9-84c0-0cc47ab39644",
            "creationtime":"Thu Jan 24 13:29:15 PST 2019",
            "cumulativeReclaimSizeMB":"0",
            "ownedsize":"0",
            "sharedSize":"0",
            "volumeSnapshotAces":{
                "readAce":"p",
                "writeAce":"p"
            }
        }
    ]
}
```

Deleting a Snapshot of a Dynamically Provisioned Volume

You can delete snapshots you created using MapR Container Storage Interface (CSI) Storage Plugin. To delete:

1. Run the following command:

```
kubectl delete -f <volume snapshot>.yaml
```

   For example:

```
# kubectl delete -f test-snapshot1.yaml
volumesnapshot.snapshot.storage.k8s.io "test-snapshot1" deleted
```

2. Log in to the cluster and verify that the snapshot was deleted by running the `volume snapshot list` on page 2030 command.

   For example:

```
# maprcli volume snapshot list -path /csisc/
csisc-securesc-txiqvsdxwu -cluster clusterA -json
{
    "timestamp":1548365417772,
    "timeofday":"2019-01-24 01:30:17.772 GMT-0800 PM",
    "status":"OK",
    "total":1,
    "data":[
        {
            "ownername":"root",
            "ownertype":"1",
            "volumeid":"234021649",
            "volumename":"csisc-securesc.txiqvsdxwu",
            "volumepath":"/csisc/csisc-securesc-txiqvsdxwu",
            "snapshotid":"256000051",

"snapshotname":"mapr-snapshot-4c5293bc-201d-11e9-84c0-0cc47ab39644",
            "creationtime":"Thu Jan 24 13:16:22 PST 2019",
            "cumulativeReclaimSizeMB":"0",
            "ownedsize":"0",
            "sharedSize":"0",
            "volumeSnapshotAces":{
                "readAce":"p",
                "writeAce":"p"
            }
        }
    ]
}
```

Creating a Volume from a Snapshot

You can restore a volume from a volume snapshot by configuring a PersistentVolumeClaim that specifies the volume snapshot as the data source.

In the following example, the snapshot named `testcsi-secure-snapshot` serves as the data source for creating a new volume named `testcsi-secure-pvc-restore`:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: testcsi-secure-pvc-restore
  namespace: test-csi
spec:
  storageClassName: testcsi-secure-sc
```

```
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 10G
  dataSource:
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
    name: testcsi-secure-snapshot
```

When creating a volume from snapshots of extra large volumes (volumes measuring hundreds of GB), you might experience timeouts or a failure to create the volume. To prevent timeouts with extra large volumes, increase the retry timeout setting of the `csi-snapshotter` sidecar container. See the `--timeout` argument in the latest `csi-maprkdf-<version>.yaml`.

For more information, see Persistent Volumes.

Enabling the Platinum POSIX Client for MapR Container Storage Interface (CSI) Storage Plugin

When you install the MapR Container Storage Interface (CSI) Storage Plugin, the Container FUSE-based POSIX client package is installed on the CSI Driver container. The CSI Driver also supports the use of the Basic, Container, and Platinum FUSE-based POSIX client. For a comparison of the POSIX client packages, see Preparing for Installation (MapR POSIX Client) on page 400.

Note: Enabling the Platinum POSIX client is not required for the Loopback NFS plug-in.

To install the Platinum POSIX client, include the `platinum` parameter in your pod spec. For example:

```
volumeAttributes:
    volumePath: "/"
    cluster: "clusterA"
    cldbHosts: "10.10.102.96"
    securityType: "secure"
    platinum: "true"
```

Release 1.0.2 and later support another method for specifying the POSIX client. You can use the `license` parameter and specify the Container, Basic, or Platinum driver. For example:

```
volumeAttributes:
    volumePath: "/"
    cluster: "clusterA"
    cldbHosts: "10.10.102.96"
    securityType: "secure"
    license: "platinum"
```

For more information, see Example: Mounting a PersistentVolume for Static Provisioning on page 3113.

Logging for the CSI Driver and Provisioner
Describes the event logs for the CSI driver and provisioner for both FUSE and Loopback NFS plugins.

Logs for the MapR Container Storage Interface (CSI) Storage Plugin can be found in `/var/log/csi-maprkdf/`. The following table shows the new log-file format. Before the FUSE 1.2.2 and Loopback NFS 1.0.2 drivers were introduced, the log files included a version in the file name.

Log File Type	FUSE or Loopback NFS	Log File	Description	Which Nodes
Driver events log	FUSE	`csi-plugin.log`	Captures CSI Driver events such as registering the driver and CSI Driver logs for node mount and unmount operations.	All Kubernetes nodes.
	Loopback NFS	`csi-nfsplugin.log`		

last-updated: Apr 11, 2022

Log File Type	FUSE or Loopback NFS	Log File	Description	Which Nodes
Provisioner events log	FUSE	`csi-provisioner.log`	Captures provisioner events such as registering the CSI provisioner and CSI Controller events such as Create/Delete volumes, Create/Delete Snapshots etc.	The Kubernetes node where the provisioner StatefulSet pod is running.
	Loopback NFS	`csi-nfsprovisioner.log`		

Note: The directory must grant `rw` permissions for creating the logs and must grant write/append permissions to the plug-in and provisioner.

Troubleshooting the MapR Container Storage Interface (CSI) Storage Plugin
This section describes how to resolve common problems you might encounter when installing and using the MapR Container Storage Interface (CSI) Storage Plugin.

Troubleshooting CSI Driver installation

Run the following commands to get the pods that are deployed for the CSI plugin and provisioner:

FUSE

```
kubectl get pods -n mapr-csi
```

Loopback NFS

```
kubectl get pods -n mapr-nfscsi
```

The installation is considered successful if the `get pods` command shows the pods in the `Running` state. For example, your output should look similar to the following when CSI plugin is deployed on three worker nodes:

FUSE

```
mapr-csi       csi-controller-kdf-0         5/5     Running     0
4h25m
mapr-csi       csi-nodeplugin-kdf-2kfrf     3/3     Running     0
4h25m
mapr-csi       csi-nodeplugin-kdf-lq5nw     3/3     Running     0
4h25m
mapr-csi       csi-nodeplugin-kdf-pkrzt     3/3     Running     0
4h25m
```

Loopback NFS

```
csi-controller-nfskdf-0       7/7     Running   0          22h
csi-nodeplugin-nfskdf-5rjt2   3/3     Running   0          18h
csi-nodeplugin-nfskdf-7d9cs   3/3     Running   0          22h
csi-nodeplugin-nfskdf-qw7kg   3/3     Running   0          22h
```

The preceding output shows the following:

FUSE

- `csi-nodeplugin-kdf-*`: Daemonset pods deployed on all the Kubernetes worker nodes
- `csi-controller-kdf-0`: StatefulSet pod deployed on a single Kubernetes worker node

Loopback NFS

- `csi-nodeplugin-nfskdf-*`: Daemonset pods deployed on all the Kubernetes worker nodes
- `csi-controller-nfskdf-0`: StatefulSet pod deployed on a single Kubernetes worker node

Troubleshoot MapR CSI Plugin Deployment Failures

If the pods show a failure in the deployment, run the following kubectl command to see the container logs:

FUSE

```
kubectl logs <csi-nodeplugin-*> -n mapr-csi -c <nodeplugin-pod-container>
```

Loopback NFS

```
kubectl logs <csi-nodeplugin-*> -n mapr-nfscsi -c <nodeplugin-pod-container>
```

If the pods show a failure in the deployment, run the following kubectl commands to see the container logs:

FUSE

```
kubectl logs <csi-nodeplugin-*> -n mapr-csi -c <nodeplugin-pod-container>
```

Loopback NFS

```
kubectl logs csi-controller-nfskdf-0 -n mapr-nfscsi -c
<controller-pod-container>
```

Here, replace `<nodeplugin-pod-container>` with the container that is failing. You can also run the following kubectl command to see the controller logs:

```
kubectl logs csi-controller-kdf-0 -n mapr-csi -c <controller-pod-container>
```

Here, replace `<controller-pod-container>` with the container which is failing.

Troubleshooting Volume Provisioning

Check the provisioner log and check for any provisioner errors:

FUSE

```
tail -100f /var/log/csi-maprkdf/csi-provisioner.log
```

Loopback NFS

```
tail -100f /var/log/csi-maprkdf/csi-nfsprovisioner.log
```

Troubleshooting Mount Operation

Check the CSI Storage plug-in log for any mount/unmount errors:

FUSE

```
tail -100f /var/log/csi-maprkdf/csi-plugin.log
```

Loopback NFS

```
tail -100f /var/log/csi-maprkdf/csi-nfsplugin.log
```


last-updated: Apr 11, 2022

If you don't see any errors, see the kubelet logs on the node where the pod is scheduled to run. Check the MapR CSI Storage plugin logs for specific errors.

Troubleshooting MapR CSI Storage Plugin Discovery with kubelet

Check the kubelet path for kubernetes deployment from the kubelet process running with `--root-dir`. The `--root-dir` is a string that contains the directory path for managing kubelet files (such as volume mounts, etc.,) and defaults to `/var/lib/kubelet`. If the Kubernetes environment has a different kubelet path, modify the CSI driver deployment `.yaml` file with the new path, and redeploy the MapR CSI Storage Plugin again.

Troubleshooting Snapshot Provisioning

See the provisioner log and check for any provisioner errors:

```
tail -100f /var/log/csi-maprkdf/csi-provisioner.log
```

If there are no errors, run the following kubectl command to check the snapshot:

```
kubectl describe volumesnapshot.snapshot.storage.k8s.io <snapshot-name> -n
<namespace-name>
```

Here:

- `<snapshot-name>`: Name of the VolumeSnapshot Object defined in yaml
- `<namespace-name>`: Namespace where the VolumeSnapshot object is created

Troubleshooting No Space on Disk Error

The devicemapper storage driver used for Docker allows only 10 GB by default resulting in "no space left on device" errors when writing to new directories for a new volume mount request. If `--maxvolumepernode` is configured to be greater than 20 and underlying docker is using devicemapper storagedriver, do the following to increase the storage size:

1. Change storagedriver to be other than devicemapper, which restricts container storage to 10 GB by default.
2. Increase default container storage to more than the default of 10 GB for devicemapper storagedriver for the Docker container running on Kubernetes worker node.

For example, do the following to increase the storage size to 50 GB:

1. In `/etc/sysconfig/docker-storage` file, add `--storage-opt dm.basesize=50G` under `DOCKER_STORAGE_OPTIONS` section.
2. Restart Docker.
3. Confirm that the setting is correctly applied by running the following command:

   ```
   docker info | grep "Base Device Size"
   ```

Kubernetes FlexVolume Driver Configuration

This section describes how to use and troubleshoot the MapR Data Platform for Kubernetes FlexVolume Driver.

For more information about the MapR Data Platform for Kubernetes, see MapR Data Fabric for Kubernetes FlexVolume Driver Overview on page 671.

Using the MapR Data Fabric for Kubernetes FlexVolume Driver

This section describes how to configure Kubernetes objects to enable persistent storage and includes example configuration files for static and dynamic provisioning.

Before You Begin

Before configuration, be sure to review the following notes about supported and unsupported features and parameters:

MapR Parameters for Dynamic Provisioning

In dynamic provisioning, you can specify MapR parameters for the MapR volume to be created. For a list of the MapR parameters that you can use, see volume create on page 1931. Note these considerations for using the MapR parameters:

- Volume attributes must be represented as a string (enclosed within quotations). Using an integer or boolean is not supported. In the following example, the `aetype` attribute will generate an error because the value (`1`) is not enclosed in quotations.

```
namePrefix: "pv"
mountPrefix: "/pv"
type: "rw"
advisoryquota: "100M"
aetype: 1
```

- The following MapR parameters are ignored because they are redundant or not supported in the Kubernetes implementation:
  - `mount`
  - `quota*`
  - `createparent`
  - `path`
  - `name`

*Specifying `resources: requests: storage` in a PersistentVolumeClaim (PVC) (see Example: Mounting a PersistentVolume for Static Provisioning Using the FlexVolume Driver on page 3155) makes it unnecessary to set the MapR `quota` parameter.

Kubernetes Access Modes

Kubernetes access modes control how a PersistentVolume (PV) is mounted on the host. Access modes can be specified on both PVs and PVCs. Only Volumes with a matching Access Mode will be bound to a PVC. Unfortunately, beyond the PVC/PV binding behavior, PVs using FlexVolume drivers ignore these access modes in the current version of Kubernetes. All access modes will work with the MapR Data Fabric for Kubernetes. However, they will appear the same. This means the ROX mode will not make the volume read only. If you want read-only behavior, specify `readOnly:` in the FlexVolume driver flags.

PersistentVolumeClaim Protection

PVC protection is a Kubernetes 1.9 alpha feature that restricts the user from deleting a PVC while it is being used by an active Pod. Alpha features are not tested for use with the volume plug-in and provisioner. However, without PVC protection, you should not delete a PVC that is still attached to Pods. If you have not turned on PVC protection, ensure that you do not delete PVC's that are in use. In the current release of the MapR Data Fabric for Kubernetes, deleting a PVC causes undefined behavior.

last-updated: Apr 11, 2022

Reclaim Policy

The Kubernetes `reclaimPolicy` parameter controls what happens to a PersistentVolume if the corresponding PersistentVolumeClaim is deleted. The `Recycle` Reclaim Policy is not supported by Kubernetes FlexVolume Drivers, so it cannot be used with KDF. The Retain Policy is currently broken in Kubernetes 1.9 StoragePolicies but not in static PersistentVolumes. The MapR Data Fabric for Kubernetes has a workaround that allows Retain policy on dynamically provisioned volumes by passing the `reclaimPolicy:` in the parameters rather than in the standard place that FlexVolumes ignore. You can specify the reclaim policy normally when you configure a persistent volume.

The following table shows the supported values for the reclaim policy:

Reclaim Policy Value	Description	Support
Delete (default value)	The PersistentVolume and the MapR volume are deleted when the user deletes the corresponding PersistentVolumeClaim.	Supported
Retain	The PersistentVolume and the MapR volume are not deleted when the user deletes the corresponding PersistentVolumeClaim.	Supported*
Recycle	Performs a basic scrub on a PersistentVolume and makes it available for a new PersistentVolumeClaim.	Not Supported by Kubernetes Flexvolumes

*Not supported for dynamic provision without a workaround.

For more information about the reclaim policy, see Change the Reclaim Policy of a PersistentVolume.

Kubernetes Mount Options

The Kubernetes `mountOptions` parameter is not supported for use with the MapR Data Fabric for Kubernetes because it is not supported for use with the FlexVolume plug-in.

Steps for Configuring the MapR Data Fabric for Kubernetes FlexVolume Driver

This page summarizes the high-level steps for configuring the MapR Data Fabric for Kubernetes FlexVolume Driver to provide static or dynamic provisioning. To learn more about static and dynamic provisioning, see Static and Dynamic Provisioning Using FlexVolume Driver on page 673.

Static Provisioning

1. Install the MapR Data Fabric for Kubernetes.

   See Installing the MapR Data Fabric for Kubernetes FlexVolume Driver on page 241.

2. In your Pod spec or as part of a separate configuration file, configure a PersistentVolume.

   See Example: Mounting a PersistentVolume for Static Provisioning Using the FlexVolume Driver on page 3155 and Persistent Volumes.

3. Do *one* of the following:

   - Annotate the Pod spec to provide information about the MapR volume.

     See Example: Statically Provisioning a MapR Volume Using the FlexVolume Plug-in on page 3153.

   - In your Pod spec or as part of a separate configuration file, configure a PersistentVolumeClaim.

     See PersistentVolumeClaims.

4. Run the Pod spec by using `kubectl` commands. See Overview of kubectl.

last-updated: Apr 11, 2022

Dynamic Provisioning

1. Install the MapR Data Fabric for Kubernetes.

   See Installing the MapR Data Fabric for Kubernetes FlexVolume Driver on page 241.

2. In your Pod spec or in a separate configuration file, create a storage class.

   See Example: Mounting a PersistentVolume for Dynamic Provisioning Using the FlexVolume Driver on page 3159 and Storage Classes.

3. In your Pod spec or in a separate configuration file, configure a PersistentVolumeClaim.

   See Example: Mounting a PersistentVolume for Dynamic Provisioning Using the FlexVolume Driver on page 3159.

4. Run the Pod spec by using `kubectl` commands. See Overview of kubectl.

Example: Statically Provisioning a MapR Volume Using the FlexVolume Plug-in

You can designate a MapR volume for use with Kubernetes by specifying the MapR FlexVolume parameters directly inside the Pod spec. In the Pod spec, you define a Kubernetes volume and add the MapR FlexVolume information to it. You can supply path information by using the `volumePath` parameter. The Kubernetes volume is only as persistent as the Pod. By defining the volume this way, when the Pod is removed, the Kubernetes volume is also immediately unmounted and removed. This approach to static provisioning is most appropriate when you want to get up and running quickly or when you want the Pod and Kubernetes volume lifecycle to be the same.

For example, a developer wants to get her application container up and running quickly with MapR. She already has a MapR path that she wants to use for the application. She only needs the data accessible to read. To make this work, she must:

1. Generate a MapR service ticket, and set the `securityType` parameter in the Pod spec to `secure`. See Generating a Service Ticket on page 1428.

2. Configure a Ticket Secret, and include the base64-encoded contents of the ticket file in the Ticket Secret. See Configuring a Secret on page 3167.

3. Set the `runAsUser` and the `fsGroup` parameters to the UID and GID of the user that created the ticket.

4. Point the `volumePath` in the `flexVolume` setting to the desired path, and fill in the `cldbHosts` and `cluster` information.

Note: The following example works for on-premise deployments. For GKE and AWS deployments, you must set a default StorageClass to the `maprfs` StorageClass. If a default StorageClass is not provided for GKE and AWS deployments, the volume is created using your default StorageClass, which might not be a good fit. For information about changing the default StorageClass, see Change the default StorageClass.

```
apiVersion: v1
kind: Pod
metadata:
  name: test-secure
  namespace: mapr-examples
spec:
  securityContext:
    runAsUser: 1000
    fsGroup: 2000
  containers:
  - name: mycontainer
```


last-updated: Apr 11, 2022

```
    image: myrepo/myorg/mycontainer
    args:
    - sleep
    - "1000000"
    imagePullPolicy: Always
    resources:
      requests:
        memory: "2Gi"
        cpu: "500m"
    volumeMounts:
    - mountPath: /mapr
      name: maprvolume
  volumes:
    - name: maprvolume
      flexVolume:
        driver: "mapr.com/maprfs"
        readOnly: true
        options:
          volumePath: "/path/to/data/in/mapr"
          cluster: "mycluster"
          cldbHosts: "cldb1 cldb2 cldb3"
          securityType: "secure"
          ticketSecretName: "mapr-ticket-secret"
          ticketSecretNamespace: "mapr-examples"
---
apiVersion: v1
kind: Secret
metadata:
  name: mapr-ticket-secret
  namespace: mapr-examples
type: Opaque
data:
  CONTAINER_TICKET: <BASE64 ENCODED VERSION OF CONTENTS OF TICKET FILE>
```

The following tables describe the parameters in the example:

Pod

Parameter	Notes
`apiVersion`	The Kubernetes API version for the Pod spec.
`kind`	The kind of object being created. For clarity, the example uses a naked Pod. Generally, it is better to use a Deployment, DaemonSet, or StatefulSet for high availability (HA) and ease of upgrade.
`metadata: name`	The Pod name.
`metadata: namespace`	The namespace in which the Pod runs.
`securityContext: runAsUser`	The user ID to run the container under. This user ID must be the same as the user ID for which the ticket was generated.
`securityContext: fsGroup`	The group ID to run the container under. This group ID must be the same as the group ID of the user for which the ticket was generated.
`volumeMounts: mountPath`	A directory inside the container that is designated as the mount path.
`volumeMounts: name`	A name that you assign to the Kubernetes `volumeMounts` resource. Matches with `Volumes: name`.

last-updated: Apr 11, 2022

`Volumes: name`	A string to identify the name of the Kubernetes `volumes` resource. Matches with `volumeMounts: name`.
`flexVolume: driver`	The MapR FlexVolume driver being used. Call it using this driver: `mapr.com/maprfs`.
`flexVolume: readOnly`	Specifies that the FlexVolume driver should tell the MapR POSIX Client to mount the volume with the read-only flag.
`volumePath`	The mount point within the MapR filesystem. This parameter specifies an existing MapR path. For example, you can specify the root volume as `"/"`, providing access to the entire filesystem.
`cluster`	The MapR cluster name.
`cldbHosts`	The DNS names or IP addresses of the CLDB hosts for the MapR cluster. You must provide at least one CLDB host. For fault-tolerance, providing multiple CLDB hosts is recommended. To specify multiple hosts, separate each name or IP address by a space.
`securityType`	A parameter that indicates whether MapR tickets are used or not used. If MapR tickets are used, specify `secure`. Otherwise, specify `unsecure`.
`ticketSecretName`	The name of the Secret that contains the ticket to use when mounting to the MapR cluster. See Configuring a Secret on page 3167.
`ticketSecretNamespace`	The namespace that contains the Secret. See Configuring a Secret on page 3167

Secret

Parameter	Notes
`apiVersion`	The Kubernetes API version.
`kind`	The type of object being created.
`name`	A string to identify the Secret.
`namespace`	The namespace in which the Secret runs.
`type`	The type of Secret being created. For type `Opaque`, clients must treat these values as opaque and pass them unmodified back to the server.
`CONTAINER_TICKET`	The contents of the MapR ticket encoded in base64. If you specified `secure` for the `securityType`, you must provide the ticket. To encode the ticket, see Converting a String to Base64 on page 3169. You may remove the ticket if the MapR cluster is not secure.

Example: Mounting a PersistentVolume for Static Provisioning Using the FlexVolume Driver

For static provisioning, configuring a PersistentVolume has some advantages over annotating Kubernetes volume information in a Pod spec:

- The configuration file can be shared for use by multiple Pod specs.
- The configuration file enables the PersistentVolume to be mounted and available even when the Pod spec that references it is removed.

last-updated: Apr 11, 2022

For example: A marketing volume exists in the MapR filesystem under the path `/Departments/Marketing`. An administrator wants to statically provision this volume and make it available to multiple users. It is critical that data access is as fast as possible. To make this work, the administrator must:

1. Create a PersistentVolume (PV).
2. Set the `AccessMode` of the PV to `ReadWriteOnce`.
3. Create a PersistentVolumeClaim (PVC) spec.
4. Set the `AccessMode` of the PVC to `ReadWriteOnce`.
5. Create the Pod spec.
6. Generate a MapR service ticket, and set the `flexVolume securityType` parameter to `secure`. For information about generating a service ticket, see Generating a Service Ticket on page 1428.
7. Configure a Ticket Secret, and include the base64-encoded contents of the ticket file in the Ticket Secret. See Configuring a Secret on page 3167.
8. Set the `runAsUser` and the `fsGroup` parameters to the UID and GID of the user that created the ticket.
9. Set the `platinum` parameter in the Pod spec to `platinum: "true"`. See Enabling the Platinum Posix Client for MapR Data Fabric for Kubernetes FlexVolume Driver on page 3166.
10. Point the `volumePath` in the `flexVolume` setting to the desired MapR path.
11. Fill in the `cldbHosts` and `cluster` information.

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-testsecure1
  namespace: mapr-examples
spec:
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  claimRef:
    namespace: mapr-examples
    name: pvc-testsecure1
  flexVolume:
    driver: "mapr.com/maprfs"
    options:
      platinum: "true"
      cluster: "mycluster"
      cldbHosts: "cldb1 cldb2 cldb3"
      volumePath: "/path/in/mapr"
      securityType: "secure"
      ticketSecretName: "mapr-ticket-secret"
      ticketSecretNamespace: "mapr-examples"
---
apiVersion: v1
kind: Pod
metadata:
  name: test-securepv
  namespace: mapr-examples
spec:
```


last-updated: Apr 11, 2022

```
  containers:
  - name: mycontainer
    image: myrepo/myorg/mycontainer
    args:
    - sleep
    - "1000000"
    resources:
      requests:
        memory: "2Gi"
        cpu: "500m"
    volumeMounts:
    - mountPath: /mapr
      name: maprvolume
  volumes:
    - name: maprvolume
      persistentVolumeClaim:
        claimName: pvc-testsecure1
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-testsecure1
  namespace: mapr-examples
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5G
---
apiVersion: v1
kind: Secret
metadata:
  name: mapr-ticket-secret
  namespace: mapr-examples
type: Opaque
data:
  CONTAINER_TICKET: <BASE64-ENCODED VERSION OF TICKET-FILE CONTENTS>
```

PersistentVolume (PV)

Parameter	Notes
`Capacity`	Specifies how big the allocated storage should be. This value is not validated against the MapR quota or advisory quota. It is up to the person creating the PV to specify this value accurately.
`accessModes`	How the PersistentVolume is mounted on the host. All modes work the same. The example uses `ReadWriteOnce`. Note that `ReadOnlyMany` does not mount read-only. (To implement a read-only mount, use the Flexvolume flag for `readOnly`, as shown in Example: Statically Provisioning a MapR Volume Using the FlexVolume Plug-in on page 3153.) It's important that the PV and PVC modes are the same so that they can bind. For more information, see Access Modes.

last-updated: Apr 11, 2022

`persistentVolumeReclaimPolicy`	Specifies what happens to the volume when it is released by its claim. The `Retain` value keeps the PVC around for manual cleanup. `Delete` deletes the PV from Kubernetes. **Note:** If this volume was created using dynamic provisioning, `Delete` causes the underlying volume to be deleted. `Recycle` is not supported by Kubernetes FlexVolumes. For more information, see Reclaiming.
`claimRef`	Specifies a default PVC to bind to. If unspecified, the PV selected for a PVC is randomly allocated based on the access mode and provides at least as much storage capacity as requested by the PVC.
`flexVolume: driver`	The MapR FlexVolume driver being used. Call it by specifying driver: `mapr.com/maprfs`.
`platinum`	If set to `platinum: "true"`, the POSIX client uses the platinum driver for better performance. Note that the platinum driver consumes more host resources and MapR Platinum licenses.
`cluster`	The MapR cluster name.
`cldbHosts`	The hostname or IP addresses of the CLDB hosts for the MapR cluster. You must provide at least one CLDB host. For fault-tolerance, providing multiple CLDB hosts is recommended. To specify multiple hosts, separate each name or IP address by a space.
`volumePath`	The mount point within the MapR filesystem. This parameter specifies an existing MapR path. For example, you can specify the root volume as `"/"`, providing access to the entire filesystem.
`securityType`	A parameter that indicates whether MapR tickets are used or not used. If MapR tickets are used, specify `secure`. Otherwise, specify `unsecure`.
`ticketSecretName`	The name of the Ticket Secret that contains the ticket to use when mounting to the MapR cluster. See Configuring a Secret on page 3167.
`ticketSecretNamespace`	The namespace that contains the Ticket Secret. Use the same namespace as the namespace used by the Pod.

Pod

Parameter	Notes
`apiVersion`	The Kubernetes API version for the Pod spec.
`kind`	The kind of object being created. The example uses a naked Pod for clarity. Generally, it is better to use a Deployment, DaemonSet, or StatefulSet for high availability and ease of upgrade.
`metadata: name`	The Pod name.
`metadata: namespace`	The namespace in which the Pod runs.
`volumeMounts: mountPath`	A directory inside the container that is designated as the mount path.

last-updated: Apr 11, 2022

`volumeMounts: name`	A name that you assign to the Kubernetes `volumeMounts` resource. This value should match `Volumes: name`.
`Volumes: name`	A string to identify the name of the Kubernetes `volumes` resource. This value should match `volumeMounts: name`.

PersistentVolumeClaim (PVC)

Parameter	Notes
`AccessMode`	How the requested PersistentVolume is mounted on the host. All modes work the same. The example uses `ReadWriteOnce`. Note that `ReadOnlyMany` does not mount read-only. (To implement a read-only mount, use the Flexvolume flag for `readOnly`, as shown in Example: Statically Provisioning a MapR Volume Using the FlexVolume Plug-in on page 3153.) It's important that the PV and PVC modes are the same so that they can bind. For more information, see Access Modes.

Secret

Parameter	Notes
`metadata: name`	The name of the Ticket Secret. See Configuring a Secret on page 3167
`metadata: namespace`	The namespace in which the Ticket Secret runs.
`CONTAINER_TICKET`	The contents of the MapR ticket encoded in base64. If you specified `secure` for the `securityType`, you must provide the ticket. To encode the ticket, see Converting a String to Base64 on page 3169. You may remove the ticket if the MapR cluster is not secure.

Example: Mounting a PersistentVolume for Dynamic Provisioning Using the FlexVolume Driver

This example also uses a PersistentVolume. However, unlike the previous example, when you use the MapR dynamic provisioner, you do not need to create a PersistentVolume manually. The PersistentVolume is created automatically based on the parameters specified in the referenced StorageClass.

Dynamic provisioning is useful in cases where you do not want MapR and Kubernetes administrators to create storage manually to store the Pod storage state.

The following example uses a PersistentVolumeClaim that references a Storage Class. In this example, a Kubernetes Administrator has created a storage class called `secure-maprfs` for Pod creators to use when they want to create persistent storage for their Pods. In this example, it is important for the created Pod storage to survive the deletion of a Pod.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
   name: secure-maprfs
   namespace: mapr-examples
provisioner: mapr.com/maprfs
parameters:
    restServers: "rest1:8443"
    cldbHosts: "cldb1 cldb2 cldb3"
    cluster: "mysecurecluster"
    securityType: "secure"
    ticketSecretName: "mapr-ticket-secret"
```


last-updated: Apr 11, 2022

```
    ticketSecretNamespace: "mapr-examples"
    maprSecretName: "mapr-provisioner-secrets"
    maprSecretNamespace: "mapr-examples"
    namePrefix: "pv"
    mountPrefix: "/pv"
    readOnly: "true"
    reclaimPolicy: "Retain"
    advisoryquota: "100M"
    readonly: "1"

---

kind: Pod
apiVersion: v1
metadata:
  name: test-secure-provisioner
  namespace: mapr-examples
spec:
  containers:
  - name: busybox
    image: busybox
    args:
    - sleep
    - "1000000"
    imagePullPolicy: Always
    volumeMounts:
      - name: maprfs-pvc
        mountPath: "/dynvolume"
  restartPolicy: "Never"
  volumes:
    - name: maprfs-pvc
      persistentVolumeClaim:
        claimName: maprfs-secure-pvc

---

kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: maprfs-secure-pvc
  namespace: mapr-examples
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: secure-maprfs
  resources:
    requests:
      storage: 300M
---

apiVersion: v1
kind: Secret
metadata:
  name: mapr-provisioner-secrets
  namespace: mapr-examples
type: Opaque
data:
  MAPR_CLUSTER_USER: CHANGETHIS!
  MAPR_CLUSTER_PASSWORD: CHANGETHIS!

---

apiVersion: v1
kind: Secret
```


last-updated: Apr 11, 2022

```
metadata:
  name: mapr-ticket-secret
  namespace: mapr-examples
type: Opaque
data:
  CONTAINER_TICKET: <BASE64 ENCODED VERSION OF CONTENTS OF TICKET FILE>
```

The following tables describe the parameters in the example:

StorageClass

Parameter	Notes
`apiVersion`	The Kubernetes APi version for the StorageClass spec.
`kind`	The kind of object being created. This is a StorageClass.
`metadata: name`	The name of the StorageClass. Administrators should specify the name carefully because it will be used by Pod authors to help select the right StorageClass for their needs.
`metadata: namespace`	The namespace in which the StorageClass runs. This namespace can be different from the namespace used by the PVC and Pod, since the StorageClass namespace can be a cross-namespace resource.
`provisioner`	The provisioner being used. for the MapR provisioner, specify `mapr.com/maprfs`.
`restServers`	A space-separated list of MapR webservers. Specify the hostname or IP address and port number of each REST server for the MapR cluster. For fault tolerance, providing multiple REST server hosts is recommended.
`cldbHosts`	The hostname or IP addresses of the CLDB hosts for the MapR cluster. You must provide at least one CLDB host. For fault-tolerance, providing multiple CLDB hosts is recommended. To specify multiple hosts, separate each name or IP address by a space.
`cluster`	The MapR cluster name.
`securityType`	A parameter that indicates whether MapR tickets are used or not used. If MapR tickets are used, specify `secure`. Otherwise, specify `unsecure`.
`ticketSecretName`	The name of the Secret that contains the ticket to use when mounting to the MapR cluster. See Configuring a Secret on page 3167.
`ticketSecretNamespace`	The namespace that contains the Secret. Use the same namespace as the namespace used by the Pod.
`maprSecretName`	The name of the Kubernetes Secret that is used to store MapR administrative credentials (user, password, and ticket information for the MapR webserver). To use the provisioner, you must configure a Secret. See Configuring a Secret on page 3167.
`maprSecretNamespace`	The namespace for the Secret containing the MapR administrative credentials (user name and password information for a MapR user that has the privileges to create MapR volumes). This namespace can be different from the namespace used by the Pod, since a

last-updated: Apr 11, 2022

Parameter	Notes
	Pod author or namespace admin might not be trusted to create administration Secrets for the MapR cluster.
`namePrefix`	A prefix for the MapR volume to be created. For example, if you specify `PV` as the `namePrefix`, the first dynamically created volume might be named `PV.bevefsescr`. The provisioner generates random names using lower-case letters. If you do not specify a prefix, the provisioner uses `maprprovisioner` as a prefix.
`mountPrefix`	The parent path of the mount in MapR filesystem. If you do not specify a mount prefix, the provisioner mounts your volume under the MapR root.
`readOnly`	This parameter specifies that the POSIX driver should mount the MapR path as read only. This is different from the `readonly` parameter for volume creation that creates the volume as read only.
`reclaimPolicy`	Kubernetes does not currently support passing a non-delete reclaim policy to the StorageClass. This parameter allows you to specify `Retain`. This ensures that provisioned volumes are not automatically deleted when their calling Pods are deleted. If you specify `Retain`, you must clean up your provisioned volumes manually.
`advisoryquota`	The advisory storage quota for the MapR volume. `advisoryquota` is one of the MapR parameters that you can specify for dynamic provisioning. For more information, see Before You Begin on page 3151.
`readonly`	When the value is `1`, this parameter specifies that the MapR volume should be created as read-only. This is different from the `readOnly` parameter that mounts the MapR path as read only.

Pod

Parameter	Notes
`apiVersion`	The Kubernetes API version for the Pod spec.
`kind`	The kind of object being created. For clarity, this example uses a naked Pod. Generally, it is better to use a Deployment, DaemonSet, or StatefulSet for high availability and ease of upgrade.
`metadata: name`	The Pod name.
`metadata: namespace`	The namespace in which the Pod runs. It should be the same namespace in which the PVC runs.
`volumeMounts: mountPath`	A directory inside the container that is designated as the mount path.
`volumeMounts: name`	A name that you assign to the Kubernetes `volumeMounts` resource. The value should match `Volumes: name`.
`Volumes: name`	A string to identify the name of the Kubernetes `volumes` resource. The value should match `volumeMounts: name`.

last-updated: Apr 11, 2022

Parameter	Notes
`persistentVolumeClaim: claimName`	The name of the PersistentVolumeClaim (PVC). For more information, see PersistentVolumeClaims.

PVC

Parameter	Notes
`apiVersion`	The Kubernetes API version for the Pod spec.
`kind`	The kind of object being created. This is a PersistentVolumeClaim (PVC).
`metadata: name`	The PVC name.
`metadata: namespace`	The namespace in which the PVC runs. This should be the same namespace used by the Pod.
`accessModes`	How the PersistentVolume is mounted on the host. (This is a limitation of the FlexVolume driver.) For more information, see Access Modes.
`storageClassName`	The name of the storage class requested by the PersistentVolumeClaim. For more information, see Dynamic Provisioning and Storage Classes.
`requests: storage`	The storage resources being requested, or that were requested and have been allocated. The Pod author can use this parameter to tell MapR how much quota is needed for the MapR volume. For the units, see Resource Model.

Provisioner Secret

In the mapr-provisioner-secrets Secret:

Parameter	Notes
`MAPR_CLUSTER_USER`	This is the base64-encoded user ID used to log in to the MapR REST server and create or delete volumes. See Converting a String to Base64 on page 3169. For more information about Secrets, see Secrets.
`MAPR_CLUSTER_PASSWORD`	This is the base64-encoded password for the MAPR_CLUSTER_USER. See Converting a String to Base64 on page 3169. For more information about Secrets, see Secrets.

Ticket Secret

In the mapr-ticket-secret Secret:

Parameter	Notes
`CONTAINER_TICKET`	The contents of the MapR ticket encoded in base64. If you specified `secure` for the `securityType`, you must provide the ticket. To encode the ticket, see Converting a String to Base64 on page 3169. You may remove the ticket if the MapR cluster is not secure. For more information about Secrets, see Secrets.

Identifying the MapR Volume Created During Dynamic Provisioning

Describes how to find the name of the MapR volume created during dynamic provisioning.

last-updated: Apr 11, 2022

In dynamic provisioning, the provisioner creates a new MapR volume with a name that is randomly generated using lower-case letters. For example, if you specify `PV` as the `namePrefix` in the StorageClass, the first dynamically created volume might be named `PV.bevefsescr`. If you do not specify a prefix, the provisioner uses `maprprovisioner` as a prefix.

To find the name of the new MapR volume and the path to the volume:

1. Use the `kubectl describe` command to get information about the PVC:

   ```
   kubectl describe pvc -n <namespace> <pvc-name>
   ```

   The command output shows the name of the PersistentVolume (PV) that was created: For example:

   ```
   kubectl describe pvc -n mapr-examples maprfs-secure-pvc109
   Name:          maprfs-secure-pvc109
   Namespace:     mapr-examples
   StorageClass:  secure-maprfs
   Status:        Bound
   Volume:        pv-ikmqxfwtjh
   Labels:        <none>
   Annotations:   control-plane.alpha.kubernetes.io/
   leader={holderIdentity":"ed60e649-0c68-11e8-acd5-36117e0e7e02","leaseDura
   tionSeconds":15, "acquireTime":"2018-02-09T22:09:43Z"
                  pv.kubernetes.io/bind-competed=yes
                  pv.kubernetes.io/bound-by-controller=yes
                  volume.beta.kubernetes.io/storage-provisioner=mapr.com/
   maprfs
   Finalizers:    []
   Capacity:      300M
   Access Modes:  RWO
   Events:
   ```

2. Use the `kubectl get` command and the PersistentVolume (PV) name to obtain a description of the PersistentVolume:

   ```
   kubectl get pv <pv-name> -o yaml
   ```

   The command output shows the path to the MapR volume. For example:

```
kubectl get pv pv-ikmqxfwtjh -o yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  annotations:
    mapr.com/description: 'Dynamically provisioned PV for MapR-FS:
pv.ikmqxfwtjh'
    mapr.com/maprProvisionerIdentity: mapr.com/maprfs
    mapr.com/provisionerVersion: v1.0.0
    mapr.com/restServers: 10.10.88.214:8443
    mapr.com/secretName: mapr-provisioner-secrets
    mapr.com/secretNamespace: mapr-examples
    mapr.com/volumeName: pv.ikmqxfwtjh
    pv.kubernetes.io/provisioned-by: mapr.com/maprfs
  creationTimestamp: 2018-02-09T22:21:22Z
  name: pv-ikmqxfwtjh
  resourceVersion: "2875820"
  selfLink: /api/v1/persistentvolumes/pv-ikmqxfwtjh
  uid: 8f11aa11-0de7-11e8-bdd6-84a9c4fbf7cb
spec:
  accessModes:
  - ReadWriteOnce
  capacity:
    storage: 300M
  claimRef:
    apiVersion: v1
    kind: PersistentVolumeClaim
    name: maprfs-secure-pvc109
    namespace: mapr-examples
    resourceVersion: "2842548"
    uid: ce5555e4-0de5-11e8 bdd6-84a9c4fbf7cb
  flexVolume:
    driver: mapr.com/maprfs
    options:
      cldbHosts: xx.xx.xx.xxx yy.yy.yy.yyy zz.zz.zz.zzz
      cluster: Test5
      mountOptions: ""
      platinum: "true"
      readOnly: "false"
      securityType: secure
      ticketSecretName: mapr-ticket-secret
      ticetSecretNamespace: mapr-examples
      volumePath: /pv/pv-ikmqxfwtjh
  persistentVolumeReclaimPolicy: Delete
  storageclassName: secure-maprfs
status:
  phase: Bound
```

Creating a Default StorageClass

As noted in Example: Statically Provisioning a MapR Volume Using the FlexVolume Plug-in on page 3153, some deployments can require a default StorageClass. A default StorageClass can reduce the effort it takes to create Pods. For example, you could use a default StorageClass to provision storage dynamically to a MapR location for any PersistentVolumeClaim that you create.

last-updated: Apr 11, 2022

If you set the `DefaultStorageClass` admission controller (see PodSecurityPolicy), and you wish to enable a MapR StorageClass as the default, follow the instructions in Change the default StorageClass.

Verifying Creation of a Kubernetes PersistentVolumeClaim and Persistent Volume

Once the Pod spec is installed, you can verify the status of a PersistentVolumeClaim or a PersistentVolume by using the Kubernetes `get` command. For example:

```
$ kubectl get pvc
NAME        STATUS  VOLUME         CAPACITY  ACCESS MODES  STORAGECLASS
AGE
Maprfs-pvc  Bound   pv-rsojpoapxy  8Mi       RWO           simple-maprfs  3d
$ kubectl get pv
NAME           CAPACITY ACCESS MODES  RECLAIM POLICY STATUS
CLAIM             ...
Pv-rsojpoapxy  8Mi      RWO           Delete         Bound  mapr-demo/
maprfs-pvc
```

For an example of creating a PersistentVolumeClaim and a PersistentVolume, see Example: Mounting a PersistentVolume for Static Provisioning Using the FlexVolume Driver on page 3155.

Enabling the Platinum Posix Client for MapR Data Fabric for Kubernetes FlexVolume Driver

When you install the MapR Data Fabric for Kubernetes FlexVolume Driver, the Basic FUSE-based POSIX client package is installed on all nodes by default. The FlexVolume Driver also supports the use of the Platinum FUSE-based POSIX client. For a comparison of the two POSIX client packages, see Preparing for Installation (MapR POSIX Client) on page 400.

To install the Platinum POSIX client, include the `platinum` parameter in your Pod spec. For example:

```
options:
cluster: "cluster2"
platinum: "true"
cldbHosts: "10.10.102.96"
```

Mounting a Read-Only Volume

This page describes how to specify a volume that should be mounted as read-only.

The following example specifies a volume that should be mounted as read-only in a MapR path or PersistentVolume during static provisioning:

```
flexVolume:
        driver: "mapr.com/maprfs"
        readOnly: true
        options:
          volumePath: "/path/to/data/in/mapr"
          cluster: "mycluster"
          cldbHosts: "cldb1 cldb2 cldb3"
          securityType: "secure"
          ticketSecretName: "mapr-ticket-secret"
          ticketSecretNamespace: "mapr-examples"
```

The following example shows how to specify that the volume should be mounted read-only in a StorageClass for dynamic provisioning. The example specifies that the POSIX driver should mount the MapR path as read only. This is different from the `readonly` parameter for volume creation that creates the volume as `readonly`.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
   name: secure-maprfs
   namespace: mapr-examples
```


last-updated: Apr 11, 2022

```
provisioner: mapr.com/maprfs
parameters:
    restServers: "rest1:8443"
    cldbHosts: "cldb1 cldb2 cldb3"
    cluster: "mysecurecluster"
    securityType: "secure"
    ticketSecretName: "mapr-ticket-secret"
    ticketSecretNamespace: "mapr-examples"
    maprSecretName: "mapr-provisioner-secrets"
    maprSecretNamespace: "mapr-examples"
    namePrefix: "pv"
    mountPrefix: "/pv"
    readOnly: "true"
    reclaimPolicy: "Retain"
    advisoryquota: "100M"
```

The following example specifies that the volume should be created as `readOnly` in a StorageClass for dynamic provisioning:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
   name: secure-maprfs
   namespace: mapr-examples
provisioner: mapr.com/maprfs
parameters:
    restServers: "rest1:8443"
    cldbHosts: "cldb1 cldb2 cldb3"
    cluster: "mysecurecluster"
    securityType: "secure"
    ticketSecretName: "mapr-ticket-secret"
    ticketSecretNamespace: "mapr-examples"
    maprSecretName: "mapr-provisioner-secrets"
    maprSecretNamespace: "mapr-examples"
    namePrefix: "pv"
    mountPrefix: "/pv"
    reclaimPolicy: "Retain"
    advisoryquota: "100M"
    readonly: "1"
```

Configuring a Secret

Kubernetes Secrets enable you to inject sensitive data into a pod. For more information about Secrets, see Secrets.

The examples in this section show how Secrets can be used in static and dynamic provisioning. Secrets are not by themselves secure. For more information about security and Secrets, see Security Properties. Specifically, it is important to turn on encryption at rest for Secrets. See Encrypting Secret Data at Rest.

During installation of the Driver, the Kubernetes token that was moved into the pod is written to the host node so that the plugin can query a Secret to pull the ticket for mounting. This Kubernetes token is sensitive and should be protected. The token is placed in `/var/run/secrets/kubernetes.io/serviceaccount`.

Here is an example of a configuration file for a Kubernetes Secret:

```
apiVersion: v1
kind: Secret
metadata:
  name: mapr-provisioner-secrets
  namespace: test-driver
type: Opaque
```


last-updated: Apr 11, 2022

```
data:
  ...
```

The following table describes the fields in the sample Secret file. For more information, see Secrets in the Kubernetes documentation.

Parameter	Notes
`apiVersion`	The Kubernetes API version.
`kind`	The type of object being created.
`name`	A string to identify the Secret.
`type`	The type of Secret being created. For type `Opaque`, clients must treat these values as opaque and pass them unmodified back to the server.

REST Secrets

For dynamic provisioning, you must use a Secret to pass the user name and password of a MapR user to the provisioner. This user must have privileges to create and delete a MapR volume. The credentials allow the provisioner to make REST calls to the MapR webserver. Secrets are protected by the Kubernetes RBAC.

The following example shows a REST secret in the Secret file:

```
apiVersion: v1
kind: Secret
metadata:
  name: mapr-provisioner-secrets
  namespace: test-driver
type: Opaque
data:
 MAPR_CLUSTER_USER: cm9vdA==
  MAPR_CLUSTER_PASSWORD: bWFwcg==
```

The following table describes the REST secret fields in the REST Secret example.

Parameter	Notes
`MAPR_CLUSTER_USER`	The base64 representation of a MapR user that has the ability to create and delete MapR volumes. See Converting a String to Base64 on page 3169.
`MAPR_CLUSTER_PASSWORD`	The base64 representation of the password for the user defined by the `MAPR_CLUSTER_USER` parameter. See .Converting a String to Base64 on page 3169
`MAPR_CLUSTER_TICKET`	The base64 representation of the ticket contents generated on the data-fabric cluster using the `maprlogin` utility. For dynamic provisioning, with the latest CSI drivers, you can configure a data-fabric ticket to authenticate to the data-fabric webserver to make REST calls. This parameter is provided as a Beta feature.

Ticket Secrets

For static and dynamic provisioning, you must specify a Secret, which is the base64 representation of the ticket, to enable the POSIX client to communicate with a secure MapR cluster. The ticket for the POSIX client can be generated on the MapR cluster using the `maprlogin on page 2130` utility.

last-updated: Apr 11, 2022

The following example shows a ticket Secret:

```
apiVersion: v1
kind: Secret
metadata:
  name: mapr-ticket-secret
  namespace: mapr-examples
type: Opaque
data:
  CONTAINER_TICKET: CHANGETHIS!
```

The following table describes the `CONTAINER_TICKET` field in the ticket Secret example.

Parameter	Notes
`CONTAINER_TICKET`	Base64-encoded ticket value. See Converting a String to Base64 on page 3169.

To create the secret:

1. Run the following command to create the Secret file:

   ```
   kubectl create -f <secret-file-name>.yaml
   ```

2. Convert sensitive data, such as a user name and password, to a base64 representation.

   See Converting a String to Base64 on page 3169.

3. Add the base64 representation of sensitive data in the Secret file.

   For more information about the format of the Secret files, see REST Secrets on page 3168 and Ticket Secrets on page 3168 earlier in this section.

4. Deploy the secret on the pod by running the following command:

   ```
   kubectl apply -f <secret-file-name>.yaml
   ```

Converting a String to Base64

Sensitive data contained in a Secret must be represented in base64. Use these steps to convert such information to the base64 representation:

For example, in Linux:

```
echo -n 'mapr' | base64
```

The output shows the base64 representation of the user name `mapr` is `bWFwcg==`.

MapR tickets include a cluster name followed by a base64-encoded string. It is not sufficient to insert the base64-encoded string into a Kubernetes Secret. You must convert *both* the cluster name and string into base64 representation and then insert the result into the Secret.

The following command shows how to convert a MapR ticket to base64 representation:

```
echo -n "cluster-name <base64-encoded ticket-value>" | base64
```

For example:

```
echo -n "cluster2 PuG0lpuPXuDxj9ERgKCTXOqsXYPTnqRJl6/
mlWJjdVKvE5r46QS2Bh9nC+I4Rcu0GtnWRUOtKBG9gp65bsZN9Kphnr/
Wp15z8D3O2go951CANes/
7QQllYVP7l2BOpGR6I1zIrC3XGwI8OQWT61qpsjSVZv8zO5oQ5GDYQTkPttI/yAk/
```


last-updated: Apr 11, 2022

```
uJBES1ohCz38n9HgYALLvMALVsBPtUtG+cNGc1ktUDDMR2q1EgVzdJbuYsOuHnZX3LO3euKDGl4C
4MCmrv9DWiWJxwiZ1yZu69GbZJlXxqLOQBlkdMoTXk=" | base64

Y2x1c3RlcjIgUHVHMGxwdVBYdUR4ajlFUmdLQ1RYT3FzWFlQVG5xUkpsNi9tbFdKamRWS3ZFNXI0
NlFTMkJoOW5DK0k0UmN1MEd0bldSVU90S0JHOWdwNjVic1pOOUtwaG5yL1dwMTV6OEQzTzJnbzk1
MUNBTmVzLzdRUWxsWVZQN2wyQk9wR1I2STF6SXJDM1hHd0k4T1FXVDYxcXBzalNWWnY4ek81b1E1
R0RZUVRrUHR0SS95QWsvdUpCRVMxb2hDejM4bjlIZ1lBTEx2TUFMVnNCUHRVdEcrY05HYzFrdFVE
RE1SMnExRWdWemRKYnVZc091SG5aWDNMTzNldUtER2w0QzRNQ21ydjlEV2lXSnh3aVoxeVp1NjlH
YlpKbFh4cUxPUUJsa2RNb1RYaz0K
```

Note: Another method for converting values to base64 is to use an Internet tool such as https://www.base64encode.org to encode or decode data.

Best Practices for Using Tickets

When using secure MapR clusters with the Kubernetes Interfaces for Data Fabric, you must generate tickets for your containers. Here are some best practices:

- Create a different user for each container.
- To avoid frequent renewals, use long-lived user tickets or servicewithimpersonation tickets. If you refresh or update a ticket, you must restart your containers.
- If you use an impersonation ticket, it is CRITICAL that you use security contexts in the pod definitions to avoid a misbehaving container impersonating all user IDs. For restrictions that apply to the use of impersonation tickets, see How Impersonation Works on page 1478 and maprlogin on page 2130.
- Match the security context `runAsUser:` ID and `fsGroup:` group to the ID or group used to create the ticket.

Here is an example of a pod spec that specifies a security context:

```
apiVersion: v1
kind: Pod
metadata:
  name: test-secure
  namespace: mapr-examples
spec:
  securityContext:
    runAsUser: 1000
    fsGroup: 2000
```

Troubleshooting the MapR Data Fabric for Kubernetes FlexVolume Driver
This section describes how to resolve common problems you might encounter when using the MapR Data Fabric for Kubernetes FlexVolume driver.

Shared Memory Lock Causes POSIX Failure

Problem	On an upgrade from a previous version of the volume plug-in, POSIX can fail with the following error in the POSIX log file: `Create/Attach to stats shared memory failed.`
Possible Cause	A shared-memory segment lock can prevent the mount from becoming available to the requested pod.
Resolution	Follow the steps in Troubleshooting MapR loopbacknfs POSIX Client Upgrades on page 329 to remove the lock. Then retry the operation.

Unable to Access MapR File System

Problem	Storage is not mounted and no errors are generated in the plugin or provisioner logs.
Possible Cause	The fusermount symlink might be broken.
Resolution	If the symlink points to a location other than `/opt/mapr/k8s/bin/fusermount`, unlink it using the following command from the command line on the host: `unlink /bin/fusermount` Then re-create the Kubernetes Pod.

Pod Container Stuck in Container Creation State During Installation

Problem	During installation, the Pod container can become stuck in the container creation state on a node, and the`/opt/mapr/k8s` directory is not created. As a result, the plug-in does not get copied to the node.
Possible Cause	Unknown.
Resolution	Check the installation logs for an indication that the installation is not completed or the `/opt/mapr/k8s` directory is not created. Restart the kubelet service in the node: `systemctl restart kubelet`

Logs for the MapR Data Fabric for Kubernetes FlexVolume Driver

Logs for the MapR Data Fabric for Kubernetes can be found in:

```
/opt/mapr/logs
```

Log File	Description	Which Nodes
`install-k8s-plugin.log`	Captures events related to the copying of files from the plug-in container to each Kubernetes host node.	All Kubernetes nodes.
`plugin-k8s.log`	Captures events from the FlexVolume plug-in.	All Kubernetes nodes. **Note:** In Azure deployments, this log is hidden in the container. See Azure AKS Considerations on page 245.
`provisioner-k8s.log`	Captures events from the provisioner.	The Kubernetes node where the provisioner Pod is running.

Useful Troubleshooting Commands

The following Kubernetes commands can help you gather information about the resources used by the MapR Data Fabric for Kubernetes:

last-updated: Apr 11, 2022

- `kubectl describe <resourcetype> <resource> -n <namespace>`
- `kubectl get <resourcetype> <resource> -n <namespace> -o yaml`
- `kubectl logs <pod-name> -n <namespace>`
- `journalctl -u kubelet -r` (on the relevant node)

kubectl describe command

In this example, the `kubectl describe` command displays information about the `mapr-kdfprovisioner-5dff68656-ln6vh` Pod. Note that the `kubectl describe` output includes an event section.

```
kubectl describe pod mapr-kdfprovisioner-5dff68656-ln6vh -n mapr-system
Name:         mapr-kdfprovisioner-5dff68656-ln6vh
Namespace:    mapr-system
Node:         qa101-139/10.10.101.139
Start Time:   Fri, 09 Feb 2018 12:58:36 -0800
Labels:       app=mapr-kdfprovisioner
              pod-template-hash=189924212
Annotations:  openshift.ix/scc-maprkdf-scc
Status:       Running
IP:           172.17.0.3
 .             .
 .             .
 .             .
Node-Selectors:  <none>
Tolerations:     <none>
Events:
  Type    Reason                 Age    From              Message
  ----    -----                  ---     ----              ---------
  Normal  Scheduled              8m    default-scheduler  Successfully
assigned mapr-kdfprovisioner-5dff68656-ln6vh to qa101-139
  Normal  SuccessfulMountVolume 8m    kubelet, qa101-139 MountVolume,SetUp
succeeded for volume "logs"
  Normal  SuccessfulMountVolume 8m    kubelet, qa101-139 MountVolume,SetUp
succeeded for volume "timezone"
  Normal  SuccessfulMountVolume 8m    kubelet, qa101-139 MountVolume,SetUp
succeeded for volume "maprkdf-token-drqtt"
  Normal  Pulling                8m    kubelet, qa101-139 pulling image
"maprtech/kdf-provisioner:1.0.0.006_centos7"
  Normal  Pulled                 8m    kubelet, qa101-139 Successfully
pulled image "maprtech/kdf-provisioner:1.0.0.006_centos7"
  Normal  Created                8m    kubelet, qa101-139 Created container
  Normal  Started                8m    kubelet, qa101-139 Started container
```

kubectl get command

In this example, the `kubectl get` returns the `.yaml` parameters for the `test-secure-provisioner86` Pod:

```
kubectl get pods test-secure-provisioner86 -n mapr-examples -o yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: 2018-02-09T00:42:06Z
  name: test-secure-provisioner86
  namespace: mapr-examples
  resourceVersion: "721689"
  selfLink: /api/v1/namespaces/mapr-examples/pods/test-secure-provisioner86
```

```
  uid: 0dd21274-0d32-11e8-bdd6-84a9c4fbf7cb
spec:
  containers:
  - args:
    - sleep
    - "1000000"
    image: busybox
    imagePullPolicy: Always
    name: busybox
    resources: {}
    terminationMessagePath: /dev/termination-log
    terminationMessagePolicy: file
    volumeMounts:
    - mountPath: /dynvolume
      name: maprfs-pvc
    - mountPath: /var/run/secrets/kubernetes.io/serviceaccount
      name: default-token-zpv69
      readOnly: true
  dnsPolicy: ClusterFirst
  nodeName: qa108-165.qa.lab
  restartPolicy: Never
  schedulerName: default-scheduler
  securityContext: {}
  serviceAccount: default
  serviceAccountName: default
  terminationGracePeriodSeconds: 30
  tolerations:
  - effect: NoExecute
    key:node.kubernetes.io/not-ready
    operator: Exists
    tolerationSeconds: 300
  - effect: NoExecute
    key: node.kubernetes.io/unreachable
    operator: Exists
    tolerationSeconds: 300
  volumes:
  -name: maprfs-pvc
```

Running the `kubectl get` command without the `-o yaml` parameter generates less output:

```
kubectl get pods test-secure-provisioner86 -n mapr-examples
NAME                        READY    STATUS   RESTARTS   AGE
test-secure-provisioner86  1/1      Running  0          14m
```

kubectl logs command

In this example, the `kubectl logs` command returns logged output for the `mapr-kdfprovisioner-5dff68656-ln6vh` Pod:

```
kubectl logs mapr-kdfprovisioner-5dff68656-ln6vh -n mapr-system
I0209 12:58:39.956822    1 controller.go:407] Starting provisioner
controller 013d58b3-0ddc-11e8-b0dd-0242acl10003!
```

journalctl -u command

In this example, the `journalctl` command returns events for the kubelet service for the node:

```
journalctl -u kubelet -r
-- Logs begin at Thu 2017-12-28 06:24:47 PST, end at Thu 2018-02-08
17:01:49 PST. --
Feb 08 17:01:49 k8s-master kubelet[26521]: E0206 17:01:49,047595  26521
```


last-updated: Apr 11, 2022

```
dns.go:121] Search Line limits were exceeded, some search paths have been
omitted, the applied search line
Feb 08 17:01:45 k8s-master kubelet[26521]: E0206 17:01:45,396253  26521
dns.go:180] CheckLimitsForResolvConf: Resolv,conf file '/etc/resolve.conf'
contains search line consisting
Feb 08 17:01:15 k8s-master kubelet[26521]: E0206 17:01:15,396023  26521
dns.go:100] CheckLimitsForResolvConf: Resolv,conf file '/etc/resolve.conf'
contains search line consisting
Feb 08 17:00:48 k8s-master kubelet[26521]: E0206 17:00:48,047555  26521
dns.go:121] Search Line limits were exceeded, some search paths have been
omitted, the applied search line
 .             .
 .             .
 .             .
```

Ecosystem Components

The following sections provide information about each open-source project that is supported by the MapR Data Platform.

This section contains documentation for each open-source project. You can learn how to configure, use, and integrate each project within the context of a MapR cluster.

Documentation Covers All Component Versions

Unless noted, the ecosystem-component information in this content hierarchy applies to all component versions included in EEPs that are supported on the core software. For a list of the supported EEPs, see EEP Support and Lifecycle Status on page 5531.

MapR Ecosystem Packs

A MapR Ecosystem Pack (EEP) provides a set of ecosystem components that work together on one or more MapR cluster versions. Each EEP contains only one version of an ecosystem component. For example, each EEP supports only one version of Hive and one version of Spark.

HPE creates a new EEP version when a new ecosystem component is available or a patch is applied to an ecosystem component that is already in a released EEP.

A single version of core can support multiple EEPs, but only one at a time. For detailed information about each EEP, see MapR Ecosystem Pack (EEP) Reference on page 6504. For a list of currently supported EEPs, see EEP Support and Lifecycle Status on page 5531.

Hadoop Ecosystem and Monitoring Components

Hadoop ecosystem components within an EEP undergo extensive interoperability testing to validate that the components can work together. Examples of Hadoop ecosystem components include Hive, Pig, Spark, and Oozie.

The following open-source components are included in the EEP for monitoring and logging use cases, but NOT for third-party use cases:

- Collectd
- Elasticsearch
- Fluentd
- Grafana
- Kibana

- OpenTSDB

Note: Developer previews for fast-moving ecosystem components continue to be available in addition to EEPs. However, developer preview releases of ecosystem components are not tested for production environments, and they do not undergo the same interoperability testing.

For more information, see the EEP Release Notes on page 5658.

AsyncHBase

MapR provides a version of AsyncHBase that is modified to work with MapR Database binary tables. The MapR version of AsyncHBase is based on the AsyncHBase library provides asynchronous Java APIs to access MapR Database binary tables. The HBase Client version is based on the current EEP and MapR version you are running. For more information, see the Interoperability Matrices on page 5519.

Configuring the Default Database for AsyncHBase

For AsynchHBase 1.7 and later, you can configure whether AsyncHBase accesses HBase tables or MapR-DB tables by default. If this value is not configured, AsyncHBase will determine the table type.

You can configure the default database in the `asynchbase.conf` file and the client application. A default database setting in the client application overrides the default database configuration in the `aynchbase.conf` file.

The process that AsyncHBase uses to access tables differs based on the default database configuration.

- When the default database is HBase, AsyncHBase accesses the table using the HBase port that was used to initialize the AsyncHBase client and the table name provided to the application.
- When the default database is MapR-DB, the table name provided to the application is translated to the MapR-DB table path, and then AsyncHBase accesses the table.
- When a default database is not configured, AsyncHBase first tries to access the table as a MapR-DB table. If that fails, it tries to access the table as an HBase table.

Set the Default Database using asynchbase.conf

To specify if AsyncHBase accesses HBase tables or MapR-DB tables:

1. Add the `mapr.hbase.default.db` parameter in the `asynchbase.conf` (`/opt/mapr/asynchbase/asynchbase-<version>/conf/asynchbase.conf`) file.
2. Set the value of `mapr.hbase.default.db` to one of the following values which will indicate the default database:
   - `hbase`
   - `maprdb`

Set the Default Database in the Client Application

Based on the database that you want as the default, add the following code in the client application:

- To access MapR-DB tables:

```
Config config = new Config();
String dbString = "maprdb";
config.overrideConfig(HBaseClient.CONFIG_PARAM_DEFAULT_DB,dbString);
HBaseClient client = new
HBaseClient(config);
```


last-updated: Apr 11, 2022

- To access HBase tables:

```
Config config = new Config();
String dbString = "hbase";
config.overrideConfig(HBaseClient.CONFIG_PARAM_DEFAULT_DB,dbString);
HBaseClient client = new
HBaseClient(config);
```

Compiling and Running AsyncHBase Applications

When you compile or run AsyncHBase applications, you need to include the required AsyncHBase libraries.

To compile the application:

```
javac -cp `asynchbase
classpath`:$APP_CLASSPATH
<ProgramName>
```

To run the application, use one of the following commands:

- ```
  java -cp `asynchbase
  classpath`:$APP_CLASSPATH
  <ProgramName>
  ```

- ```
  asynchbase $APP_CLASSPATH
  <ProgramName>
  ```

To include the AsyncHBase library in your maven project:

1. Add MapR's maven repository to the list of repositories in your project's pom.xml:

   ```
   <repository>
   <id>mapr-releases</id>
   <url>https://repository.mapr.com/
   maven/</url>
   <snapshots><enabled>true</
   enabled></snapshots>
   <releases><enabled>true</enabled></
   releases>
   </repository>
   ```

2. Add the following dependency to the list of dependencies:

   ```
   <dependency> <groupId>org.hbase</
   groupId>
   <artifactId>asynchbase</
   artifactId>
   <version><AsyncHBaseVersion>-mapr-<
   MapREcoVersion></version> </
   dependency>
   ```

   Note: For example, if you are using AsyncHBase 1.7-1603, configure the following for the version dependency: `<version>1.7.0-mapr-1603</version>`

AsyncHBase Script

MapR provides an AsyncHBase script that you can use to run applications and generate the AsyncHBase classpath.

The asynchbase script has the following syntax:

```
asynchbase <command> [<args>]
Commands:
classpath Dump AsyncHBase CLASSPATH
CLASSNAME Run the class named CLASSNAME
```

Parameters	Description
classpath	Dumps the AsyncHBase classpath. For example, you can use `aynchbase classpath` when you compile an application: `javac -cp `asynchbase classpath`:$APP_CLASSPATH <ProgramName>`
CLASSNAME	Runs the named class. For example: `asynchbase <path to application> CLASSNAME`

AsyncHBase Behavior with MapR Database Binary Tables

After you install AsyncHBase, you can use the AsyncHBase libraries to provide asynchronous access to MapR Database binary tables. However, it is important to note the behavior that is specific to using AsyncHBase with MapR Database.

The `Scanner.setMaxNumKeyValues` method, when run against MapR Database binary tables, does not behave as documented.	According to the AsyncHBase documentation, this method sets "the maximum number of KeyValues the server is allowed to return in a single RPC response." When you use this method with MapR Database binary tables, the value for the maximum number of key values is ignored and the full set of KeyValues is always returned.
List<RegionClientStats> regionStats() is not supported	As of AsyncHBase 1.7-1603, `List<RegionClientStats> regionStats()` is not supported and when it is used the API does not return statistics.
MapR Database ignores HBase configurations in the asynchbase.conf file	As of AsyncHBase 1.7-1603, the conf object can be used to override Hbase properties that were previously only configured in the asynchbase.conf file. The asynchbase.conf file is located in the asynchbase installation directory. MapR Database does not use these Hbase configurations and therefore they are ignored by MapR Database

Using OpenTSDB with AsyncHBase

OpenTSDB can use MapR's AsyncHBase to perform time-series data-plots on MapR Database binary tables.

The OpenTSDB software package provides a time-series database that collects user-specified data.

To use OpenTSDB with AsyncHBase, install and configure OpenTSBD from source files or from a package.

Installing OpenTSDB from Source Files
The following steps describe how to install OpenTSDB from source files.

Be sure to install the OpenTSDB version that is required for your AsyncHBase version. AsyncHBase 1.6 requires OpenTSDB 2.0. AsyncHBase 1.7 requires OpenTSDB 2.2.

1. Clone the opentsdb.git project and check out the OpenTSDB branch that you require.

   Note:

   For example:

   ```
   $ git clone https://github.com/OpenTSDB/opentsdb.git
   Cloning into 'opentsdb'...
   remote: Counting objects: 5625, done.
   remote: Compressing objects: 100% (76/76), done.
   remote: Total 5625 (delta 51), reused 64 (delta 30)
   Receiving objects: 100% (5625/5625), 27.15 MiB | 2.67 MiB/s, done.
   Resolving deltas: 100% (3755/3755), done.
   Checking connectivity... done.
   $ cd opentsdb
   $ git tag -l
   mapr-1.1.0-release+5
   v1.0.0
   v2.0.0
   ...
   $ git checkout v2.0.0
   Switched to a new branch 'v2.0.0'
   ```

2. Install dependencies for graph generation:

   ```
   $ yum install autoconf automake gnuplot
   ```

3. Replace the `asynchbase.jar` file with the MapR version of that file:

   ```
   $ yum install mapr-asynchbase
   ```

4. Run the build script:

   ```
   ./build.sh
   ```

5. If you want to use OpenTSDB with MapR Database tables, open the create_table.sh file (<OPENTSDB_ROOT_INSTALL_DIR>/src/create_table.sh) and add "/" before the table names so that MapR recognizes them as MapR Database tables: See Example: create_table.sh on page 3179.

6. Create tables:

   ```
   env COMPRESSION=NONE;HBASE_HOME=/opt/mapr/hbase/hbase-<version>
   <OPENTSDB_ROOT_INSTALL_DIR>/src/create_table.sh
   ```

7. Run the following command to verify that the tables are created successfully:

   ```
   hadoop fs -ls /
   ```

8. Create a simple metric to store, such as “sys.cpu.user”:

```
./build/tsdb mkmetric sys.cpu.user --table=/tsdb --uidtable=/tsdb-uid
```

9. Run the OpenTSDB daemon (tsd).

```
./build/tsdb tsd --port=4242 --staticroot=build/staticroot
--cachedir=/tmp/opentsdb_tmp --zkquorum=10.10.101.50:5181 --table=/tsdb
--uidtable=/tsdb-uid
```

Note: Instead of providing these options on command line, you can configure the values in the opentsdb.conf file. This file must be in the root folder so the option settings are read when tsd is run. Also note that the staticroot argument points to the static UI files. You do not need to create cachedir because openstdb creates it automatically. Specifying the destination cachedir argument is enough. You do need to explicitly specify tsdb tables (tsdb, tsdb-uid) and Zookeeper quorum nodes.

10. Log into the web UI: `http://<TSD_Installed_Node_IP>:<Port>`
    For example: `http://10.10.10.230:4242/`

11. Run a simple test program that generates data and sends repeated puts for the metric over a socket connection: <UI-IP>:<UI-Port> . See Data Generator Program on page 3180.

12. Check the plot in the UI.
    a) Select **From date** and check **autoreload**.
    b) Fill in the metric (in this case, `sys.cpu.user`) and the Tag keys (`cpu, host`) values (`webserver 0, webserver 1`). You should see a graph with a random plot.

Example: create_table.sh
create_table.sh is used to set up MapR Database to accepts `puts` from OpenTSDB. This example create_table.sh script was updated to work with MapR Database tables.

Note the changed sections for the `*_TABLE` variables.

```
#!/bin/sh
# Small script to setup the HBase tables used by OpenTSDB.

test -n "$HBASE_HOME" || {
  echo >&2 'The environment variable HBASE_HOME must be set'
  exit 1
}
test -d "$HBASE_HOME" || {
  echo >&2 "No such directory: HBASE_HOME=$HBASE_HOME"
  exit 1
}

TSDB_TABLE=${TSDB_TABLE-'/tsdb'}
UID_TABLE=${UID_TABLE-'/tsdb-uid'}
TREE_TABLE=${TREE_TABLE-'/tsdb-tree'}
META_TABLE=${META_TABLE-'/tsdb-meta'}
BLOOMFILTER=${BLOOMFILTER-'ROW'}
# LZO requires lzo2 64bit to be installed + the hadoop-gpl-compression jar.
COMPRESSION=${COMPRESSION-'LZO'}
# All compression codec names are upper case (NONE, LZO, SNAPPY, etc).
COMPRESSION=`echo "$COMPRESSION" | tr a-z A-Z`

case $COMPRESSION in
```

```
  (NONE|LZO|GZIP|SNAPPY)  :;;  # Known good.
  (*)
    echo >&2 "warning: compression codec '$COMPRESSION' might not be
supported."
    ;;
esac

# HBase scripts also use a variable named `HBASE_HOME', and having this
# variable in the environment with a value somewhat different from what
# they expect can confuse them in some cases.  So rename the variable.
hbh=$HBASE_HOME
unset HBASE_HOME
exec "$hbh/bin/hbase" shell <<EOF
create '$UID_TABLE',
  {NAME => 'id', COMPRESSION => '$COMPRESSION', BLOOMFILTER =>
'$BLOOMFILTER'},
  {NAME => 'name', COMPRESSION => '$COMPRESSION', BLOOMFILTER =>
'$BLOOMFILTER'}

create '$TSDB_TABLE',
  {NAME => 't', VERSIONS => 1, COMPRESSION => '$COMPRESSION', BLOOMFILTER
=> '$BLOOMFILTER'}

create '$TREE_TABLE',
  {NAME => 't', VERSIONS => 1, COMPRESSION => '$COMPRESSION', BLOOMFILTER
=> '$BLOOMFILTER'}

create '$META_TABLE',
  {NAME => 'name', COMPRESSION => '$COMPRESSION', BLOOMFILTER =>
'$BLOOMFILTER'}
EOF
```

Data Generator Program

This simple test program generates data and sends repeated puts for the metric over a socket connection.

```
import java.io.PrintWriter;
import java.net.Socket;
import java.util.Date;
import java.util.Random;

public class TestOpenTsdbAPI {
   public static Random random = new Random();
   public static long timeStamp = new Date().getTime()/1000; //in secs
   public static void testTSDBConnection() throws Exception {
       Socket sock = null;
       PrintWriter pw = null;
       String hostname = "10.10.10.230";
       int port = 4242;
       int count=1;
       while(true) {
           if(null==sock) {
               sock = new Socket(hostname, port);
               pw = new PrintWriter(sock.getOutputStream(), true);
           }
           pw.println(dataGen(0, 0, count));
           pw.flush();
           pw.println(dataGen(0, 1, count));
           pw.flush();
           pw.println(dataGen(1, 0, count));
           pw.flush();
           pw.println(dataGen(1, 1, count));
           pw.flush();
```

```
            if(++count==Integer.MAX_VALUE) break;
            Thread.sleep(60000);
        }
    }
    public static void main(String [] args) {
        try {
            testTSDBConnection();
        } catch(Exception ex) {
            ex.printStackTrace();
        }
    }

    public static String dataGen(int web, int cpu, int count) {
        int Low = 1;
        int High = 99;
        int val = random.nextInt(High-Low) + Low;
        long timeStamp1 = new Date().getTime()/1000;
        String dat = "put sys.cpu.user "+(timeStamp1)+" "+val+"
host=webserver"+ web +" cpu="+cpu;//(timeStamp+count)
        System.out.println(dat);
        return dat;
    }
}
```

For example, this program tries to put metrics for 2 hosts (webserver 0 and webserver 1). Each host has 2 CPUs (cpu 0 and cpu 1). Sample puts look like this:

```
put sys.cpu.user 1415300810 87 host=webserver0 cpu=0
put sys.cpu.user 1415300810 66 host=webserver0 cpu=1
put sys.cpu.user 1415300810 18 host=webserver1 cpu=0
put sys.cpu.user 1415300810 26 host=webserver1 cpu=1

put <metric> <timestamp> <value> <tag1>=<> <tag2>=<>
```

When you run the program, you should see entries that indicate that the tags for the metric were created, and they should auto-complete on the UI.

```
UniqueId: Creating an ID for kind='tagv' name='webserver0'
```

You can also verify this from command line instead of the UI:

```
<OpenTSDB-Root>/build/tsdb query 1y-ago sum sys.cpu.user
```

Installing OpenTSDB with a Package

The following steps describe how to install OpenTSDB from a package.

Be sure to install the OpenTSDB version that is required for your AsyncHBase version. AsyncHBase 1.6 requires OpenTSDB 2.0. AsyncHBase 1.7 requires OpenTSDB 2.2.

1. Install the OpenTSDB RPM:
   a) `mkdir /root/opentsdbrpm`

   b) `cd /root/opentsdbrpm`

   c) Download the version of OpenTSDB that you required.

      For OpenTSDB 2.0: `wget` https://github.com/OpenTSDB/opentsdb/releases/download/v2.0.0/opentsdb-2.0.0.noarch.rpm `-O opentsdb-2.0.0.noarch.rpm`

      For OpenTSDB 2.2: `wget`https://github.com/OpenTSDB/opentsdb/releases/download/v2.2.0/opentsdb-2.2.0.noarch.rpm`-o opentsdb-2.2.0.noarch.rpm`

d) `rpm -ivh opentsdb-<version>.noarch.rpm`

2. Configure OpenTSDB to work with MapR:

a) Edit the following `tsdb` scripts to cover MapR-specific dependencies: `/usr/share/opentsdb/bin/tsdb` and `/usr/bin/tsdb`

```
# Base of MapR installation
BASEMAPR=${MAPR_HOME:-/opt/mapr}

# Add MapR hadoop jars to classpath
if test -d "$BASEMAPR/hadoop/hadoop-0.20.2/lib"; then
 # hadoop conf directory to beginning of classpath (for core-site.xml)
 CLASSPATH="$BASEMAPR/hadoop/hadoop-0.20.2/conf:$CLASSPATH"

 for jar in "$BASEMAPR"/hadoop/hadoop-0.20.2/lib/*.jar; do
   if [ "`echo $jar | grep slf4j`" != "" ]; then
     continue
   fi
   CLASSPATH="$CLASSPATH:$jar"
 done
fi
```

b) Replace the asynchbase jar file (provide the current jar file name in the `cp` command):

```
cp
/opt/mapr/asynchbase/asynchbase-<version>/
asynchbase-<version>-mapr-*.jar
/usr/share/opentsdb/lib/
rm -f /usr/share/opentsdb/lib/asynchbase-<previous_version>.jar
```

c) Configure the `opentsdb.conf` files: These files must have the following settings:

```
/usr/share/opentsdb/etc/opentsdb/opentsdb.conf
/etc/opentsdb/opentsdb.conf
```

```
tsd.network.port = 4242
tsd.http.staticroot = /usr/share/opentsdb/static/
tsd.core.auto_create_metrics = false (for testing purposes only)
tsd.storage.hbase.data_table = /tsdb
tsd.storage.hbase.uid_table = /tsdb-uid
tsd.storage.hbase.zk_quorum = <zookeeperNode>:<zookeeperP>
```

d) Edit the `<OPENTSDB_ROOT_INSTALL_DIR>/src/create_table.sh` file and add "/" before the table names so that MapR recognizes them as MapR Database tables. Then, create tables in MapR Database: .

```
export COMPRESSION=NONE; export HBASE_HOME=/opt/mapr/hbase/
hbase-<version>; /usr/share/opentsdb/tools/create_table.sh
```

See Example: create_table.sh on page 3179

e) Confirm that the tables are created:

```
hadoop fs -ls /
tr--------   3 root root            2 2014-12-12 01:47 /tsdb
tr--------   3 root root            2 2014-12-12 01:47 /tsdb-meta
tr--------   3 root root            2 2014-12-12 01:47 /tsdb-tree
tr--------   3 root root            2 2014-12-12 01:47 /tsdb-uid
```

3. Start the `tsd` daemon. You can give executable permissions to the tsdb script in `/usr/share/opentsdb/bin`, or you can directly use `tsdb` (because of the dependencies you added earlier).

```
chmod +x /usr/share/opentsdb/bin/tsdb
/usr/share/opentsdb/bin/tsdb tsd --port=4242
--staticroot="/usr/share/opentsdb/static/"
--cachedir="/tmp/opentsdb" --auto-metric
```

4. Create a metric: `/usr/share/opentsdb/bin/tsdb mkmetric mymetric.stock`

5. Test the metric:
   a) Run a Test Program for OpenTSDB on page 3184 that reads from the tmp_input on page 3183 file and sends put requests to opentsdb, which saves the data to a MapR Database table (tsdb/tsdb-uid).

   b) Run aggregation queries (such as SUM) from the command line: `/usr/share/opentsdb/bin/tsdb query 1y-ago sum mymetric.stock` or `tsdb query 1y-ago sum mymetric.stock`

   c) When you run the SUM command, the results should look like the following:

```
====
mymetric.stock 1407165399000 680.500015 {}
mymetric.stock 1407165401000 904.625000 {}
mymetric.stock 1407165402000 904.612495 {}
mymetric.stock 1407165403000 904.599991 {}
mymetric.stock 1407165404000 904.599991 {}
mymetric.stock 1407165405000 904.599991 {}
mymetric.stock 1407165406000 904.599991 {}
mymetric.stock 1407165407000 904.599991 {}
mymetric.stock 1407165408000 904.599991 {}
mymetric.stock 1407165409000 904.599991 {}
mymetric.stock 1407165410000 904.599991 {}
mymetric.stock 1407165411000 904.599991 {}
mymetric.stock 1407165412000 904.599991 {}
mymetric.stock 1407165413000 904.599991 {}
mymetric.stock 1407165414000 904.599991 {}
mymetric.stock 1407165415000 904.599991 {}
mymetric.stock 1407165416000 904.599991 {}
mymetric.stock 1407165417000 904.599991 {}
mymetric.stock 1407165418000 904.599991 {}
mymetric.stock 1407165419000 904.599991 {}
mymetric.stock 1407165422000 904.678749 {}
mymetric.stock 1407165423000 484.255005 {}
====
```

tmp_input

```
=====
put mymetric.stock 1407165399 196.30 symbol=VOD.L
```

```
put mymetric.stock 1407165399 484.20 symbol=BP.L
put mymetric.stock 1407165401 224.15 symbol=BARC.L
put mymetric.stock 1407165402 196.30 symbol=VOD.L
put mymetric.stock 1407165403 484.15 symbol=BP.L
put mymetric.stock 1407165404 224.15 symbol=BARC.L
put mymetric.stock 1407165405 196.30 symbol=VOD.L
put mymetric.stock 1407165405 484.15 symbol=BP.L
put mymetric.stock 1407165406 224.15 symbol=BARC.L
put mymetric.stock 1407165407 196.30 symbol=VOD.L
put mymetric.stock 1407165408 484.15 symbol=BP.L
put mymetric.stock 1407165409 224.15 symbol=BARC.L
put mymetric.stock 1407165410 196.30 symbol=VOD.L
put mymetric.stock 1407165411 484.15 symbol=BP.L
put mymetric.stock 1407165412 224.15 symbol=BARC.L
put mymetric.stock 1407165413 196.30 symbol=VOD.L
put mymetric.stock 1407165414 484.15 symbol=BP.L
put mymetric.stock 1407165415 224.15 symbol=BARC.L
put mymetric.stock 1407165416 196.30 symbol=VOD.L
put mymetric.stock 1407165417 484.15 symbol=BP.L
put mymetric.stock 1407165417 224.15 symbol=BARC.L
put mymetric.stock 1407165418 196.30 symbol=VOD.L
put mymetric.stock 1407165419 484.15 symbol=BP.L
put mymetric.stock 1407165422 224.15 symbol=BARC.L
put mymetric.stock 1407165422 196.30 symbol=VOD.L
put mymetric.stock 1407165423 484.255 symbol=BP.L
====
```

Test Program for OpenTSDB

```
public static void testTSDBConnection() throws Exception {
       Socket sock = null;
       PrintWriter pw = null;
       String hostname = "10.10.10.220"; //replace with the node where tsd
runs
       int port = 4242; //replace with your port
       sock = new Socket(hostname, port);
       pw = new PrintWriter(sock.getOutputStream(), true);
       File dir = new File(".");
       File fin = new File(dir.getCanonicalPath() + File.separator +
"tmp_input");
       BufferedReader br = new BufferedReader(new FileReader(fin));
String line = null;
while ((line = br.readLine()) != null) {
           System.out.println(line);
           pw.println(line);
           pw.flush();
}
br.close();
}
```

GetRequest API

MapR includes an additional constructor for the GetRequest class which takes an extra qualifier.

This additional constructor allows you to use one GetRequest to retrieve the key and value for tables that consist of multiple column families with different qualifiers:

```
public GetRequest(final byte[] table,
                  final byte[] key,
                  final byte[][] families,
                  final byte[][][] qualifiers) {
    super(table, key);
```

```
    this.families(families);
    this.qualifiers(qualifiers);
  }
```

Cascading

Cascading™ is a Java application framework produced by Concurrent, Inc. that enables developers to quickly and easily build rich enterprise-grade Data Processing and Machine Learning applications that can be deployed and managed across private or cloud-based Hadoop clusters.

Note: Cascading is ***not*** part of the MapR Data Platform distribution and not supported by HPE. However, like many other open source technologies, it can be used with MapR Data Platform. The following information provides relevant details about using Cascading with MapR Data Platform.

The `mt` command is the wrapper around Cascading. Multitool, a command line tool for processing large text files and datasets (like sed and grep on unix). The `mt` command is located in the `/opt/mapr/contrib/multitool/bin` directory.

Related Links

For information about working with Cascading, see:

- Cascading project at Concurrent, Inc.
- Forum posts related to Cascading
- Search HPE Blog for Cascading topics

Apache Drill

Drill is a low-latency distributed query engine for large-scale datasets, including structured and semi-structured/nested data. Inspired by Google's Dremel, Drill is designed to scale to several thousands of nodes and query petabytes of data at interactive speeds that BI/Analytics environments require.

Drill includes a distributed environment, purpose built for large-scale data processing. At the core of Drill is the "Drillbit" service which is responsible for accepting requests from the client, processing the queries, and returning results to the client.

Installing Drill

You can install Drill on one node or multiple nodes in a cluster. When Drill runs on each data node in a cluster, Drill can maximize data locality without moving data over the network or between nodes. Drill uses ZooKeeper to maintain cluster membership and health check information.

last-updated: Apr 11, 2022

See Installing Drill on page 177 for instructions and additional information.

Configuring Data Source Connections

Drill connects to data sources through storage plugins. Drill can connect to several types of data sources including databases, local or distributed filesystems, and Hive metastores.

See Connecting Drill to Data Sources on page 3251 and Connect a Data Source for instructions and additional information.

Accessing Drill

After you install Drill and configure connections to your data sources, you can access Drill from any of the following user interfaces:

- Drill shell (SQLLine)
- Drill Web Console
- ODBC
- JDBC
- C++ API
- REST API

Additional Resources

Drill documentation is accessible from following the locations:

- Drill Release Notes on page 5747
- Apache Drill

Drill Tutorial

Drill is included as part of the Hadoop distribution. The sandbox with Drill is a fully functional single-node cluster that simulates Drill in a Hadoop environment. Business and technical analysts, product managers, and developers can use the sandbox environment to get a feel for the power and capabilities of Drill by performing various types of queries. Refer to the Drill web site and Drill documentation for more details.

Hadoop is not a prerequisite for Drill and users can start learning Drill by running SQL queries directly on the local filesystem.

To complete the tutorial on the sandbox with Drill, work through lessons in following order:

Installing the Drill Sandbox

Prerequisites

The sandbox with Drill runs on VMware Player and VirtualBox (**recommended**), free desktop applications that you can use to run a virtual machine on a Windows, Mac, or Linux PC.

Before you install the sandbox with Drill, verify that the host system meets the following prerequisites:

- VMware Player or VirtualBox is installed.
- At least 20 GB free hard disk space, at least 4 physical cores, and 8 GB of RAM is available. Performance increases with more RAM and free hard disk space.
- Uses one of the following 64-bit x86 architectures:

last-updated: Apr 11, 2022

- • A 1.3 GHz or faster AMD CPU with segment-limit support in long mode
  - A 1.3 GHz or faster Intel CPU with VT-x support
- If you have an Intel CPU with VT-x support, verify that VT-x support is enabled in the host system BIOS. The BIOS settings that must be enabled for VT-x support vary depending on the system vendor. See the VMware knowledge base article at http://kb.vmware.com/kb/1003944 for information about how to determine if VT-x support is enabled.

VM Player Downloads

For Linux, Mac, or Windows, download the free VMware Player or VirtualBox. Optionally, you can purchase VMware Fusion for Mac.

VM Player Installation

The following list provides links to the virtual machine installation instructions:

- To install the VMware Player, see the VMware documentation. Use of VMware Player is subject to the VMware Player end user license terms. VMware does not provide support for VMware Player. For self-help resources, see the VMware Player FAQ.
- To install VirtualBox, see the Oracle VM VirtualBox User Manual. By downloading VirtualBox, you agree to the terms and conditions of the respective license.

Warning:

The sandbox for Drill on VirtualBox comes with NAT port forwarding enabled. The sandbox for Drill on VMware Player supports NAT, but does not support port forwards.

Click the link appropriate for the virtual machine player running on your machine:

Installing the MapR Sandbox with Drill on VMware Player/VMware Fusion
Download the MapR sandbox for Drill and import the virtual machine into VirtualBox. Configure the network setting, and start the MapR sandbox for Drill.

Complete the following steps to install the MapR sandbox with Drill on VMware Player or VMware Fusion:

1. Download the MapR sandbox for Drill file to a directory on your machine. To access the file, go to https://package.mapr.hpe.com/releases/, and select the directory for the latest release. The file is located in the `sandbox/` directory for the release, as shown:
   ```
   https://package.mapr.com/releases/v6.1.0/sandbox/
   MapR-Sandbox-For-Apache-Drill-1.14.0-6.1.0-vmware.ova
   ```
2. Open the virtual machine player, and select the **Open a Virtual Machine** option. If you are running VMware Fusion, select **Import**.
3. Navigate to the directory where you downloaded the MapR sandbox with Drill file, and select `MapR-Sandbox-For-Apache-Drill-<version>-vmware.ova`. The *Import Virtual Machine* dialog appears.
4. Click **Import**. The virtual machine player imports the sandbox.
5. Select **Edit virtual machine settings** and then select the **Network Adapter** option.

   Note: The correct setting depends on your network connectivity when you run the sandbox. In general, if you are going to use a wired Ethernet connection, select **NAT**. If you use ODBC or JDBC on a remote host, select **Bridged Adapter**. If you are going to use a wireless network, select **Host-only**.

last-updated: Apr 11, 2022

6. Click **OK** to save the settings.
7. Select `MapR-Sandbox-For-Apache-Drill-<version>_VM`, and click **Play virtual** machine.
8. When you see the `maprdemo login:_` prompt, enter `mapr` and then enter `mapr` as the password.
   The message, *Welcome to your Demo virtual machine,* appears.
9. The client must be able to resolve the actual hostname of the Drill node(s) with the IP(s). Verify that a DNS entry was created on the client machine for the Drill node(s). If a DNS entry does not exist, create the entry for the Drill node(s).
   - For Windows, create the entry in %WINDIR%\system32\drivers\etc\hosts.
   - For Linux and Mac, create the entry in /etc/hosts.

   Example: `127.0.1.1 maprdemo`
10. Go to `http://localhost:8047` or `http://127.0.0.1:8047` to experience the Drill Web Console.

    Note: If you used a bridged adapter, the network generates and assigns an IP to the sandbox that you use in place of 127.0.0.1 or localhost. To get the IP address, enter `root` as the username and `mapr` as the password from the sandbox command line. Run the `ifconfig` command to get the sandbox IP address. The IP address is located in the inet address field. You can access the Drill Web Console using the IP address and port 8047, as shown in the following example:

    ```
    http://10.250.56.216:8047
    ```

    You can access the command line on the virtual machine by pressing Alt+F2 on Windows or Option+F5 on Mac. You can access the sandbox via SSH, as described in Getting to Know the Drill Sandbox

After downloading and installing the sandbox, continue with the tutorial by Getting to Know the Drill Setup.
Installing the MapR Sandbox with Drill on VirtualBox
Download the MapR sandbox for Drill and import the virtual machine into VirtualBox. Configure the network setting, and start the MapR Sandbox for Drill.

Note: The MapR sandbox for Drill on VirtualBox comes with NAT port forwarding enabled, which allows you to access the sandbox using localhost as the hostname.

Complete the following steps to install the MapR sandbox with Drill:

1. Download the MapR sandbox for Drill file to a directory on your machine. To access the file, go to https://package.mapr.hpe.com/releases/, and select the directory for the latest MapR release. The file is located in the `sandbox/` directory for the release, as shown:
   `https://package.mapr.com/releases/v6.1.0/sandbox/MapR-Sandbox-For-Apache-Drill-1.14.0-6.1.0.ova`
2. Open VirtualBox, and select **File** > **Import Appliance**. The *Import Appliance* window appears.
3. Go to the directory where you downloaded the MapR sandbox for Drill file, select the `MapR-Sandbox-For-Apache-Drill-<version>.ova` file, and click **Next**. The *Import Virtual Appliance* window appears.
4. In the *Import Virtual Appliance* window, select the option to **Reinitialize the MAC address of all network cards** and then click **Import**. The Import Appliance imports the sandbox.

5. When the import completes, select the MapR sandbox for Drill and then click **Settings**. The *Settings* window appears.

6. In the *Settings* window, select **Network**. The correct setting depends on your network connectivity when you run the sandbox. In general, if you are going to use a wired Ethernet connection, use the default, **NAT**. If you use ODBC or JDBC on a remote host, select **Bridged Adapter**. If you are going to use a wireless network, select **Host-only Adapter** or **Bridged Adapter** with the **Intel(R) Dual Band Wireless-AC 8260** option.

7. Click **OK** to continue.

8. Select the `MapR-Sandbox-For-Apache-Drill-<version>` in VirtualBox, and click **Start**.

9. When you see `maprdemo login:_` prompt, enter `mapr` and then enter `mapr` as the password.

   The message, *Welcome to your Demo virtual machine,* appears.

10. The client must be able to resolve the actual hostname of the Drill node(s) with the IP(s). Verify that a DNS entry was created on the client machine for the Drill node(s). If a DNS entry does not exist, create the entry for the Drill node(s).

    - For Windows, create the entry in %WINDIR%\system32\drivers\etc\hosts.
    - For Linux and Mac, create the entry in /etc/hosts.

    Example: `127.0.0.1 maprdemo`

11. Go to http://localhost:8047 or http://127.0.0.1:8047 to experience the Drill Web Console, and enter `mapr` as the username and password.

    Note: If you used a bridged adapter, the network generates and assigns an IP to the sandbox that you use in place of localhost or 127.0.0.1. To get the IP address, run `ifconfig` from the sandbox command line to get the sandbox IP address. The IP address is located in the inet address field. You can access the Drill Web Console using the IP address and port 8047, as shown in the following example:

    ```
    http://10.250.56.216:8047
    ```

    You can access the command line on the virtual machine by pressing Alt+F2 on Windows or Option+F5 on Mac. You can access the sandbox via SSH, as described in Getting to Know the Drill Sandbox.

After downloading and installing the sandbox, continue with the tutorial by Getting to Know the Drill Setup.

Getting to Know the Drill Setup

This section describes the configuration of the Drill system that you have installed and introduces the overall use case for the tutorial.

Storage Plugins Overview

The Hadoop cluster within the sandbox is set up with MapR File System, MapR Database, and Hive, which all serve as data sources for Drill in this tutorial. Before you can run queries against these data sources, you need to connect to the data source through an interface called a storage plugin. A storage plugin defines interfaces to read/write and get metadata from the data source. Each storage plugin also exposes optimization rules for Drill to leverage for efficient query execution.

Jump directly to the queries in Lesson 1: Learn About the Data Set, or first, get some important background information about pre-configured storage plugins by following these steps:

1. Start the Drill Web Console.

last-updated: Apr 11, 2022

2. Go to the Storage tab.
3. Open the configured storage plugins one at a time by clicking Update. You will see the following plugins configured.

dfs

This is a storage plugin configuration for the MapR File System in the sandbox. The connection attribute indicates the type of distributed filesystem: in this case, MapR File System. Drill can work with any distributed system, including HDFS, S3, and so on.

The configuration also includes a set of workspaces; each one represents a location in MapR File System:

- root: access to the root filesystem location
- clicks: access to nested JSON log data
- logs: access to flat (non-nested) JSON log data in the logs directory and its subdirectories
- views: a workspace for creating views

A workspace in Drill is a location where users can easily access a specific set of data and collaborate with each other by sharing artifacts. Users can create as many workspaces as they need within Drill.

Each workspace can also be configured as “writable” or not, which indicates whether users can write data to this location and defines the storage format in which the data will be written (parquet, csv, json). These attributes become relevant when you explore Drill SQL commands, especially CREATE TABLE AS (CTAS) and CREATE VIEW.

Drill can query files and directories directly and can detect the file formats based on the file extension or the first few bits of data within the file. However, additional information around formats is required for Drill, such as delimiters for text files, which are specified in the “formats” section as follows.

```
{
 "type": "file",
 "enabled": true,
 "connection": "maprfs:///",
 "workspaces": {
   "root": {
     "location": "/mapr/demo.mapr.com/data",
     "writable": false,
     "storageformat": null
   },
   "clicks": {
     "location": "/mapr/demo.mapr.com/data/nested",
     "writable": true,
     "storageformat": "parquet"
   },
   "logs": {
     "location": "/mapr/demo.mapr.com/data/flat",
     "writable": true,
     "storageformat": "parquet"
   },
   "views": {
     "location": "/mapr/demo.mapr.com/data/views",
     "writable": true,
     "storageformat": "parquet"
 },
 "formats": {
   "psv": {
     "type": "text",
     "extensions": [
       "tbl"
```


last-updated: Apr 11, 2022

```
      ],
      "delimiter": "|"
    },
    "csv": {
      "type": "text",
      "extensions": [
        "csv"
      ],
      "delimiter": ","
    },
    "tsv": {
      "type": "text",
      "extensions": [
        "tsv"
      ],
      "delimiter": "\t"
    },
    "parquet": {
      "type": "parquet"
    },
    "json": {
      "type": "json"
    }
  }
}
```

hive

A storage plugin configuration for a Hive data warehouse within the sandbox. Drill connects to the Hive metastore by using the configured metastore thrift URI. Metadata for Hive tables is automatically available for users to query.

```
{
 "type": "hive",
 "enabled": true,
 "configProps": {
   "hive.metastore.uris": "thrift://localhost:9083",
   "hive.metastore.sasl.enabled": "false"
 }
}
```

Client Application Interfaces

Drill also provides additional application interfaces for the client tools to connect and access from Drill. The interfaces include the following.

ODBC/JDBC drivers

Drill provides ODBC/JDBC drivers to connect from BI tools such as Tableau, MicroStrategy, SQUirrel, and Jaspersoft; refer to Drill Interfaces Introduction to learn more.

SQLLine

SQLLine is a JDBC application that comes packaged with Drill. In order to start working with it, you can use the command line on the demo cluster to log in as root, then enter `sqlline`. Use `mapr` as the login password. For example:

```
$ ssh root@localhost -p 2222
Password:
Last login: Mon Sep 15 13:46:08 2014 from 10.250.0.28
Welcome to your Mapr Demo virtual machine.
```


last-updated: Apr 11, 2022

```
[root@maprdemo ~]# sqlline
sqlline version 1.1.6
0: jdbc:drill:>
```

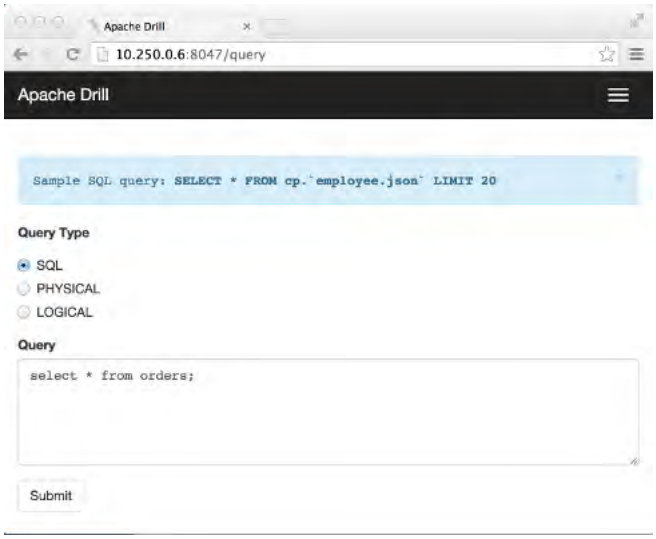
Drill Web UI

The Drill Web UI is a simple user interface for configuring and manage Drill. This UI can be launched from any of the nodes in the Drill cluster. The configuration for Drill includes setting up storage plugins that represent the data sources on which Drill performs queries. The sandbox comes with storage plugins configured for Hive, MapR Database binary tables, MapR File System, and local filesystem.

Users and developers can get the necessary information for tuning and performing diagnostics on queries, such as the list of queries executed in a session and detailed query plan profiles for each.

Detailed configuration and management of Drill is out of scope for this tutorial.

The following Web Console for Drill also provides a query UI where users can submit queries to Drill and observe results.



What's Next

Start running queries by going to Lesson 1: Learn About the Data Set.

Lesson 1: Learn About the Data Set

Goal

This lesson is simply about discovering what data is available, in what format, using simple SQL SELECT statements. Drill is capable of analyzing data without prior knowledge or definition of its schema. This means that you can start querying data immediately (and even as it changes), regardless of its format.

The data set for the tutorial consists of:

- Transactional data: stored as a Hive table
- Product catalog and primary customer data: stored as MapR Database binary tables
- Clickstream and logs data: stored in the MapR File System as JSON files

Queries in This Lesson

This lesson consists of select * queries on each data source.

Before You Begin

Start sqlline

If sqlline is not already started, use a Terminal or Command window to log into the demo VM as root, then enter `sqlline`:

```
$ ssh root@10.250.0.6
Password:
Last login: Mon Sep 15 13:46:08 2014 from 10.250.0.28
Welcome to your Mapr Demo virtual machine.
[root@maprdemo ~]# sqlline
sqlline version 1.1.6
0: jdbc:drill:>
```

You can run queries from this prompt to complete the tutorial. To exit from `sqlline`, type:

```
0: jdbc:drill:> !quit
```

Note that though this tutorial demonstrates the queries using SQLLine, you can also execute queries using the Drill Web UI.

Enable the DECIMAL Data Type

This tutorial uses the DECIMAL data type in some examples. The DECIMAL data type is disabled by default in this release, so enable the DECIMAL data type before proceeding:

```
alter session set `planner.enable_decimal_data_type`=true;

+-------+--------------------------------------------+
|  ok   |                  summary                   |
+-------+--------------------------------------------+
| true  | planner.enable_decimal_data_type updated.  |
+-------+--------------------------------------------+
1 row selected
```

List the available workspaces and databases:

```
0: jdbc:drill:> show databases;
+-------------+
| SCHEMA_NAME |
+-------------+
| hive.default |
| dfs.default |
| dfs.logs    |
| dfs.root    |
| dfs.views   |
| dfs.clicks  |
| dfs.data    |
| dfs.tmp     |
| sys         |
| maprdb      |
| cp.default  |
| INFORMATION_SCHEMA |
+-------------+
12 rows selected
```

Note that this command exposes all the metadata available from the storage plugins configured with Drill as a set of schemas. This includes the Hive and MapR Database databases as well as the workspaces configured in the filesystem. As you run queries in the tutorial, you will switch among these schemas

last-updated: Apr 11, 2022

by submitting the USE command. This behavior resembles the ability to use different database schemas (namespaces) in a relational database system.

Query Hive Tables

The orders table is a six-column Hive table defined in the Hive metastore. This is a Hive external table pointing to the data stored in flat files on the MapR File System. The orders table contains 122,000 rows.

Set the schema to hive:

```
0: jdbc:drill:> use hive;
+------------+------------+
| ok | summary |
+------------+------------+
| true | Default schema changed to 'hive' |
+------------+------------+
```

You will run the USE command throughout this tutorial. The USE command sets the schema for the current session.

Describe the table:

You can use the DESCRIBE command to show the columns and data types for a Hive table:

```
0: jdbc:drill:> describe orders;
+-------------+------------+-------------+
| COLUMN_NAME | DATA_TYPE  | IS_NULLABLE |
+-------------+------------+-------------+
| order_id    | BIGINT     | YES         |
| month       | VARCHAR    | YES         |
| cust_id     | BIGINT     | YES         |
| state       | VARCHAR    | YES         |
| prod_id     | BIGINT     | YES         |
| order_total | INTEGER    | YES         |
+-------------+------------+-------------+
```

The DESCRIBE command returns complete schema information for Hive tables based on the metadata available in the Hive metastore.

Select 5 rows from the orders table:

```
0: jdbc:drill:> select * from orders limit 5;
+------------+------------+------------+------------+------------+-------------+

| order_id | month | cust_id | state | prod_id | order_total |
+------------+------------+------------+------------+------------+-------------+

| 67212 | June | 10001 | ca | 909 | 13 |
| 70302 | June | 10004 | ga | 420 | 11 |
| 69090 | June | 10011 | fl | 44 | 76 |
| 68834 | June | 10012 | ar | 0 | 81 |
| 71220 | June | 10018 | az | 411 | 24 |
+------------+------------+------------+------------+------------+-------------+
```

Because orders is a Hive table, you can query the data in the same way that you would query the columns in a relational database table. Note the use of the standard LIMIT clause, which limits the result set to the specified number of rows. You can use LIMIT with or without an ORDER BY clause.

Drill provides seamless integration with Hive by allowing queries on Hive tables defined in the metastore with no extra configuration. Note that Hive is not a prerequisite for Drill, but simply serves as a storage

plugin or data source for Drill. Drill also lets users query all Hive file formats (including custom serdes). Additionally, any UDFs defined in Hive can be leveraged as part of Drill queries.

Because Drill has its own low-latency SQL query execution engine, you can query Hive tables with high performance and support for interactive and ad-hoc data exploration.

Query MapR Database Binary Tables

The customers and products tables are MapR Database binary tables. MapR Database is an enterprise in-Hadoop NoSQL database. It exposes the HBase API to support application development. Every MapR Database binary table has a row_key, in addition to one or more column families. Each column family contains one or more specific columns. The row_key value is a primary key that uniquely identifies each row.

Drill allows direct queries on MapR Database binary tables. Unlike other SQL on Hadoop options, Drill requires no overlay schema definitions in Hive to work with this data. Think about a MapR Database table with thousands of columns, such as a time-series database, and the pain of having to management duplicate schemas for it in Hive!

Products Table

The products table has two column families.

Column Family	Columns
details	name category
pricing	price

The products table contains 965 rows.

Customers Table

The Customers table has three column families.

Column Family	Columns
address	state
loyalty	agg_rev membership
personal	age gender

The customers table contains 993 rows.

Set the workspace to maprdb:

```
0: jdbc:drill:> use maprdb;
+------------+------------+
| ok | summary |
+------------+------------+
| true | Default schema changed to 'maprdb' |
+------------+------------+
```


last-updated: Apr 11, 2022

Describe the tables:

```
0: jdbc:drill:> describe customers;
+-------------+------------+-------------+
| COLUMN_NAME | DATA_TYPE  | IS_NULLABLE |
+-------------+------------+-------------+
| row_key     | ANY        | NO          |
| address     | (VARCHAR(1), ANY) MAP | NO          |
| loyalty     | (VARCHAR(1), ANY) MAP | NO          |
| personal    | (VARCHAR(1), ANY) MAP | NO          |
+-------------+------------+-------------+

0: jdbc:drill:> describe products;
+-------------+------------+-------------+
| COLUMN_NAME | DATA_TYPE  | IS_NULLABLE |
+-------------+------------+-------------+
| row_key     | ANY        | NO          |
| details     | (VARCHAR(1), ANY) MAP | NO          |
| pricing     | (VARCHAR(1), ANY) MAP | NO          |
+-------------+------------+-------------+
```

Unlike the Hive example, the DESCRIBE command does not return the full schema up to the column level. Column-oriented NoSQL databases such as MapR Database can be schema-less by design; every row has its own set of column name-value pairs in a given column family, and the column value can be of any data type, as determined by the application inserting the data.

A “MAP” complex type in Drill represents this variable column name-value structure, and “ANY” represents the fact that the column value can be of any data type. Observe the row_key, which is also simply bytes and has the type ANY.

Select 5 rows from the products table:

```
0: jdbc:drill:> select * from products limit 5;
+------------+------------+------------+
| row_key | details | pricing |
+------------+------------+------------+
| [B@a1a3e25 | {"category":"bGFwdG9w","name":"IlNvbnkgbm90ZWJvb2si"} |
{"price":"OTU5"} |
| [B@103a43af |
{"category":"RW52ZWxvcGVz","name":"IzEwLTQgMS84IHggOSAxLzIgUHJlbWl1bSBEaWFnb
25hbCBTZWFtIEVudmVsb3Blcw=="} | {"price":"MT |
| [B@61319e7b |
{"category":"U3RvcmFnZSAmIE9yZ2FuaXphdGlvbg==","name":"MjQgQ2FwYWNpdHkgTWF4a
SBEYXRhIEJpbmRlciBSYWNrc1BlYXJs"} | {"price" |
| [B@9bcf17 | {"category":"TGFiZWxz","name":"QXZlcnkgNDk4"} |
{"price":"Mw=="} |
| [B@7538ef50 | {"category":"TGFiZWxz","name":"QXZlcnkgNDk="} |
{"price":"Mw=="} |
```

Given that Drill requires no up front schema definitions indicating data types, the query returns the raw byte arrays for column values, just as they are stored in MapR Database. Observe that the column families (details and pricing) have the map data type and appear as JSON strings.

In Lesson 2, you will use CAST functions to return typed data for each column.

Select 5 rows from the customers table:

```
0: jdbc:drill:> select * from customers limit 5;
+------------+------------+------------+------------+
| row_key | address | loyalty | personal |
+------------+------------+------------+------------+
```

```
| [B@284bae62 | {"state":"Imt5Ig=="} |
{"agg_rev":"IjEwMDEtMzAwMCI=","membership":"ImJhc2ljIg=="} |
{"age":"IjI2LTM1Ig==","gender":"Ik1B |
| [B@7ffa4523 | {"state":"ImNhIg=="}
| {"agg_rev":"IjAtMTAwIg==","membership":"ImdvbGQi"} |
{"age":"IjI2LTM1Ig==","gender":"IkZFTUFMRSI= |
| [B@7d13e79 | {"state":"Im9rIg=="} |
{"agg_rev":"IjUwMS0xMDAwIg==","membership":"InNpbHZlciI="} |
{"age":"IjI2LTM1Ig==","gender":"IkZFT |
| [B@3a5c7df1 | {"state":"ImtzIg=="} |
{"agg_rev":"IjMwMDEtMTAwMDAwIg==","membership":"ImdvbGQi"} |
{"age":"IjUxLTEwMCI=","gender":"IkZF |
| [B@e507726 | {"state":"Im5qIg=="}
| {"agg_rev":"IjAtMTAwIg==","membership":"ImJhc2ljIg=="} |
{"age":"IjIxLTI1Ig==","gender":"Ik1BTEUi" |
+------------+------------+------------+------------+
```

Again the table returns byte data that needs to be cast to readable data types.

Query the File System

Along with querying a data source with full schemas (such as Hive) and partial schemas (such as MapR Database), Drill offers the unique capability to perform SQL queries directly on filesystem. The filesystem could be a local filesystem, or a distributed filesystem such as MapR File System, HDFS, or S3.

In the context of Drill, a file or a directory is considered as synonymous to a relational database “table.” Therefore, you can perform SQL operations directly on files and directories without the need for up-front schema definitions or schema management for any model changes. The schema is discovered on the fly based on the query. Drill supports queries on a variety of file formats including text, CSV, Parquet, and JSON in the 0.5 release.

In this example, the clickstream data coming from the mobile/web applications is in JSON format. The JSON files have the following structure:

```
{"trans_id":31920,"date":"2014-04-26","time":"12:17:12","user_info":
{"cust_id":22526,"device":"IOS5","state":"il"},"trans_info":{"prod_id":
[174,2],"purch_flag":"false"}}
{"trans_id":31026,"date":"2014-04-20","time":"13:50:29","user_info":
{"cust_id":16368,"device":"AOS4.2","state":"nc"},"trans_info":{"prod_id":
[],"purch_flag":"false"}}
{"trans_id":33848,"date":"2014-04-10","time":"04:44:42","user_info":
{"cust_id":21449,"device":"IOS6","state":"oh"},"trans_info":{"prod_id":
[582],"purch_flag":"false"}}
```

The clicks.json and clicks.campaign.json files contain metadata as part of the data itself (referred to as “self-describing” data). Also note that the data elements are complex, or nested. The initial queries below do not show how to unpack the nested data, but they show that easy access to the data requires no setup beyond the definition of a workspace.

Query nested clickstream data

Set the workspace to dfs.clicks:

```
 0: jdbc:drill:> use dfs.clicks;
+------------+------------+
| ok | summary |
+------------+------------+
| true | Default schema changed to 'dfs.clicks' |
+------------+------------+
```


last-updated: Apr 11, 2022

In this case, setting the workspace is a mechanism for making queries easier to write. When you specify a filesystem workspace, you can shorten references to files in the FROM clause of your queries. Instead of having to provide the complete path to a file, you can provide the path relative to a directory location specified in the workspace. For example:

```
"location": "/mapr/demo.mapr.com/data/nested"
```

Any file or directory that you want to query in this path can be referenced relative to this path. The clicks directory referred to in the following query is directly below the nested directory.

Select 2 rows from the clicks.json file:

```
0: jdbc:drill:> select * from `clicks/clicks.json` limit 2;
+------------+------------+------------+------------+------------+
|  trans_id  |    date    |    time    | user_info  | trans_info |
+------------+------------+------------+------------+------------+
| 31920      | 2014-04-26 | 12:17:12
| {"cust_id":22526,"device":"IOS5","state":"il"} | {"prod_id":
[174,2],"purch_flag":"false"} |
| 31026      | 2014-04-20 | 13:50:29
| {"cust_id":16368,"device":"AOS4.2","state":"nc"} | {"prod_id":
[],"purch_flag":"false"} |
+------------+------------+------------+------------+------------+
2 rows selected
```

Note that the FROM clause reference points to a specific file. Drill expands the traditional concept of a “table reference” in a standard SQL FROM clause to refer to a file in a local or distributed filesystem.

The only special requirement is the use of back ticks to enclose the file path. This is necessary whenever the file path contains Drill reserved words or characters.

Select 2 rows from the campaign.json file:

```
0: jdbc:drill:> select * from `clicks/clicks.campaign.json` limit 2;
+------------+------------+------------+------------+------------+------------+

|  trans_id  |    date    |    time    | user_info  |  ad_info   |
trans_info |
+------------+------------+------------+------------+------------+------------+

| 35232      | 2014-05-10 | 00:13:03   |
{"cust_id":18520,"device":"AOS4.3","state":"tx"} | {"camp_id":"null"} |
{"prod_id":[7,7],"purch_flag":"true"} |
| 31995      | 2014-05-22 | 16:06:38   |
{"cust_id":17182,"device":"IOS6","state":"fl"} | {"camp_id":"null"} |
{"prod_id":[],"purch_flag":"false"} |
+------------+------------+------------+------------+------------+------------+

2 rows selected
```

Notice that with a select * query, any complex data types such as maps and arrays return as JSON strings. You will see how to unpack this data using various SQL functions and operators in the next lesson.

Query Logs Data

Unlike the previous example where we performed queries against clicks data in one file, logs data is stored as partitioned directories on the filesystem. The logs directory has three subdirectories:

- 2012

last-updated: Apr 11, 2022

- 2013
- 2014

Each of these year directories fans out to a set of numbered month directories, and each month directory contains a JSON file with log records for that month. The total number of records in all log files is 48000.

The files in the logs directory and its subdirectories are JSON files. There are many of these files, but you can use Drill to query them all as a single data source, or to query a subset of the files.

Set the workspace to dfs.logs:

```
 0: jdbc:drill:> use dfs.logs;
+------------+------------+
| ok | summary |
+------------+------------+
| true | Default schema changed to 'dfs.logs' |
+------------+------------+
```

Select 2 rows from the logs directory:

```
0: jdbc:drill:> select * from logs limit 2;
+------------+------------+------------+------------+------------+------------+------------+---------
| dir0 | dir1 | trans_id | date | time | cust_id | device | state | camp_id
| keywords | prod_id | purch_fl |
+------------+------------+------------+------------+------------+------------+------------+---------
| 2014 | 8 | 24181 | 08/02/2014 | 09:23:52 | 0 | IOS5 | il | 2 | wait | 128
| false |
| 2014 | 8 | 24195 | 08/02/2014 | 07:58:19 | 243 | IOS5 | mo | 6 | hmm |
107 | false |
+------------+------------+------------+------------+------------+------------+------------+---------
```

Note that this is flat JSON data. The dfs.clicks workspace location property points to a directory that contains the logs directory, making the FROM clause reference for this query very simple. You do not have to refer to the complete directory path on the file system.

The column names dir0 and dir1 are special Drill variables that identify subdirectories below the logs directory. In Lesson 3, you will do more complex queries that leverage these dynamic variables.

Find the total number of rows in the logs directory (all files):

```
0: jdbc:drill:> select count(*) from logs;
+------------+
| EXPR$0 |
+------------+
| 48000 |
+------------+
```

This query traverses all of the files in the logs directory and its subdirectories to return the total number of rows in those files.

What's Next

Go to Lesson 2: Run Queries with ANSI SQL.

last-updated: Apr 11, 2022

Lesson 2: Run Queries with ANSI SQL

Goal

This lesson shows how to do some standard SQL analysis in Drill: for example, summarizing data by using simple aggregate functions and connecting data sources by using joins. Note that Drill provides ANSI SQL support, not a “SQL-like” interface.

Queries in This Lesson

Now that you know what the data sources look like in their raw form, using select * queries, try running some simple but more useful queries on each data source. These queries demonstrate how Drill supports ANSI SQL constructs and also how you can combine data from different data sources in a single SELECT statement.

- Show an aggregate query on a single file or table. Use GROUP BY, WHERE, HAVING, and ORDER BY clauses.
- Perform joins between Hive, MapR Database, and filesystem data sources.
- Use table and column aliases.
- Create a Drill view.

Aggregation

Set the schema to hive:

```
0: jdbc:drill:> use hive;
+------------+------------+
|     ok     |  summary   |
+------------+------------+
| true       | Default schema changed to 'hive' |
+------------+------------+
1 row selected
```

Return sales totals by month:

```
0: jdbc:drill:> select `month`, sum(order_total)
from orders group by `month` order by 2 desc;
+------------+------------+
| month | EXPR$1 |
+------------+------------+
| June | 950481 |
| May | 947796 |
| March | 836809 |
| April | 807291 |
| July | 757395 |
| October | 676236 |
| August | 572269 |
| February | 532901 |
| September | 373100 |
| January | 346536 |
+------------+------------+
```

Drill supports SQL aggregate functions such as SUM, MAX, AVG, and MIN. Standard SQL clauses work in the same way in Drill queries as in relational database queries.

Note that back ticks are required for the “month” column only because “month” is a reserved word in SQL.

last-updated: Apr 11, 2022

Return the top 20 sales totals by month and state:

```
0: jdbc:drill:> select `month`, state, sum(order_total) as sales from
orders group by `month`, state
order by 3 desc limit 20;
+------------+------------+------------+
|   month    |   state    |   sales    |
+------------+------------+------------+
| May        | ca         | 119586     |
| June       | ca         | 116322     |
| April      | ca         | 101363     |
| March      | ca         | 99540      |
| July       | ca         | 90285      |
| October    | ca         | 80090      |
| June       | tx         | 78363      |
| May        | tx         | 77247      |
| March      | tx         | 73815      |
| August     | ca         | 71255      |
| April      | tx         | 68385      |
| July       | tx         | 63858      |
| February   | ca         | 63527      |
| June       | fl         | 62199      |
| June       | ny         | 62052      |
| May        | fl         | 61651      |
| May        | ny         | 59369      |
| October    | tx         | 55076      |
| March      | fl         | 54867      |
| March      | ny         | 52101      |
+------------+------------+------------+
20 rows selected
```

Note the alias for the result of the SUM function. Drill supports column aliases and table aliases.

HAVING Clause

This query uses the HAVING clause to constrain an aggregate result.

Set the workspace to dfs.clicks

```
0: jdbc:drill:> use dfs.clicks;
+------------+------------+
|     ok     |  summary   |
+------------+------------+
| true       | Default schema changed to 'dfs.clicks' |
+------------+------------+
1 row selected
```

Return total number of clicks for devices that indicate high click-throughs:

```
0: jdbc:drill:> select t.user_info.device, count(*) from `clicks/
clicks.json` t
group by t.user_info.device
having count(*) > 1000;
+------------+------------+
|   EXPR$0   |   EXPR$1   |
+------------+------------+
| IOS5       | 11814      |
| AOS4.2     | 5986       |
| IOS6       | 4464       |
| IOS7       | 3135       |
| AOS4.4     | 1562       |
```


last-updated: Apr 11, 2022

```
| AOS4.3     | 3039       |
+------------+------------+
```

The aggregate is a count of the records for each different mobile device in the clickstream data. Only the activity for the devices that registered more than 1000 transactions qualify for the result set.

UNION Operator

Use the same workspace as before (dfs.clicks).

Combine clicks activity from before and after the marketing campaign

```
0: jdbc:drill:> select t.trans_id transaction, t.user_info.cust_id customer
from `clicks/clicks.campaign.json` t
union all
select u.trans_id, u.user_info.cust_id  from `clicks/clicks.json` u limit 5;
+-------------+------------+
| transaction |  customer  |
+-------------+------------+
| 35232       | 18520      |
| 31995       | 17182      |
| 35760       | 18228      |
| 37090       | 17015      |
| 37838       | 18737      |
+-------------+------------+
```

This UNION ALL query returns rows that exist in two files (and includes any duplicate rows from those files): `clicks.campaign.json` and `clicks.json`.

Subqueries

Set the workspace to hive:

```
0: jdbc:drill:> use hive;
+------------+------------+
|     ok     |  summary   |
+------------+------------+
| true       | Default schema changed to 'hive' |
+------------+------------+
```

Compare order totals across states:

```
0: jdbc:drill:> select ny_sales.cust_id, ny_sales.total_orders,
ca_sales.total_orders
    from
    (select o.cust_id, sum(o.order_total) as total_orders
    from hive.orders o where state = 'ny' group by o.cust_id) ny_sales
    left outer join
    (select o.cust_id, sum(o.order_total) as total_orders
    from
    hive.orders o where state = 'ca' group by o.cust_id) ca_sales
    on ny_sales.cust_id = ca_sales.cust_id
    order by ny_sales.cust_id
    limit 20;
+------------+------------+------------+
|  cust_id   |  ny_sales  |  ca_sales  |
+------------+------------+------------+
| 1001       | 72         | 47         |
| 1002       | 108        | 198        |
```


last-updated: Apr 11, 2022

```
| 1003       | 83         | null       |
| 1004       | 86         | 210        |
| 1005       | 168        | 153        |
| 1006       | 29         | 326        |
| 1008       | 105        | 168        |
| 1009       | 443        | 127        |
| 1010       | 75         | 18         |
| 1012       | 110        | null       |
| 1013       | 19         | null       |
| 1014       | 106        | 162        |
| 1015       | 220        | 153        |
| 1016       | 85         | 159        |
| 1017       | 82         | 56         |
| 1019       | 37         | 196        |
| 1020       | 193        | 165        |
| 1022       | 124        | null       |
| 1023       | 166        | 149        |
| 1024       | 233        | null       |
+------------+------------+------------+
```

This example demonstrates Drill support for correlated subqueries. This query uses a subquery in the select list and correlates the result of the subquery with the outer query, using the cust_id column reference. The subquery returns the sum of order totals for California, and the outer query returns the equivalent sum, for the same cust_id, for New York.

The result set is sorted by the cust_id and presents the sales totals side by side for easy comparison. Null values indicate customer IDs that did not register any sales in that state.

CAST Function

Use the maprdb workspace:

```
0: jdbc:drill:> use maprdb;
+------------+------------+
|     ok     |  summary   |
+------------+------------+
| true       | Default schema changed to 'maprdb' |
+------------+------------+
1 row selected
```

Return customer data with appropriate data types

```
0: jdbc:drill:> select cast(row_key as int) as cust_id,
cast(t.personal.name as varchar(20)) as name,
cast(t.personal.gender as varchar(10)) as gender, cast(t.personal.age as
varchar(10)) as age,
cast(t.address.state as varchar(4)) as state, cast(t.loyalty.agg_rev as
dec(7,2)) as agg_rev,
cast(t.loyalty.membership as varchar(20)) as membership
from customers t limit 5;
+------------+------------+------------+------------+------------+------------+---------

|  cust_id   |    name    |   gender   |    age     |   state    |
agg_rev   | membership |
+------------+------------+------------+------------+------------+------------+---------

| 10001      | "Corrine Mecham" | "FEMALE"   | "15-20"    | "va"       |
197.00     | "silver"   |
| 10005      | "Brittany Park" | "MALE"     | "26-35"    | "in"       |
230.00     | "silver"   |
| 10006      | "Rose Lokey" | "MALE"     | "26-35"    | "ca"       |
```


last-updated: Apr 11, 2022

```
250.00     | "silver"   |
| 10007      | "James Fowler" | "FEMALE"   | "51-100"   | "me"       |
263.00     | "silver"   |
| 10010      | "Guillermo Koehler" | "OTHER"    | "51-100"   | "mn"       |
202.00     | "silver"   |
+------------+------------+------------+------------+------------+------------+---------

5 rows selected
```

Note the following features of this query:

- The CAST function is required for every column in the table. This function returns the MapR Database/HBase binary data as readable integers and strings. Depending on what encoding is used while populating the MapR Database binary tables/HBase tables, you might have to use CONVERT_TO/CONVERT_FROM functions to decode them.
- The row_key column functions as the primary key of the table (a customer ID in this case).
- The table alias t is required; otherwise the column family names would be parsed as table names and the query would return an error.

Remove the quotes from the strings:

You can use the regexp_replace function to remove the quotes around the strings in the query results. For example, to return a state name va instead of “va”:

```
0: jdbc:drill:> select cast(row_key as int),
regexp_replace(cast(t.address.state as varchar(10)),'"','')
from customers t limit 1;
+------------+------------+
|   EXPR$0   |   EXPR$1   |
+------------+------------+
| 10001      | va         |
+------------+------------+
1 row selected
```

CREATE VIEW Command

Use a mutable workspace:

```
0: jdbc:drill:> use dfs.views;
+------------+------------+
| ok | summary |
+------------+------------+
| true | Default schema changed to 'dfs.views' |
+------------+------------+
```

A mutable (or writable) workspace is a workspace that is enabled for “write” operations. This attribute is part of the storage plugin configuration. You can create Drill views and tables in mutable workspaces.

Create a view on a MapR Database binary table

```
0: jdbc:drill:> create or replace view custview as select cast(row_key as
int) as cust_id,
cast(t.personal.name as varchar(20)) as name,
cast(t.personal.gender as varchar(10)) as gender,
cast(t.personal.age as varchar(10)) as age,
cast(t.address.state as varchar(4)) as state,
cast(t.loyalty.agg_rev as dec(7,2)) as agg_rev,
cast(t.loyalty.membership as varchar(20)) as membership
```

```
from maprdb.customers t;
+------------+------------+
|     ok     |  summary   |
+------------+------------+
| true       | View 'custview' replaced successfully in 'dfs.views' schema |
+------------+------------+
1 row selected
```

Drill provides CREATE OR REPLACE VIEW syntax similar to relational databases to create views. Use the OR REPLACE option to make it easier to update the view later without having to remove it first. Note that the FROM clause in this example must refer to maprdb.customers. The MapR Database binary tables are not directly visible to the dfs.views workspace.

Unlike a traditional database where views typically are DBA/developer-driven operations, filesystem-based views in Drill are very lightweight. A view is simply a special file with a specific extension (.drill). You can store views even in your local filesystem or point to a specific workspace. You can specify any query against any Drill data source in the body of the CREATE VIEW statement.

Dril provides a decentralized metadata model. Drill is able to query metadata defined in data sources such as Hive, HBase, and the filesystem. Drill also supports the creation of metadata in the filesystem.

Query data from the view:

```
0: jdbc:drill:> select * from custview limit 1;
+------------+------------+------------+------------+------------+------------+---------

|  cust_id   |    name    |   gender   |    age     |   state    |
agg_rev   | membership |
+------------+------------+------------+------------+------------+------------+---------

| 10001      | "Corrine Mecham" | "FEMALE"   | "15-20"    | "va"       |
197.00     | "silver"   |
+------------+------------+------------+------------+------------+------------+---------
```

Once the users know what data is available by exploring it directly from the file system, views can be used as a way to read the data into downstream tools such as Tableau and MicroStrategy for analysis and visualization. For these tools, a view appears simply as a “table” with selectable “columns” in it.

Query Across Data Sources

Continue using dfs.views for this query.

Join the customers view and the orders table:

```
0: jdbc:drill:> select membership, sum(order_total) as sales from
hive.orders, custview
where orders.cust_id=custview.cust_id
group by membership order by 2;
+------------+------------+
| membership |   sales    |
+------------+------------+
| "basic"    | 380665     |
| "silver"   | 708438     |
| "gold"     | 2787682    |
+------------+------------+
3 rows selected
```

In this query, we are reading data from a MapR Database binary table (represented by custview) and combining it with the order information in Hive. When doing cross data source queries such as this, you need to use fully qualified table/view names. For example, the orders table is prefixed by “hive,” which

is the storage plugin name registered with Drill. We are not using any prefix for “custview” because we explicitly switched the dfs.views workspace where custview is stored.

Note: Note: If the results of any of your queries appear to be truncated because the rows are wide, set the maximum width of the display to 10000:

```
0: jdbc:drill:> !set maxwidth 10000
```

Do not use a semicolon for this SET command.

Join the customers, orders, and clickstream data:

```
0: jdbc:drill:> select custview.membership, sum(orders.order_total) as
sales from hive.orders, custview,
dfs.`/mapr/demo.mapr.com/data/nested/clicks/clicks.json` c
where orders.cust_id=custview.cust_id and
orders.cust_id=c.user_info.cust_id
group by custview.membership order by 2;
+------------+------------+
| membership |   sales    |
+------------+------------+
| "basic"    | 372866     |
| "silver"   | 728424     |
| "gold"     | 7050198    |
+------------+------------+
3 rows selected
```

This three-way join selects from three different data sources in one query:

- hive.orders table
- custview (a view of the HBase customers table)
- clicks.json file

The join column for both sets of join conditions is the cust_id column. The views workspace is used for this query so that custview can be accessed. The hive.orders table is also visible to the query.

However, note that the JSON file is not directly visible from the views workspace, so the query specifies the full path to the file:

```
dfs.`/mapr/demo.mapr.com/data/nested/clicks/clicks.json`
```

What's Next

Go to Lesson 3: Run Queries on Complex Data Types

Lesson 3: Run Queries on Complex Data Types

Goal

This lesson focuses on queries that exercise functions and operators on self-describing data and complex data types. Drill offers intuitive SQL extensions to work with such data and offers high query performance with an architecture built from the ground up for complex data.

Queries in This Lesson

Now that you have run ANSI SQL queries against different tables and files with relational data, you can try some examples including complex types.

- Access directories and subdirectories of files in a single SELECT statement.
- Demonstrate simple ways to access complex data in JSON files.
- Demonstrate the repeated_count function to aggregate values in an array.

Query Partitioned Directories

You can use special variables in Drill to refer to subdirectories in your workspace path:

- dir0
- dir1
- …

Note that these variables are dynamically determined based on the partitioning of the file system. No up-front definitions are required on what partitions exist.

Set workspace to dfs.logs

```
0: jdbc:drill:> use dfs.logs;
+------------+------------+
| ok | summary |
+------------+------------+
| true | Default schema changed to 'dfs.logs' |
+------------+------------+
```

Query logs data for a specific year

```
0: jdbc:drill:> select * from logs where dir0='2013' limit 10;
+------------+------------+------------+------------+------------+------------+--------

| dir0 | dir1 | trans_id | date | time | cust_id | device | state | camp_id
| keywords | prod_id | purch_flag |
+------------+------------+------------+------------+------------+------------+--------

| 2013 | 11 | 12119 | 11/09/2013 | 02:24:51 | 262 | IOS5 | ny | 0 | chamber
| 198 | false |
| 2013 | 11 | 12120 | 11/19/2013 | 09:37:43 | 0 | AOS4.4 | il | 2 | outside
| 511 | false |
| 2013 | 11 | 12134 | 11/10/2013 | 23:42:47 | 60343 | IOS5 | ma | 4 | and |
421 | false |
| 2013 | 11 | 12135 | 11/16/2013 | 01:42:13 | 46762 | AOS4.3 | ca | 4 |
here's | 349 | false |
| 2013 | 11 | 12165 | 11/26/2013 | 21:58:09 | 41987 | AOS4.2 | mn | 4 | he
| 271 | false |
| 2013 | 11 | 12168 | 11/09/2013 | 23:41:48 | 8600 | IOS5 | in | 6 | i |
459 | false |
| 2013 | 11 | 12196 | 11/20/2013 | 02:23:06 | 15603 | IOS5 | tn | 1 | like
| 324 | false |
| 2013 | 11 | 12203 | 11/25/2013 | 23:50:29 | 221 | IOS6 | tx | 10 | if |
323 | false |
| 2013 | 11 | 12206 | 11/09/2013 | 23:53:01 | 2488 | AOS4.2 | tx | 14 |
unlike | 296 | false |
| 2013 | 11 | 12217 | 11/06/2013 | 23:51:56 | 0 | AOS4.2 | tx | 9 | can't |
54 | false |
+------------+------------+------------+------------+------------+------------+--------
```

This query constrains files inside the subdirectory named 2013. The variable dir0 refers to the first level down from logs, dir1 to the next level, and so on. So this query returned 10 of the rows for February 2013.

last-updated: Apr 11, 2022

Further constrain the results using multiple predicates in the query

```
0: jdbc:drill:> select dir0 as yr, dir1 as mth, cust_id from logs
where dir0='2013' and dir1='8' and device='IOS5' and purch_flag='true'
order by `date`;
+------------+------------+------------+
| yr | mth | cust_id |
+------------+------------+------------+
| 2013 | 8 | 4 |
| 2013 | 8 | 521 |
| 2013 | 8 | 1 |
| 2013 | 8 | 2 |
| 2013 | 8 | 4 |
| 2013 | 8 | 549 |
| 2013 | 8 | 72827 |
| 2013 | 8 | 38127 |
...
```

This query returns a list of customer IDs for people who made a purchase via an IOS5 device in August 2013.

Return monthly counts per customer for a given year

```
0: jdbc:drill:> select cust_id, dir1 month_no, count(*) month_count from
logs
where dir0=2014 group by cust_id, dir1 order by cust_id, month_no limit 10;
+------------+------------+-------------+
|  cust_id   |  month_no  | month_count |
+------------+------------+-------------+
| 0          | 1          | 143         |
| 0          | 2          | 118         |
| 0          | 3          | 117         |
| 0          | 4          | 115         |
| 0          | 5          | 137         |
| 0          | 6          | 117         |
| 0          | 7          | 142         |
| 0          | 8          | 19          |
| 1          | 1          | 66          |
| 1          | 2          | 59          |
+------------+------------+-------------+
10 rows selected
```

This query groups the aggregate function by customer ID and month for one year: 2014.

Query Complex Data

Drill provides some specialized operators and functions that you can use to analyze nested data natively without transformation. If you are familiar with JavaScript notation, you will already know how some of these extensions work.

Set the workspace to dfs.clicks

```
0: jdbc:drill:> use dfs.clicks;
+------------+------------+
| ok | summary |
+------------+------------+
| true | Default schema changed to 'dfs.clicks' |
+------------+------------+
```

Explore clickstream data

```
0: jdbc:drill:> select * from `clicks/clicks.json` limit 5;
+------------+------------+------------+------------+------------+
| trans_id | date | time | user_info | trans_info |
+------------+------------+------------+------------+------------+
| 31920 | 2014-04-26 | 12:17:12
| {"cust_id":22526,"device":"IOS5","state":"il"} | {"prod_id":
[174,2],"purch_flag":"false"} |
| 31026 | 2014-04-20 | 13:50:29
| {"cust_id":16368,"device":"AOS4.2","state":"nc"} | {"prod_id":
[],"purch_flag":"false"} |
| 33848 | 2014-04-10 | 04:44:42
| {"cust_id":21449,"device":"IOS6","state":"oh"} | {"prod_id":
[582],"purch_flag":"false"} |
| 32383 | 2014-04-18 | 06:27:47
| {"cust_id":20323,"device":"IOS5","state":"oh"} | {"prod_id":
[710,47],"purch_flag":"false"} |
| 32359 | 2014-04-19 | 23:13:25 |
{"cust_id":15360,"device":"IOS5","state":"ca"} | {"prod_id":
[0,8,170,173,1,124,46,764,30,711,0,3,25],"purch_flag":"true"} |
+------------+------------+------------+------------+------------+
```

Note that the user_info and trans_info columns contain nested data: arrays and arrays within arrays. The following queries show how to access this complex data.

Unpack the user_info column

```
0: jdbc:drill:> select t.user_info.cust_id as custid, t.user_info.device as
device,
t.user_info.state as state
from `clicks/clicks.json` t limit 5;
+------------+------------+------------+
| custid | device | state |
+------------+------------+------------+
| 22526 | IOS5 | il |
| 16368 | AOS4.2 | nc |
| 21449 | IOS6 | oh |
| 20323 | IOS5 | oh |
| 15360 | IOS5 | ca |
+------------+------------+------------+
```

This query uses a simple table.column.column notation to extract nested column data. For example:

```
t.user_info.cust_id
```

where `t` is the table alias provided in the query, `user_info` is a top-level column name, and `cust_id` is a nested column name.

The table alias is required; otherwise column names such as `user_info` are parsed as table names by the SQL parser.

Unpack the trans_info column

```
0: jdbc:drill:> select t.trans_info.prod_id as prodid,
t.trans_info.purch_flag as
purchased
from `clicks/clicks.json` t limit 5;
+------------+------------+
| prodid | purchased |
+------------+------------+
```

```
| [174,2] | false |
| [] | false |
| [582] | false |
| [710,47] | false |
| [0,8,170,173,1,124,46,764,30,711,0,3,25] | true |
+------------+------------+
5 rows selected
```

Note that this result reveals that the prod_id column contains an array of IDs (one or more product ID values per row, separated by commas). The next step shows how you to access this kind of data.

Query Arrays

Now use the [n] notation, where n is the position of the value in an array, starting from position 0 (not 1) for the first value. You can use this notation to write interesting queries against nested array data.

For example:

`trans_info.prod_id[0]`

refers to the first value in the nested prod_id column and

`trans_info.prod_id[20]`

refers to the 21st value, assuming one exists.

Find the first product that is searched for in each transaction:

```
0: jdbc:drill:> select t.trans_id, t.trans_info.prod_id[0] from `clicks/
clicks.json` t limit 5;
+------------+------------+
|  trans_id  |   EXPR$1   |
+------------+------------+
| 31920      | 174        |
| 31026      | null       |
| 33848      | 582        |
| 32383      | 710        |
| 32359      | 0          |
+------------+------------+
5 rows selected
```

For which transactions did customers search on at least 21 products?

```
0: jdbc:drill:> select t.trans_id, t.trans_info.prod_id[20]
from `clicks/clicks.json` t
where t.trans_info.prod_id[20] is not null
order by trans_id limit 5;
+------------+------------+
|  trans_id  |   EXPR$1   |
+------------+------------+
| 10328      | 0          |
| 10380      | 23         |
| 10701      | 1          |
| 11100      | 0          |
| 11219      | 46         |
+------------+------------+
5 rows selected
```

This query returns transaction IDs and product IDs for records that contain a non-null product ID at the 21st position in the array.

last-updated: Apr 11, 2022

Return clicks for a specific product range:

```
0: jdbc:drill:> select * from (select t.trans_id, t.trans_info.prod_id[0]
as prodid,
t.trans_info.purch_flag as purchased
from `clicks/clicks.json` t) sq
where sq.prodid between 700 and 750 and sq.purchased='true' order by
sq.prodid;
+------------+------------+------------+
| trans_id | prodid | purchased |
+------------+------------+------------+
| 21886 | 704 | true |
| 20674 | 708 | true |
| 22158 | 709 | true |
| 34089 | 714 | true |
| 22545 | 714 | true |
| 37500 | 717 | true |
| 36595 | 718 | true |
...
```

This query assumes that there is some meaning to the array (that it is an ordered list of products purchased rather than a random list).

Perform Operations on Arrays

Rank successful click conversions and count product searches for each session:

```
0: jdbc:drill:> select t.trans_id, t.`date` as session_date,
t.user_info.cust_id as
cust_id, t.user_info.device as device, repeated_count(t.trans_info.prod_id)
as
prod_count, t.trans_info.purch_flag as purch_flag
from `clicks/clicks.json` t
where t.trans_info.purch_flag = 'true' order by prod_count desc;
+------------+--------------+------------+------------+------------+------------+

| trans_id | session_date | cust_id | device | prod_count | purch_flag |
+------------+--------------+------------+------------+------------+------------+

| 37426 | 2014-04-06 | 18709 | IOS5 | 34 | true |
| 31589 | 2014-04-16 | 18576 | IOS6 | 31 | true |
| 11600 | 2014-04-07 | 4260 | AOS4.2 | 28 | true |
| 35074 | 2014-04-03 | 16697 | AOS4.3 | 27 | true |
| 17192 | 2014-04-22 | 2501 | AOS4.2 | 26 | true |
...
```

This query uses a Drill SQL extension, the repeated_count function, to get an aggregated count of the array values. The query returns the number of products searched for each session that converted into a purchase and ranks the counts in descending order. Only clicks that have resulted in a purchase are counted.

Store a Result Set in a Table for Reuse and Analysis

To facilitate additional analysis on this result set, you can easily and quickly create a Drill table from the results of the query.

Continue to use the dfs.clicks workspace

```
0: jdbc:drill:> use dfs.clicks;
+------------+------------+
```


last-updated: Apr 11, 2022

```
| ok | summary |
+------------+------------+
| true | Default schema changed to 'dfs.clicks' |
+------------+------------+
```

Return product searches for high-value customers:

```
0: jdbc:drill:> 0: jdbc:drill:> select o.cust_id, o.order_total,
t.trans_info.prod_id[0] as prod_id
from
hive.orders as o
join `clicks/clicks.json` t
on o.cust_id=t.user_info.cust_id
where o.order_total > (select avg(inord.order_total)
                       from hive.orders inord
                       where inord.state = o.state);
+----------+-------------+----------+
| cust_id  | order_total  | prod_id  |
+----------+-------------+----------+
| 1328     | 73           | 26       |
| 1328     | 146          | 26       |
| 1328     | 56           | 26       |
| 1328     | 91           | 26       |
| 1328     | 74           | 26       |
...
+----------+-------------+----------+
107,482 rows selected (14.863 seconds)
```

This query returns a list of products that are being searched for by customers who have made transactions that are above the average in their states.

Materialize the result of the previous query:

```
0: jdbc:drill:> 0: jdbc:drill:> create table product_search as select
o.cust_id, o.order_total, t.trans_info.prod_id[0] as prod_id
from
hive.orders as o
join `clicks/clicks.json` t
on o.cust_id=t.user_info.cust_id
where o.order_total > (select avg(inord.order_total)
                       from hive.orders inord
                       where inord.state = o.state);
+-----------+----------------------------+
| Fragment  | Number of records written  |
+-----------+----------------------------+
| 0_0       | 107482                     |
+-----------+----------------------------+
1 row selected
```

This example uses a CTAS statement to create a table based on a correlated subquery that you ran previously. This table contains all of the rows that the query returns (107,482) and stores them in the format specified by the storage plugin (Parquet format in this example). You can create tables that store data in csv, parquet, and json formats.

Query the new table to verify the row count

```
0: jdbc:drill:> select count(*) from product_search;
+------------+
|   EXPR$0   |
+------------+
```

```
| 107482     |
+------------+
1 row selected
```

This example simply checks that the CTAS statement worked by verifying the number of rows in the table.

Find the storage file for the table

```
[root@maprdemo product_search]# cd /mapr/demo.mapr.com/data/nested/
product_search
[root@maprdemo product_search]# ls -la
total 451
drwxr-xr-x. 2 mapr mapr      1 Sep 15 13:41 .
drwxr-xr-x. 4 root root      2 Sep 15 13:41 ..
-rwxr-xr-x. 1 mapr mapr 460715 Sep 15 13:41 0_0_0.parquet
```

Note that the table is stored in a file called `0_0_0.parquet`. This file is stored in the location defined by the `dfs.clicks` workspace:

```
"location": "/mapr/demo.mapr.com/data/nested"
```

with a subdirectory that has the same name as the table you created.

Summary

This tutorial introduced Drill and its ability to run ANSI SQL queries against various data sources, including Hive tables, MapR Database binary tables, and filesystem directories. The tutorial also showed how to work with and manipulate complex and multi-structured data commonly found in Hadoop/NoSQL systems.

Now that you are familiar with different ways to access the sample data with Drill, you can try writing your own queries against your own data sources. Refer to the Apache Drill documentation for more information.

Drill-on-YARN

You can install and run Drill under Warden or you can install and run Drill under YARN. YARN (Yet Another Resource Negotiator) is a cluster management tool that automates the resource sharing process in a cluster.

Note: The MapR default security feature introduced in 6.0 is not supported with Drill-on-YARN.

The following sections provide information about Drill-on-YARN, including overview material, installation and configuration instructions, and additional information related to Drill-on-YARN:

Drill-on-YARN Overview

Running Drill as a YARN application (Drill-on-YARN) enables Drill to work alongside other applications, such as Hadoop and Spark, in a YARN-managed cluster. If you are currently running Drill under Warden, you can upgrade Drill and continue to run Drill under Warden, or you can migrate Drill to run under YARN. See Migrate Drill to Run Under YARN for instructions.

YARN assigns resources, such as memory and CPU, to applications in the cluster and eliminates the manual steps associated with installation and resource allocation for stand-alone applications in a multi-tenant environment. YARN automatically deploys (localizes) the Drill software onto each Drill node and manages the Drill cluster. Drill becomes a long-running application with YARN. You can monitor the Drill-on-YARN cluster using the Application Master web UI.

Resource Usage

By design, Drill aggressively uses all of the resources available to run queries at optimal speed. When Drill runs under YARN, you must inform YARN of the resources that Drill needs. The resource settings are descriptive, not proscriptive. Drill does not limit itself to the YARN settings, instead the YARN settings

last-updated: Apr 11, 2022

inform YARN of the resources that Drill will consume so that YARN does not over-allocate those resources to other tasks.

YARN manages CPU, memory, and disks. YARN calls settings for memory and CPU “vcores." You configure Drill’s memory and then inform YARN of the Drill configuration. Drill uses all available disk I/O and CPU.

Components

Several software components work together to run Drill as a YARN application. Drill, YARN, and the Drill-on-YARN application collectively provide the components required to run Drill under YARN.

The following table lists the software components with their descriptions:

Software	Component	Description
YARN	Resource Manager	The Resource Manager manages the set of applications running on the cluster. Each cluster must have one Resource Manager.
	Node Manager	The Node Manager manages the application tasks running on a particular node. Each node in a cluster must have one Node Manager.
Drill	Drillbit	A Drillbit is the Drill daemon software that YARN runs on each node in the cluster. SeeDrill Query Execution. The Drillbit process on the node acts as the application task.
	Client	A client, such as JDBC, ODBC, or SQLLine sends queries to a Drillbit. The client uses ZooKeeper to discover a Drillbit that the client treats as the Foreman. SeeDrill Clients.
	Drill distribution archive	The Drill distribution archive is a Drill distribution .tar.gz file included with the Drill installation. Drill-on-YARN uploads this archive to the distributed filesystem (DFS). YARN downloads (localizes) the file to each worker node.
ZooKeeper	ZooKeeper	ZooKeeper is the service that tracks the available set of Drillbit processes. The Foreman for a query uses ZooKeeper to identify the set of available drill nodes that can run the query. SeeDrill Query Execution.
Drill-on-YARN Application	Application Master (AM)	The Application Manager requests containers from the Resource Manager and launches Drillbits using those containers. The AM monitors Drillbits, detects failures, and restarts failed Drillbits. The AM also provides a web UI to manage the Drill cluster.

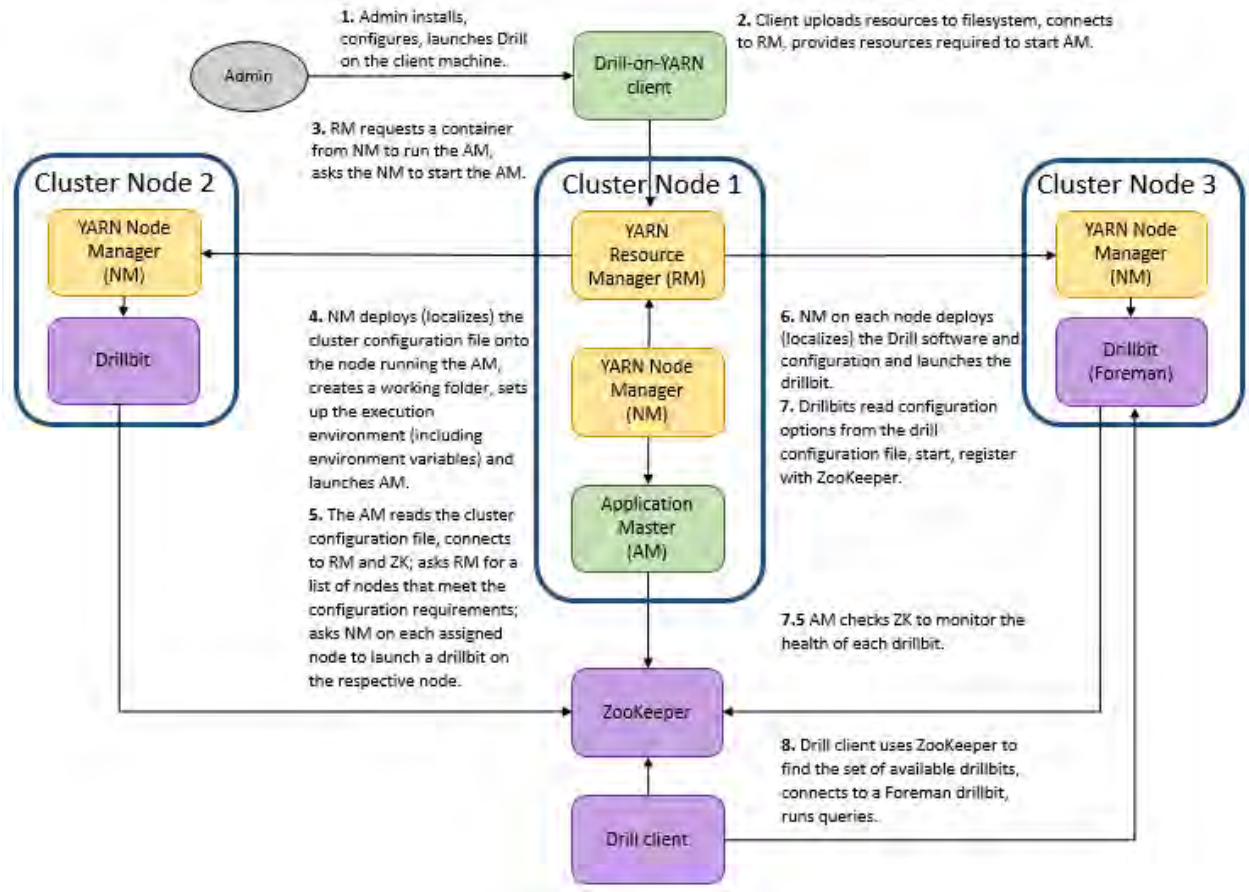
last-updated: Apr 11, 2022

	Drill-on-YARN client	The Drill-on-YARN client is a command-line program that starts, stops, and monitors the Drill cluster. The client provides the information that YARN needs to start the Application Master. The client can run on any machine that has both the Drill and YARN client software. The client does not have to be part of the YARN cluster.
	drill-on-yarn.conf	The drill-on-yarn.conf configuration file provides the information that Drill-on-YARN needs to manage the Drill cluster. This file is separate from the configuration files for Drill itself.

Component Workflow

Running Drill as a YARN application is mostly an automated process carried out by the Drill and YARN components. After an administrator installs, configures, and launches Drill from the Drill-on-YARN client, YARN deploys (localizes) Drill on to designated nodes and starts the Drill process on each node.

The following diagram shows the workflow between the components in a cluster with the steps that Drill and YARN complete to deploy and run Drill as a YARN application:



Configuring Drill to Run Under YARN

To run Drill under YARN, you must have the YARN version of Drill installed on the node designated as the Drill-on-YARN client. If you have not already planned your cluster and installed the YARN-ready version of

Drill, see Install Drill to Run Under YARN and then return to this topic to configure Drill to run as a YARN application.

Configuring Drill to run under YARN requires modifications to Drill, Drill-on-YARN, and YARN configuration files. The following sections provide the information needed to make the changes to the configuration files, as well as information about how to launch Drill under YARN and validate the cluster configuration and status of the Drill nodes.

Complete the following steps to configure Drill to run under YARN, launch Drill as a YARN application, and validate the cluster:

Step 1: Configure Drill

Drill configuration under Drill-on-YARN differs from Drill configuration under Warden. You must create a site directory to contain the site-specific files for Drill. Drill-on-YARN copies the site directory to every node so that each node has the configuration settings. Having a site directory also simplifies upgrades because you can just delete the old Drill distribution and install the new one while the site-specific files remain unchanged in the site directory.

The drill-env.sh file contains only custom configurations. MapR-specific configuration settings reside in distrib-env.sh, a file separate from the site-specific settings. When you migrate an existing Drill installation to run under YARN, you must modify the drill-env.sh to remove the Drill and settings, leaving only your site-specific settings.

When you finish configuring Drill, use the site directory to test Drill, including starting, checking status, and stopping Drill.

Note: If you installed the mapr-drill-yarn package on nodes other than the Drill-on-YARN client in order to make SQLLine accessible to users, the site directory must be accessible from all nodes. You can copy the configurations across all the nodes, as you did when you ran Drill under the Warden service. Alternatively, you can put the site directory in a shared filesystem nfs mount to extend the configuration. When users launch SQLLine, they should provide the ZooKeeper connection string to launch Drill.

Create the Site Directory

To create the `site` directory, complete the following steps as the user that installed Drill and will run the Drill-on-YARN client application:

1. Create the site directory and an environment variable for the directory:

```
export DRILL_SITE=/opt/mapr/drill/site
mkdir $DRILL_SITE
```

Note: The variable is not required. It is used for convenience in the documentation.

last-updated: Apr 11, 2022

2. Copy the drill-env.sh, drill-override.conf, drill-on-yarn.conf, and distrib-env.sh files from $DRILL_HOME/conf/ into the `site` directory. In the following example, $DRILL_HOME is the location of the new Drill installation (usually /opt/mapr/drill/drill-<version>).

```
cp $DRILL_HOME/conf/drill-override.conf $DRILL_SITE
cp $DRILL_HOME/conf/drill-env.sh $DRILL_SITE
cp $DRILL_HOME/conf/drill-on-yarn.conf $DRILL_SITE
cp $DRILL_HOME/conf/distrib-env.sh $DRILL_SITE
```

Note: Copy any configuration changes from drill-env.sh file in the previous Drill installation over to the drill-env.sh file in the `site` directory. Do not include the memory settings when you copy over your previous configurations. These changes must be made in the drill-on-yarn.conf file described in Step 3: Configure YARN to Run Drill.

Note: Never modify `distrib-env.sh`. The `distrib-env.sh` script contains MapR settings that you should not change. You copy this file to the `site` directory because it often contains values set during Drill installation. When you upgrade Drill, replace the file with the latest version from $DRILL_HOME/conf.

3. If you developed custom code (data sources or user-defined functions), place the Java JAR files in $DRILL_SITE/jars. If you have code from your prior Drill installation, copy the JAR files from $PREV_DRILL/jars/3rdparty to $DRILL_SITE/jars.

```
cp $PREV_DRILL/jars/3rdparty/yourJarName.jar $DRILL_SITE/jars
```

Note: Only copy the JAR files that you added. Do not copy JAR files that shipped with the prior Drill version.

4. Add native libraries to the `site` directory. If you used a native library, such as the JPAM library in prior versions of Drill, place the native libraries in $DRILL_SITE/lib to enable YARN to automatically copy (localize) them to each node that runs Drill.

```
cp native_libraries $DRILL_SITE/lib
```

Modify the drill-env.sh File

Copy any configuration changes from the drill-env.sh file in the previous Drill installation over to the drill-env.sh file in the `site` directory. Memory settings under Drill-on-YARN are now part of the drill-on-yarn.conf file. Modify the memory settings in $DRILL_SITE/drill-env.sh, as shown below, to ensure that the Drill memory settings match the amount of memory that Drill-on-YARN requires.

To modify drill-env.sh, complete the following steps:

1. Review each line in $PREV_DRILL/conf/drill-env.sh for settings you added, and copy them into the new $DRILL_SITE/drill-env.sh file.

Note: If you do not recall whether you customized settings, you can compare your file with the original version of drill-env.sh that shipped with the prior Drill version.

last-updated: Apr 11, 2022

2. Locate the following lines in `drill-env.sh` and note the values:

```
DRILL_MAX_DIRECT_MEMORY="<value>"
DRILL_HEAP="<value>"
```

Replace those lines with the following lines, substituting the values in the new lines with the values from the old lines:

```
export DRILL_MAX_DIRECT_MEMORY=${DRILL_MAX_DIRECT_MEMORY:-"<value>"}
export DRILL_HEAP=${DRILL_HEAP:-"<value>"}
```

Note: If you do not intend to run Drill outside of YARN, you can remove the two lines shown above from drill-env.sh.

Note: If you do not make this change, Drill ignores the memory settings in the drill-on-yarn.conf file. If you are installing Drill fresh, and do not have an existing file, you can skip this step. Files in Drill 1.8 and later have the correct format.

Note: When you install Drill, the MapR Installer automatically adds the HADOOP_HOME variable, which points the current MapR-provided Hadoop to your distrib-env.sh. If HADOOP_HOME is located elsewhere, change this location in drill-env.sh

Use the Site Directory to Test Drill

You will use the `site` directory each time you start Drill using the `--site` or `--config` option. Use the option to verify that the configuration works by starting Drill as a stand-alone service on a single node.

```
drillbit.sh --site $DRILL_SITE start
```

Wait a few seconds and then verify that Drill continues to run:

```
drillbit.sh --site $DRILL_SITE status
```

You can also use the Drill Web Console for the Drillbit to verify that Drill has the proper settings. Once satisfied that the configuration is connect, stop Drill:

```
drillbit.sh --site $DRILL_SITE stop
```

Note: If you run a Drilbit with the `--site` (`--config`) option and you want to use SQLLine, you must add the option to SQLLine:

```
sqlline --site $DRILL_SITE
```

Tip: If you find that specifying the `--site` option becomes tedious, you can set the DRILL_CONF_DIR variable in your environment:

```
export DRILL_CONF_DIR="$DRILL_SITE"
drillbit.sh start
```

Step 2: Configure Drill-on-YARN

To configure Drill to run as a YARN application, modify the $DRILL_SITE/drill-on-yarn.conf cluster configuration file to suit the needs of your cluster. This file is a “starter” configuration file that corresponds to the simplest Drill cluster. The drill-on-yarn.conf file is in the same HOCON format as drill-override.conf.

last-updated: Apr 11, 2022

Consult the $DRILL_HOME/conf/drill-on-yarn-example.conf file as an example. However, do not just copy the example file. Instead, copy only the specific configuration settings that you need; the others will automatically take the Drill-defined default values.

Note: Make sure that resources can accommodate the Drill memory, CPU, and disk requirements.

The following sections list the configuration settings required to run Drill under YARN:

Drill Memory Settings

The following configuration sets the Java heap size and amount of direct memory the node can allocate to Drill:

```
drillbit: { heap: "<value>" max-direct-memory: "<value>" }
```

When you add the configuration, use the same values set in the following parameters of the `drill-env.sh` file, if you did not remove these lines when you modified `drill-env.sh`:

```
export DRILL_MAX_DIRECT_MEMORY=${DRILL_MAX_DIRECT_MEMORY:-"<value>"}
export DRILL_HEAP=${DRILL_HEAP:-"<value>"}
```

Drill-on-YARN copies these values into the environment variables when launching each Drillbit. Drill also uses additional JVM memory. For example, Drill uses a code cache to hold classes generated at runtime. The default size of the cache is 1 GB:

```
drillbit: { code-cache: "1G" }
```

Typically, you do not need to change the code cache size, but you must account for it when computing the YARN container size.

YARN Container Size

The following configuration sets the YARN container size required to run Drill as a YARN application.

```
drillbit: {
   memory-mb: 14336
 }
```

The default value is 14GB. Typically, this size is the sum of the heap and direct memory. However, if you use custom libraries that perform their own memory allocation, or launch sub-processes, you must account for that memory usage as well. Note that YARN memory is expressed in MB. To compute the container size, start with the values used for the heap, direct memory, and code cache settings, as shown in the following example:

```
drillbit: {
    heap: "4G"
    max-direct-memory: "8G"
    code-cache: "1G"
    memory-mb: 14336
  }
```

The values shown above are the Drill defaults. You may use larger values. Although the three values account for the bulk of Drill memory, the JVM itself also has a certain overhead. Assume that the overhead is about 1 GB, though the amount varies depending on the workload.

Add the four values together to get a memory requirement in GB.

```
Total memory = 8G + 4G + 1G + 1G = 14G
```


last-updated: Apr 11, 2022

YARN sizes containers in megabytes. Convert GB to MB:

```
Container size = 14G * 1024 = 14336 MB
```

Set this size in drill-on-yarn.conf:

```
drillbit: { memory-mb: 14336 }
```

CPU

The following configuration sets the CPU to allocate to Drill:

```
drillbit: { vcores: <value> }
```

Drill is a CPU-intensive operation and greatly benefits from each additional core. YARN does not limit the number of cores used by an application. Rather, this number reports to YARN the average CPU usage of Drill so that YARN can use the number when deciding how many other applications to run on the same node.

Drillbit Cluster Configuration

The following configuration sets the cluster group:

```
cluster: [ { name: "drillbits" type: "basic" count: 1 } ]
```

Drill-on-YARN uses the concept of a “cluster group” of Drillbits to describe the set of drillbits to launch. Currently, only the “basic” type of group is supported. A basic group launches drillbits anywhere in the YARN cluster where a container is available. For a basic group, specify the group type and the number of drillbits to launch.

Note: The syntax says that cluster is a list that contains cluster group objects contained in braces. Drill currently supports just one group.

The name is optional. It appears in the Application Master web UI. Type must be set to “basic.” Set the count to the number of hosts on which Drill is to run at launch time. You can resize the Drill cluster after the cluster is launched.

YARN Queue Labels

The following configuration sets the YARN queue labels that identify the cluster nodes that run Drill:

```
yarn: { queue: "<queue_name>" }
```

The MapR distribution of YARN provides queue labels for assigning YARN applications to specific queues. See Label-Based Scheduling for YARN Applications. You can use queue labels with Drill to identify the YARN queue that should run Drill.

To use queue labels, complete the following steps:

1. Create a node label.
2. Assign the label to the nodes that are to run Drill.
3. Create a Drill-specific queue that uses the node label.
4. Configure Drill-on-YARN to use the queue.

last-updated: Apr 11, 2022

Suppose you create a queue called “drill.” Setting the following configuration causes Drill-on-YARN to launch through the drill queue:

```
yarn: { queue: "drill" }
```

Set queue to the name a name of your choice. When Drill-on-YARN launches, both the Application Master and drillbits run only on nodes with the same node label as the queue.

DFS Location

The following configuration sets the dfs location:

```
dfs: { app-dir: "/user/drill" }
```

Drill copies the archive in to the MapR filesystem in a location you provide. The default is /user/drill, however you can specify a different location. You do not have to specify the MapR File System connection information; MapR defines this automatically.

Step 3: Configure YARN to Run Drill

YARN default settings are optimized for MapReduce applications. MapReduce applications use a limited amount of memory; however, Drill is long-running and consumes a significant amount of resources. Adjust the YARN memory configuration to allow YARN to allocate containers large enough to run Drill. Exclude the YARN container directory from `systemd-tmpfiles` to prevent `systemd-tmpfiles` from removing Drill’s container files while Drill runs.

Increase Maximum Container Size

YARN provides a number of parameters to control the amount of resources available to applications. The MapR distribution of YARN sets most of these parameters automatically, except for the maximum container size, which is left at the Apache YARN default of 8 GB. Typical Drill configurations use significantly more memory. Therefore, you must increase the YARN maximum container size on each node to suit the needs of Drill. You can use the YARN Resource Manager web UI to determine the amount of memory available on each node.

Note: YARN resource requirements match the Drill resource requirements.

To increase the maximum container size, determine the required container size from the Drill setting in `drill-on-yarn.conf`, which you previously set in step 2:

```
drillbit: {
   memory-mb: 14336
 }
```

Use this number to set the `yarn.scheduler.maximum-allocation-mb` parameter in `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop.<version>`, substituting the number of the version you have installed.

Edit `yarn-site.xml` to add the following:

```
<property>
   <name>yarn.scheduler.maximum-allocation-mb</name>
   <value>14336</value>
   <description>Set to allow Drill containers 14G.</description>
 </property>
```

Note: You must update this configuration on every YARN node.

Restart the YARN Resource Manager to pick up change, and use the YARN Resource Manager UI to verify that the maximum container size shows the new value.

Exclude the YARN Container Directory from systemd-tmpfiles

The system puts the YARN Node Manager container files in the `/tmp` directory. Most system administrators configure `systemd-tmpfiles` to periodically remove files in `/tmp`. Since Drill-on-YARN is a long-running YARN application, `systemd-tmpfiles` can remove Drill's container files while Drill runs. If this occurs, you must manually shut down the Drill cluster because `systemd-tmpfiles` will have removed the `pid` file that YARN needs to manage Drill.

You can prevent `systemd-tmpfiles` from cleaning up Drill's container files by adding a new configuration file to `/etc/tmpfiles.d/`, for example `/etc/tmpfiles.d/exclude-nm-local-dir.conf`, with the following configuration:

```
x /tmp/hadoop-mapr/nm-local-dir/*
```

Note: This configuration prevents `systemd-tmpfiles` from cleaning the `/nm-local-dir` directory when cleaning `/tmp`.

Example

```
$ cat /etc/tmpfiles.d/exclude-nm-local-dir.conf
x /tmp/hadoop-mapr/nm-local-dir/*
```

Step 4: Launch Drill Under YARN

Now that the Drill and YARN configuration is complete, you can issue the `start` command from the Drill-on-YARN client to launch Drill under YARN. Launching Drill-on-YARN from the client starts Drill and brings Drill up on other nodes.

Issuing the start command starts the YARN Application Master, which then works with YARN to start the Drillbits. The Application Master provides a web UI to monitor the cluster.

Note: To simplify debugging, you can set the cluster size to a single node. Once you confirm that a single node works, increase the node count.

Launch Drill under YARN as the `mapr` user. For example, if you installed Drill as `mapr` launch Drill as the `mapr` user.

Note: If you launch Drill as `root` and the system returns an error failing the launch attempt, launch Drill as a user with permissions, such as `mapr`.

Issue the following command to start Drill under YARN:

```
$DRILL_HOME/bin/drill-on-yarn.sh --site $DRILL_SITE start
```

Note: To run SQLLine, you must also add the `--site` option:

```
$DRILL_HOME/bin/sqlline --site $DRILL_SITE
```

Tip: To avoid typing the `site` argument each time you launch Drill under YARN, set an environment variable:

```
export DRILL_CONF_DIR=$DRILL_SITE
$DRILL_HOME/bin/drill-on-yarn.sh start
```

After you issue the start command, a number of lines describing the start-up process print. The tool automatically archives and uploads the site directory, which YARN copies (along with the Drill software) onto each node. A URL that includes both the host and the port number displays. Enter the URL in a web browser to access the Application Master web UI.

Note: When you launch Drill from the Drill-on-YARN client, the Application Master can come up on any node. Save the provided URL to share with other users so they can also access the Application Master. Alternatively, you can run the `status` command to see the URL or go to the YARN Resource Manager UI to get the link.

See Drill-on-YARN Command Line Tool for additional commands, including stop, status, and resize. See Application Master Web UI for cluster monitoring information.

Step 5: Validate Cluster Configuration and Status

The Drill-on-YARN command line tool prints a URL, for the Drill Application Master process, that you can use to monitor the cluster. The Drillbits should be up and running, unless the upload or launch failed.

Failed upload	If the upload fails, the most likely reason is that the HADOOP_HOME variable is not set, or the user launching Drill-on-YARN does not have permission to create or write to the DFS location set in the Drill-on-YARN configuration file.
Failed launch	If the launch fails, verify that the YARN maximum container size was set to be at least as large as the Drill container size and that YARN can provide the number of vcores and disks you have requested for Drill.

Monitor Cluster Activity

When the cluster launches successfully, you can verify the status of the components in the Drill cluster from the command line or you can copy the Application Master URL into your browser to use the Application Master web UI.

To check the status from the command line, issue the following command:

```
$DRILL_HOME/bin/drill-on-yarn.sh --site $DRILL_SITE status
```

- Verify that the Drillbits are in the Running state (or transition to that state after a few moments.) If the Drillbits remain in the Requesting state, the likely cause is that YARN cannot provide a container of the requested size. You can access this information on the Drillbits page in the Application Master web UI.
- Verify that the Application Master has correctly picked up the YARN-related configuration from drill-override.conf and drill-on-yarn.conf. You can access this information on the Configuration page in the Application Master web UI.
- Verify that the Application Master is running. The Drill Application Master uses ZooKeeper to verify that only one Application Master runs per Drill cluster. The Application Master fails if ZooKeeper is down, is configured incorrectly in drill-override.conf, or if another Application Master is running for the same cluster.
- Verify that that the configurations in are correct in drill-override.conf if there are Drillbit failures. Drill-on-YARN detects Drillbit failures and retries each Drillbit a few times. If the Drillbit continues to fail, the node is black-listed for that run of Drill-on-YARN. You can see failures in the History page of the web UI. If your Drillbits fail, the most likely reason is misconfiguration within drill-override.conf. Use YARN to locate and view the Drill logs for the failed container.

last-updated: Apr 11, 2022

Resize the Cluster

If the cluster works well for a single node, you can increase the number of nodes.

Note: In the Drill 1.8 release, adding nodes can be done while users run queries. However, removing nodes from the cluster causes queries to fail, and so should only be done when the cluster is idle.

To make a permanent change, modify the cluster size in drill-on-yarn.conf:

```
cluster: [
   {
     ...
     count: 5
   }
 ]
```

You can also resize the cluster dynamically. To add two nodes from the command line:

```
$DRILL_HOME/bin/drill-on-yarn.sh --site $DRILL_SITE resize +2
```

To set the cluster size to a total of five nodes:

```
$DRILL_HOME/bin/drill-on-yarn.sh --site $DRILL_SITE resize 5
```

To remove one of the nodes:

```
$DRILL_HOME/bin/drill-on-yarn.sh --site $DRILL_SITE resize -1
```

You can also resize your cluster using the Manage page of the Application Master web UI, by entering the number of desired nodes and clicking **Go**.

Stop the Cluster

You can stop the cluster from the command line or from the Manage page in the Application Master web UI.

To stop the cluster from the command line, issue the following command:

```
$DRILL_HOME/bin/drill-on-yarn.sh --site $DRILL_SITE stop
```

Migrate Drill to Run Under YARN

Explains how to migrate Drill to run under YARN instead of the MapR Warden service.

When you migrate Drill to run under YARN, you must back up configurations, including files and storage plugins, shutdown the Drill cluster running under Warden, and uninstall Drill on all Drill nodes in the cluster. You may also want to determine which system settings have been changed from the Drill defaults.

Note: Drill-on-YARN is an advanced feature used to manage a production Drill cluster. Only skilled Drill and MapR administrators, familiar with YARN, should configure Drill to run under YARN. If you are new to Drill, consider running Drill under the MapR Warden service until you are familiar with Drill and Drill cluster management.

The sections below provide the tasks required to migrate Drill under YARN:

Tasks

Complete the following tasks when you want to migrate Drill to run under YARN:

Backup configurations and UDFs	Back up configuration files, storage plugin configurations, and UDFs (user-defined functions) or

last-updated: Apr 11, 2022

custom JAR files. Back up the configuration files located in `/opt/mapr/drill/drill-<version>/conf`, including `drill-override.conf` and `drill-env.sh`, to preserve your ZooKeeper configuration and any options or custom configurations specified in the files. Also back up `logback.xml` if you configured the file for the Lilith software.

- To back up configuration files, go to `/opt/mapr/drill/drill-<version>/conf`, and copy the files to a location outside of the Drill installation directory.

  ```
  cp <file_name> drill-env.sh /
  path/to/directory
  ```

- To back up custom JARs or UDFs, go to `/opt/mapr/drill/drill-<version>/jars/3rdparty` and copy the JARs or UDFs to a location outside of the Drill installation directory.
- To back up storage plugin configurations, complete the following steps:
  1. Start the Web Console. The Drill node that you use to access the Web Console must be a node that is currently running the Drillbit process.
  2. Click **Storage**.
  3. Click **Update** next to a storage plugin.
  4. Copy the configuration to a text file, and save the file.
  5. Repeat steps 3 and 4 for each storage plugin configuration that you want to save.

Verify system option settings

Drill should save set system options when you migrate Drill. However, you may want to verify which system options were changed from the default settings beforehand. You can run the following query to see which system options were changed from the defaults:

```
select * from sys.options where
status = 'CHANGED';
```

Example:

```
select * from sys.options where
status = 'CHANGED';
+-------------------------------------+----------+

|                name
|   kind   |  type   |  status  |
num_val  | string_val  | bool_val  |
float_val  |
+-------------------------------------+----------+

| exec.errors.verbose
| BOOLEAN  | SYSTEM  | CHANGED  |
null     | null        | true      |
```


last-updated: Apr 11, 2022

```
null       |
| new_view_default_permissions
| STRING   | SYSTEM  | CHANGED  |
null     | 777         | null      |
null       |
| planner.enable_decimal_data_type
| BOOLEAN  | SYSTEM  | CHANGED  |
null     | null        | true      |
null       |
| planner.enable_limit0_optimization
| BOOLEAN  | SYSTEM  | CHANGED  |
null     | null        | true      |
null       |
+------------------------------------+----------+

4 rows selected (0.541 seconds)
```

After you migrate, run the query again to verify that the system options are still set. If not, set the options.

Shutdown the Drill cluster

Issue the following command to shutdown the existing Warden-managed Drill cluster:

```
maprcli node services -name
drill-bits -action stop -nodes <node
host names separated by a space>
```

Note: Do not shutdown nodes when queries are in progress.

Verify that Drillbits stopped

Run the following command to verify that the Drillbit service is no longer running on each node:

```
ps -ef | grep -i drill
```

No Drill processes should print to screen when you run this command.

You can also log in to the Control System at `https://<host name>:8443` to verify the status of the Drillbit service. You should not see Drillbit as a listed service.

Uninstall Drill

Uninstall Drill and then run `configure.sh -R` to refresh the configuration.

Issue the command appropriate for your system as root or using sudo to uninstall the mapr-drill package:

Operating System	Command
RedHat/CentOS	`yum remove mapr-drill`
Ubuntu	`apt-get remove mapr-drill`

Note: Verify that Drillbit processes stopped.

Additional Drill-on-YARN Configuration Options

You can include additional configuration options in the `$DRILL_SITE/drill-on-yarn.conf` file for specialized cases. For example, you can customize the Application Master web UI port or Application Master settings.

last-updated: Apr 11, 2022

Refer to the `drill-on-yarn-example.conf` file in $DRILL_HOME/conf to see examples of the additional options. Do not use the example file.

The following list describes the changes that you can make for several of the Drill-on-YARN components:

Application Name

You can customize the application name that appears when starting or stopping the Drill cluster and in the Drill-on-YARN web UI. Change the value of the following option to a name you prefer:

```
app-name: "My Drill Cluster"
```

Application Master Web UI Port

If you run multiple Drill clusters in a YARN cluster, YARN may assign two Drill Application Master processes on the same node. To avoid port conflicts, change the HTTP port for one or both of the Drill clusters. Change the value of the following option to a different port number:

```
drill.yarn:
 http: {
   port: 12345
 }
}
```

Application Master Settings

You can customize certain Application Master properties. All of the Application Master properties are prefixed with `drill.yarn.am`, for example `drill.yarn.am.heap`.

The following table lists the Application Master properties with their default settings:

Name	Description	Default
memory-mb	Memory, in MB, to allocate to the Application Master.	14336
vcores	Number of CPUS to allocate to the Application Master.	1
heap	Java heap for the Application Master.	450M

Drillbit

You can customize certain properties that control the Drillbit processes. All of the Drillbit properties are prefixed with `drill.yarn.drillbit`, for example `drill.yarn.drillbit.disks`.

Note: You can specify Drill disk usage to YARN, however Drill uses all disks regardless of the setting.

The following table lists the Drillbit properties with their default settings:

Name	Description	Default

last-updated: Apr 11, 2022

code-cache	Code cache that holds classes generated at runtime.	1G
memory-mb	Memory, in MB, to allocate to the Drillbit.	13000
vcores	Number of CPUS to allocate to the AM.	4
disks	Number of disk equivalents consumed by Drill (on versions of YARN that support disk resources.)	1
heap	Java heap memory.	4G
max-direct-memory	Direct (off-heap) memory for the Drillbit.	4G
log-gc	Enables Java garbage collector logging.	false
class-path	Additional class-path entries.	blank

Mapping of drill-env.sh to drill-on-yarn.conf Options

When you run Drill as a standalone application, you set startup options, such as Drillbit memory, in the $DRILL_HOME/conf/drill-env.sh start up script. Under YARN, Drill still reads $DRILL_SITE/drill-env.sh for configuration options, however Drill-on-YARN provides the $DRILL_SITE/drill-on-yarn.conf file to configure options that were formerly set in drill-env.sh.

The following table maps the drill-env.sh environment variables to their equivalent configuration parameters in drill-on-yarn.conf:

drill-env.sh Environment Variable	**drill-on-yarn.conf Configuration Parameter**
DRILL_MAX_DIRECT_MEMORY *	drill.yarn.drillbit.max-direct-memory
DRILL_HEAP *	drill.yarn.drillbit.heap
DRILL_JAVA_OPTS	drill.yarn.drillbit.vm-args (Added to those in drill-env.sh.)
SERVER_GC_OPTS (to add GC logging)	Drill.yarn.drillbit.log-gc (To enable GC logging)
DRILL_HOME	Set automatically when files are localized (drill.yarn.drill-install. localize is true), else drill.yarn.drill-install. drill-home.
DRILL_CONF_DIR	Set automatically when files are localized, else uses the normal defaults.

DRILL_LOG_DIR	Set automatically to point to YARN's log directory unless disabled by setting drill.yarn.drillbit. disable-yarn-logs to false. If disabled, uses the normal Drill defaults.
DRILL_CLASSPATH_PREFIX *	Drill.yarn.drillbit. prefix-class-path
HADOOP_CLASSPATH *	drill.yarn.hadoop.class-path (or, better drill.yarn.drillbit. extn-class-path.)
HBASE_CLASSPATH *	Drill.yarn.hadoop. hbase-class-path (or, better drill.yarn.drillbit. extn-class-path.)
EXTN_CLASSPATH * (New in Drill 1.8.)	Drill.yarn.drillbit. extn-class-path
DRILL_CLASSPATH *	drill.yarn.drillbit.class-path

Multiple Drill Clusters within YARN

You can define multiple Drill clusters within a single YARN cluster. Each Drill cluster is a collection of Drillbits that work as an independent unit. For example, you might define one test Drill cluster that consists of a few machines on the same physical cluster that runs larger Drill clusters for development and marketing.

You must assign each Drill cluster a distinct ZooKeeper entry. Drill uses ZooKeeper to coordinate activities. Each Drill cluster also needs a distinct set of ports because YARN may launch Drillbits from different clusters on the same physical node.

The following steps summarize the process for defining multiple Drill clusters on a single YARN cluster:

1. Create a new site directory.
2. Configure Drill.
3. Configure Drill-on-YARN.
4. Start the cluster.

The following task provides instructions for each of the steps required to configure and run multiple Drill clusters under YARN:

Defining Multiple Drill Clusters within YARN

Complete the following steps to define multiple Drill clusters under YARN:

1. Create a new "site" directory under $DRILL_HOME, and create an environment variable for the directory.

   ```
   mkdir $DRILL_HOME/<site_name>
   export $<SITE_NAME>_SITE=$DRILL_HOME/<site_name>
   ```

2. Copy the following configuration files from the existing "site" directory into the new "site" directory:
   - drill-override.conf
   - drill-env.sh
   - drill-on-yarn.conf
   - distrib-env.sh

3. Modify the settings in the `drill.exec` section of `$DRILL_SITE/drill-override.conf` to configure the new Drill cluster to act independently of the other Drill cluster(s), or share settings, such as storage plugin configurations. For the new Drill cluster to act independently, give `zk.root` a distinct name from the existing clusters. In the more advanced scenario where the clusters share configurations, give `zk.root` the same name as the existing Drill clusters. When the clusters share the same root, they must have distinct cluster-id values. The user, bit, and http ports must have values distinct from all the other Drill clusters.

   The following example shows a number 1 added to the first digit of the default port numbers, however you can choose any available ports.

   ```
   drill.exec: {
    cluster-id: "drillbits",
    zk: {
      root: "<site_name>"
      connect: "zk-host:5181"
    }

    rpc {
      user.server.port: 41010
      bit.server.port: 41011
    }
    http.port: 9047
   }
   ```

4. Modify the `drill-on-yarn.conf` configuration file for the new cluster. The new cluster must have a distinct name, a distinct upload directory in the filesystem, and a distinct port number.

   The following settings in the `drill.yarn` section must have distinct values for the new cluster:

   ```
   drill.yarn: {
    app-name: "Distinct Cluster Name"

    dfs: {
      app-dir: "/upload/directory"
    }

    http : {
      port: <distinct port number>
    }
   }
   ```

5. Start the new cluster from the "site" directory that correlates with the new cluster.

   ```
   $DRILL_HOME/bin/drill-on-yarn.sh --site $NEW_SITE start
   ```

Drill-on-YARN Command Line Tool

Run the Drill-on-YARN command line tool from the Drill-on-YARN client and use it to start, stop, resize, and check the status of the Drill cluster. When you launch Drill from the command line, the tool automatically archives and uploads the “site” directory, which YARN deploys (along with Drill) onto each node.

You can access the Drill-on-YARN command line tool in the following directory:

```
$DRILL_HOME/bin/drill-on-yarn.sh --site $DRILL_SITE command
```

where *command* is the operation you want to perform, such as `start`.

To avoid having to type the site argument for each command, set an environment variable:

```
export DRILL_CONF_DIR=$DRILL_SITE
```

The following example shows the start command after setting the environment variable:

```
$DRILL_HOME/bin/drill-on-yarn.sh start
```

Command Summary

The following table lists the commands and provides a brief summary for each:

Command	Description
start	Starts the Drill cluster. Prints the startup status followed by a summary of the application.
status	Retrieves basic information about the Drill cluster.
stop	Stops the Drill cluster.
resize <value> resize + <increase_node_count_by> resize - <decrease_ node_count_by>	Adds or removes nodes in the Drill cluster while the cluster runs. You can specify the exact number of nodes you want to run, or you can use +/- to increase or decrease the current node count by a certain amount.
clean	Removes the cached Drill archive from the designated DFS directory.

Commands

The following sections provide detailed information and examples for each of the commands listed in the command summary:

start

The start command launches Drill and provides a startup status followed by a summary of the application.

The first line in the summary displays the cluster name from the configuration file to confirm which cluster is starting.

```
Launching Drill-on-YARN...
```

The next line shows the YARN application ID and tracks the job status from Accepted to Running.

```
Application ID:
application_1462842354064_0001
Application State: ACCEPTED
Starting......
Application State: RUNNING
```

Once the job starts, you see the YARN job tracking URL with the Drill-on-YARN web UI URL.

```
Application Master URL: http://
<YARN_Job_Tracking_URL>:8048/
```


last-updated: Apr 11, 2022

Once the application starts, the Drill-on-YARN writes an “appid” file into the Drill installation directory:

```
ls /opt/mapr/drill/drill-<version>
…
drillbits1.appid
```

The file name is the same as the Drill cluster ID. The file contains the ID of the Drill-on-YARN application. The other commands use this ID. You can run only one Drill application at a time. If you attempt to start a second from the same client machine on which you started the first, the client command complains that the appid file already exists. If you attempt to start the cluster from a different node, the second application detects a conflict and shuts down again.

Example

```
$DRILL_HOME/bin/drill-on-yarn.sh start

Launching Drill-on-YARN...
Application ID:
application_1462842354064_0001
Application State: ACCEPTED
Starting......
Application State: RUNNING
Tracking
URL: http://10.250.50.31:8088/proxy/
application_1462842354064_0001/
Application Master URL: http://
10.250.50.31:8048/
```

status

The status command retrieves basic information about the Drill cluster and provides a status summary.

The first several lines of the status summary provide information about the state of YARN, which includes the application ID, the application state, and YARN’s tracking URL for the application.

```
Application ID:
application_1462842354064_0001
Application State: RUNNING
Host: yosemite/10.250.50.31
Tracking URL:
```

Following the state of YARN information is the host on which the Drill application is running, the queue on which the application was placed, and the user who submitted the application. The start time tells you when YARN started the application.

```
http://10.250.50.31:8088/proxy/
application_1462842354064_0001/
Queue: default
User: drilluser
Start Time: 2016-05-09 16:56:40
```

The next few lines are specific to Drill, including the name of the application (which you configured in the drill-on-yarn.conf configuration file), the Drill

Application Master URL, the number of Drillbits you requested to run, and the number actually running.

```
Application Name: Drill-on-YARN
AM State: LIVE
Target Drillbit Count: 1
Live Drillbit Count: 1
```

Finally, the last line provides the URL for the Drill-on-YARN web UI.

```
For more information, visit: http://
10.250.50.31:8048/
```

Example

```
$DRILL_HOME/bin/drill-on-yarn.sh
status

Application ID:
application_1462842354064_0001
Application State: RUNNING
Host: yosemite/10.250.50.31
Tracking
URL: http://10.250.50.31:8088/proxy/
application_1462842354064_0001/
Queue: default
User: drilluser
Start Time: 2016-05-09 16:56:40
Application Name: Drill-on-YARN
AM State: LIVE
Target Drillbit Count: 1
Live Drillbit Count: 1
For more information, visit: http://
10.250.50.31:8048/
```

stop

The stop command stops the Drill cluster. This command is forceful and kills any in-flight queries. The output tracks the shutdown and displays the final YARN application status.

Example

```
$DRILL_HOME/bin/drill-on-yarn.sh stop

Stopping Application ID:
application_1462842354064_0001
Stopping...
Stopped.
Final status: SUCCEEDED
```

resize

The resize command changes the number of nodes in the cluster. You can use this command to add or remove nodes in the Drill cluster as it runs. You can specify the change either by giving the number of nodes you want to run, or by using the + or - to specify the change in node count.

Drill adds nodes only if additional nodes are available from YARN. If you request to stop more nodes than are running, Drill stops all of the running nodes.

last-updated: Apr 11, 2022

Example

```
$DRILL_HOME/bin/drill-on-yarn.sh
resize 10
$DRILL_HOME/bin/drill-on-yarn.sh
resize +2
$DRILL_HOME/bin/drill-on-yarn.sh
resize -3
```

clean

The clean command removes the cached Drill archive from the designated DFS directory. If you run Drill-on-YARN for a temporary cluster, Drill leaves the Drill software archive in your designated DFS directory. Specifically, the first start uploads the Drill archive to DFS. Stop leaves the archive in DFS. Subsequent start commands reuse the cached archive if it is the same size as the version on the local disk. Clean removes the cached file, forcing Drill to upload a fresh copy if you again restart the Drill cluster.

Example

```
$DRILL_HOME/bin/drill-on-yarn.sh clean
```

Application Master Web UI

Drill, running as a YARN application, provides the Drill-on-YARN Application Master (AM) process to manage the Drill cluster. The Drill AM provides a web UI where you can monitor cluster status and perform simple operations, such as increasing or decreasing cluster size, or stopping the cluster.

When you launch Drill using the Drill-on-YARN command line tool, the tool signals YARN to launch the AM, which in turn launches the Drillbits in the cluster. When Drill starts, you can access the web UI using the URL provided at startup.

The following sections describe the information that the Application Master web UI provides:

Main

The main page provides the following information about the Drill cluster:

Drill Cluster Status

The Drill cluster status show the state of the Drill cluster, which is one of the following:

- LIVE: This is the normal state and shows that the Drill cluster is running.
- ENDING: The cluster is in the process of shutting down

There is no “ENDED.” state. When the cluster shuts down, the web UI is no longer available.

Target Drillbit Count

The target Drillbit count is the number of Drillbits to run in the cluster. The actual number may be less if Drillbits have not yet started, or if YARN cannot allocate enough containers.

Live Drillbit Count

The live Drillbit count is the number of Drillbits that are ready for use. These have successfully started, have registered with ZooKeeper, and are ready for use. You can see the detail of all Drillbits (including those in the process of starting or stopping) using the Drillbits page. Each Drillbit must run on a separate node, so

last-updated: Apr 11, 2022

	this is also the number of nodes in the cluster running Drill.
Total Drillbit Memory and Virtual Cores	The total number of YARN resources currently allocated to running Drillbits.
YARN Node Count, Memory, and Virtual Cores	Reports general information about YARN itself including the number of nodes, the total cluster memory, and total number of virtual cores.
Groups	Lists the cluster groups defined in the configuration file (only one is currently supported), along with the target and actual number of Drillbits in that group.

Configuration

The configuration page shows the complete set of configuration values used for the current run. The values come from the configurations you set and the Drill-provided defaults. Use this page to diagnose configuration-related issues. Names are shown in fully-expanded form. That is the name “drill.yarn.http.port” refers to the parameter defined, as follows, in your configuration file:

```
drill.yarn:
 http: {
   port: 8048
 }
}
```

Drillbits

The Drillbits page provides the following information about each of the Drillbits:

ID	A sequential number assigned to each new Drillbit. Numbers may not start with 1 if you have previously shut down some Drillbits.
Group	The cluster group that started the Drillbit. Cluster groups configured in drill-on-yarn.conf.
Host	The host name or IP address on which the Drillbit runs. If the Drillbit is in a normal operating state, this field is also a hyperlink to the Web UI for the Drillbit.
State	The operating state of the Drillbit. The normal state is “Running.” The Drillbit passes through a number of states as YARN allocates a container and launches a process, as the AM waits for the Drillbit to become registered in ZooKeeper, and so on. Similarly, the Drillbit passes through a different set of states during shutdown. Use this value to diagnose problems. If the Drillbit is in a live state, this field shows an “[X]” link that you can use to kill this particular Drillbit. Use this if the Drillbit has startup problems or seems unresponsive. During the shut-down process, the kill link disappears and is replaced with a “Cancelled” note.
ZK State	The ZooKeeper handshake state. Normal state is “START_ACK”, meaning that the Drillbit has registered with ZooKeeper. This state is useful when diagnosing problems.
Container ID	The YARN-assigned container ID for the Drillbit task. The ID is a link that takes you to the YARN Node Manager UI for the Drillbit task.

last-updated: Apr 11, 2022

Memory and Virtual Cores (vcores)	The amount of resources actually allocated to the Drillbit by YARN.
Start Time	The date and time (in your local time-zone, displayed in ISO format) when the Drillbit launch started. This page also displays unmanaged Drillbits, if present. An unmanage Drillbit is one that is running, has registered with ZooKeeper, but was not started by the Application Master. Likely, the Drillbit was launched using the Drillbit.sh script directly. Use the host name to locate the machine running the Drillbit if you want to convert the Drillbit to run under YARN.

Manage

The Manage page provides options to resize or stop the Drill cluster. You can resize the cluster by adding or removing Drillbits or setting the cluster to a specific size.

Drill is a long-running application. Typically, Drill runs indefinitely, and you would only shut down the Drill cluster to perform an upgrade of the Drill software or to change configuration options. When you terminate the Drill cluster, any in-progress queries fail. Therefore, best practice is to perform the shutdown with users so that Drill is not processing any queries at the time of the shut-down.

When removing or shutting-down the cluster, you receive a confirmation page asking if you really do want to stop Drillbit processes. Click **Confirm** to continue.

History

The History page lists all failed, killed, and restarted Drillbits. You can detect failures and diagnose problems using the information on this page. Use the YARN container ID listed on this page to locate the log files for the Drillbit.

Enabling Application Master Web UI Security
By default, the Application Master Web UI is not secure and open to everyone. You can configure user authentication or implement a simple, predefined user name and password to secure the UI. Modify the drill-on-yarn.conf configuration file to enable Drill-on-YARN security.

The following sections describe how to enable security for the Application Master web UI.

User Authentication

You must enable user authentication in Drill if you want Drill-on-YARN to use this feature for security purposes. When user authentication is enabled, the user name and password must match that of the user that started the Drill-on-YARN application.

To secure the Application Master web UI by way of user authentication, modify `drill-on-yarn.conf` to include the following section with the `auth-type` set to `drill`:

```
drill.yarn.http: {
   auth-type: "drill"
}
```

Simple Security

Define a username and password in `drill-on-yarn.conf` and then restart the Drill-on-YARN Application Master to implement simple security for the Application Master web UI.

last-updated: Apr 11, 2022

Modify the `drill-on-yarn.conf` configuration file to include the following section, replacing the user-name and password settings with yours, and then restar the Drill-on-YARN Application Master:

```
drill.yarn.http: {
   auth-type: "simple"
   user-name: "tsmith"
   password: "secret"
}
```

When you visit the web UI, a login page prompts you for the username and password that you configured. These are the only valid credentials.

Drill-on-YARN Limitations

Drill-on-YARN has the following limitations:

Hanging requests	Drill-on-YARN and YARN “hang” if YARN cannot fulfill a container request. YARN provides no information about why a request “hangs.”
/tmp directory	The default MapR YARN settings cause Drillbits to become unmanaged within a short amount of time due to a /tmp directory issue. See the Exclude the YARN Container Directory from tmpwatch section in Step 3: Configure YARN to Run Drill for information on how to resolve the issue.
Container size	The default MapRYARN settings do not allow a default Drill cluster to run due to the default YARN container size. See the Increase Maximum Container Size section in Step 3: Configure YARN to Run Drill for information on how to resolve the issue.
Drill disk usage	You can specify Drill disk usage to YARN, but Drill will use all disks regardless of the setting. There is no effective way to manage a Drill cluster that: • resizes based on load. • is rack-aware in its smaller state. YARN chooses arbitrary nodes perhaps resulting in large network reads. (MD-1028, MD-1089)
Node Labels	Although the Apache YARN documentation states that you can associate node labels with YARN container requests, some people have noticed that the feature does not work in practice. While Drill-on-YARN configuration has settings to associate Drillbit container requests with node labels, doing so is not supported. To use node labels, associate node labels with YARN queues as described in the YARN configuration step in the Migrate Drill to Run Under YARN documenation.

Configuring Drill

Lists the MapR-specific configuration for Drill.

Drill is highly configurable. This document focuses on MapR-related configurations and refers to the open source Apache Drill documentation for generic information. Key things to configure are:

Drill memory	Determine the amount of heap and direct memory allocated to a Drillbit for query processing in a Drill cluster. See Configuring Drill Memory on page 3239.

last-updated: Apr 11, 2022

Parquet block size	Change the Parquet block size to match the MapR filesystem chunk size. See Configuring the Parquet Block Size on page 3241.
Resources for a shared Drillbit	Configure queues and parallelization for supporting multiple users sharing a Drillbit. Support separate Drillbits running on different nodes in the cluster. See Configuring Resources for a Shared Drillbit.
Multitenancy	Configure a multitenant cluster to account for resources required for Drill. See Configuring a Multitenant Cluster on page 3246.
User Impersonation	Configure impersonation to allow a service to act on behalf of a client while performing the action requested by the client. See User Impersonation on page 3280.
User authentication and encryption	Configure user authentication when you want the identity of a user, before permitting the user access to a process running on a system. See MapR Security (Tickets) on page 3292 .
SSL/TLS for Encryption	Enable and configure SSL/TLS for encryption when you need to use Plain authentication. See SSL/TLS for Encryption on page 3312.
Drill impersonation with Hive authorization	Configure Drill impersonation to work with Hive impersonation to authorize access to metadata in the Hive metastore repository and data in the Hive warehouse. See User Impersonation with Hive on page 3286.
Volumes to use for spooling	Use the drill.exec.spill.directories option to set MapReduce volumes or local volumes for spooling to improve performance and stripe data across as many disks as possible.
Persistent configuration storage	See Persistent Configuration Storage and Configuring the ZooKeeper PStore Location on page 3247.
Access rights	Configure access rights if you have 777 file-level permissions to a table, and a query returns no results. See Configuring Access Rights.

Drill typically runs along side other workloads, including the following:

- MapReduce
- Yarn
- Hive and Pig
- Spark

You need to plan and configure these resources for use with Drill and other workloads:

- Memory
- CPU
- Disk

Configuring Access Rights

If the security in your organization limits access to MapR Database tables, you might experience a problem querying the tables. If you have 777 file-level permissions to a table, yet a query returns no results, you might need to add your user name to the maprcli *ACL*.

last-updated: Apr 11, 2022

Adding a Drill Node to a Cluster

To add a new node to a MapR cluster that provides the Drillbit service, add the node to the cluster first, and then install Drill on the new node. If you install Drill first, and then add the node to the cluster, the cluster cannot detect ZooKeeper information and the cluster is misnamed. These problems require some work to resolve. You need to edit the drill-override.conf file in the `mapr/drill` directory and modify the name of the cluster. Avoid this extra work, and install Drill only after adding the node to the cluster by performing steps in the following order:

1. Follow instructions for adding a node to a cluster.
2. Reconfigure the cluster as described in the same instructions.
3. Verify that the new node is up and running.
4. Stop Warden, as shown:

   ```
   $ service mapr-warden stop
   ```

5. Stop ZooKeeper service, as shown:

   ```
   $ service mapr-zookeeper stop
   ```

6. Configure the repository to add the ecosystem repository.
7. Install Drill on the new node.
8. Reconfigure the node.

   ```
   $ /opt/mapr/server/configure.sh -R
   ```

9. Start ZooKeeper if the node is a ZooKeeper node.
10. Start warden to make configuration changes effective.

Verify that the Drillbit service is running on the node. It might take a minute or so for the Drillbit to start after starting warden.

Configuring Drill Memory

A system administrator can modify the amount of system memory that Warden allocates to the Drill service on each node in the `warden.drill-bits.conf` file. Drill users, with file permissions, can modify the amount of heap and direct memory allocated to the Drill service on each node in the `drill-env.sh` file.

Note: The cumulative memory allocation in `drill-env.sh` cannot exceed the memory allocation in `warden.drill-bits.conf`.

After modifying `drill-env.sh`, restart Drill:

```
$ maprcli node services -name drill-bits -action restart -nodes
<space-separated-list-of-drill-hostnames>
```

After modifying `warden.drill-bits.conf`, run the configuration script, configure.sh, to update the node configuration and then restart Drill:

```
/opt/mapr/server/configure.sh -R
$ maprcli node services -name drill-bits -action restart -nodes
<space-separated-list-of-drill-hostnames>
```


last-updated: Apr 11, 2022

The following sections describe the `warden.drill-bits.conf` and `drill-env.sh` files in detail.

Drill Memory Allocation in a Warden-Managed Cluster

If you install and run Drill under the Warden service, Warden manages the amount of system memory that Drill can use. By default, Warden allocates 20% of the system memory on a node to the Drill service. For example, if a node has 50GB of memory, Warden allocates 10GB (20% of 50GB) to the Drill service.

A system administrator can define the amount of memory that Warden allocates to Drill by changing the value of the `DRILLBIT_MAX_PROC_MEM` variable in `/opt/mapr/drill/drill-<version>/conf/warden.drill-bits.conf`.

When starting, Drill verifies that the amount of memory configured in `drill-env.sh` does not exceed the limit set by the `service.env=DRILLBIT_MAX_PROC_MEM` variable in `warden.drill-bits.conf`. If the settings in `drill-env.sh` exceed the setting in Warden, the system prints a message stating the issue; Warden does not start the Drill service on the node.

The `warden.drill-bits.conf` file contains the following settings:

Note: Drill automatically configures the `service.heapsize` parameters. Do not modify them.

```
#Default Drill Mem Distrib: 20% of System memory
service.env=DRILLBIT_MAX_PROC_MEM=20%
//Specifies the maximum amount of memory that Warden will allocate to the
Drill service on the node. You can set this value as a percentage of system
memory or as an absolute value in GB. Memory configured in drill-env.sh
cannot exceed this memory setting.

service.heapsize.min=5120
//Minimum heap size. Do not change this value. The value is auto-populated
and represents the minimum memory that the Drillbit process will take.

service.heapsize.max=13312
//Maximum heap size. Do not change this value. The value is auto-populated
and represents the maximum memory that the Drillbit process will take.

#Warden will allocate 20% of memory for Drill
service.heapsize.percent=20
//Do not change this value. Total heap size available based on the value
set for the DRILLBIT_MAX_PROC_MEM variable. If the variable is defined in
absolute values, it is represented as a percent of the system memory.
```

Drill Memory Allocation in drill-env.sh

You can configure the amount of heap and direct memory allocated to Drill on each node in the`/opt/mapr/conf/conf.d/drill-env.sh` file. If you do not manually configure the heap and direct memory, Drill calculates these values based on the amount of system memory that Warden allocates to Drill and auto-populates the settings for the variables.

The cumulative amount of memory allocated to Drill in `drill-env.sh` cannot exceed the amount of memory that Warden allocates to Drill, which is set by the `DRILLBIT_MAX_PROC_MEM` variable in `warden.drill-bits.conf`.

Note: The values in `drill-env.sh`, such as 13G, are examples and do not indicate the default memory limits for Drill. By default, Warden allocates 20% of the system memory on a node to Drill.

The `drill-env.sh` file contains the following memory variables that you can uncomment and modify:

```
#export DRILLBIT_MAX_PROC_MEM=${DRILLBIT_MAX_PROC_MEM:-"13G"}
//Specifies the maximum amount of system memory that the Drill service
```


last-updated: Apr 11, 2022

```
can use on a node. Must be equal to or less than the value set for
DRILLBIT_MAX_PROC_MEM in warden.drill-bits.conf. You can set this value as
a percentage of system memory or as an absolute value in GB. If you define
this variable, without defining the heap and direct memory variables, Drill
automatically calculates the heap and direct memory values.

#export DRILL_HEAP=${DRILL_HEAP:-"4G"}
//Maximum theoretical heap limit for the JVM per node.

#export DRILL_MAX_DIRECT_MEMORY=${DRILL_MAX_DIRECT_MEMORY:-"8G"}
//Java direct memory limit per node.

#export DRILLBIT_CODE_CACHE_SIZE=${DRILLBIT_CODE_CACHE_SIZE:-"1G"}
//The memory limit for the compiled code generated by the JVM JIT compiler.
Do not modify. The value for this parameter is auto-computed based on the
heap size and cannot exceed 1GB.
```

Note: If performance is an issue, add -Dbounds=false, as shown:

```
export DRILL_JAVA_OPTS="$DRILL_JAVA_OPTS -Dbounds=false"
```

Configuring the Parquet Block Size

The default value for the `store.parquet.block-size` parameter is 268435456 (256 MB), the same size as MapR filesystem chunk sizes. In previous versions of Drill, the default value was 536870912 (512 MB).

If you change the MapR filesystem chunk size, change the Parquet block size to match using the ALTER SYSTEM or SET commands, as shown:

```
ALTER SYSTEM SET `store.parquet.block-size` = <value>;
[ALTER SESSION] SET `store.parquet.block-size` = <value>
```

Alternatively, you can override the default setting in the `<DRILL_HOME>/conf/drill-override.conf` file, as shown:

```
drill.exec: {
  ...
  options.store.parquet.block-size = 268435456
}
```

For information about setting the MapR filesystem chunk size, see Setting Chunk Size.

Configuring Multiple Drill Clusters and Designating One Cluster as an OJAI Distributed Query Service

As of MapR 6.0 and Drill 1.11, you can run operational queries through the OJAI Distributed Query Service, as well as analytical queries through Drill. If you want to run operational and analytical workloads in your MapR cluster, you must configure multiple Drill clusters within the MapR cluster and then configure a Drill cluster as the OJAI Distributed Query Service. Restricting each workload to its own cluster improves query performance.

Note: Installing Drill and the OJAI Distributed Query Service together through the MapR Installer is not currently supported. Only one of these services running in the cluster is supported unless you manually install and configure multiple Drill clusters, as instructed here.

Data Distribution

If you install both Drill and the OJAI Distributed Query Service through the MapR Installer, both workloads get processed across the entire MapR cluster. When both services run together in the cluster, the system

replicates data across the entire cluster, causing remote reads and impairing performance, which can lead to missed SLAs and memory issues.

Memory Allocation

The amount of memory allocated to Drill and the OJAI Distributed Query Service differ. By default, when you install Drill, 13 GB of memory is allocated to the Drillbit service running on a node:

- 8 GB direct
- 4 GB heap
- 1 GB core cache

The OJAI Distributed Query Service less memory than Drill. By default, the OJAI Distributed Query Service is allocated ~ 5 GB of memory:

- 1 GB direct
- 3 GB heap
- 512 MB core cache

If you use the MapR Installer and select both Drill and the OJAI Distributed Query Service, memory is configured for Drill. If you only run operational queries, which do not use as much memory as analytical queries, you unnecessarily lose an additional 8 GB of memory.

How to Run Drill and the OJAI Distributed Query Service Together in a MapR Cluster

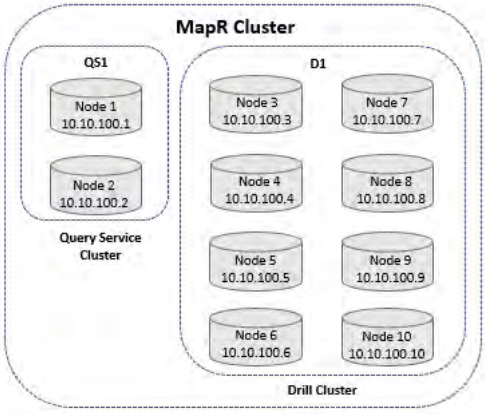
You can manually install Drill on several nodes and divide the nodes into multiple topologies (Drill clusters). For each of the topologies, create and mount a volume. Then, create directories within each volume to store your data. Configure these directories as workspaces in the Drill dsf storage plugin. Finally, configure a Drill cluster to run as an OJAI Distributed Query Service.

The following topics provide instructions for each of the required steps:

Step 1: Plan the Clusters
Decide which nodes in the MapR cluster you want to run Drill and which nodes you want to run the OJAI Distributed Query Service.

The nodes you select to run Drill can form one or more Drill clusters, while the nodes you select to run the OJAI Distributed Query Service can form another Drill cluster. You can configure multiple Drill clusters.

For example, if you have a ten node MapR cluster, you can configure one Drill cluster to run analytical queries and one OJAI Distributed Query Service cluster to run OJAI operational queries, as shown:



last-updated: Apr 11, 2022

Track the nodes that you want to group into a cluster, as this information is needed to configure node topology and volumes. Also note the memory requirements of each service. Only non-overlapping Drill clusters are supported. You cannot install more than one Drillbit on a MapR server node.

Step 2: Manually Install Drill on All Nodes
Manually install Drill on all nodes, including the nodes designated to run the OJAI Distributed Query Service.

For Drill installation instructions, see Installing Drill. You can install Drill to run under Warden or YARN, as described in the following following topics:

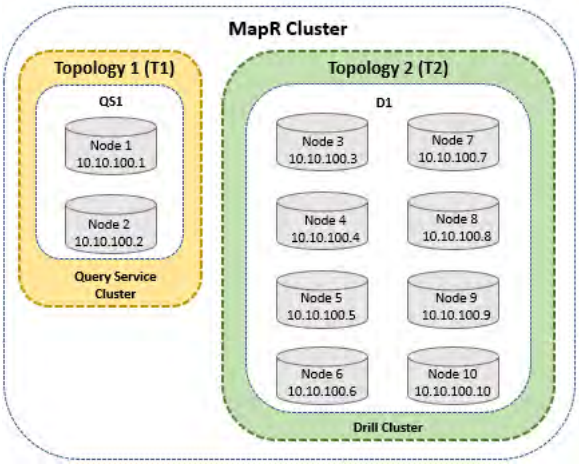
- Install Drill to Run Under Warden
- Install Drill to Run Under YARN

Step 3: Define Node Topologies
Node topologies restrict data to a designated set of nodes.

When you define a topology, data is only replicated on the nodes within the topology. Node topologies improve query performance because data is localized to the nodes specified in the topology instead of being distributed across the entireMapR cluster.

You can create node topologies for the Drill clusters in the UI or from the CLI. See Changing Topology for One or More Nodes for instructions.

When you create a topology, you define the nodes that form a Drill cluster, as shown in the following image:



The image shows two node topologies, T1 and T2. T1 is the Drill cluster to be configured as the OJAI Distributed Query Service. T2 is the Drill cluster that will remain a Drill service cluster.

Step 4: Create Volumes
Volumes organize data and manage cluster performance. Create and mount a volume to each of the topologies (Drill clusters) you created.

For example, you can create a volume named "operational" and mount it to the T1 topology and then create a volume named "analytical" and mount it to the T2 topology.

See Administering Volumes for volume information and Creating a Volume on page 864 for instructions.

Once you create the volumes, you have a place where you can create directories and store data. For example, you can create and store operational data in the `/operational/data/here` directory and analytical data in the `/analytical/data/here` directory.

Use these directories to configure the workspaces in the Drill dfs storage plugin configuration.

last-updated: Apr 11, 2022

Step 5: Configure Multiple Drill Clusters
Update the /opt/mapr/drill/drill-<version>/conf/drill-override.conf file on each Drill node that is part of a cluster with the cluster ID and a ZooKeeper entry to define the Drill cluster. Each Drill cluster should have a unique cluster ID and ZooKeeper entry to separate the clusters.

Note: Each Drill node in a cluster must have the same configuration.

The Drillbit process reads the configuration file and communicates with ZooKeeper to see if the cluster it belongs to exists. If the cluster exists, ZooKeeper says to join the cluster. If the cluster does not exist, the Drillbit initiates a new Drill cluster based on the cluster ID.

The following table provides an example of unique cluster IDs and ZooKeeper entries based on the topologies in the image shown in step 3, Define Node Topologies:

Cluster	Nodes	Cluster ID	ZooKeeper Entry
QS1	10.10.100.1 10.10.100.2	cluster-id: "drillbits"	zk.root: "drill"
D1	10.10.100.3 10.10.100.4 10.10.100.5 10.10.100.6 10.10.100.7 10.10.100.8 10.10.100.9 10.10.100.10	cluster-id: "drillbits2"	zk.root: "drill2"

For QS1, drill-override.conf must include the following configuration:

```
drill.exec: {
  zk.root: "drill",
  cluster-id: "drillbits",
  zk.connect: "<zk-node-ip-address>:5181",
}
```

For D1, drill-override.conf must include the following configuration:

```
drill.exec: {
  zk.root: "drill2",
  cluster-id: "drillbits2",
  zk.connect: "<zk-node-ip-address>:5181",
}
```

Note: If you installed Drill to run under YARN, follow the steps in Defining Multiple Drill Clusters Under YARN for each Drill node. Drill running under YARN requires some additional steps, such changing ZooKeeper ports.

Step 6: Configure Workspaces
You must configure a workspace on one Drill node in each Drill cluster that points to the volume directory where data is stored. When you create a workspace, you must include the volume mount point.

last-updated: Apr 11, 2022

For example, if you created a volume with the mount point `/operational` for the OJAI Distributed Query Service cluster and stored your data in `/data/here/` within that volume, you would configure the workspace, as shown:

```
 {
     "type": "file",
     "enabled": true,
     "connection": "file:///",
     "workspaces": {
       "root": {
         "location": "/",
         "writable": false,
         "defaultInputFormat": null
       },
       "tmp": {
         "location": "/tmp",
         "writable": true,
         "defaultInputFormat": null
       },
       "operational": {
           "location": "/operational/data/here",
           "writable": true,
           "defaultInputFormat": null
          }
     }
```

Likewise, if you also created a volume with a mount point `/analytical` for the other Drill cluster that runs analytical queries and stored your data in `/data/here/` within that volume, you would configure the workspace, as shown:

```
{
     "type": "file",
     "enabled": true,
     "connection": "file:///",
     "workspaces": {
       "root": {
         "location": "/",
         "writable": false,
         "defaultInputFormat": null
       },
       "tmp": {
         "location": "/tmp",
         "writable": true,
         "defaultInputFormat": null
       },
       "analytical": {
         "location": "/analytical/data/here",
         "writable": true,
         "defaultInputFormat": null
        }
     }
```

You can define a workspace in the Drill dfs storage plugin configuration on the Storage page in the Drill Web UI at `https://<drill-node-ip-address>:8047`. You only need to configure the workspace on one Drill node in each Drill cluster.

See Plugin Configuration Basics and Workspaces for more information.

Step 7: Register a Drill Cluster as an OJAI Distributed Query Service
You can select any of the configured Drill clusters to act as the OJAI Distributed Query Service provider for operational queries, by running the `queryservice setconfig` command.

last-updated: Apr 11, 2022

When you register the Drill cluster as the OJAI Distributed Query Service, adjust the memory setting on each node. The default Drill memory setting of 13 GB is unnecessarily high for the OJAI Distributed Query Service, which only requires ~ 5 GB. You must restart the Drillbits after you update the memory settings.

Registering a Drill Cluster as the OJAI Distributed Query Service

To register a Drill Cluster as an OJAI Distributed Query Service, run the following command:

```
maprcli cluster queryservice setconfig -enabled true -clusterid
<name_of_cluster> -storageplugin dfs -znode <zk_setting>
```

For example, drillbits2 and drill2 are the cluster ID and ZooKeeper settings used in examples in previous steps. For these configurations, the command is:

```
maprcli cluster queryservice setconfig -enabled true -clusterid
drillbits2 -storageplugin dfs -znode drill2
```

See queryservice setconfig for more information about the command.

Configuring Memory for the OJAI Distributed Query Service

Modify the memory settings on each node in the OJAI Distributed Query Service cluster and then restart Drill. See Configuring Drill Memory on page 3239 for instructions.

Configuring a Multitenant Cluster

Drill operations are memory and CPU-intensive. Currently, Drill resources are managed outside of any cluster management service, such as the MapR Warden service. In a multi-tenant or any other type of cluster, YARN-enabled or not, you configure memory and memory usage limits for Drill by modifying `drill-env.sh` as described in the section, "Configuring Drill Memory" in Apache Drill documentation.

Configure a multitenant cluster to account for resources required for Drill. For example, on a MapR cluster, ensure warden accounts for resources required for Drill. Configuring `drill-env.sh` allocates resources for Drill to use during query execution, while configuring the following properties in `warden-drill-bits.conf` prevents warden from committing the resources to other processes.

```
service.heapsize.min=<some value in MB>
service.heapsize.max=<some value in MB>
service.heapsize.percent=<a whole number>
```

Set the `service.heapsize` properties in `warden.drill-bits.conf` regardless of whether you changed defaults in `drill-env.sh` or not.

"Configuring Drill in a YARN-enabled MapR Cluster" shows an example of setting the `service.heapsize` properties. The `service.heapsize.percent` is the percentage of memory for the service bounded by minimum and maximum values. Typically, users change `service.heapsize.percent` because using a percentage setting increases or decreases resources according to different node configurations. For more information about the `service.heapsize` properties, see the section, "warden.<servicename>.conf."

You need to statically partition the cluster to designate which partition handles which workload. To configure resources for Drill in a MapR cluster, modify one or more of the files created by the installation process in `/opt/mapr/conf/conf.d:`

```
warden.drill-bits.conf
warden.nodemanager.conf
warden.resourcemanager.conf
```

Configure Drill memory by modifying `warden.drill-bits.conf` in YARN and non-YARN clusters. Configure other resources by modifying `warden.nodemanager.conf` and `warden.resourcemanager.conf` in a YARN-enabled cluster.

Configuring Drill in a YARN-enabled MapR Cluster

To add Drill to a YARN-enabled cluster, change memory resources to suit your application. For example, you have 120G of available memory that you allocate to following workloads in a Yarn-enabled cluster:

File system = 20G Yarn = 20G OS = 8G

If Yarn does most of the work, give Drill 20G, for example, and give Yarn 60G. If you expect a heavy query load, give Drill 60G and Yarn 20G.

YARN consists of two main services:

- ResourceManager: There is at least one instance in a cluster, more if you configure high availability.
- NodeManager: There is one instance per node.

The `warden.resourcemanager.conf` and `warden.nodemanager.conf` files set ResourceManager and NodeManager memory to the following defaults:

```
service.heapsize.min=64
service.heapsize.max=325
service.heapsize.percent=2
```

Change these settings for NodeManager and ResourceManager to reconfigure the total memory required for YARN services to run. If you want to place an upper limit on memory, set the YARN_NODEMANAGER_HEAPSIZE or YARN_RESOURCEMANAGER_HEAPSIZE environment variable in

```
/opt/mapr/hadoop/hadoop-2.5.1/etc/hadoop/yarn-env.sh
```

You do not set the `-Xmx` option, allowing memory to grow as needed.

MapReduce Version 2 and other Resources

You configure memory for each service by setting three values in `warden.conf`.

```
service.command.<servicename>.heapsize.percent
service.command.<servicename>.heapsize.max
service.command.<servicename>.heapsize.min
```

Configure memory for other services in the same manner. For more information about managing memory in a MapR cluster, see the following sections:

- Memory Allocation for Nodes
- Cluster Resource Allocation

How to Manage Drill CPU Resources

Currently, you do not manage CPU resources within Drill. Use Linux cgroups to manage the CPU resources.

Configuring the ZooKeeper PStore Location

By default, the ZooKeeper PStore offloads query profile data to `maprfs:///apps/drill/profiles`. You can override the default location in the `drill-override.conf` file.

last-updated: Apr 11, 2022

When query profile data is stored on a distributed system, like the MapR filesystem, you can see a global query list (view of query profiles coordinated by all Drill nodes in one Web UI).

To change the ZooKeeper PStore location, update the `drill.exec` block in `/opt/mapr/drill/drill-<version>/conf/drill-override.conf with the following` configuration, as shown:

```
drill.exec: {
    cluster-id: "my_cluster_com-drillbits",
    zk.connect: "<zkhostname>:5181",
    sys.store.provider.zk.blobroot: "maprfs:///new/storage/location/"
   }
```

Note: By default, the MapR filesystem replicates the data three times. If you are concerned about storage consumption, you can create a new volume specifically for query profile data, and set the replication value to 1 for that volume. After you create the volume, update `sys.store.provider.zk.blobroot` to point to the volume. See Creating a Volume on page 864 for additional information.

After you modify `drill-override.conf`, restart Drill:

```
maprcli node services -name drill-bits -action restart -nodes
<drill-hostnames-separated-by-a-space>
```

Configuring cgroups to Control CPU Usage
Starting in Drill 1.13, you can configure a Linux cgroup (control group) to enforce CPU limits on the Drillbit service running on a node. Linux cgroups enable you to limit system resources to defined user groups or processes. You can use the cgconfig service to configure a Drill cgroup to control CPU usage and then set the CPU limits for the Drill cgroup on each Drill node in the `/etc/cgconfig.conf` file.

Note: Cgroups V2 is recommended.

Before You Begin

Each Drill node must have the `libcgroup` package installed to configure CPU limits for a Drill cgroup. The `libcgroup` package installs the cgconfig service required to configure and manage the Drill cgroup.

Install the `libcgroup` package using the `yum install` command, as shown:

```
yum install libcgroup
```

Enable Drill to Directly Manage CPU Resources

Starting in Drill 1.14, Drill can directly manage CPU resources through the start-up script, `drill-env.sh`, which means that you no longer have to manually add the PID (Drill process ID) to the `cgroup.procs` file each time a Drillbit restarts. This step occurs automatically upon restart. The start-up script checks for the specified cgroup, such as drillcpu, and then applies the cgroup to the launched Drillbit JVM. The Drillbit CPU resource usage is then managed under the cgroup, drillcpu.

For Drill to directly manage CPU resources, you must enable (uncomment) the following variables in the `drill-env.sh` script:

Variable	Description
export DRILLBIT_CGROUP=${DRILLBIT_CGROUP:-"drillcpu"}	Sets the cgroup to which the Drillbit belongs when running as a daemon using drillbit.sh start. Drill uses the cgroup for CPU enforcement only.

export SYS_CGROUP_DIR=${SYS_CGROUP_DIR:-"/sys/fs/cgroup"}	Drill assumes the default cgroup mount location set by systemd (the system and service manager for Linux operating systems). If your cgroup mount location is in a different location, change the setting to match your location.
export DRILL_PID_DIR=${DRILL_PID_DIR:-$DRILL_HOME}	The location of the Drillbit PID file when Drill is running as a daemon using drillbit.sh start. By default, this location is set to $DRILL_HOME.

Important: If you have Drill 1.13 running on the node, or you have Drill 1.14 running on the node and you do not want to enable Drill to directly manage the CPU resources through `drill-env.sh`, you must manually update the `/cgroup/cpu/drillcpu/cgroup.procs` file with the PID (Drill process ID), as shown, each time a Drillbit restarts to enforce the CPU limit for the Drillbit service:

```
echo 25809 > /cgroup/cpu/drillcpu/cgroup.procs
```

Set the CPU Limit for the Drillbit Service

You can set the CPU limit as a soft or hard limit, or both. You set the limits with parameters in the `/etc/cgconfig.conf` file. The hard limit takes precedence over the soft limit. When Drill hits the hard limit, in-progress queries may not complete. Review the following sections that describe the soft and hard limit parameters and then configure CPU limits.

Soft Limit Parameter

You set the soft limit with the `cpu.shares` parameter. This parameter takes an integer value, which specifies a relative share of CPU time available to the tasks in a cgroup. For example, if there are two tasks and `cpu.shares` is set to 100, each task receives half of the CPU time. The value must be 2 or greater. When you set a soft limit, Drill can exceed the CPU allocated if extra CPU is available for use on the system. Drill can continue to use CPU until there is contention with other processes over the CPU or Drill hits the hard limit.

Hard Limit Parameters

You set the hard limit on the amount of CPU time that the Drill process can use through the `cpu.cfs_period_us` and `cpu.cfs_quota_us` parameters.

- `cpu.cfs_period_us`

  Specifies a period in microseconds (represented by `us` for µs) to indicate how often a cgroup's access to CPU resources should be reallocated. For example, if you want tasks in a cgroup to have access to a single CPU for 0.2 seconds in a 1 second window, set `cpu.cfs_quota_us` to 200000 and `cpu.cfs_period_us` to 1000000. The upper limit of the `cpu.cfs_quota_us` parameter is 1 second and the lower limit is 1000 microseconds.

- `cpu.cfs_quota_us`

  Specifies the total amount of runtime in microseconds (represented by `us` for µs), for which all tasks in the Drill cgroup can run during one period (as defined by `cpu.cfs_period_us`). When tasks in the Drill cgroup use up all the time specified by the quota, the tasks are throttled for the remainder of the time specified by the period and they cannot run until the next period. For example, if tasks in the Drill cgroup can access a single CPU for 0.2 seconds out of every 1 second, set `cpu.cfs_quota_us` to 200000 and `cpu.cfs_period_us` to 1000000. Setting the `cpu.cfs_quota_us` value to -1 indicates that the group does not have any restrictions on CPU. This is the default value for every cgroup, except for the root cgroup.

Configuring CPU Limits

Complete the following steps to set a hard and/or soft CPU limit for the Drill process running on the node:

1. Start the cgconfig service:

```
service cgconfig start
```

2. Add a cgroup for Drill in the `/etc/cgconfig.conf` file:

```
group drillcpu {
                     cpu {
                            cpu.shares = 320;
                            cpu.cfs_quota_us = 400000;
                            cpu.cfs_period_us = 100000;
                            }
                     }
```

In the configuration example above, the `cpu.shares` parameter sets the soft limit. The other two parameters, `cpu.cfs_quota_us` and `cpu.cfs_period_us`, set the hard limit. If you prefer to set only one type of limit, remove the parameters that do not apply. When setting a soft limit, allocate a specific number of CPU shares to the Drill cgroup in the configuration. Calculate the CPU shares as:

```
1024 (CPU allocated to Drill/Total available CPU)
```

In the example, CPU shares is calculated as:

```
1024 (10/32) = 320
```

When setting a hard limit, add limits to the `cpu.cfs_quota_us` and `cpu.cfs_period_us` parameters. In the example, the Drill process can fully utilize 4 CPU.

Tip:

The hard limit parameter settings persist after each cgroup service restart. Alternatively, you can set the parameters at the session level using the following commands:

```
echo 400000 > /cgroup/cpu/drillcpu/cpu.cfs_quota_us
       echo 100000 > /cgroup/cpu/drillcpu/cpu.cfs_period_us
```

3. (Optional) If you want the cgconfig service to automatically restart upon system reboots, run the following command:

```
chkconfig cgconfig on
```

Related information

https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/resource_management_guide/sec-cpu

Working with Drill

For general information about working with Drill, refer to the following key topics:

Connecting Drill to Data Sources
Choose and configure storage plugins to enable Drill to connect to a data source.

Drill serves as a query layer that connects to data sources through storage plugins. A storage plugin is a software module for connecting Drill to data sources. A storage plugin typically optimizes execution of Drill queries, provides the location of the data, and configures the workspace and file formats for reading data.

What you can do with Storage Plugins

Several storage plugins are installed with Drill that you can configure to suit your environment. Through a storage plugin, Drill connects to a data source, such as a database, a file on a local or distributed filesystem, or a Hive metastore. See the Drill Storage and Format Plugin Support Matrix.

You can modify the default configuration of a storage plugin and give the new configuration a unique name. This document refers to Y as a different storage plugin, although it is actually just a reconfiguration of original interface.

On the Storage tab of the Web Console, you can view and reconfigure a storage plugin if you have permission. You can access each node running a Drillbit by starting the Drill Web Console. The way you start the Drill Web Console depends on your security setup.

When you install Drill using the `mapr-drill` package, storage plugin configurations are available for the following data sources:

- MapR File System
- MapR Database

  Note: To access MapR Database tables, use the dfs storage plugin with the maprdb format plugin.
- Hive
- Kafka

Connecting Drill to HBase

As of the MapR 6.0 and Drill 1.11, HBase is no longer supported, therefore the communication path between Drill and HBase is also not supported. If you have an hbase storage plugin configured in Drill, you should disable it.

Default Storage Plugin Configurations

The Drill documentation describes the attributes and definitions that you can configure for storage plugins, except for the MapR Database format. See MapR Database Format Plugin for Drill.

The Drill Web Console includes some default storage plugin configurations. The following table lists the default configurations and their descriptions:

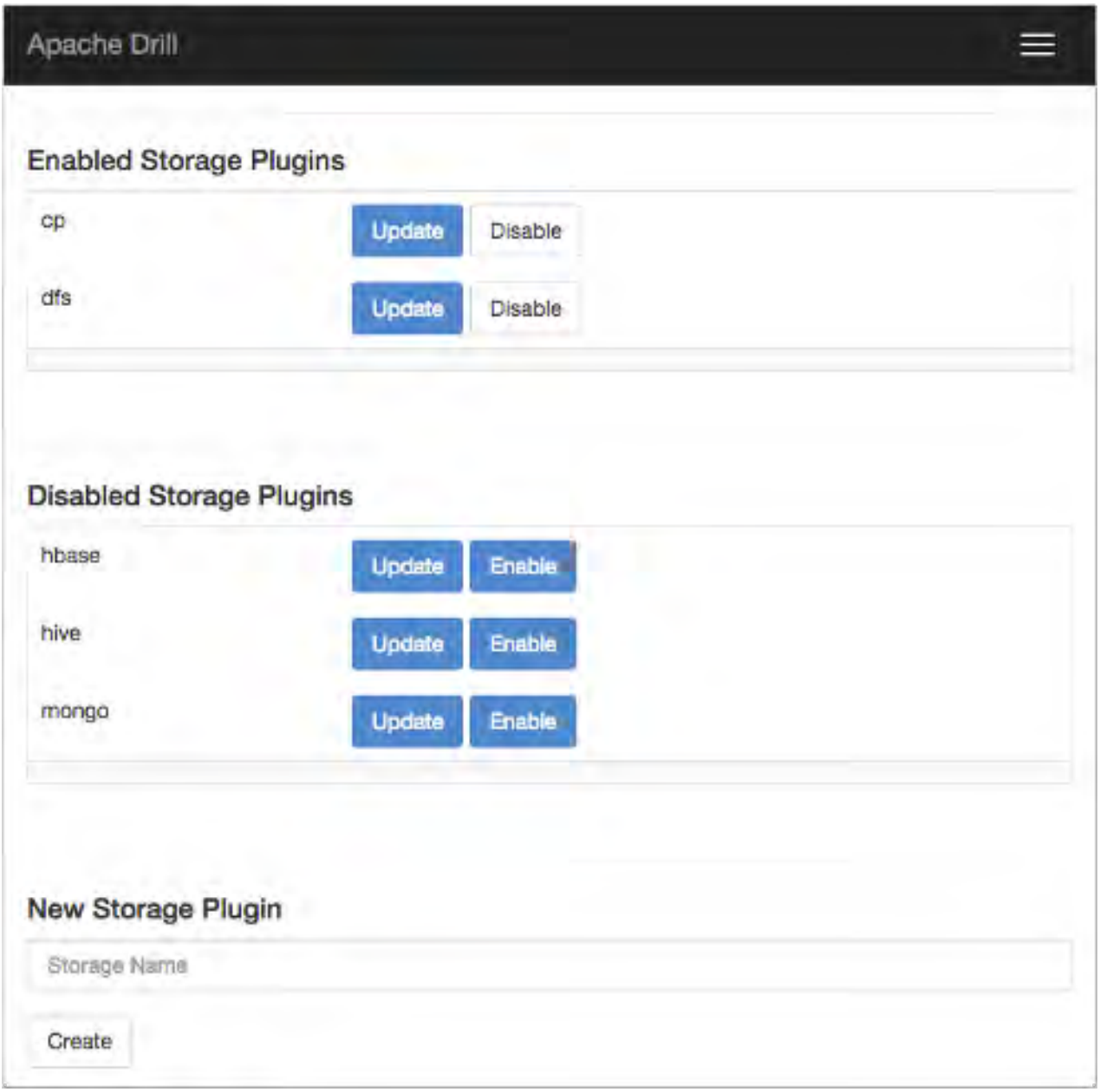
last-updated: Apr 11, 2022

Instance	Description
cp	Points to a JAR file in the Drill classpath that contains the Transaction Processing Performance Council (TPC) benchmark schema TPC-H that you can query.
dfs	Points to MapR File System by default. Drill automatically configures this instance when you install Drill in a the MapR cluster. Includes a maprdb format plugin for MapR Database.
hive	Integrates Drill with the Hive metadata abstraction of files, MapR Database, and libraries to read data and operate on SerDes and UDFs.

When you add or update a storage plugin configuration on one Drill node in a Drill cluster, Drill broadcasts the information to all of the other Drill nodes. All nodes have identical storage plugin configurations. You do not need to restart any Drillbits when you add or update a storage plugin configuration.h

Configuring Storage Plugin Instances

You can add, remove, or update Drill storage plugin configurations using the Web Console. The following image shows the default storage plugin configurations in the Drill Web UI:



If you click **Update** next to `dfs`, the following default configuration appears :

```
{
  "type": "file",
  "enabled": true,
  "connection": "maprfs:///",
  "workspaces": {
    "root": {
      "location": "/",
      "writable": false,
      "defaultInputFormat": null
    },
    "tmp": {
      "location": "/tmp",
      "writable": true,
      "defaultInputFormat": null
    }
  },
```


last-updated: Apr 11, 2022

```
  "formats": {
    "psv": {
      "type": "text",
      "extensions": [
        "tbl"
      ],
      "delimiter": "|"
    },
    "csv": {
      "type": "text",
      "extensions": [
        "csv"
      ],
      "delimiter": ","
    },
    "tsv": {
      "type": "text",
      "extensions": [
        "tsv"
      ],
      "delimiter": "\t"
    },
    "parquet": {
      "type": "parquet"
    },
    "json": {
      "type": "json"
    },
    "maprdb": {
      "type": "maprdb"
    }
  }
}
```

The `dfs` configuration includes the storage plugin type, connection information, default workspaces, and file formats that the data source supports. You can add and remove workspaces and file formats.

Changing the Connection Attribute

You can also change the connection if you want the configuration to point to a different cluster.

By default, Drill connects to the cluster that the Drill node belongs to. You do not need to modify the connection unless you want to connect Drill to a different cluster. To connect to a different cluster, edit the connection to include the name of the cluster that you want to connect to.

Example:

```
"connection": "maprfs://<cluster_name>/"
```

Drill Storage and Format Plugin Support Matrix

You can deploy Drill without Hadoop in a standalone configuration on a single node, however multi-node standalone cluster deployments of Drill are not supported. Note that Drill itself does not require Hadoop.

The following table lists the supported and unsupported data sources and formats in Drill:

Data Source	Storage Plugin Type	Formats	Supported
MapR File System	dfs	Text (CSV, TSV, PSV)	Yes
		Parquet	Yes
		JSON	Yes

		Avro	No
MapR Database	dfs	Binary	Yes
		JSON	Yes
HBase	hbase	Binary	No (as of Drill 1.11 and MapR 6.0)
Hive	hive	Text (CSV, TSV, PSV)	Yes
		Parquet	Yes
		JSON	Yes
		Avro	Yes
		Other Hive built-in SerDes	Yes (Not recommended due to the memory overhead and performance implications.)
S3	s3	Supports the same formats as the dfs storage plugin.	Yes
MongoDB	mongodb	N/A	No
RDBMS	jdbc	N/A	No
Kudu	kudu	N/A	No
Kafka	kafka	JSON	No  **Note:** The kafka storage plugin on the MapR Streams is in the Alpha testing phase and not officially supported. See Configuring the Kafka Storage Plugin for more information.
OpenTSDB	openTSDB	N/A	 **Note:** The openTSDB storage plugin is not officially supported. See OpenTSDB Storage Plugin for more information.

 **Note:** As of the MapR 6.0 and Drill 1.11, HBase is no longer supported, therefore the communication path between Drill and HBase is also not supported. If you have an hbase storage plugin configured in Drill, you should disable it.

maprdb Format Plugin for Drill
Drill supports access to MapR Database JSON and binary tables through the maprdb format plugin.

When you install Drill, the maprdb format is automatically defined within the default dfs storage plugin configuration to make MapR Database a consumable data source for Drill. You can access the dfs storage plugin configuration from the Storage page in the Drill Web UI.

When you install Drill, you will see some options specific to the maprdb format plugin that you can change or configure. You can modify these options in the following places:

- dfs storage plugin configuration

last-updated: Apr 11, 2022

- drill-override.conf file
- SET command

For additional information about storage plugins, see Plugin Configuration Basics.

Modifying the maprdb Format Settings within the dfs Storage Plugin Configuration

You can add or modify certain maprdb format plugin settings within the dfs storage plugin configuration on the Storage page in the Drill Web UI.

The following table lists the maprdb format options that you can set within the dfs storage plugin configuration:

Option	Description	Value
allTextMode	When enabled, Drill reads all values as type varchar. Useful when the underlying data set has type values of mixed scalar types, such as integers, floating point, varchars, date, time, and timestamp. Disabled by default.	true|false
disableCountOptimization	When enabled, this option disables optimization for queries with the COUNT (*) aggregate function. Disabled by default.	true|false
enablePushdown	When enabled, Drill pushes down filters to MapR Database. Disabling this option is not recommended unless you intend to use it for troubleshooting purposes. Enabled by default.	true|false
ignoreSchemaChange	When enabled, Drill ignores schema changes. Disabled by default.	true|false
nonExistentColumnsProjection	When enabled, Drill can distinguish between null and non-existent fields. Disabled by default.	true|false
readNumbersAsDouble	When enabled, Drill reads all numeric values as type double. Useful when the underlying data set has type values of mixed numeric types, such as integers and floating point. Disabled by default.	true|false

The following example configuration shows you how to include the options in the maprdb format configuration within the dfs storage plugin configuration:

```
{
  "type": "file",
  "enabled": true,
  "connection": "maprfs:///",
  "config": null,
  "workspaces": {
    "root": {
      "location": "/",
      "writable": false,
      "defaultInputFormat": "maprdb",
      "allowAccessOutsideWorkspace": false
    },
  "formats": {
    "maprdb": {
```


last-updated: Apr 11, 2022

```
      "type": "maprdb",
      "allTextMode": true,
      "disableCountOptimization": true,
      "enablePushdown": false
    },
    "parquet": {
      "type": "parquet"
    },
    "json": {
      "type": "json",
      "extensions": [
        "json"
      ]
    }
  }
}
```

See Plugin Configuration Basics and File System Storage Plugin for more information.

Overriding Default maprdb Format Plugin Settings in drill-override.conf

You can override the default maprdb format plugin settings that control the level of parallelism in Drill and the media type in the `drill-override.conf` file.

To override the default maprdb format plugin settings, add the options to the `format-maprdb.json` configuration in the `/opt/mapr/drill/drill-<version>/conf/drill-override.conf` file, as shown:

```
format-maprdb: {
                json: {
                    scanSizeMB: 512,
                    restrictedScanSizeMB: 4096
                    mediaType: HDD
                    }
                }
```

The following sections describe how to modify the options in the configuration shown above:

Configuring the Level of Parallelism in Drill

The size of data chunks and number of minor fragments affect the level of parallelism in Drill. When querying JSON tables, Drill creates minor fragments that scan the chunks of data that MapR Database passes to Drill. Minor fragments are logical units of work that determine the level of parallelism in Drill. The level of parallelism increases with the number of minor fragments. See Drill Query Execution for more information about how Drill executes a query.

Modifying the Size of Data Chunks

The `format-maprdb.json.scanSizeMB` option changes the size of data chunks that MapR Database passes to Drill when querying JSON tables. Drill creates approximately one minor fragment per data chunk when querying JSON tables. For example, if a table has 4 GB of data and the chunk size is set to 128 MB, Drill creates approximately 32 minor fragments to scan the data chunks.

The default setting for data chunks is 128 MB, however you can override this default in the drill-override.conf file. The value of the `format-maprdb.json.scanSizeMB` option can range from 32 MB to 8192 MB (8 GB). Adjust the setting based on the size of your tables. Use a higher setting for larger tables and a lower setting for smaller tables. The right setting can reduce latency and increase throughput.

Modifying the Number of Minor Fragments Created

The `format-maprdb.json.restrictedScanSizeMB` option determines the number of minor fragments that Drill creates to scan the data and do the join-back to a JSON table when executing a non-covering index plan.

last-updated: Apr 11, 2022

The default setting for this option is 4096 MB, however you can override this setting in the drill-override.conf file. The value of this option can range from 32 MB to 8192 MB (8 GB). Adjust the setting based on the size of your tables, keeping in mind that due to the random I/O nature of the join-back, a smaller setting (increased parallelism) may not necessarily increase throughput.

Note: The `planner.slice_target` option in Drill determines the number of minor fragments that can run in parallel. See Modifying Query Planning Options for more information.

Configuring the Media Type

Drill is optimized for SSDs, however MapR Database and Drill can run on HDDs. If you run Drill and MapR Database on HDDs, use the mediaType option in the `drill-override.conf` file to override the default setting. The `mediaType` option accepts HDD or SSD (default) as the value. Specify SSD or HDD in upper case as the value in the `drill-override.conf` file.

Drill Options for the maprdb Format Plugin

You can enable certain Drill options for the maprdb format plugin from the Options page in the Drill Web UI or from the command line using the SET and ALTER SYSTEM commands.

To enable the options from the Drill Web UI, go to `http(s)://<drill-hostname-or-ip-address>:8047`, and select **Options** in the menu bar. Alternatively, enable options from the command line using the SET or ALTER SYSTEM commands, as shown:

```
SET `store.hive.maprdb_json.optimize_scan_with_native_reader` = true;
```

You can enable the following Drill options for the maprdb format plugin:

store.hive.maprdb_json.optimize_scan_with_native_reader	Starting in Drill 1.14 (EEP 6.0), enable the `store.hive.maprdb_json.optimize_scan_with_native_reader` option if you want Drill to use the native Drill reader to read Hive MapR-DB JSON tables. When you enable the native Drill reader, Drill typically performs faster reads of data and applies filter pushdown optimizations.
store.hive.maprdb_json.read_timestamp_with_timezone_offset	Starting in Drill 1.16, you can enable Drill to read timestamp values with a timezone offset when the hive plugin is used and the Drill native MaprDB JSON reader is enabled through the `store.hive.maprdb_json.optimize_scan_with_native_reader` option.

last-updated: Apr 11, 2022

Important: Internally, Drill stores timestamp values in UTC format, for example 2018-01-01T20:12:12.123Z. When you enable the timezone offset option, select on a table returns different timestamp values. If you filter on timestamp values when this option is enabled, you must include the new timestamp value in the filter condition. For example, look at the timestamp values when the store.hive.maprdb_json.read_timestamp_with_timezone_offset option is disabled (set to 'false'):

```
select * from dfs.`/tmp/timestamp`;
----------------------------------------------------------
_id        datestring              datetimestamp
----------------------------------------------------------
1    2018-01-01 12:12:12.123    2018-01-01 20:12:12.123
2    9999-12-31 23:59:59.999    10000-01-01 07:59:59.999
----------------------------------------------------------
```

When the option is enabled (set to 'true'), you can see the difference in the timestamp values returned:

```
select * from dfs.`/tmp/timestamp`;
----------------------------------------------------------
_id     datestring            datetimestamp
----------------------------------------------------------
1    2018-01-01 12:12:12.123    2018-01-01 12:12:12.123
2    9999-12-31 23:59:59.999    9999-12-31 23:59:59.999
----------------------------------------------------------
```

When the option is enabled, queries that filter on timestamp values must include the new timestamp value in the filter condition, as shown:

```
select * from dfs.`/tmp/timestamp` where datetimestamp=timestamp
'2018-01-01 12:12:12.123';
----------------------------------------------------------
_id    datestring             datetimestamp
----------------------------------------------------------
1    2018-01-01 12:12:12.123   2018-01-01 12:12:12.123
----------------------------------------------------------
```

Notice that the WHERE clause uses the `2018-01-01 12:12:12.123` format versus the `2018-01-01 20:12:12.123` format.

MapR Database Tables

The `maprdb` format plugin enables you to query binary and JSON tables like you would query files in a file system because MapR Database and the MapR File System share the same namespace.

Binary tables differ from JSON tables in that they store a multi-dimensional map in which both keys and values are a sequence of bytes. JSON tables store OJAI documents. JSON tables support rich data types, including complex and repeated types, that enable database servers to evaluate filter conditions for optimized query execution. Binary tables can pose performance limitations because the table columns do not contain the necessary type information.

You can query tables stored in any directory in MapR Database using the same syntax that you use to select from files in the MapR File System. Instead of including the path of a file in a query, you include the table path. The user running the query must have read permission to access the table.

last-updated: Apr 11, 2022

Note: MapR Database is a case sensitive data source. To ensure that your queries return results, use the case that corresponds to the column names in the JSON tables and views that you query. For example, if you query the “age” column in a JSON table, the query must reference the column as “age” and cannot reference the column as “AGE” or the query will not return results. If you have a dataset where a column name has mixed cases, such as “age” and “AGE,” cleanse the data set so column names consist of one case and then reimport the data into the table to ensure a complete result set when you query the column.

The following sections describe the types of tables that MapR Database supports, provide examples of Drill queries on each type of table, and show you how to load data into a JSON table from JSON files.

JSON Tables

A JSON table is a collection of JSON documents stored in an optimized format in MapR Database. JSON tables support complex schema, like JSON files including nested and repeated types, but with additional support for more data types.

JSON tables leverage the OJAI API to natively support Drill data types making it possible for MapR Database to recognize, store, and interpret each of the Drill data types. This alleviates the need to encode data when an application writes to tables or use conversion functions when running queries against tables. For example, if a number or date is stored in a JSON table, you do not need to use the CAST or CONVERT functions for the query to return the actual values.

MapR Database's native support for Drill data types enables Drill to push down filters and projections to MapR Database which optimizes performance.

Querying a JSON Table

Querying JSON tables is simpler than querying binary tables because you do not have to include conversion functions in the queries to change the byte sequences into specific data types, and you do not have to include column families.

The following query examples show query results on a JSON table named “students” in MapR Database. Note that Drill returns human readable values without having to include the CAST or CONVERT functions in the queries.

Example 1

```
SELECT * FROM dfs.`/user/root/json/students`;
```

```
+-----------+-------------+--------+--------+-------------------+----------+
|    _id    |    date     |  name  | state  |      street       | zipcode  |
+-----------+-------------+--------+--------+-------------------+----------+
| student1  | 2016-01-15  | Alice  | CA     | 123 Ballmer Av    | 12345    |
| student2  | 2016-03-08  | Bob    | CA     | 1 Infinite Loop   | 12345    |
| student3  | 2015-12-22  | Frank  | CA     | 435 Walker Ct     | 12345    |
| student4  | 2015-09-15  | Mary   | CA     | 56 Southern Pkwy  | 12345    |
+-----------+-------------+--------+--------+-------------------+----------+
4 rows selected (0.233 seconds)
```

Example 2

```
SELECT _id, `date`, name, state, street, zipcode FROM dfs.`/user/root/json/
students`;
```

```
+-----------+-------------+--------+--------+-------------------+----------+
|    _id    |    date     |  name  | state  |      street       | zipcode  |
+-----------+-------------+--------+--------+-------------------+----------+
| student1  | 2016-01-15  | Alice  | CA     | 123 Ballmer Av    | 12345.0  |
| student2  | 2016-03-08  | Bob    | CA     | 1 Infinite Loop   | 12345.0  |
| student3  | 2015-12-22  | Frank  | CA     | 435 Walker Ct     | 12345.0  |
| student4  | 2015-09-15  | Mary   | CA     | 56 Southern Pkwy  | 12345.0  |
```


last-updated: Apr 11, 2022

```
+----------+-------------+--------+--------+------------------+----------+
4 rows selected (1.033 seconds)
```

Loading JSON Documents into a MapR Database Table with dbshell Commands

You can use the INSERT command in the `mapr dbshell` to load JSON documents into a MapR Database table.

The INSERT command is useful when inserting a small number of JSON documents into a JSON table.

Note: Alternatively, you can put JSON documents in a flat text file and use the `mapr importJSON` command to import the JSON documents into a table. The `mapr importJSON` command is useful when you need to insert many documents into a table. Refer to MapR Database JSON ImportJSON on page 5322 for instruction.

Note: The examples in this document use the student data in theQuerying HBase Tutorial to recreate the binary “students” table as a JSON table in MapR Database.

To load JSON documents into a MapR Database table through the `mapr dbshell`, complete the following steps:

1. Run the following command to start the `mapr dbshell`:

   ```
   mapr dbshell
   ```

2. Run the following command to create a table:

   ```
   create <table-name>

   Example:
    create students
   ```

   Note: By default, the table is stored in the default directory. The default directory is the current directory on the MapR Filesystem, which is set to the user’s home directory when the `mapr dbshell` starts. Include a file path, as shown, if you do not want the table stored in the default directory:

   ```
   create /file/path/table-name
   ```

3. Load the JSON documents into the table using the INSERT command, as shown:

   ```
   insert <table-name> --value '{JSON-document}'

   Example:
   insert students --value '{"_id":"student1", "name":"Alice",
   "street":"123 Ballmer Av", "zipcode":12345, "state":"CA"}'
   insert students --value '{"_id":"student2", "name":"Bob", "street":"1
   Infinite Loop", "zipcode":12345, "state":"CA"}'
   insert students --value '{"_id":"student3", "name":"Frank",
   "street":"435 Walker Ct", "zipcode":12345, "state":"CA"}'
   insert students --value '{"_id":"student4", "name":"Mary", "street":"56
   Southern Pkwy", "zipcode":12345, "state":"CA"}'
   ```

4. Run the following command to verify that the table was created:

```
find <table-name>

Example:
find students
```

5. Run the following command to close the `mapr dbshell`:

```
exit
```

6. If you need to start or restart Drill, run the following command:

```
maprcli node services -name drill-bits -action start|restart -nodes
<space-separated-list-of-drill-hostnames>
```

7. Run the following command to start the Drill shell (SQLLine):

```
sqlline
```

You can query the MapR Database JSON table from the Drill shell. If you did not include a file path when you created the table, the table was created in the current directory on the MapR Filesystem, which is set to the user's home directory. For example, the following query specifies the default directory if the root user created the table without indicating a file path:

```
SELECT * FROM dfs.`/user/root/table-name`;

Example:
SELECT * FROM dfs.`/user/root/students`;
```

If the user created the table in a specific directory, the query must include the directory in which the table was created, as shown:

```
SELECT * FROM dfs.`/file/path/table-name`;
```

Binary Tables

Binary tables store data in a flat table structure where the table consists of columns and column values. Every field in a binary table is stored as a sequence of bytes.

Binary tables do not store data type information. You manage the encoding for binary tables when storing data and then convert the sequence of bytes into a specific data type using the CAST or CONVERT functions when you run queries against the tables.

For example, if a string is stored in binary format, such as a UTF-8 encoded string, you must use the CAST function for the query to return a string type. If an integer is stored in binary format, such as 4-byte little endian encoding, you must use the CONVERT function for the query to return the integer value instead of the 4-byte sequence.

Querying a Binary Table

The following examples, from the Querying HBase Tutorial, display query results when the binary table "/user/root/binary/students" with two column families, "account" and "address," is queried without using a conversion function to convert the binary table data into specific data types.

Example 1

```
SELECT * FROM `/user/root/binary/students` students;
```

```
+--------------+-------------------------------------------------+-----------------------
|   row_key    |                    account                      |
address                                  |
+--------------+-------------------------------------------------+-----------------------

| [B@78dfa1ed  | {"date":"MjAxNi0wMS0xNQ==","name":"QWxpY2U="}  |
{"state":"Q0E=","street":"MTIzIEJhbGxtZXIgQXY=","zipcode":"MTIzNDU="}      |
| [B@22000c9a  | {"date":"MjAxNi0wMy0wOA==","name":"Qm9i"}       |
{"state":"Q0E=","street":"MSBJbmZpbml0ZSBMb29w","zipcode":"MTIzNDU="}      |
| [B@313b63e6  | {"date":"MjAxNS0xMi0yMg==","name":"RnJhbms="}  |
{"state":"Q0E=","street":"NDM1IFdhbGtlciBDdA==","zipcode":"MTIzNDU="}      |
| [B@321baa4a  | {"date":"MjAxNS0wOS0xNQ==","name":"TWFyeQ=="}  |
{"state":"Q0E=","street":"NTYgU291dGhlcm4gUGt3eQ==","zipcode":"MTIzNDU="}  |
+--------------+-------------------------------------------------+-----------------------

4 rows selected (0.612 seconds)
```

In example 2, using the CONVERT_FROM and CAST functions in a query on the same table converts the binary table data to typed data.

Example 2

```
SELECT CONVERT_FROM(row_key, 'UTF8') AS studentid,
CONVERT_FROM(students.account.name, 'UTF8') AS name,
CONVERT_FROM(students.address.state, 'UTF8') AS state,
CONVERT_FROM(students.address.street, 'UTF8') AS street,
CONVERT_FROM(students.address.zipcode, 'UTF8') AS zipcode,
CAST(students.account.`date` as date) AS `date` FROM dfs.`/user/root/binary/
students` students;
```

```
+------------+--------+--------+-------------------+----------+-------------+
| studentid  |  name  | state  |      street       | zipcode  |    date
|
+------------+--------+--------+-------------------+----------+-------------+

| student1   | Alice  | CA     | 123 Ballmer Av    | 12345    | 2016-01-15
|
| student2   | Bob    | CA     | 1 Infinite Loop   | 12345    | 2016-03-08
|
| student3   | Frank  | CA     | 435 Walker Ct     | 12345    | 2015-12-22
|
| student4   | Mary   | CA     | 56 Southern Pkwy  | 12345    | 2015-09-15
|
+------------+--------+--------+-------------------+----------+-------------+

4 rows selected (0.702 seconds)
```

Configuring the Hive Storage Plugin

You can connect Drill to a Hive data source through the hive storage plugin configuration in the Drill Web UI. Once configured, you can use Drill to query data stored in Hive.

Drill can work with only one version of Hive in a given cluster. To access Hive tables using custom SerDes or InputFormat/OutputFormat, all nodes running Drill must have the SerDes or InputFormat/OutputFormat JAR files in the following location: `<drill_installation_directory>/jars/3rdparty`.

To query across multiple versions of Hive, install each version of Hive on a separate Drill cluster. You must define separate storage plugins, each corresponding to the specific Hive version of the metastore.

Note: In EEP 6.0, Drill requires Hive version 2.3.3-mapr or later to successfully query Hive data sources.

Note: You can update the hive storage plugin configuration through the configuration script, configure.sh. If the hive storage plugin is disabled, and the configuration in the Drill Web UI displays "null," you must rerun configure.sh with the `-hiveMetastoreHost` argument. See configure.sh for details.

Configuring a Hive Remote Metastore

A remote Hive metastore configuration runs as a separate service outside of Hive. The metastore service communicates with the Hive database over JDBC. Point Drill to the Hive metastore service address, and provide the connection parameters in the hive storage plugin configuration to configure a connection to Drill. The hive storage plugin (located on the Storage tab in the Drill Web UI) has the following default configuration when you install Drill:

```
{
 "type": "hive",
 "enabled": true,
 "configProps": {
  "hive.metastore.uris": "",
  "javax.jdo.option.ConnectionURL": "jdbc:derby:;databaseName=../
sample-data/drill_hive_db;create=true",
  "hive.metastore.warehouse.dir": "/tmp/drill_hive_wh",
  "fs.default.name": "file:///",
  "hive.metastore.sasl.enabled": "false",
  "datanucleus.schema.autoCreateAll": "true"
 }
}
```

Complete the following steps to modify the default hive storage plugin configuration for your MapR File System environment:

1. Verify that Hive is running.
2. Issue the following command to start the Hive metastore service on the system specified in the `hive.metastore.uris`: `hive --service metastore`
3. Start the Drill Web UI.
4. Select the **Storage** tab. If Web UI security is enabled, you must have administrator privileges to perform this step.
5. In the list of disabled storage plugins in the Drill Web UI, click **Update** next to hive.
6. Update the following hive storage plugin parameters to match the system environment:
   - `"hive.metstore.uris"`
   - `"jdbc:<database>://<host:port>/<metastore database>"`
   - Change the default location of files to suit your environment. For example, change `"fs.default.name": "file:///"` to the MapR File System location: `maprfs:///`
   - To run Drill and Hive in a secure MapR cluster, change the `"hive.metastore.sasl.enabled"` parameter to `"true"`.
   - Change the `"datanucleus.schema.autoCreateAll"` property setting for your system environment. When enabled, `"datanucleus.schema.autoCreateAll"` initializes the Hive metastore schema.

last-updated: Apr 11, 2022

- In a production environment, remove the `"datanucleus.schema.autoCreateAll"` property from the hive storage plugin configuration; the property is not required because the preferred schema information is already created for the Hive metastore service.
- In a test environment with an embedded Hive metastore, you can disable (set to false) this property after the first query on the Hive data source that you submit from Drill. Alternatively, use the Hive schema tool to initialize or upgrade the Hive metastore schema. Using the Hive schema tool is recommended for queries on transactional tables. Run the `schematool` command as an initialization step, as shown:

```
/opt/mapr/hive/hive-<version>/bin/schematool -dbType
<databaseType> -initSchema
```

7. Click **Enable** in the Web UI to enable the Hive storage plugin configuration.

Configuring the Kafka Storage Plugin
The Kafka storage plugin is not officially supported for Drill; however, if you choose to configure Kafka as a data source in Drill, you must update the `<drill_home>/jars/3rdParty` directory such that it contains the required JAR files and then restart Drill before you configure the `kafka` storage plugin in the Drill Web UI.

Verify that the nodes in your cluster meet the requirements and then complete the steps listed.

Requirements

The Kafka storage plugin requires:

- A MapR 6.1 cluster.
- Drill 1.14 installed on nodes.
- The MapR Kafka client package (kafka-1.1.1) installed on at least one node. The Kafka client installation provides the following kafka JAR files that you copy into the `<drill_home>/jars/3rdParty` directory (step 4):
  - kafka_2.11-1.1.1-mapr-1808.jar
  - kafka-clients-1.1.1-mapr-1808.jar

Steps

Complete the following steps to query Kafka Streams from Drill:

Note: Do not perform step 2 if you installed Drill using the MapR RPM or Debian packages. Step 2 is only required if you installed Drill using a TAR file.

1. Remove the specified JAR files from the `<drill_home>/jars/3rdParty` directory based on the Drill installation method:
   - If you installed Drill using MapR RPM or Debian packages, only remove JAR files that start with kafka, such as `kafka-clients-<version>.jar` and `kafka_<version>.jar`, from the `<drill_home>/jars/3rdParty` directory.
   - If you installed Drill using a TAR file, remove all the JAR files that start with `mapr` and `kafka`, such as `maprdb-<version>-mapr.jar, maprfs-<version>-mapr.jar`, `kafka_<version>-mapr.jar`, and `kafka-clients-<version>.jar`, from the `<drill_home>/jars/3rdParty` directory.

2. (Only perform this step if you installed Drill using a TAR file.) Copy the following JAR files from the `/opt/mapr/lib directory` into `<drill_home>/jars/3rdParty` directory:
   - `maprdb-6.1.0-mapr.jar`
   - `maprdb-6.1.0-mapr-tests.jar`
   - `maprfs-6.1.0-mapr.jar`
   - `maprfs-6.1.0-mapr-tests.jar`
   - `mapr-hbase-6.1.0-mapr.jar`
   - `mapr-hbase-6.1.0-mapr-tests.jar`
   - `mapr-streams-6.1.0-mapr.jar`
3. Copy the `mapr-streams-6.1.0-mapr.jar` file from the `/opt/mapr/lib` directory into the `<drill_home>/jars/3rdParty` directory.
4. Copy the following kafka JAR files from the `/opt/mapr/kafka/kafka-1.1.1/libs` directory into the `<drill_home>/jars/3rdParty` directory:
   - `kafka_2.11-1.1.1-mapr-1808.jar`
   - `kafka-clients-1.1.1-mapr-1808.jar`
5. Issue the following command to restart Drill:

   ```
   $ maprcli node services -name drill-bits -action restart -nodes <node
   hostnames separated by a space>
   ```

6. Log in to the Drill Web UI, and configure the kafka storage plugin. See Kafka Storage Plugin for instructions.

   Note: When configuring the kafka storage plugin, you must also include the following parameter in the storage plugin configuration:

   ```
   "streams.consumer.default.stream": "<path-to-stream>"
   ```

Usage Example

This example shows a Drill query on a MaR Streams data set made accessible to Drill through the kafka storage plugin.

For this example, tables that contain Yelp stream topics reside in a directory named /YelpStream. The kakfa storage plugin is configured with the "streams.consumer.default.stream" parameter pointing to the /YelpStream directory, as shown:

```
"streams.consumer.default.stream": "/YelpStream"
```

The USE command tells Drill to access data from only the kafka data source:

```
use kafka;
+-----+-----------------------------------+
| ok | summary |
+-----+-----------------------------------+
```

```
| true | Default schema changed to [kafka] |
+-----+-----------------------------------+
```

The SHOW TABLES command lists the tables in the /YelpStream directory configured for the kafka data source:

```
show tables;
+-------------+--------------------------+
| TABLE_SCHEMA | TABLE_NAME |
+-------------+--------------------------+
| kafka | /YelpStream:UserTable |
| kafka | /YelpStream:ReviewTable |
| kafka | /YelpStream:BusinessTable |
+-------------+--------------------------+
```

The query selects all the data from the BusinessTable in the `/YelpStream` directory, limiting the results to one row data:

```
select * from `/YelpStream:BusinessTable` limit 1;
+---+---------+----------+---------+----+-----------+-----+--------+---------+----+-

| _id | attributes | business_id | categories | city | full_address | hours
| latitude | longitude | name | neighborhoods | open | review_count | stars
| state | type | kafkaTopic | kafkaPartitionId | kafkaMsgOffset |
kafkaMsgTimestamp | kafkaMsgKey |
+---+---------+----------+---------+----+-----------+-----+--------+---------+----+-

| --1emggGHgoG6ipd_RMb-g | {"Accepts Credit Cards":"true","Parking":
{"garage":"false","lot":"true","street":"false","valet":"false","validated":
"false"},"Price Range":"1","Ambience":{},"Good For":{},"Music":{}}
| --1emggGHgoG6ipd_RMb-g | ["Food","Convenience Stores"] | Las Vegas | 3280
S Decatur Blvd
Westside
Las Vegas, NV 89102 | {"Friday":{},"Monday":{},"Saturday":{},"Sunday":
{},"Thursday":{},"Tuesday":{},"Wednesday":{}} | 36.1305306 | -115.2072382 |
Sinclair | ["Wes
```

Start the Drill Web UI

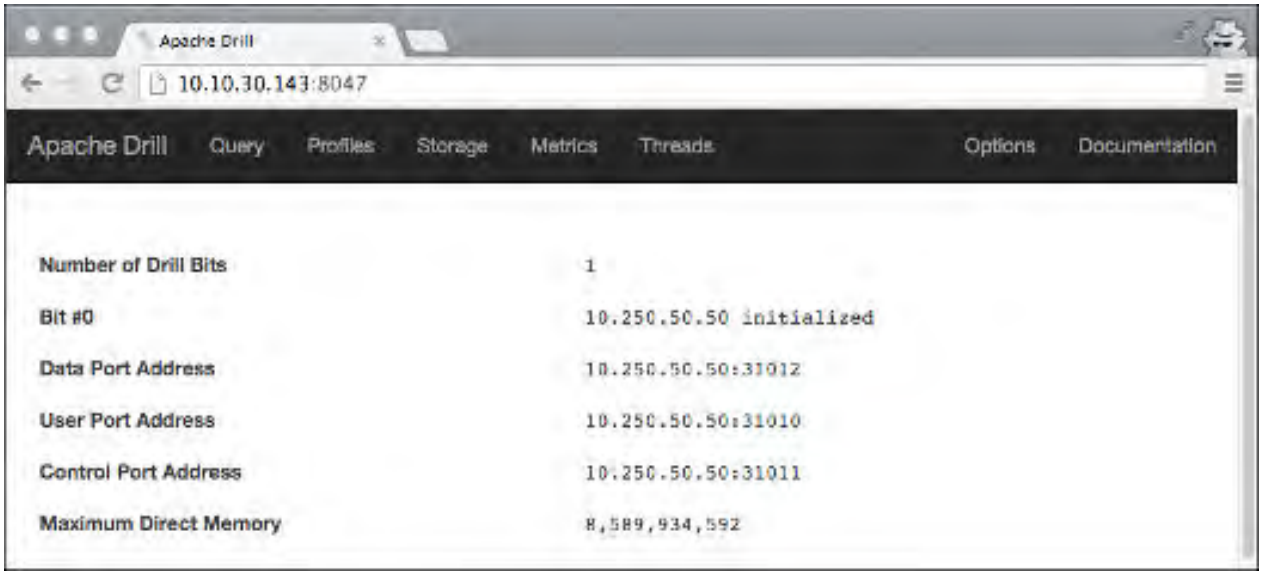
The Drill Web UI is one of several client interfaces that you can use to access Drill.

Accessing the Drill Web UI

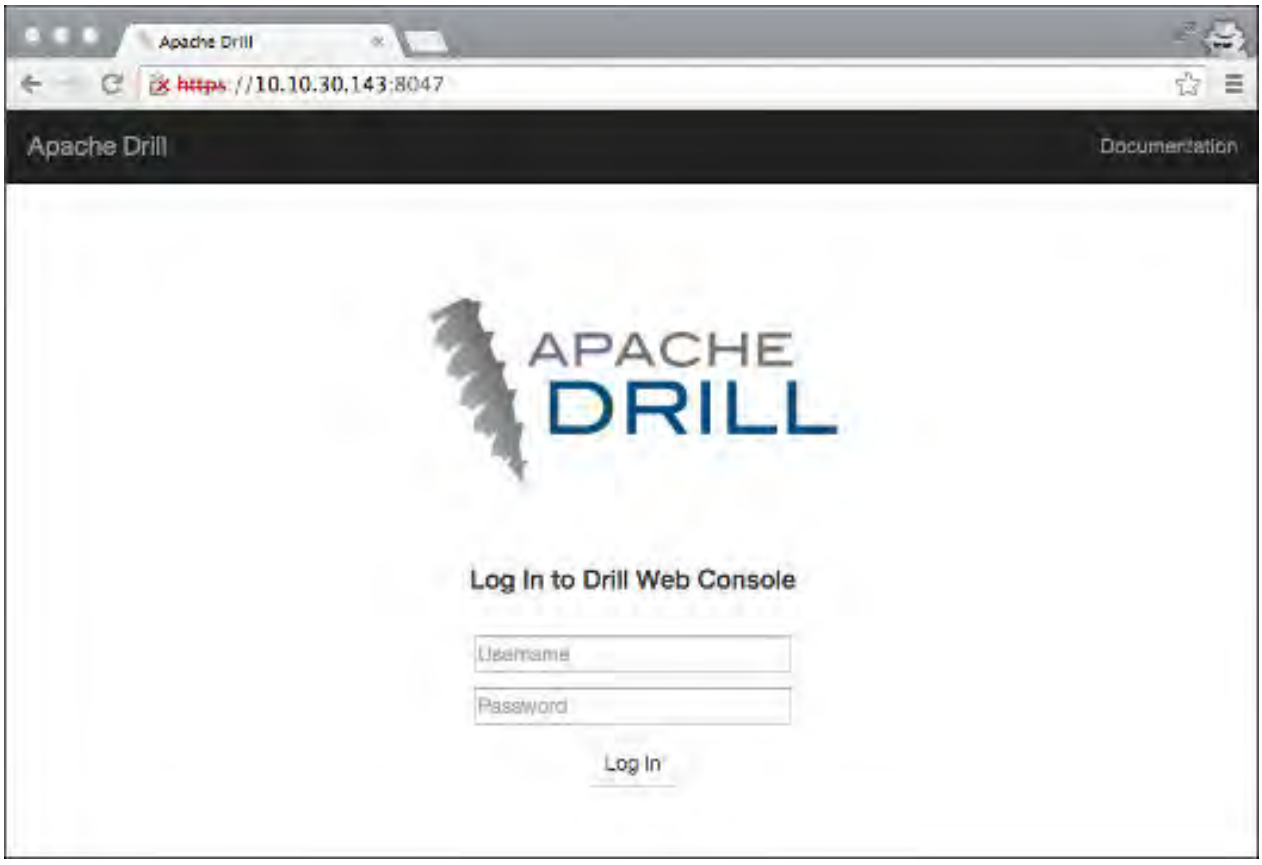
To open the Drill Web UI, launch a web browser, and go to one of the following URLs:

- If HTTPS support is disabled, use the default URL: http://<IP address or host name>:8047
- If HTTPS support is enabled, use this URL: https://<IP address or host name>:8047

If user authentication is *not* enabled, all the Drill Web UI controls appear to users and administrators, including Query, Profiles, Storage, Metrics, and Threads:

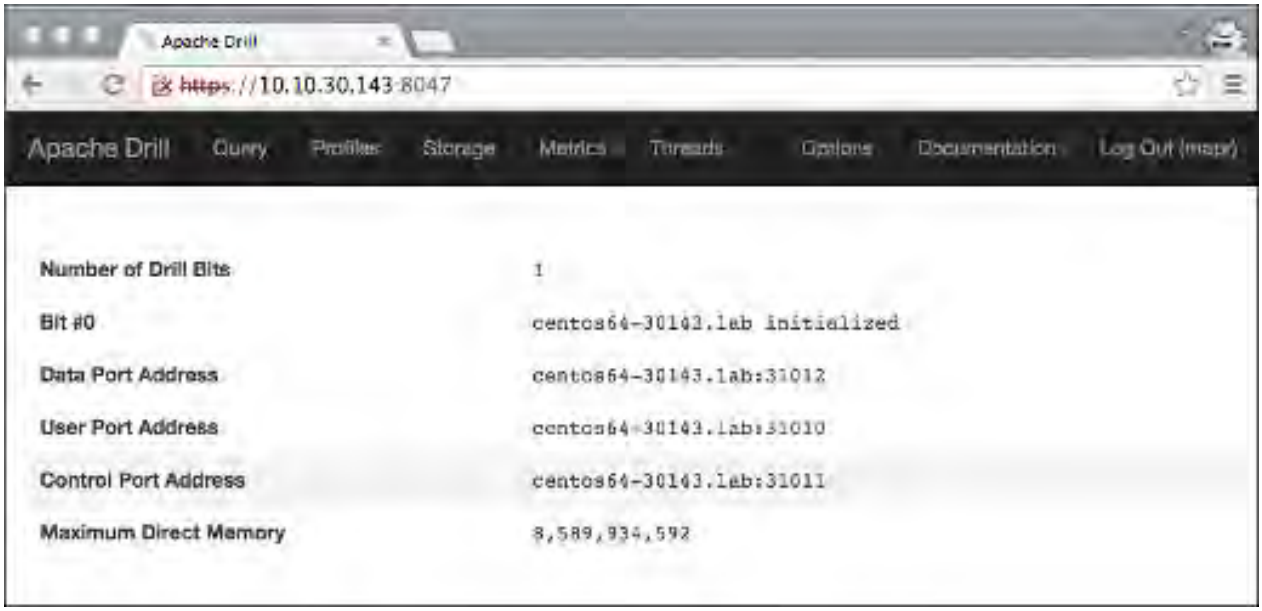


If user authentication is enabled, Drill prompts you for a user name and password:

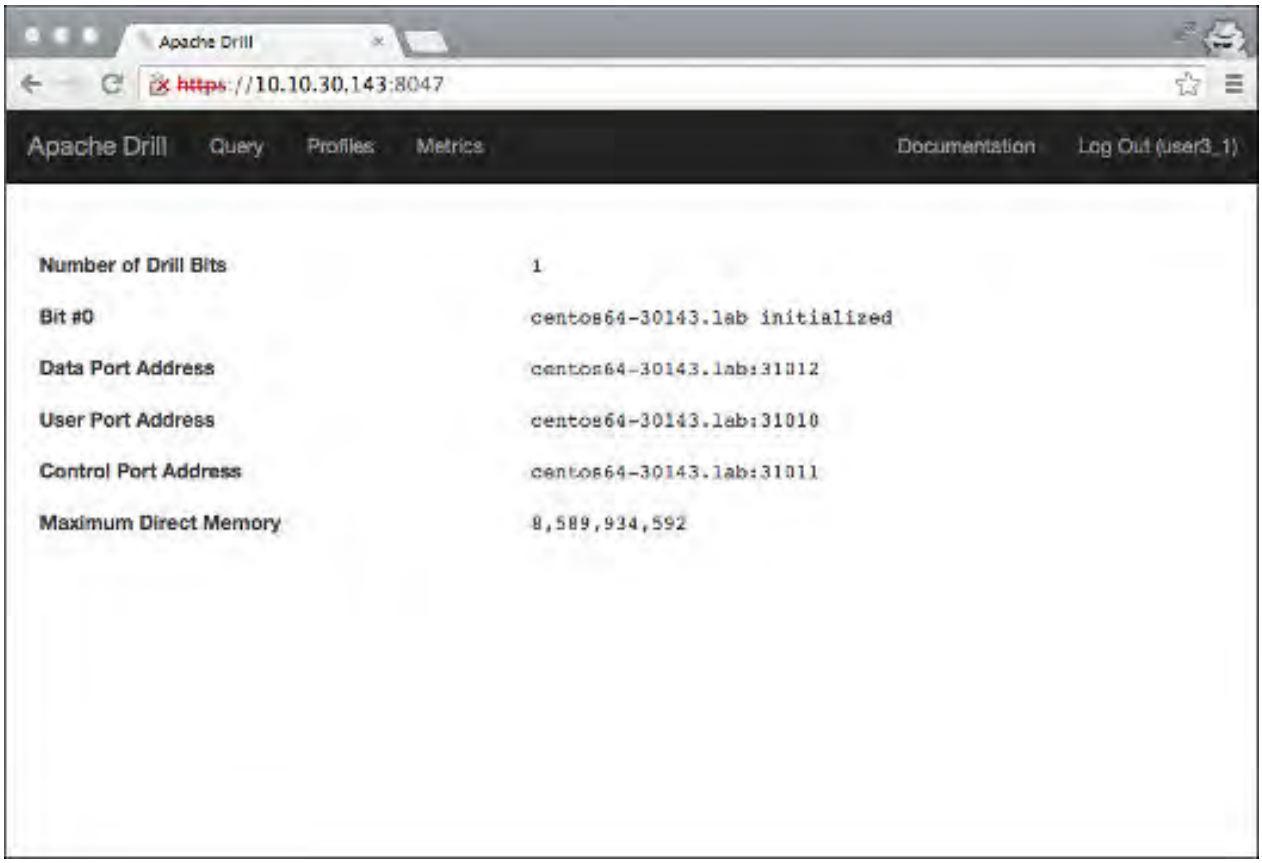


If an administrator logs in, all the Drill Web UI controls appear, including Query, Profiles, Storage, Metrics, Threads, and Options. For administrators, the Profiles page contains the profiles of all queries executed on a cluster. Only administrators can see and use the Storage tab to view, update, or add a new storage plugin configuration. Only administrators can see and use the Threads tab, which provides information about threads running in Drill.

last-updated: Apr 11, 2022



If a non-administrative user logs in, the Drill Web UI controls are limited to Query, Metrics, and Profiles. The Profiles tab for a non-administrative user contains the profiles of all queries the user issued either through ODBC, JDBC, or the Drill Web UI.



Related information

https://drill.apache.org/docs/architecture-introduction/#drill-clients
https://drill.apache.org/docs/securing-drill/

last-updated: Apr 11, 2022

Start the Drill Shell (SQLLine)
SQLLine is a JDBC application packaged with Drill that serves as the Drill shell. When you issue queries from the SQLLine, the SQLLine client sends the queries to the connected Drillbit (Drill node).

You can connect to Drill through SQLLine directly or through a connection-property file. If want to avoid exposing credentials, connecting through the connection-property file is recommended.

A JDBC connection string supplies the connection information to a Drill node or ZooKeeper cluster. When connecting to a ZooKeeper cluster, ZooKeeper selects the Drillbit that SQLLine connects to.

JDBC Connection String

This is an example of a JDBC connection string that connects SQLLine to drillnode1:

```
jdbc:drill:drillbit=drillnode1:31010
```

The default port on any Drill node is 31010.

Starting SQLLine

You start SQLLine from the Drill installation directory, as shown in the following example where SQLLine connects directly to a Drill node named `drillnode1`:

```
/opt/mapr/drill/drill-<version>/bin/sqlline -u
jdbc:drill:drillbit=drillnode1:31010
```

Connection Parameters

You can include SQLLine connection parameters in the connection string and run various shell commands, as described in Configuring the Drill Shell.

In the following example, -u is the connection parameter for the JDBC connection string, -n is the parameter for the username, and –p is the parameter for the password:

```
/opt/mapr/drill/drill-<version>/bin/sqlline -u
"jdbc:drill:drillbit=drillnode1:31010" -n mapr -p mapr
```

In the following example, the `!connect` shell command is used to hide the password when making an authenticated connection:

```
//From /opt/mapr/drill/drill-<version>/, run:

bin/sqlline

//The sqlline prompt appears. At the prompt, provide the connection string
with the !connect property:

sqlline> !connect jdbc:drill:drillbit=drillnode1:31010
//The system prompts you for the username and password.
Enter username for jdbc:drill:drillbit=drillnode1:31010: mapr
Enter password for jdbc:drill:drillbit=drillnode1:31010: ********
```

Notice: In Drill 1.15, the SQLLine `!connect` command incorrectly requests a username and password when connecting to a secure cluster via MAPRSASL or KERBEROS authentication:

```
sqlline> !connect jdbc:drill:drillbit=drillnode1:31010;auth=MAPRSASL

//!connect usage: connect <url> <username> <password> [driver]
//Driver is optional. Driver is the Apache Drill driver class,
org.apache.drill.jdbc.Driver.
```

To workaround this issue, provide your username when you connect and press Enter when prompted for the password:

```
sqlline> !connect jdbc:drill:drillbit=drillnode1:31010;auth=MAPRSASL
mapr
Enter password for jdbc:drill:drillbit=drillnode1:31010;auth=MAPRSASL:
```

Alternatively, you can use an empty quote in place of a username:

```
sqlline> !connect jdbc:drill:drillbit=drillnode1:31010;auth=MAPRSASL ""
```

Configuration Options

You can also include configuration options, such as `schema` and `auth` (if authentication is enabled):

```
/opt/mapr/drill/drill-<version>/bin/sqlline -u "jdbc:drill:drillbit
drillnode1:31010;schema=dfs;auth=MAPRSASL"
```

Schema	The `schema` is the name of a storage plugin configuration to use as the default for queries. If you indicate the schema in the connection string, you do not have to run the `USE <schema>;` query to switch to the schema you want to use. All queries run against the schema indicated in the JDBC connection string.
Authentication	If authentication is enabled (Plain, MAPRSASL, Kerberos), include the `auth` option in the connection string. If Drill is installed on a cluster secured by the default security, set `auth=MAPRSASL`. If using Plain authentication, include the username and password, as shown: `/opt/mapr/drill/drill-<version>/bin/ sqlline -u "jdbc:drill:drillbit drillnode1:31010;schema=dfs;auth=MAPRSASL"`

Connecting to a Specific Drill Node

Indicate which Drill node you want SQLLine to connect to in the JDBC connection string, using the following JDBC connection string format:

```
jdbc:drill:drillbit=<host>:<port>
```

Note that properties are case-sensitive. The `host` is the DNS or IP address of the server (Drill node). The default connection port is 31010.

Example

last-updated: Apr 11, 2022

The following example shows you how to start SQLLine with a JDBC connection string that includes the username, password, and auth parameters to authenticate to the server with Plain authentication:

```
/opt/mapr/drill/drill-<version>/bin/sqlline -u
"jdbc:drill:drillbit=<ip-address>:<port>;auth=PLAIN" -n <username> -p
<password>
```

If you installed Drill on a a cluster secured by default security, set the `auth` type to `maprsasl`:

```
/opt/mapr/drill/drill-<version>/bin/sqlline -u
"jdbc:drill:drillbit=<ip-address>:<port>;auth=MAPRSASL"
```

Connecting to ZooKeeper

When you include the ZooKeeper nodes in the JDBC connection string, ZooKeeper selects an available Drill node for SQLLine to connect to.

Indicate the ZooKeeper cluster you want SQLLine to connect to in the JDBC connection string, using the following JDBC connection string format:

```
jdbc:drill:zk=<zk-server-list>/drill/<clustername>
```

The `zk-server-list` is a comma-separated list of the ZooKeeper nodes in the cluster. The `clustername` is the unique name of the Drillbit cluster that you want to connect to.

You can locate the name of the Drillbit cluster in `/opt/mapr/drill/drill-<version>/conf/drill-distrib.conf`. The default name of the Drillbit cluster is `drillbits1`. The name is set by the `cluster-id` property. If you have multiple Drill clusters, you may want to override the Drillbit cluster name in `drill-override.conf`. However, first back-up your storage plugin configurations, as they may reset to the defaults when you change the cluster name. Restart Drill after you edit `drill-override.conf`.

Example

The following example shows you how to configure the JDBC connection string to connect SQLLine to the ZooKeeper cluster:

```
/opt/mapr/drill/drill-<version>/bin/sqlline
jdbc:drill:zk=<node-ip>:<port>,<node-ip>:<port>,=<node-ip>:<port>/drill/
drillbits1;auth=PLAIN -n <username> -p <password>
```

The default port for ZooKeeper nodes is 5181.

If you installed Drill on a secure cluster, set the `auth` type to `MAPRSASL`:

```
/opt/mapr/drill/drill-<version>/bin/sqlline
jdbc:drill:zk=<node-ip>:<port>,<node-ip>:<port>,=<node-ip>:<port>/drill/
drillbits1;auth=MAPRSASL
```

Using a Connection-Property File with SQLLine

Make sure you restrict access to the connection-property file to specific users.

Create a connection-property file named `login.properties`, as shown:

```
url:<jdbc-connection-url>
user:<username>
password:<password>

//Example
cat login.properties
```

```
url:jdbc:drill:schema=dfs;drillbit=drill-lab-node01
user:drilluser
password:letsdrill
```

To connect to Drill, run SQLLine as shown:

```
sqlline <sqlline args> <path/to/login.properties file>
```

The following examples show you how you can use the connection-property file to connect to Drill:

Example 1: Connecting to Drill via the connection-property file

Run SQLLine from `/opt/mapr/drill/drill-<version>/bin`:

```
sqlline login.properties

//List the active connection:
0: jdbc:drill:schema=dfs> !list
1 active connection:
 #0  open
jdbc:drill:schema=dfs;drillbit=drill-l
ab-node01

//Exit SQLLine:
0: jdbc:drill:schema=dfs>!q
```

Example 2: Submitting a query when connecting to Drill via the connection-property file

Run SQLLine from `/opt/mapr/drill/drill-<version>/bin`:

```
sqlline -q "SELECT version FROM
sys.version" login.properties

//Run query:
0: jdbc:drill:schema=dfs> select
version from sys.version;
+----------+
| version  |
+----------+
| 1.16.0   |
+----------+
1 row selected (0.295 seconds)
```

Example 3: Use the properties command to connect to Drill via the connection-property

Run SQLLine from `/opt/mapr/drill/drill-<version>/bin`:

```
sqlline

//At sqlline the prompt, run:
sqlline> !properties /home/drilluser/
login.properties
0: jdbc:drill:schema=dfs>
0: jdbc:drill:schema=dfs> !list
1 active connection:
 #0  open
jdbc:drill:schema=dfs;drillbit=drill-l
ab-node01
0: jdbc:drill:schema=dfs>
```


last-updated: Apr 11, 2022

Verify that Login Details are Secure

Run the following command to verify that login details are not exposed to other users:

```
ps -ef | grep sqlline

drilluser     18938 21924 99 14:14
pts/0    00:00:03 /opt/
jdk1.8.0_141/bin/
java -XX:MaxPermSize=512M -Djava.secur
ity.auth.login.config=/opt/mapr/conf/
mapr.login.conf \
-Dzookeeper.sasl.client=false -Dhadoop
.login=simple -Dlog.path=/opt/mapr/
drill/drill-1.10.0/logs/
sqlline.log -Dlog.query.path=/opt/
mapr/drill/drill-1.16.0/logs/
sqlline_queries.json \
-cp /opt/mapr/drill/drill-1.10.0/
conf:/opt/mapr/drill/drill-1.16.0/
jars/*:/opt/mapr/drill/drill-1.16.0/
jars/ext/*:/opt/mapr/drill/
drill-1.16.0/jars/3rdparty/*:/opt/
mapr/drill/drill-1.16.0/jars/classb/*
sqlline.SqlLine -d
org.apache.drill.jdbc.Driver --maxWidt
h=10000 --color=true login.properties
drilluser     20119  1691  0 14:14
pts/1    00:00:00 grep sqlline
```

Exit SQLLine

To exit SQLLine, run `!quit`.

Start|Stop the Drill Process

You can start|stop|restart the Drill process on one or more nodes using the Control System or the following command:

```
maprcli node services -name drill-bits -action start|restart|stop -nodes
<node host names separated by a space>
```

Use the host name if possible. Using host names instead of IP addresses is a best practice.

Related concepts

Drill Drivers on page 3333
HPE Ezmeral Data Fabric provides Drill ODBC and JDBC drivers that you can download and use to connect Drill to BI tools. The drivers are updated periodically to include support for new functionality in Drill.

Drill JDBC Drivers on page 3333
Download the Drill JDBC driver and use it on all platforms to connect BI tools, such as SQuirreL and Spotfire, to Drill. Drill also includes an embedded, open-source JDBC driver.

Hive to Drill Type Mapping

Using Drill you can read tables created in Hive that use data types in the Hive-to-Drill type mapping table. Currently, the Apache Hive version used by Drill does not support the timestamp in Unix Epoch format. The workaround is to use the JDBC format for the timestamp, which Hive accepts and Drill uses, as shown in the type mapping example.

For more information about connecting Drill to data sources, refer to Connect to Data Sources on the Apache Drill documentation web site. For information about workspaces, refer to Workspaces.

last-updated: Apr 11, 2022

Securing Drill

An administrator can install Drill with the default security configuration or manually configure custom security for Drill.

Drill supports several security features that secure the communication paths between Drill clients (such as ODBC/JDBC) and Drillbits and also between Drillbits. The following sections briefly describe the security configuration options for Drill and provide links to additional information and instructions.

MapR Default Security Configuration

Starting in MapR 6.0 and Drill 1.11 (EEP 4.0), Drill is automatically secured when you install Drill on a MapR cluster that was installed with the default MapR security configuration. The default MapR security configuration provides authentication, authorization, and encryption through the MapR-SASL mechanism, except for HTTPS, which uses SSL/TLS with form-based authentication. See Drill Default Security and SSL/TLS for Encryption for more information. You may also want to reference the following topics:

- Installing Drill, which describes some Drill installation security scenarios.
- Drill Drivers on page 3333, where you can access the JDBC and ODBC driver information and downloads required to connect to Drill when using the default security configuration.

Note: The default MapR security configuration does not include Kerberos or Plain authentication; however, you can manually configure these security mechanisms in addition to the default MapR security configuration.

Security Features Supported in a Custom Configuration

Drill supports several security features that an administrator can manually configure to secure the communication paths between the Drill client and Drillbit and also between Drillbits.

The following table lists the security features and mechanisms supported by Drill, as well as the communication paths secured by each mechanism:

Note: In the following table, Drill client refers to the Drill ODBC and JDBC clients. See Drill Drivers for ODBC and JDBC driver information.

Security Features	Supported Mechanisms	Communication Paths Secured
Authentication	MapR Security (MapR-SASL/Tickets)	• Drill client to Drillbit • Drillbit to Drillbit • Drillbit to ZooKeeper **Note:** The Drillbit creates znodes, for which ZooKeeper ACLs provide security. See Security Between ZooKeeper and Drillbits for more information.
	Kerberos	• Drill client to Drillbit • Drillbit to Drillbit
	Plain (username and password)	• Drill client to Drillbit

last-updated: Apr 11, 2022

Security Features	Supported Mechanisms	Communication Paths Secured
	Form-based	• Web client/REST API to Drillbit  **Note:** You can configure SSL/TLS for encryption.
	SPNEGO for HTTP	• Web client/REST API to Drillbit  **Note:** You can configure SSL/TLS for encryption.
Encryption	MapR Security (MapR/ Tickets)	• Drill client to Drillbit • Drillbit to Drillbit
	Kerberos	• Drill client to Drillbit • Drillbit to Drillbit
	SSL/TLS	• Drill client to Drillbit • Web client/REST API to Drillbit
Authorization	Based on filesystem permissions.	• Drill client to Drillbit
Impersonation	User Impersonation	• Drill client to Drillbit  **Note:** Drill supports user impersonation, inbound impersonation, and user impersonation with Hive authorization.
	Inbound impersonation	• Drill client to Drillbit  **Note:** Supports setting inbound impersonation policies, which are used to verify whether the user (set as the DelegationUID parameter passed in the client connection URL) can be impersonated by the connection user or not.

Views and File ACEs

In addition to the listed security features, you can create views on data to limit access to the data. You can also create file ACEs on the view definition files to protect the views.

Related concepts

Drill Drivers on page 3333
HPE Ezmeral Data Fabric provides Drill ODBC and JDBC drivers that you can download and use to connect Drill to BI tools. The drivers are updated periodically to include support for new functionality in Drill.

Start the Drill Shell (SQLLine) on page 3270
SQLLine is a JDBC application packaged with Drill that serves as the Drill shell. When you issue queries from the SQLLine, the SQLLine client sends the queries to the connected Drillbit (Drill node).

Connection URLs for Kerberos using JDBC Drivers to connect via SQLLine on page 3304

last-updated: Apr 11, 2022

You can use client-side connection URL parameters for Kerberos authentication in multiple combinations to authenticate a client with Drill.

Connection URL for Plain Authentication using the Apache JDBC Driver to connect via SQLLine on page 3311
When Plain authentication is enabled, each user that accesses the Drillbit process through a client, must provide username and password credentials for access.

SSL/TLS for Encryption on page 3312
You can enable SSL for Drill in a secure or unsecure cluster. SSL (Secure Sockets Layer), more recently called TLS, is a security mechanism that encrypts data passed between the Drill client and Drillbit (server). SSL also provides one-way authentication through which the Drill client verifies the identity of the Drillbit.

Configuring Drill Web UI and Web API Security on page 3323
The Drill web client and web API communicate with web browsers or web tools, like curl, through the HTTP or HTTPS. Drill uses HTTP by default.

SPNEGO for HTTP Authentication on page 3328
Drill 1.13 and later supports the Simple and Protected GSS-API Negotiation mechanism (SPNEGO) to extend the Kerberos-based single sign-on authentication mechanism to HTTP. An administrator configures the web server (Drillbit) to use SPNEGO for authentication. Depending on the system, either the administrator or the user configures the client (web browser or web client tool) to use SPNEGO for authentication.

Roles and Privileges

Drill has USER and ADMIN roles. Each role can perform different functions in Drill.

Access in the Drill Web UI differs between users and administrators. Certain pages are exposed based on privilege. For example, only administrators can see the Storage tab and edit a storage plugin configuration.

The following sections describe a few additional differences between a user and an administrator in Drill.

USER Role

The following list notes the functions that a user can perform in Drill:

- Users can run queries on data to which they have access.
- Users can view and cancel their own queries in the Profiles tab of the Drill Web UI.
- Users can create views on data to provide granular access to that data.

Note: Each data source manages the read/write permissions.

ADMIN Role

When authentication is enabled, only Drill users assigned the administrator (ADMIN) role can perform the following tasks:

- Change system-level options by issuing the ALTER SYSTEM command or through the options tab in the Drill Web UI.
- Update a storage plugin configuration through the REST API or Drill Web UI.
- View the profiles of all queries run by all users.
- Cancel running queries that were launched by any user in the cluster.
- Shut down the Drillbit in the Drill Web UI.

Configuring USER and ADMIN Roles

You can define administrative users through the `security.admin.user_groups` and `security.admin.users` options.

The default value for `admin.users` is the `drill_process_user`. The default value for `admin.user_groups` is `drill_process_user_groups`. These options accept a comma-separated list of users or user groups.

To edit these options, use the SET command, as shown in the following examples:

```
ALTER SYSTEM SET `security.admin.user_groups` = 'drill,
%drill_process_user_groups%';
ALTER SYSTEM SET `security.admin.users` = 'user1, %drill_process_user%';
ALTER SYSTEM SET `security.admin.users` = 'user1, user2';
```

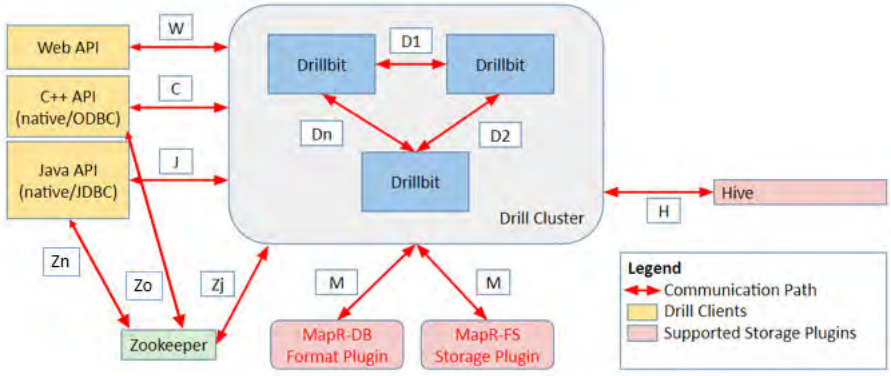
Drill Default Security

The default security configuration uses MapR-SASL (tickets) for authentication, authorization, and encryption to automatically secure the MapR cluster and ecosystem components when you install them manually or using the MapR Installer.

The default security configuration automatically secures all Drill communication paths with the following exceptions:

- The path between the web client and web server (W) uses SSL/TLS with form-based authentication.
- The path between the ODBC/JDBC client and ZooKeeper (Zn, Zo) is unsecure.

The following diagram shows the secured communication paths:



The following table describes the security support for each communication path in the diagram, along with the components involved in the communication:

Type of Security Supported	Communication Path	Component Communication
Authentication and encryption using MapR-SASL (tickets)	C	ODBC client/C++ API to Drillbits
	J	JDBC client/Java API to Drillbits
	D1, D2, Dn	Drillbit to Drillbit
	M	Drillbit to MapR Database/MapR File System

last-updated: Apr 11, 2022

	H	Drillbit to Hive **Note:** The Hive storage plugin is not secured by default and requires that you manually modify the configuration to enable security. See Configuring the Hive Storage Plugin on page 3263.
Plain authentication with SSL encryption (HTTPS enabled)	W	Web client/Web API to Web server **Note:** The HTTPS channel (Web client) uses Plain authentication to authenticate a Web client with SSL/TLS for encryption. This is configured by default in a secure 6.x cluster with Drill 1.11 or later installed. Plain authentication does not support encryption. You must enable SSL to encrypt the communication channels when using Plain authentication. See Configuring Drill Web UI and Web API Security on page 3323.
Authentication with MapR security (no encryption)	Zj	Drillbit to ZooKeeper **Note:** The Drillbit creates znodes, for which ZooKeeper *ACL*s provide security. See Security Between ZooKeeper and Drillbits on page 3322 for more information.
No security support	Zo, Zn	ODBC/JDBC client to ZooKeeper **Note:** Only znodes created for Drillbit endpoints in Zookeeper are readable by the client. All other znodes (not required by the client) are secured using ZooKeeper *ACL*s, and are only readable by Drillbits.

Note the following information:

- Kerberos and Plain authentication are not enabled or configured as part of the default security configuration. However, you can manually configure these security mechanisms in addition to the defaults. If you enable Plain authentication, you must use SSL/TLS for encryption.
- Drill clients running Drill 1.10 and earlier do not support encryption and cannot connect to Drillbits installed with the default MapR security configuration.

Connecting Drill

See Drill Drivers on page 3333. Alternatively, you can use SQLLine, the Drill shell, as shown:

Disabling Security

You can turn off the default MapR security configuration across the entire MapR cluster.

Note: If you unsecure a cluster, you must backup the Drill znodes. After the switch to unsecured, update the ACL on the Drill znodes so that Drill in an unsecured cluster can access all Drill znodes. See Security Between ZooKeeper and Drillbits on page 3322 for more information.

last-updated: Apr 11, 2022

To disable the default security configuration across an entire MapR cluster, run `configure.sh` with the `-unsecure` parameter, as shown:

```
/opt/mapr/server/configure.sh -forceSecurityDefaults [ -unsecure | -secure ]
 -C <CLDB_node> -Z <ZK_node>
```

Alternatively, you can enable security across an entire MapR cluster with the `-secure` parameter.

See Installing Drill on page 177 and configure.sh on page 2053 for more information.

Additional Notes

Performance	The default security configuration enables encryption for all network channels, which can affect Drill performance. If performance is your highest priority, install MapR and Drill without security enabled and have your security expert manually configure cluster security. Alternatively, you can install MapR and Drill with security enabled, and then disable individual Drill security settings. For example, you can edit the `drill-override.conf` file and disable encryption, leaving authentication enabled. **Note:** Manually configuring security settings when default security is enabled is not recommended.
Drill Configuration Files	The default security configuration introduces new Drill configuration files. In addition to drill-override.conf, distrib-env.sh, and drill-env.sh, Drill includes a drill-distrib.conf file. See Drill Configuration Files on page 3346 for more information. Note that modifying drill distribution-specific files is highly discouraged. To customize any Drill configuration, use `drill-override.conf` and `drill-env.sh`.
HBase	As of MapR 6.0 and Drill 1.11, HBase is no longer supported; therefore, the communication path between Drill and HBase is also not supported.

User Impersonation

Impersonation allows a service to act on behalf of a client while performing the action requested by the client. By default, user impersonation is disabled in Drill. You can configure user impersonation in the `/opt/mapr/drill/drill-<version>/drill-override.conf` file.

When you enable impersonation, Drill executes all the client requests as the user logged in to the client. Drill passes the user credentials to the filesystem, and the filesystem checks to see if the user has permission to access the data. When you enable authentication, Drill uses the pluggable authentication module (PAM) to authenticate a user's identity before the user can access the Drillbit process.

If impersonation is disabled, Drill executes all of the client requests against the file system as the user that started the Drillbit service on the node. This is typically a privileged user. The filesystem verifies that the system user has permission to access the data.

User Impersonation Example

When impersonation is disabled and user Bob issues a query through the SQLLine client, SQLLine passes the query to the connecting Drillbit. The Drillbit executes the query as the system user that started the Drill process on the node. For the purpose of this example, we will assume that the system user has full access to the filesystem. Drill executes the query and returns the results back to the client.

When impersonation is enabled and user Bob issues a query through the SQLLine client, the Drillbit uses Bob's credentials to access data in the filesystem. The filesystem checks to see if Bob has permission to access the data. If Bob has permission, Drill returns the query results to the client. If Bob does not have permission, Drill returns an error.

Impersonation Support

Drill supports impersonation with the following clients, storage plugins, and types of queries:

- **Clients**
  - ODBC
  - JDBC
  - REST API
  - Drill Web UI
- **Storage plugins**
  - MapR File System
  - MapR Database
  - Hive
- **Types of queries**

  Note: When you enable impersonation, the setting applies to queries on data and metadata. For example, if you issue the SHOW SCHEMAS command, Drill impersonates the user logged into the client to access the requested metadata. If you issue a SELECT query on a workspace, Drill impersonates the user logged in to the client to access the requested data.

  Drill applies impersonation to queries issued using the following commands:

  - SHOW SCHEMAS
  - SHOW DATABASES
  - SHOW TABLES
  - CTAS
  - SELECT
  - CREATE VIEW
  - DROP VIEW

last-updated: Apr 11, 2022

- SHOW FILES.

Note: To successfully run the CTAS and CREATE VIEW commands, a user must have write permissions on the directory where the table or view will exist. Running these commands creates artifacts on the filesystem.

Impersonation and Views
You can use views with impersonation to provide granular access to data and protect sensitive information.

When you create a view, Drill stores the view definition in a file and suffixes the file with `view.drill`. For example, if you create a view named myview, Drill creates a view file named `myview.view.drill` and saves it in the current workspace or the workspace specified, such as `dfs.views.myview`. See CREATE VIEW.

You can create a view and grant read permissions on the view to give other users access to the data that the view references. When a user queries a view on which s/he has read access, Drill impersonates the view owner to access the underlying data. If the user tries to query the data directly (instead of using the view), Drill returns a permission denied error. A user with read access to a view can create new views from the originating view to further restrict access on data.

View Permissions

A user must have write permission on a directory or workspace to create a view, as well as read access on the table(s) and/or view(s) that the view references. When a user creates a view, permission on the view is set to owner by default. Users can query an existing view or create new views from the view if they have read permissions on the view file and the directory or workspace where the view file is stored.

When users query a view, Drill accesses the underlying data as the user that created the view. If a user does not have permission to access a view, the query fails and Drill returns an error. Only the view owner or a superuser can modify view permissions to change them from owner to group or world.

The view owner or a superuser can modify permissions on the view file directly or they can set view permissions at the system or session level prior to creating any views. Any user that alters view permissions must have write access on the directory or workspace in which they are working.

Modifying Permissions on a View File

Only a view owner or a super user can modify permissions on a view file to change them from owner to group or world readable. Before you grant permission to users to access a view, verify that they have access to the directory or workspace in which the view file is stored.

Use the `chmod` and `chown` commands with the appropriate octal code to change permissions on a view file:

```
hadoop fs -chmod <octal code> <file_name>
hadoop fs -chown <user>:<group> <file_name>
//hadoop fs -chmod 750 employees.view.drill
```

Modifying SYSTEM|SESSION Level View Permissions

Use the `ALTER SESSION|SYSTEM` command with the `new_view_default_permissions` parameter and the appropriate octal code to set view permissions at the system or session level prior to creating a view.

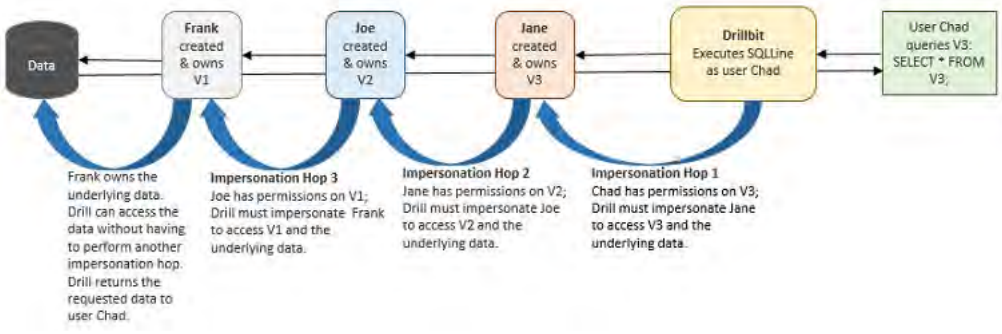
```
ALTER SESSION SET `new_view_default_permissions` = '<octal_code>';
ALTER SYSTEM SET `new_view_default_permissions` = '<octal_code>';
//ALTER SESSION SET `new_view_default_permissions` = '777';
```

After you set this parameter, Drill applies the same permissions on each view created during the session or across all sessions if set at the system level.

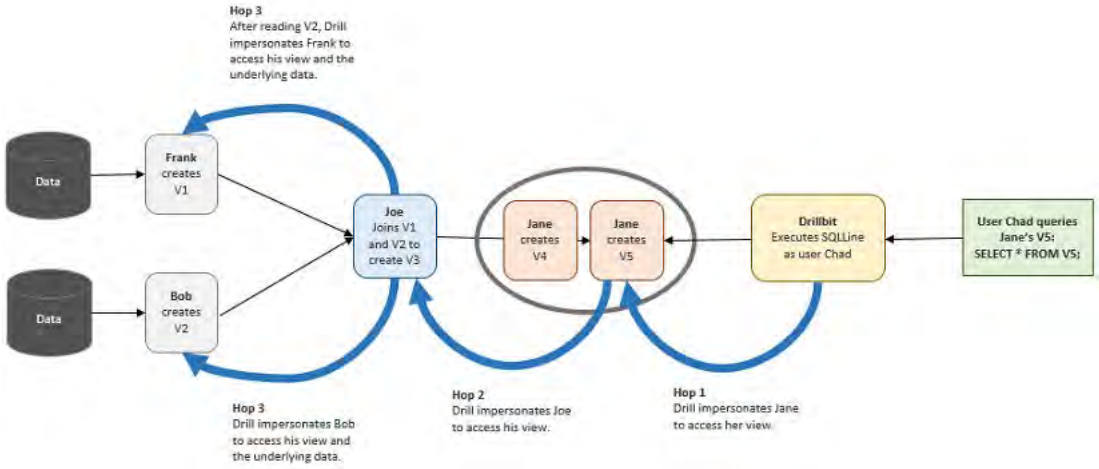
Chained Impersonation
You can configure Drill to allow chained impersonation on views when you enable impersonation in the `drill-override.conf` file. Chained impersonation controls the number of identity transitions that Drill can make when a user queries a view. Each identity transition is equal to one hop.

An administrator can set the maximum number of hops for impersonation to limit the number of times that Drill can impersonate a different user when other users query a view. The default maximum number of hops is set at 3. When the maximum number of hops is set to 0, Drill does not allow impersonation chaining, and a user can only read data for which they have direct permission to access. An administrator may set the chain length to 0 to protect highly sensitive data.

The following diagram depicts a scenario where the maximum hop number is set to 3, and Drill must impersonate three users to access data when Chad queries a view that Jane created:



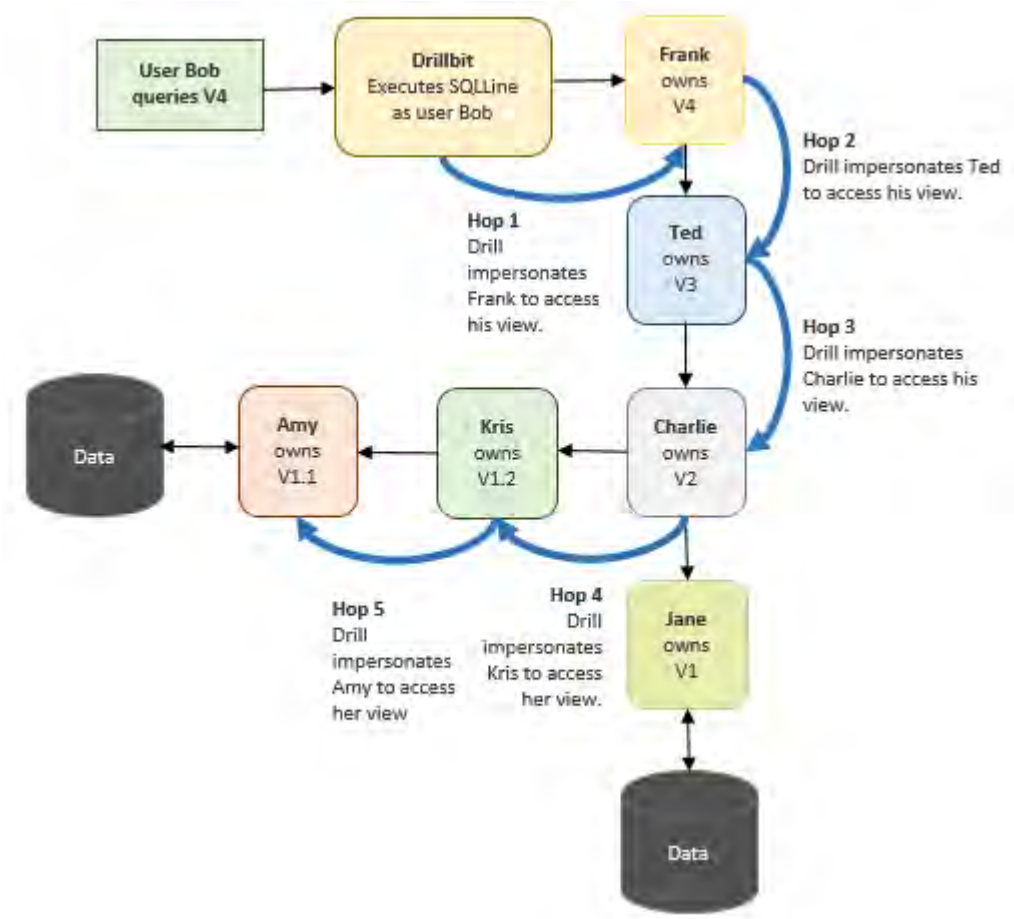
In the previous example, Joe created V2 from the view that user Frank created. In the following example, Joe created V3 by joining a view that Frank created with a view that Bob created.



Although V3 was created by joining two different views, the number of hops remains at 3 because Drill does not read the views at the same time. Drill reads V2 first and then reads V1.

In the next example, Bob queries V4 which was created by Frank. Frank's view was created from several underlying views. Charlie created V2 by joining Jane's V1 with Kris's V1.2. Kris's V1.2 was created from Amy's V1.1, increasing the complexity of the chaining. Assuming that the hop limit is set at 4, this scenario exceeds the limit.

last-updated: Apr 11, 2022



When Bob queries Franks's view, Drill returns an error stating that the query cannot complete because the number of hops required to access the data exceeds the maximum hop setting of 4.

If users encounter this error, the administrator can increase the maximum hop setting to accommodate users running queries on views.

Configuring Impersonation and Chaining

Impersonation allows a service to act on behalf of a client while performing the action requested by the client. Chaining is a system-wide setting that applies to all views. Currently, Drill does not provide an option to allow different chain lengths for different views.

Complete the following steps on each Drillbit node to enable user impersonation, and set the maximum number of chained user hops that Drill allows:

1. Navigate to `<drill_installation_directory>/conf/` and edit `drill-override.conf`.

last-updated: Apr 11, 2022

2. Under `drill.exec`, add the following:

```
drill.exec.impersonation: {
                  enabled: true,
                  max_chained_user_hops: 3
  }
```

   Alternatively, you can nest impersonation within the `drill.exec` block, as shown in the following example:

```
drill.exec: {
           cluster-id: "cluster_name",
           zk.connect:
"<hostname>:<port>,<hostname>:<port>,<hostname>:<port>",
           sys.store.provider.zk.blobroot: "hdfs://",
           impersonation: {
             enabled: true,
             max_chained_user_hops: 3
           }
         }
```

3. Set the maximum number of chained user hops.

4. In `<drill_installation_directory>/conf/drill-env.sh`, add one of the following lines:

   - If the underlying filesystem has MapR security enabled, add the following line: `export MAPR_TICKETFILE_LOCATION=/opt/mapr/conf/maprusertickет`
   - If the underlying filesystem is not secure, add the following line: `export MAPR_IMPERSONATION_ENABLED=true`

5. Restart the Drillbit process on each Drill node.

   ```
   maprcli node services -name drill-bits -action restart -nodes
   <node-hostnames-separated-by-a-space> -f
   ```

Example: Impersonation and Chaining
This example demonstrates how to use impersonation and chaining to limit access to data. Impersonation allows a service to act on behalf of a client while performing the action requested by the client. Chaining controls the number of identity transitions that Drill can make when a user queries a view.

Note: The number of identity transitions in controled by the `max_chained_user_hops` option in the `drill-override.conf` file. See Chained Impersonation and Configuring Impersonation and Chaining for more information.

Frank is a senior HR manager at a company. Frank has access to all of the employee data because he is a member of the hr group. Frank created a table named “employees” in his home directory to store the employee data he uses. Only Frank has access to this table.

```
drwx------ frank:hr /user/frank/employees
```

Each record in the employees table consists of the following information: emp_id, emp_name, emp_ssn, emp_salary, emp_addr, emp_phone, emp_mgr

Frank needs to share a subset of this information with Joe who is an HR manager reporting to Frank. To share the employee data, Frank creates a view called emp_mgr_view that accesses a subset of the data. The emp_mgr_view filters out sensitive employee information, such as the employee social security

last-updated: Apr 11, 2022

numbers, and only shows data for the employees that report directly to Joe. Frank and Joe both belong to the mgr group. Managers have read permission on Frank's directory.

```
rwxr----- frank:mgr /user/frank/emp_mgr_view.view.drill
```

The emp_mgr_view.view.drill file contains the following view definition:

```
(view definition: SELECT emp_id, emp_name, emp_salary, emp_addr, emp_phone
FROM `/user/frank/employee` WHERE emp_mgr = 'Joe')
```

When Joe issues SELECT * FROM emp_mgr_view, Drill impersonates Frank when accessing the employee data, and the query returns the data that Joe has permission to see based on the view definition. The query results do not include any sensitive data because the view protects that information. If Joe tries to query the employees table directly, Drill returns an error or null values.

Because Joe has read permissions on the emp_mgr_view, he can create new views from it to give other users access to the employee data even though he does not own the employees table and cannot access the employees table directly.

Joe needs to share employee contact data with his direct reports, so he creates a special view called emp_team_view to share the employee contact information with his team. Joe creates the view and writes it to his home directory. Joe and his reports belong to a group named joeteam. The joeteam group has read permissions on Joe's home directory so they can query the view and create new views from it.

```
rwxr----- joe:joeteam /user/joe/emp_team_view.view.drill
```

The emp_team_view.view.drill file contains the following view definition:

```
(view definition: SELECT emp_id, emp_name, emp_phone FROM `/user/frank/
emp_mgr_view.drill`);
```

When anyone on Joe's team issues SELECT * FROM emp_team_view, Drill impersonates Joe to access the emp_team_view and then impersonates Frank to access the emp_mgr_view and the employee data. Drill returns the data that Joe's team has can see based on the view definition. If anyone on Joe's team tries to query the emp_mgr_view or employees table directly, Drill returns an error or null values.

Because Joe's team has read permissions on the emp_team_view, they can create new views from it and write the views to any directory for which they have write access. Creating views can continue until Drill reaches the maximum number of impersonation hops (chained impersonation).

User Impersonation with Hive

You can configure Drill impersonation with Hive impersonation to authorize access to metadata in the Hive metastore repository and data in the Hive warehouse. Drill impersonation works with Hive when Hive has impersonation enabled and optionally, storage based or SQL standard based authorization enabled. Drill impersonation can also work with Hive when the Hive metastore has Kerberos enabled on a secure cluster. Currently, Drill does not support Hive configured with Sentry authorization.

Storage Based Authorization

Hive storage based authorization is a remote metastore server security feature that uses the underlying filesystem permissions to determine permissions on databases, tables, and partitions. The permissions a user or group has on directories in the filesystem determines access to data. Because the filesystem controls access at the directory and file level, storage based authorization cannot control access to data at the column or view level.

You manage user and group privileges through permissions and access controls in the distributed filesystem. DDL statements that manage permissions, such as GRANT and REVOKE, do not have any effect on permissions in the storage based authorization model.

For more information, see Storage Based Authorization in the Metastore Server.

SQL Standard Based Authorization

The SQL standard based authorization model can control which users have access to columns, rows, and views. SQL standard based authorization is configured in HiverServer2 and enforced during query processing. Users with the appropriate permissions can issue the GRANT and REVOKE statements to manage privileges from Hive.

For more information, see SQL Standard Based Hive Authorization.

Prerequisites

To configure user impersonation with Hive, the system must meet the following requirements:

- MapR version 4.1 or later
- Drill installed with Drillbits running as the `mapr` user
- Supported version of Hive installed with the following:
  - User impersonation enabled
  - Configured Hive remote metastore repository
  - (Optional) SQL standard based authorization or storage based authorization configured

Note: See EEP Components and OS Support on page 5536 for supported versions of Hive.

Configuration

Complete the steps listed in Configuring User Impersonation with Hive.

Configuring User Impersonation with Hive

Complete the following steps on a secure or insecure MapR cluster to configure user impersonation with Hive:

Step 1: Modify drill-env.sh

Modify `<DRILL_HOME>/conf/drill-env.sh` to include the required environment variables on each Drill node.

Insecure Cluster	On an insecure cluster, include the following environment variable: `export MAPR_IMPERSONATION_ENABLED=true`

Secure Cluster	On a secure cluster, include the following environment variables:
	`export DRILL_JAVA_OPTS="$DRILL_JAVA_OPTS -Djava.security.auth.login.config=/opt/mapr/conf/mapr.login.conf -Dzookeeper.sasl.client=true"`
	`export DRILL_JAVA_OPTS="$DRILL_JAVA_OPTS -Dmapr_sec_enabled=true -Dhadoop.login=maprsasl_keytab -Dzookeeper.saslprovider=com.mapr.security.maprsasl.MaprSaslProvider -Dmapr.library.flatclass"`

last-updated: Apr 11, 2022

```
export MAPR_TICKETFILE_LOCATION=/opt/
mapr/conf/maprusertickет
```

Step 2: Modify drill-override.conf
For secure and inecure clusters, modify `<DRILL_HOME>/conf/drill-override.conf` on each Drill node to enable impersonation in Drill, and set the maximum number of chained user hops that Drill allows.

Add the following configuration properties to the `drill.exec` block in `drill-override.conf`:

```
drill.exec: {
 cluster-id: "<drill_cluster_name>",
 zk.connect: "<hostname>:5181,<hostname>:5181,<hostname>:5181"
 impersonation: {
       enabled: true,
       max_chained_user_hops: 3
 }
}
```

Step 3: Modify the Hive Storage Plugin in Drill
Modify the Hive storage plugin configuration in the Drill Web UI based on the authorization and security scenario for the cluster. You can only access the Drill Web UI for a running Drillbit.

Complete the following steps to modify the Hive storage plugin configuration:

1. Navigate to `http://<drillbit_hostname>:8047`, and select the **Storage** tab.
2. Click **Update** next to the hive option.
3. In the configuration window, add the required properties based on the authorization type and security scenario:

Storage Based Authorization or No Authorization Enabled

For a *insecure cluster*, add the following properties to the configuration:

```
{
 type:"hive",
 enabled: true,
 configProps : {

"hive.metastore.uris" : "thrift://
<metastore_hostname>:9083",
   "fs.default.name" : "maprfs:///",
   "hive.metastore.sasl.enabled" :
"false",
   "hive.server2.enable.doAs" :
"true",

"hive.metastore.execute.setugi" :
"true"
 }
}
```

For a *secure cluster*, add the following properties to the configuration:

```
{
"type": "hive",
"enabled": true,
```

```
"configProps": {
  "hive.metastore.uris": "thrift://
<metastore_hostname>:9083",
  "fs.default.name": "maprfs:///",
  "hive.server2.enable.doAs": "true"
   }
}
```

Add the following additional properties if the Hive metastore is configured with Kerberos in a secure cluster; include a comma after each line except for the last:

```
"hive.metastore.kerberos.principal":
 "hive/<metastore_thrift_server>"
"hive.metastore.sasl.enabled":
"true"
```

SQL Standard Based Authorization

For an *insecure cluster*, add the following properties to the configuration:

```
{
 type:"hive",
 enabled: true,
 configProps : {
   "hive.metastore.uris" :
"thrift://
<metastore_hostname>:9083",
   "fs.default.name" : "maprfs:///",

"hive.security.authorization.enabled
" : "true",

"hive.security.authenticator.manager
" :
"org.apache.hadoop.hive.ql.security.
SessionStateUserAuthenticator",

"hive.security.authorization.manager
" :
"org.apache.hadoop.hive.ql.security.
authorization.plugin.sqlstd.SQLStdHi
veAuthorizerFactory",
   "hive.metastore.sasl.enabled" :
"false",
   "hive.server2.enable.doAs" :
"false",

"hive.metastore.execute.setugi" :
"false"
 }
}
```

For a *secure cluster*, add the following properties to the configuration:

```
{
"type": "hive",
"enabled": true,
"configProps": {
```

```
  "hive.metastore.uris": " thrift://
<metastore_hostname>:9083",
  "fs.default.name": "maprfs:///",

"hive.security.authorization.enabled
": "true",

"hive.security.authenticator.manager
":
"org.apache.hadoop.hive.ql.security.
SessionStateUserAuthenticator",

"hive.security.authorization.manager
":
"org.apache.hadoop.hive.ql.security.
authorization.plugin.sqlstd.SQLStdHi
veAuthorizerFactory",
  "hive.server2.enable.doAs":
"false",
  "hive.metastore.execute.setugi":
"true"
 }
}
```

Add the following additional properties if the Hive metastore is configured with Kerberos in a secure cluster; include a comma after each line except for the last:

```
"hive.metastore.kerberos.principal":
 "hive/<metastore_thrift_server>"
"hive.metastore.sasl.enabled":
"true"
```

Step 4: Restart Warden
Run the following command on all nodes to restart the Warden service:

```
service mapr-warden restart
```

If you have `clush` installed, you can run the following command to restart Warden on all nodes at once:

```
clush -a "service mapr-warden restart"
```

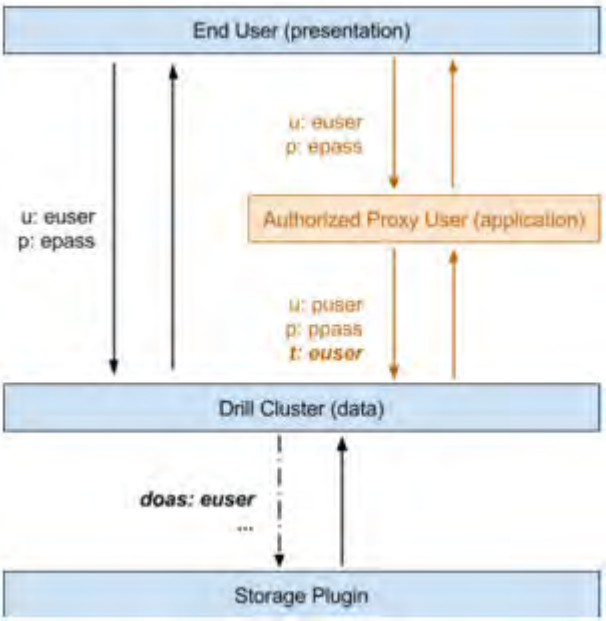
Inbound Impersonation
An administrator can define inbound impersonation policies to impersonate the end user.

Drill supports user impersonation where queries run as the user that created a connection. However, this user is not necessarily the end user who submits the queries. For example, in a classic three-tier architecture, the end user interacts with Tableau Desktop, which communicates with a Tableau Server, which in turn communicates with a Drill cluster. In this scenario, a proxy user creates a connection, and the queries are submitted to Drill by the proxy user on behalf of the end user, and not by the end user directly. In this particular case, the query needs run run as the end user.

The proxy user must be authorized to submit queries on behalf of the specified end user. Otherwise, any user can impersonate another user. The query runs as the end user, and data authorization is based on this user's access permissions. Note that without authentication enabled in both communication channels, a user can impersonate any other user.

Drill trusts proxy users to provide the correct end user identity information. Drill does not authenticate the end user. The proxy user (application) is responsible for end user authentication, which is usually enabled.

The following diagram shows how identity is propagated through various layers (with authentication enabled). The flow on the left is Drill with user impersonation enabled. The flow on the right is Drill with user impersonation and inbound impersonation enabled. `t:euser` is a property on the connection (u is `username`, pis `password`, `t` is `impersonation_target`).



The following topic provides instructions for configuring inbound impersonation:

Configuring Inbound Impersonation

Administrators can configure inbound impersonation in the `drill-override.conf` file.

Complete the following steps to enable inbound impersonation:

1. If user impersonation is not enabled, you must enable it before configuring inbound impersonation. To enable user impersonation, edit `/opt/mapr/drill/drill-<version>/drill-override.conf` and set the option to true, as shown:

```
{
      drill.exec.impersonation.enabled: true,
      ...
      }
```

2. Define inbound impersonation policies. For example, the following ALTER SYSTEM statement authorizes:

   - `puser1` to impersonate any user (use * as a wildcard character)
   - `puser2` to impersonate `euser1` and all users in `egroup2`
   - all users in `pgroup3` to impersonate all users in `egroup3`

```
ALTER SYSTEM SET `exec.impersonation.inbound_policies`='[
  { proxy_principals : { users: ["puser1"] },
    target_principals: { users: ["*"] } },
  { proxy_principals : { users: ["puser2"] },
    target_principals: { users: ["euser1"], groups :  ["egroup2"] } },
  { proxy_principals : { groups: ["pgroup3"] },
    target_principals: { groups: ["egroup3"] } } ]';
```


last-updated: Apr 11, 2022

Policy format:

```
{ proxy_principals : { users : ["...", "..."], groups : ["...", "..."] },
  target_principals: { users : ["...", "..."], groups : ["...",
"..."] } }
```

3. Ensure that the proxy user (application) passes the username of the impersonation target user to Drill when creating a connection through the `impersonation_target` connection property. For example, through sqlline:

```
bin/sqlline -u
"jdbc:drill:schema=dfs;zk=myclusterzk;impersonation_target=euser1" -n
puser1 -p ppass1
```

Note: In this example, `puser1` is the user submitting the queries. This user is authenticated. Since this user is authorized to impersonate any user, queries through the established connection are run as `euser1`.

Related information

https://drill.apache.org/docs/configuration-options-introduction/#system-options

MapR Security (Tickets)

Drill supports authentication and encryption through the MapR Security (tickets) security mechanism. Authentication is the process of establishing confidence of authenticity. Encryption is the process of converting information or data from plain text into ciphertext to prevent unauthorized access. An administrator can manually configure Drill to use MapR Security. When MapR Security is enabled, all Drill clients, such as JDBC and ODBC, must connect to Drillbits through MapR Security.

The MapR Security mechanism secures the communication path between the Drill client, such as JDBC/ODBC and Drillbit, Drillbit and ZooKeeper, and also between Drillbits.

Note: The Drill web communication path (web client to web server) does not support MapR Security-based authentication and encryption.

Note: The Apache JDBC driver packaged with Drill does not support MapR Security.

Configuration parameters in the Drill startup configuration file, `/opt/mapr/drill/drill-<version>/conf/drill-override.conf`, enable or disable authentication and encryption.

Prerequisites

- Ensure that your MapR cluster is secure. To manually configure secure clusters with MapR Security, see Enable Wire-Level Security.
- When you configure Drill to use encryption, authentication must also be configured and enabled with the encryption-specific configurations.
- For encryption and authentication to work together, the Drill client and Drillbits must all run Drill 1.11 or later. Drill clients running earlier versions of Drill cannot connect to Drillbits when encryption is enabled.
- The client-side should have created a user mapr ticket for the authenticating user. See maprlogin for more information.

Post-requisite

You must restart the Drillbit process on each node after you enable security and/or modify the configuration options, as shown:

```
$ maprcli node services -name drill-bits -action restart -nodes <node host
names separated by a space>
```

Download and configure the MapR-specific JDBC or ODBC Drill drivers. See Drill Drivers for more information.

The following topics provide configuration information to enable authentication and encryption in Drill:

Configuring Authentication
An administrator can enable MapR Security as the only authentication mechanism, or in addition to other mechanisms, such as Kerberos and Plain authentication in `drill-override.conf`.

Note: When Drill is installed on the MapR Data Platform, Drill distribution defaults are stored in the `drill-distrib.conf` file. To override the defaults, you must explicitly disable them in the `drill-override.conf` file.

The following sections provide configuration examples for several configuration scenarios:

Note: For client-side configuration, see Drill Drivers.

Example 1: Drill Client to Drillbit Authentication using MapR Security Only

```
drill.exec:{
              security: {
                 user.auth.enabled: true,
                 auth.mechanisms : ["MAPRSASL"]
          }
}
```

Note: Drill executes all queries as a service or process user when impersonation is disabled.

Example 2: Drill Client to Drillbit Authentication with User Impersonation using MapR

```
drill.exec:{
                security: {
                   user.auth.enabled: true,
                   auth.mechanisms : ["MAPRSASL"],
                    }
                impersonation: {
                   enabled: true,
                   max_chained_user_hops: 3
              }
        }
```

Note: Drill executes all queries as the authenticated (ticket) user when impersonation is enabled. The client to Drillbit communication path will not be encrypted.

Example 3: Drill Client to Drillbit using Multiple Authentication Mechanisms

```
drill.exec:{
                 security: {
                     user.auth.enabled: true,
                     user.auth.impl: "pam4j",
                     security.user.auth.packages +=
"org.apache.drill.exec.rpc.user.security",
                     user.auth.pam_profiles: ["sudo", "login",
```

```
"mapr-admin"],
                     auth.mechanisms : ["MAPRSASL", "KERBEROS", "PLAIN"],
                     auth.principal : "mapr/_host@REALM.COM",
                     auth.keytab : "/opt/mapr/conf/mapr.keytab"
                   },
                  impersonation: {
                    enabled: true,
                    max_chained_user_hops: 3
              }

          }
```

Example 4: Drillbit to Drillbit Authentication using MapR Security

```
drill.exec:{
              security: {
                  auth.mechanisms : ["MAPRSASL"],
                  bit.auth.enabled : true
                  bit.auth.mechanism : "MAPRSASL"
       }
}
```

Example 5: Drill Client to Drillbit and Drillbit to Drillbit Authentication using MapR Security

```
drill.exec {
              security: {
                  user.auth.enabled: true,
                  auth.mechanisms : ["MAPRSASL"],
                  bit.auth.enabled : true,
                  bit.auth.mechanism : "MAPRSASL"
                     },
              impersonation: {
                   enabled: true,
                   max_chained_user_hops: 3
              }
            }
```

Configuring Encryption
An administrator can enable encryption with MapR Security (tickets).

Note: When the `sasl_encrypt` (for JDBC) or `EnforceSaslEncrypt` (for ODBC) connection parameter is set to `"true"` or `1`, the Drill client only accepts encrypted connections. If the client tries connecting to a Drillbit with encryption disabled, the connection fails.

Note: For client-side configuration, see Drill Drivers.

Set the encryption options to `"true"` in `/opt/mapr/drill/drill-<version>/conf/drill-override.conf`.

The following table lists the encryption configuration options with their descriptions and default values:

Note: If you installed Drill on a MapR cluster that was installed with the default MapR security configuration, the following options are set to "true" by default.

Option	Description	Default
drill.exec.security.user.encryption.sasl.enabled	Determines if encryption on the server is enabled for negotiating privacy with the Drill client.	false

drill.exec.security.bit.encryption.sasl.enabled	Determines if the server is enabled for negotiating privacy with another Drillbit.	false

The following sections provide configuration examples for Drill client to Drillbit encryption and Drillbit to Drillbit encryption.

Example 1: Drill Client to Drillbit Connection with MapR Security Authentication and Encryption
In the following server configuration, the Drill client connection to the Drillbit is encrypted using the MapR Security mechanism when the client is running with encryption support.

Note: Drill clients running Drill 1.10 and earlier cannot connect to the Drillbit through MapR Security with encryption enabled.

```
drill.exec {
            security: {
                user.auth.enabled: true,
                auth.mechanisms : ["MAPRSASL"]
                user.encryption.sasl.enabled : true
                    }
                }
```

Note: Drill executes all queries as a service or process user when impersonation is disabled.

Example 2: Drillbit to Drillbit Connection with MapR Security Authentication and Encryption
The following configuration authenticates and encrypts the path between Drillbits using the MapR Security mechanism.

```
drill.exec {
              security: {
                  auth.mechanisms : ["MAPRSASL"],
                  bit.auth.enabled : true
                  bit.auth.mechanisms : "MAPRSASL"
                  bit.encryption.sasl.enabled : true
        }
}
```

Example 3: Drill Client to Drillbit and Drillbit to Drillbit Connection with MapR Security Authentication and Encryption
The following configuration authenticates and encrypts the path between the Drill client and Drillbit, and between Drillbits using the MapR Security mechanism.

```
drill.exec {
              security: {
                 user.auth.enabled: true,
                 auth.mechanisms : ["MAPRSASL"],
                 user.encryption.sasl.enabled : true

                 bit.auth.enabled : true
                 bit.auth.mechanism : "MAPRSASL"
                 bit.encryption.sasl.enabled : true
                    }
                }
```

Note: Drill executes all queries as a service or process user when impersonation is disabled.

last-updated: Apr 11, 2022

Example 4: Drill Client to Drillbit and Drillbit to Drillbit Connection with MapR Security Authentication and Encryption and Impersonation Enabled
The following configuration authenticates and encrypts the path between the Drill client and Drillbit, and between Drillbits using the MapR Security mechanism.

```
drill.exec {
              security: {
                 user.auth.enabled: true,
                 auth.mechanisms : ["MAPRSASL"],
                 user.encryption.sasl.enabled : true

                 bit.auth.enabled : true
                 bit.auth.mechanism : "MAPRSASL"
                 bit.encryption.sasl.enabled : true
                    },
            impersonation: {
                   enabled: true,
                   max_chained_user_hops: 3
              }
          }
```

Note: Drill executes all queries as the authenticated (ticket) user when impersonation is enabled.

Example 5: Drill Client to Drillbit Authentication and Encryption Enabled using Multiple Mechanisms and Drillbit to Drillbit Authentication using MapR Security
The following configuration authenticates and encrypts the connection between the Drill client and Drillbit using multiple authentication mechanisms, and also authenticates and encrypts the connection between Drillbits using the MapR security mechanism.

Note: Plain authentication not supported in this configuration.

```
drill.exec {
              security: {
                  user.auth.enabled: true,
                  auth.mechanisms : ["MAPRSASL", "KERBEROS"],
                  auth.principal : "mapr/_host@REALM.COM",
                  auth.keytab : "/opt/mapr/conf/mapr.keytab",
                  user.encryption.sasl.enabled : true,
                  bit.auth.enabled : true,
                  bit.auth.mechanism : "MAPRSASL",
                  bit.encryption.sasl.enabled : true
                      }
              impersonation: {
                  enabled: true,
                  max_chained_user_hops: 3
              }
           }
```

Note: Drill executes all queries as a service or process user when impersonation is disabled.

Kerberos
Drill supports Kerberos v5 network security authentication and encryption. Kerberos is a network authentication protocol built on symmetric-key cryptography. Kerberos eliminates the need to store passwords locally or send them over the network and reduces the risk of impersonation.

Kerberos provides a security infrastructure called a Kerberos Realm. A Kerberos Realm is comprised of clients, services or hosts, and a KDC (key-distribution center). The KDC is a trusted third-party service that

generates tickets to coordinate authentication between a client and server or host. Tickets are cached on the client machine, which allows for single sign-on.

Clients use a password or a special file called a “keytab” to get tickets from the KDC. Clients exchange the tickets and secret keys with the KDC and service or host to prove their identity for access to the requested service. This authentication process of exchanging tickets and secret keys runs in the background, unseen by the user trying to access the service. When a client request to access a service is granted, a unique session key is established between the client and service. The unique session key proves the authenticity of the user. The session key is used for all communication between the client and service. Kerberos also supports encryption between the client and server to prevent data theft from a man-in-the-middle attack during communication.

A KDC administrator must create the password or keytab for the clients and servers, as well as a principal (a name for the user or server identity) to securely authenticate using the Kerberos infrastructure.

Note: Proper setup, configuration, administration, and usage of a Kerberos environment is beyond the scope of this documentation. See the MIT Kerberos documentation for more detailed information about Kerberos.

The following sections list the prerequisites for using Kerberos with Drill and describe the authentication process.

Prerequisites

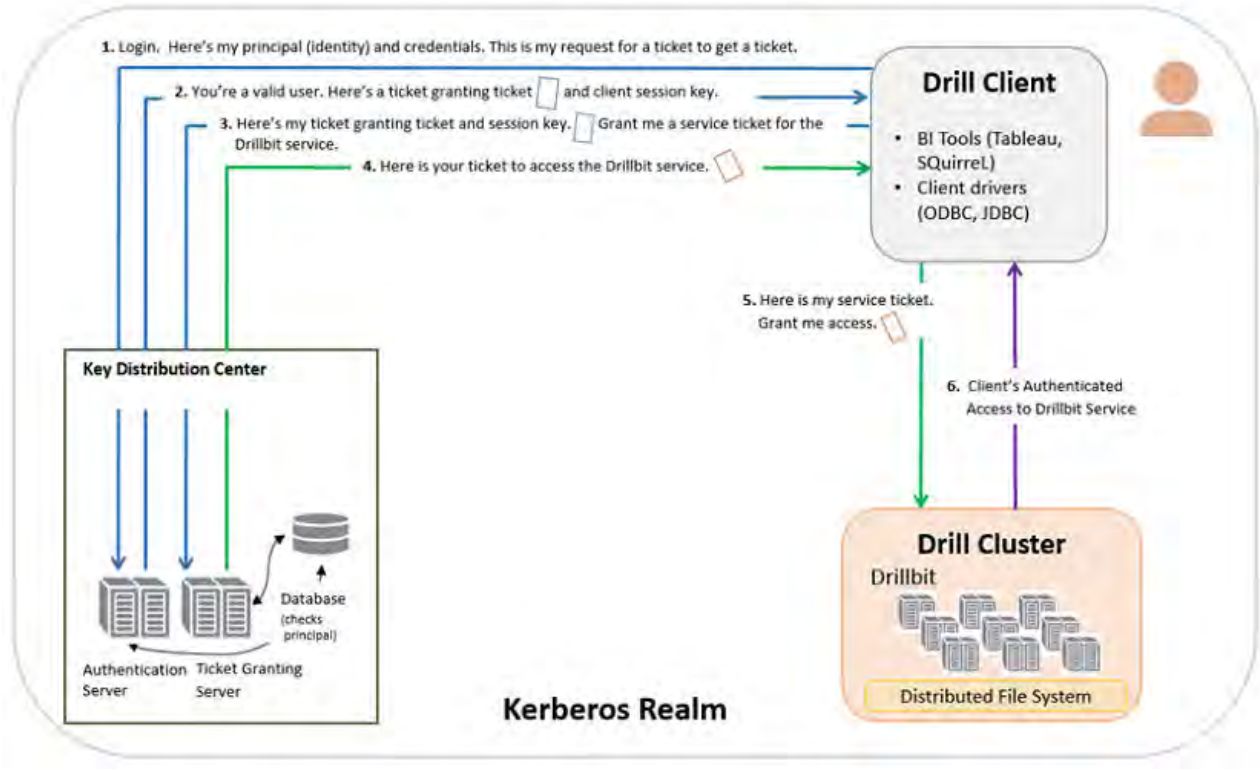
- The MapR Drill driver includes the required Kerberos plugin to authenticate to secure Kerberos Drill clusters. To use Kerberos with Drill, you must have a working Kerberos infrastructure, which Drill does not provide.
- Either a ticket granting ticket (TGT) is pre-generated on the client node, or a keytab file and the client principal is available to provide in the connection URL for Kerberos authentication between the Drill client and Drill server. Drill does not generate the TGT.
- You must be working in a Linux-based or Windows Active Directory (AD) Kerberos environment with secure clusters and have a Drill server configured for Kerberos.

Client Authentication Process

This section provides a high-level overview of the Kerberos client authentication process. For this overview, assume that Kerberos credentials are present in the client.

The following diagram shows the process of authenticating a client:

last-updated: Apr 11, 2022



1. The client sends a request for a ticket granting ticket that contains the user principal to the Kerberos KDC, a network service that supplies tickets and temporary session keys.
2. The authentication server validates the principal's identity and sends the client a ticket granting ticket and session key encrypted with a secret key. A session key is a temporary encryption key used for one login session.
3. Using the ticket granting ticket, the principal requests access to a Drillbit service from the ticket granting server.
4. The ticket granting server checks for a valid ticket granting ticket and the principal identity. If the request is valid, the ticket granting server returns a ticket granting service ticket.
5. The client uses the service ticket to request access to the Drillbit.
6. The Drillbit service has access to the keytab, a file that contains a list of keys for principals. The key allows the service to decrypt the client's ticket granting service ticket, identify the principal, and grant access.

Server Authentication Process

For Kerberos server authentication information, see the MIT Kerberos administration documentation.

Configuring Drill with Kerberos

The topics listed below provide configuration and connection information.

Configuring Authentication and Encryption
To enable authentication and encryption, you must create a Kerberos principal identity and a keytab file. You add the principal and keytab file to `<DRILLINSTALL_HOME>/conf/drill-override.conf` with the specified configuration parameters. In addition, you can configure a mapping from a Kerberos principal to a Drill user account. This mapping is used by a Drillbit to convert an authenticated client principal to

last-updated: Apr 11, 2022

a corresponding Kerberos short name, which is used to determine administrator privileges for the client principal. After you complete the configuration steps, restart the Drillbit.

To enable authentication and encyption using the Kerberos mechanism, configure the following Kerberos-specific parameters in `drill-override.conf`:

Note: Only Drill 1.11 and later supports encryption.

Note: For client-side configuration, see Drill Drivers.

Parameters	Communication Path	Description	Default
drill.exec.security.auth.principal	Drill client (ODBC/ JDBC) to Drillbit	String representation of the Kerberos principal used by the Drillbit service.	N/A
drill.exec.security.auth.keytab	Drill client (ODBC/ JDBC) to Drillbit	Location of the keytab file for the configured Drillbit service principal. **Note:** The Kerberos keytab file that contains the encrypted key for the Drillbit service principal. The file should be readable by the Drillbit process user.	N/A
drill.exec.security.auth.auth_to_local	Drill client (ODBC/ JDBC) to Drillbit	Custom rules to convert the Kerberos principal to the Kerberos short name. **Note:** Drill uses a Hadoop Kerberos name and rules to transform the Kerberos principal provided by client to the one it will use internally as the client's identity. This client identity is used to determine administrator privileges. See Mapping from Kerberos Principal to OS user account in the Hadoop in Secure Mode documentation for details about how the rule works.	The primary name of the Kerberos principal. By default, this mapping rule extracts the first part from the provided principal. For example, if the principal format is `<Name1>/ <Name2>@realm`, the default rule extracts only `Name1` from the principal and `Name1` as the client's identity on server side.
drill.exec.security.user.encryption.sasl.enabled	Drill client (ODBC/ JDBC) to Drillbit	Enables/disables encryption for the communication path between the Drill client and Drillbit.	false
drill.exec.security.bit.auth.use_login_principal	Drilbit to Drillbit	When set to true, the Drillbit uses the same logged in service principal configured with drill.exec.security.auth.principal for the Drillbit to Drillbit communication paths. When this parameter is set to false, a principal is constructed using the hostname from ZooKeeper for the remote Drillbit and keeping the primary and realm information the same as the logged in principal set by drill.exec.security.auth.principal .	false

last-updated: Apr 11, 2022

Parameters	Communication Path	Description	Default
drill.exec.security.bit.encryption.sasl.enabled	Drillbit to Drillbit	Enables/disables encryption for the communication path between the Drillbits.	false

Steps to Enable Kerberos Authentication and Encryption

Complete the following steps to enable Drill to use Kerberos for authentication and encryption:

1. Create a Kerberos principal identity and a keytab file. You can create one principal for each Drillbit or one principal for all Drillbits in a cluster.

   Note: The adminstrator must own the `drill.keytab` file and have the ability to read the file.

   - For a single principal per node in cluster:

     ```
     # kadmin
     : addprinc -randkey <username>/<FQDN>@<REALM>.COM
     : ktadd -k /opt/mapr/conf/drill.keytab <username>/
     <FQDN>@<REALM>.COM
     ```

   - For a single principal per cluster, use `<clustername>` instead of `<FQDN>`:

     ```
     # kadmin
     : addprinc -randkey <username>/<clustername>@<REALM>.COM
     : ktadd -k /opt/mapr/conf/drill.keytab <username>/
     <clustername>@<REALM>.COM
     ```

   Note: The instance name must be lowercase. If _HOST is set as the instance name in the principal, it is replaced with the fully qualified domain name of that host for the instance name. For example, if a Drillbit running on `host01.aws.lab` uses `drill/_HOST@<EXAMPLE>.COM` as the principal, the canonicalized principal is `drill/host01.aws.lab@<EXAMPLE>.COM`.

2. Add the Kerberos principal identity, keytab file, and parameters specific to Kerberos to the `drill-override.conf` file. You can use the following configuration examples for enabling authentication, encryption, or both between the Drill client and Drillbit and between Drillbits.

Example 1: Enabling Kerberos Authentication Between the Drill Client and Drillbit

```
      drill.exec: {
        cluster-id: "drillbits1",
        zk.connect:
"qa102-81.qa.lab:5181,qa102-82.qa.la
b:5181,qa102-83.qa.lab:5181",
        impersonation: {
          enabled: true,
          max_chained_user_hops: 3
        },
        security: {

user.auth.enabled:true,
                auth.mechanisms:
["KERBEROS"],

auth.principal:"drill/
<clustername>@<REALM>.COM",
                auth.keytab:"/etc/
drill/conf/drill.keytab"
```


last-updated: Apr 11, 2022

```
            }
        }
```

Example 2: Enabling Kerberos Authentication and Encryption Between the Drill Client and Drillbit

```
      drill.exec: {
        cluster-id: "drillbits1",
        zk.connect:
"qa102-81.qa.lab:5181,qa102-82.qa.la
b:5181,qa102-83.qa.lab:5181",
        impersonation: {
          enabled: true,
          max_chained_user_hops: 3
        },
        security: {

user.auth.enabled:true,
                auth.mechanisms:
["KERBEROS"],

auth.principal:"drill/
<clustername>@<REALM>.COM",
                auth.keytab:"/etc/
drill/conf/drill.keytab",

user.encryption.sasl.enabled: true
              }
            }
        }
```

Example 3: Enabling Kerberos Authentication Between Drill Client and Drillbits and Between Drillbits

```
      drill.exec: {
        cluster-id: "drillbits1",
        zk.connect:
"qa102-81.qa.lab:5181,qa102-82.qa.la
b:5181,qa102-83.qa.lab:5181",
        impersonation: {
          enabled: true,
          max_chained_user_hops: 3
        },
        security: {

user.auth.enabled:true,
                auth.mechanisms:
["KERBEROS"],

auth.principal:"drill/
<clustername>@<REALM>.COM",
                auth.keytab:"/etc/
drill/conf/drill.keytab"
              }
        security.bit: {
                auth.enabled: true,
                auth.mechanism:
"Kerberos",

auth.use_login_principal: true
               }
        }
```

Example 4: Enabling Kerberos Authentication and Encryption Between Drill Client and Drillbits and Between Drillbits

```
      drill.exec: {
        cluster-id: "drillbits1",
        zk.connect:
"qa102-81.qa.lab:5181,qa102-82.qa.la
b:5181,qa102-83.qa.lab:5181",
        impersonation: {
          enabled: true,
          max_chained_user_hops: 3
        },
        security: {

user.auth.enabled:true,
                auth.mechanisms:
["KERBEROS"],

auth.principal:"drill/
<clustername>@<REALM>.COM",
                auth.keytab:"/etc/
drill/conf/drill.keytab",

user.encryption.sasl.enabled: true
              }
        security.bit: {
                auth.enabled: true,
                auth.mechanism:
"Kerberos",

auth.use_login_principal: true,

encryption.sasl.enabled: true
              }
        }
```

Note: In examples 3 and 4 above, the Drillbit will use the same logged in service principal as configured in `drill.exec.security.auth.principal`.

Example 5: Enabling Kerberos Authentication and Encryption Between Drill Client and Drillbits and Between Drillbits. For Drillbit to Drillbit authentication, where the service principal is created using the hostname from ZooKeeper for a remote Drillbit as an instance name. The primary and the realm component of the service principal is used from the drill.exec.security.auth.principal parameter.

```
      drill.exec: {
        cluster-id: "drillbits1",
        zk.connect:
"qa102-81.qa.lab:5181,qa102-82.qa.la
b:5181,qa102-83.qa.lab:5181",
        impersonation: {
          enabled: true,
          max_chained_user_hops: 3
        },
        security: {

user.auth.enabled:true,
                auth.mechanisms:
["KERBEROS"],

auth.principal:"drill/
<clustername>@<REALM>.COM",
                auth.keytab:"/etc/
drill/conf/drill.keytab",

user.encryption.sasl.enabled: true
```


last-updated: Apr 11, 2022

```
                }
        security.bit: {
                auth.enabled: true,
                auth.mechanism:
"Kerberos",

encryption.sasl.enabled: true
                }
        }
```

Note: For the configuration in example 5, if the hostname of the remote Drillbit known to ZooKeeper is **host01.aws.lab**, then the service principal used by a Drillbit to authenticate with the remote Drillbit will be **drill/host01.aws.lab@<REALM>.COM**.

3. Restart the Drillbit process on each Drill node.

```
$ maprcli node services -name drill-bits -action restart -nodes <node
host names separated by a space>
```

Related concepts

Plain Authentication on page 3309
An administrator can configure Drill to use the Linux pluggable authentication module (PAM) for Plain (username and password) authentication. PAM provides an authentication module that interfaces with any installed PAM authentication entity, such as the local operating system password file (/etc/passwd) or LDAP.

Configuring Drill to Use Kerberos with Hive Metastore

To configure Drill to use Kerberos with the Hive metastore, modify the hive storage plugin in the Drill Web UI and then restart the Warden service.

Note: When you configure Drill to use Kerberos with the Hive metastore, Drill submits requests to the Hive metastore as the `mapr` superuser. If you want Drill to submit requests to the Hive metastore as any other user, configure Drill impersonation with Hive instead of performing this task. Drill impersonation works with or without Kerberos configured for the Hive metastore.

Prerequisites

The configurations described in this document have the following dependencies:

- MapR cluster.
- Drill installed with Drillbits running as the `mapr` user.
- Supported version of Hive installed with the following:
  - Hive Metastore configured to use Kerberos authentication
  - Configured Hive remote metastore repository

Note: See the Drill Support Matrix on page 5629 for supported versions of Hive.

Modify the Hive Storage Plugin in Drill

Modify the Hive storage plugin configuration in the Drill Web UI based on the authorization and security scenario for the cluster. You can only access the Drill Web UI for a running Drillbit.

last-updated: Apr 11, 2022

Complete the following steps to configure Drill to use Kerberos with Hive Metastore:

1. Navigate to `http://<drillbit_hostname>:8047`, and select the **Storage** tab.

   Note: You can only access the Drill Web UI for a running Drillbit.

2. Click **Update** next to the hive option.

3. In the configuration window, add the `hive.metastore.sasl.enabled`, `hive.metastore.kerberos.principal`, and `hive.security.authorization.enabled` properties, as shown below, if configuration does not contain them already. Note that other properties shown may or may not be required in your environment:

```
{
  "type": "hive",
  "enabled": true,
  "configProps": {
    "hive.metastore.uris": "thrift://<metastore_hostname>:9083",
    "fs.default.name": "maprfs:///",
    "hive.server2.enable.doAs": "false",
    "hive.metastore.sasl.enabled": "true",
    "hive.metastore.kerberos.principal":
"<metastore_server_principal_name>",
    "hive.security.authorization.enabled": "true"
  }
}
```

Restart Warden

Issue the following command on all nodes to restart the Warden service:

```
service mapr-warden restart
```

If you have `clush` installed, you can run the following command to restart Warden on all nodes at once:

```
clush -a "service mapr-warden restart"
```

Connection URLs for Kerberos using JDBC Drivers to connect via SQLLine
You can use client-side connection URL parameters for Kerberos authentication in multiple combinations to authenticate a client with Drill.

Client Credentials

A client can provide its credentials in two ways:

- With a ticket granting ticket (TGT) generated on client side. The TGT must be present on client node; Drill does not generate the TGT.
- With a keytab file and the client principal provided in the user property of the connection URL.

Configuration Options

The following table lists configuration options for connection URLs. See the Connection URL Examples section for sample URLs.

last-updated: Apr 11, 2022

Connection Parameter (Apache Drill JDBC Driver)	Description	Connection Parameter (Drill JDBC Driver)	Mandatory/Optional	Default Value
auth	Authentication deduced if not provided. A Drill client can also explicitly specify a particular authentication mechanism to use using this parameter. For example, for Kerberos along with service_name, service_host or principal and for the Plain authentication with username and password.	Auth	Optional	The preference order is Kerberos and Plain.
principal	Drillbit service principal. Required if the principal is not provided using this configuration. For Kerberos, the Drill service principal is derived if the value is not provided. The service_name (primary) and service_host (instance) are used to generate a valid principal. Since the ticket or keytab contains the realm information, the realm is optional.	Principal	Optional	

keytab	For Kerberos, if you authenticate using a keytab, set the keytab parameter to the location of the keytab file. This is passed through the user parameter. A Kerberos ticket is used as the default credential (It is assumed to be present on the client-side. The Drill client does not generate the required credentials.)	If the client chooses to use a keytab rather than a ticket, the client principal must be provided	Optional	
sasl_encrypt	When set to true, ensures that a client connects to a server with encryption capabilities. For example, Drill 1.11 Drillbits, which support client-to-drillbit encryption.		Optional	FALSE
service_name	Primary name of the Drillbit service principal.		Optional	drill
service_host	Instance name of the Drillbit service principal.		Optional	Since this value is usually the hostname of the node where a Drillbit is running, the default value is the Drillbit hostname is provided either through ZooKeeper or through a direct connection string.
realm	Kerberos realm of the principal. The realm name for the Drillbit service ticket or keytab contains the realm information.		Optional	

Client Encryption

A client can specify that it requires a server with encryption capabilities only by setting the

`sasl_encrypt` connection parameter to "true." If the cluster to which client is connecting has encryption disabled, the client will fail to connect to that server.

```
drill.exec {
 security:  {
      user.auth.enabled: true,
      auth.mechanisms: ["KERBEROS"],
      auth.principal: "drill/serverhostname@REALM.COM",
      auth.keytab: "/etc/drill/conf/drill.keytab",
      user.encryption.sasl.enabled: true
          }
}
```

Connection URL Examples

The following five examples show the JDBC connection URL that the embedded JDBC client uses for Kerberos authentication. The first section, Example of a Simple Connection URL, includes a simple connection string and the second section, Examples of Connection URLs Used with Previously Generated TGTs, includes examples to use with previously generated TGTs.

Example of a Simple Connection URL

Example 1: TGT for Client Credentials

The simplest way to connect using Kerberos is to generate a TGT on the client side. Only specify the service principal in the JDBC connection string for the Drillbit the user wants to connect to.

```
jdbc:drill:drillbit=10.10.10.10;principal=<principal for host 10.10.10.10>
```

In this example, the Drill client uses the:

- Default `service_name`, which is **`drill`**.
- `service_host` from the Drillbit name provided in the connection URL, which is **`10.10.10.10`**.

The service principal format is `<primary>/<instance>@<realm from TGT>`. The service principal is **`principal for host 10.10.10.10`**.

Examples of Connection URLs Used with Previously Generated TGTs

If you do not provide a service principal in the connection string when using Kerberos authentication, then use the `service_name` or `service_host` parameters. Since these parameters are optional, their default values will be used internally (if not provided) to create a valid principal.

Examples 2 through 4 show a valid connection string for Kerberos authentication if a client has previously generated a TGT. Realm information will be extracted from the TGT if it is not provided.

Note: For end-to-end authentication to function, it is assumed that the proper principal for the Drillbit service is configured in the KDC.

Example 2: Drillbit Provided by Direct Connection String and Configured with a Unique Service Principal

This type of connection string is used when:

- Each Drillbit in the cluster is configured with its own service principal.
- The instance component is the host address of the Drillbit.

  ```
  jdbc:drill:drillbit=host1;auth=kerberos
  ```

In this example, the Drill client uses the:

- Default `service_name`, which is **`drill`**.

- `service_host`, which is the Drillbit name provided in the connection URL (**`host1`**).

The internally created service principal will be **`drill/host1@<realm from TGT>`**.

Example 3: Drillbit Selected by ZooKeeper and Configured with Unique Service Principal

This type of connection string is used when the Drillbit is chosen by ZooKeeper instead of directly from the connection string.

```
jdbc:drill:zk=host01.aws.lab:5181;auth=kerberos;service_name=myDrill
```

In this example, the Drill client uses the:

- Provided `service_name`, which is **`myDrill`** as the primary name of the principal.
- `service_host` as the address of the Drillbit, which is chosen from the list of active drillbits that ZooKeeper provides (**`host01.aws.lab:5181`**).

The internally created service principal will be **`myDrill/<host address from zk>@<realm from TGT>`**.

Example 4: Drillbit Selected by Zookeeper and Configured with a Common Service Principal

This type of connection string is used when all Dillbits in a cluster use the same principal.

```
jdbc:drill:zk=host01.aws.lab:5181;auth=kerberos;service_name=myDrill;service
_host=myDrillCluster
```

In this example, the Drill client uses the:

- Provided `service_name`, which is **`myDrill`**.
- `service_host`, which is **`myDrillCluster`**.

The internally created service principal, which will be **`myDrill/myDrillCluster@<realm from TGT>`**.

Example 5: Keytab for Client Credentials

If a client chooses to provide its credentials in a keytab instead of a TGT, it must also provide a principal in the user parameter. In this case, realm information will be extracted from the `/etc/krb5.conf` file on the node if it is not provided in the connection URL. All other parameters can be used as shown in the preceding examples (1-4). This connection string is for the case when all Drillbits in a cluster use the same principal.

```
jdbc:drill:zk=host01.aws.lab:5181;auth=kerberos;service_name=myDrill;service
_host=myDrillCluster;keytab=<path to keytab file>;user=<client principal>
```

In this example, the Drill client:

- Will authenticate itself with the:
  - Keytab (**`path to keytab file`**) and
  - Principal provided in the user parameter (**`client principal`**)
- Uses the:
  - Provided `service_name`, which is **`myDrill`**.
  - `service_host`, which is **`myDrillCluster`**.

The internally created service principal will be **`myDrill/myDrillCluster@<realm from krb5.conf>`**.

Plain Authentication

An administrator can configure Drill to use the Linux pluggable authentication module (PAM) for Plain (username and password) authentication. PAM provides an authentication module that interfaces with any installed PAM authentication entity, such as the local operating system password file (/etc/passwd) or LDAP.

Note: Starting in EEP 5.0, Drill supports form-based authentication between the web client and Drillbit. Form-based authentication is like Plain authentication in that a user is presented with a web form where s/he enters a username and password to access restricted web pages. Configuring Drill to Use libpam4j includes configuration details. When using form-based authentication, you can also configure Drill to use SPNEGO. See SPNEGO for HTTP Authentication.

When using PAM for authentication, each user with permission to run Drill queries must exist in the list of users that resides on each Drill node in the cluster. The username (including uid) and password for each user must be identical across all Drill nodes.

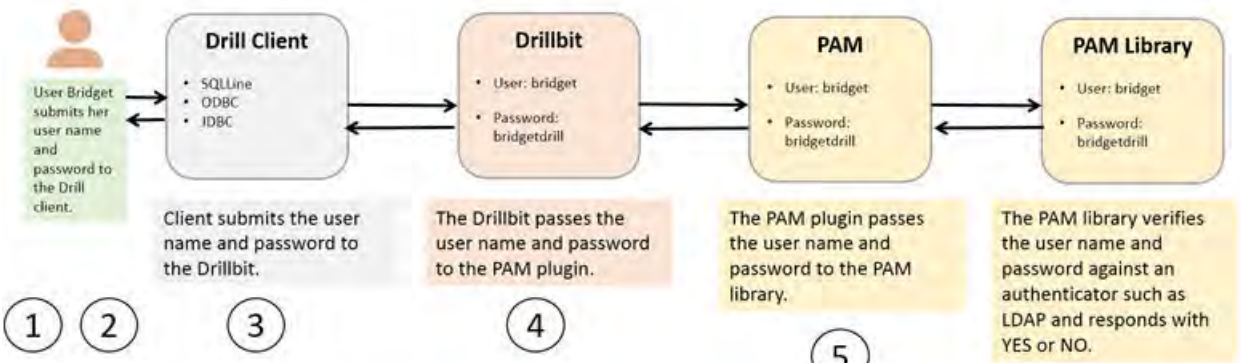
If you use PAM with /etc/passwd for authentication, verify that the users permitted to start the Drill process are part of the shadow user group on all nodes in the cluster. This enables Drill to read the /etc/shadow file for authentication.

Note: Plain authentication does not support SASL encryption. You can use SSL/TLS for encryption when Plain authentication is enabled. You can also enable user impersonation and create views to limit user access to data.

Authentication Process Overview

During the authentication process, the client passes a username and password to the Drillbit as part of the connection request, which then passes the credentials to PAM. If PAM authenticates the user, the connection request passes the authentication phase, and the connection is established. The user will be authorized to access Drill and issue queries against the filesystem or other storage plugins, such as Hive.

The following image illustrates the PAM user authentication process in Drill:



Plain (Username and Password) Authentication Process

If PAM cannot authenticate the user, the connection request does not pass the authentication phase and the user will not be authorized to access Drill. The connection is terminated as AUTH_FAILED.

For more PAM information (including a JPAM User Guide), see JPAM.

Configuring Plain Authentication in Drill

Drill supports the libjpam and libpam4j libraries. In Drill 1.12 and later, the libpam4j library is packaged with Drill. There is no download or external dependency required to use libpam4j. Using libpam4j is recommended for Drill 1.12 and later.

last-updated: Apr 11, 2022

Note: You can configure Drill to use multiple types of authenication mechanisms. For example, you can configure Drill to use Plain, Kerberos, and MapR-SASL; however, only SSL/TLS is supported for encryption when Plain authentication is configured with other authentication mechanisms.

The following sections provide information for configuring Drill to use libpam4j or libjpam, as well as instructions for connecting to Drill from SQLLine and BI tools when Plain authentication is enabled:

Configuring Drill to Use libpam4j
You can configure Drill to use libpam4j for Plain authentication between a client, such as ODBC, and the Drillbit.

Starting in EEP 5.0, you can configure Drill to use libpam4j for form-based authentication between a web client and Drillbit (web server). Form-based authentication is like Plain authentication in that a user is presented with a web form where s/he enters a username and password to access restricted web pages. When using form-based authentication, you can also configure Drill to use SPNEGO. See SPNEGO for HTTP Authentication.

Complete the following steps to configure Plain authentication (for JDBC/ODBC clients) and form-based authentication (for the web client) in Drill:

1. Add the following configurations to the `/opt/mapr/drill/drill-<version>/conf/drill-override.conf` file:

```
drill.exec:{
   cluster-id:"drillbits1",

zk.connect:"<zk-node-hostname>:5181,<zk-node-hostname>:5181,<zk-node-host
name>:5181",
   security:{
      auth.mechanisms:[ "PLAIN"],
   },
   security.user.auth:{
      enabled:true,
      packages += "org.apache.drill.exec.rpc.user.security",
      impl:"pam4j",
      pam_profiles:[ "sudo", "login" ]
   },
   http.auth.mechanisms:[ "FORM" ]
}
```

2. (Optional) To add or remove different PAM profiles, add or delete the profile names in the `pam_profiles` array portion of the configuration:

```
pam_profiles: [ "sudo", "login" ]
```

3. Restart the Drillbit process on each Drill node, as shown:

```
/opt/mapr/drill/drill-<version>/bin/drillbit.sh restart
```

Configuring Drill to Use libjpam
You can configure Drill to use libjpam for Plain authentication between a client, such as ODBC, and the Drillbit.

To configure Drill to use libjpam, complete the following steps:

1. Copy the libjpam.so file from /opt/mapr/lib to a directory that does not contain other Hadoop components, for example /opt/pam/.

last-updated: Apr 11, 2022

2. Add the following line to /opt/mapr/drill/drill-<version>/conf/drill-env.sh, including the directory where the libjpam.so file is located, as shown:

```
export
DRILLBIT_JAVA_OPTS="$DRILLBIT_JAVA_OPTS -Djava.library.path=<directory>"
Example: export
DRILLBIT_JAVA_OPTS="$DRILLBIT_JAVA_OPTS -Djava.library.path=/opt/pam/"
```

3. Add the following configuration to the drill.exec block in /opt/mapr/drill/drill-<version>/conf/drill-override.conf:

```
drill.exec: {
             cluster-id: "drillbits1",
             zk.connect:
"qa102-81.qa.lab:5181,qa102-82.qa.lab:5181,qa102-83.qa.lab:5181",
             impersonation: {
               enabled: true,
               max_chained_user_hops: 3
          },
             security: {
                auth.mechanisms : ["PLAIN"],
                 },
              security.user.auth: {
                enabled: true,
                packages += "org.apache.drill.exec.rpc.user.security",
                impl: "pam",
                pam_profiles: [ "sudo", "login", "mapr-admin" ]
        }
       }
```

4. (Optional) To add or remove different PAM profiles, add or delete the profile names in the pam_profiles array portion of the configuration:

```
pam_profiles: [ "sudo", "login" ]
```

5. Restart the Drillbit process on each Drill node, as shown:

```
/opt/mapr/drill/drill-<version>/bin/drillbit.sh restart
```

Connection URL for Plain Authentication using the Apache JDBC Driver to connect via SQLLine
When Plain authentication is enabled, each user that accesses the Drillbit process through a client, must provide username and password credentials for access.

Connecting to Drill from SQLLine

Include the `-n` and `-p` parameters with your username and password when launching SQLLine, as shown in the following example:

```
sqlline -u jdbc:drill:zk=10.10.11.112:5181 -n <username> -p <password>
```

Alternatively, you can launch SQLLine and then issue the `!connect` command to hide the password.

Complete the following steps to hide the password:

1. Run the sqlline script, as shown:

```
$ /etc/drill/bin/sqlline
```

2. At the prompt, enter the `!connect` command followed by

```
jdbc:drill:zk=zk=<zk name>[:<port>][,<zk name2>[:<port>]...]
```

, as shown:

```
`sqlline> !connect jdbc:drill:zk=localhost:5181 scan complete in 1385m`s
```

3. When prompted, enter a username and password; the password is hidden as it is typed, as shown:

```
Enter username for jdbc:drill:zk=localhost:5181: yourusername
Enter password for jdbc:drill:zk=localhost:5181: *************
```

Connecting to Drill from BI Tools

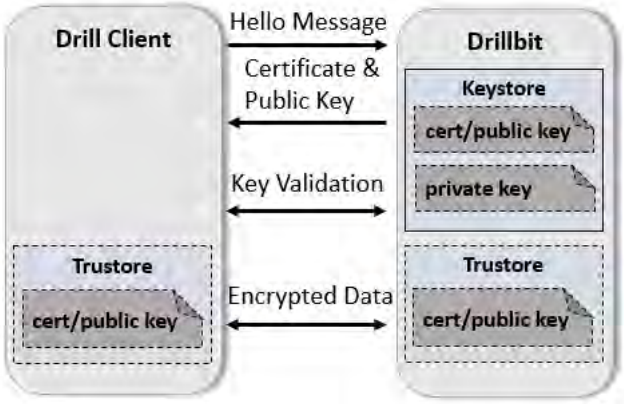
When you connect to Drill from a BI tool, such as Tableau, the ODBC driver prompts you for the authentication type, username, and password. For PAM, select **Basic Authentication** in the Authentication Type drop-down menu.

SSL/TLS for Encryption

You can enable SSL for Drill in a secure or unsecure cluster. SSL (Secure Sockets Layer), more recently called TLS, is a security mechanism that encrypts data passed between the Drill client and Drillbit (server). SSL also provides one-way authentication through which the Drill client verifies the identity of the Drillbit.

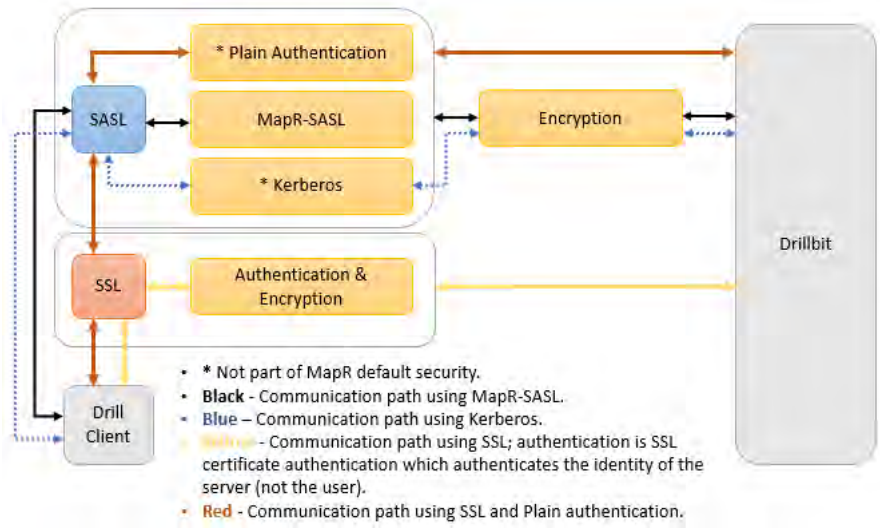
Authentication occurs during the SSL handshake when the Drillbit (server) presents its certificate to the client, and the client checks if the certificate exists in its truststore or if the certificate is signed by a trusted CA (Certificate Authority) that exists in its truststore.

The following diagram depicts the communication between the Drill client and the Drillbit (server):



The SASL feature in Drill provides authentication and an option to encrypt data, however the encryption feature is not available when using Plain authentication. If you need to use Plain authentication (certain BI tools only use Plain authentication), you can enable SSL to encrypt data. Using SSL and SASL encryption together is strongly discouraged.

The following diagram depicts the SSL communication paths between the Drill client and Drillbit (server), including the scenario where Plain authentication is used:



Note: The REST API supports HTTPS. SSL is not supported for communication between Drillbits.

The following sections provide information about how to use certificates in secure and unsecure clusters, enabling and configuring SSL, connection parameters, and common SSL issues.

Related concepts

SSL Certificates on page 689
Describes how certificates are used to perform authentication and encryption for websites that use the HTTPS protocol.

SSL Certificates in Secure and Insecure Clusters

The Drill server requires an SSL certificate. The certificate can be self-signed or signed by a CA (Certificate Authority).

The sections below describe how to use SSL certificates in secure and insecure MapR Data Platform clusters.

SSL in a Secure Cluster

By default, SSL is configured in a secure MapR cluster, but not enabled. In a secure cluster the keystore is configured for you. The security in a MapR cluster uses a self-signed certificate. If you have a certificate signed by a certificate authority, follow the instructions for Importing a Certificate Authority Signed (CA Signed) SSL Certificate Into a MapR Cluster and then enable and configure SSL.

To use SSL, enable the SSL option and then modify any of the available configuration options as needed.

- To enable SSL for the ODBC/JDBC client to Drillbit communication path, you must enable SSL on the client side and Drillbit. See Drill Drivers for client instructions. See Configuring SSL/TLS for the Drillbit.
- To enable SSL for the Drill Web UI, see Configuring the Drill Web UI and Web API Security.

After you modify the configuration options, restart Drill, as shown:

```
$ maprcli node services -name drill-bits -action restart -nodes <node host
names separated by a space>
```

SSL in an Insecure Cluster

Before you can enable SSL in an insecure cluster, you must either get or generate a certificate and then import the certificate into the Java keystore. You can do this using the Java keytool utility. See To Use keytool to Create a Server Certificate for instructions.

last-updated: Apr 11, 2022

If you have a custom certificate, you can import it using the method described in Importing a Certificate Authority Signed (CA Signed) SSL Certificate Into a MapR Cluster. You may also want to reference this document.

After you generate or import a server certificate, add the path (and password) to the keystore in the SSL configuration for Drill. See Configuring SSL for information on how to update the SSL configuration.

Restart Drill after you modify the configuration options, as shown:

```
$ maprcli node services -name drill-bits -action restart -nodes <node host
names separated by a space>
```

Configuring SSL/TLS
Enable SSL in `<DRILL_INSTALL_HOME>/conf/drill-override.conf`. You can use several configuration options to customize SSL/TLS.

You must restart the Drillbit process on each node after you modify the configuration options, as shown:

```
$ maprcli node services -name drill-bits -action restart -nodes <node host
names separated by a space>
```

The following sections provide information and instructions for enabling and configuring SSL:

Enabling SSL

When SSL is enabled, all Drill clients, such as JDBC and ODBC, must connect to Drill servers using SSL. Enable SSL in the Drill startup configuration file, `drill-override.conf`, located in `/opt/mapr/drill/drill-<version>/conf`.

To enable SSL for Drill, set the `drill.exec.security.user.encryption.ssl.enabled` option in `drill-override.conf` to `"true."`

Configuring SSL

You can customize SSL on a Drillbit through the SSL configuration options. You can set the options from the command-line (using Java system properties), in the `drill-override.conf` file, or in the property file to which the Hadoop parameter `hadoop.ssl.server.conf` points (recommended).

Note: Specifying values in `drill-override.conf` can expose the security parameters to end users. Administrators should set these values in the Hadoop security file and restrict permissions on that file.

If a parameter is specified in multiple places, the value in the Hadoop configuration takes precedence over the Drill configuration, which takes precedence over the system property.

The Hadoop configuration is specified in the file pointed to by the `hadoop.ssl.server.conf` parameter in the Hadoop `core-site.xml` file. Typically, this parameter points to `$HADOOP_CONF/ssl-server.xml`, which contains the property names to configure SSL. Both the `core-site.xml` file and the `ssl-server.xml` file must exist in Drill's classpath. Drill's SSL configuration picks up the Hadoop SSL configuration.

Note: Since the Drillbit implementation is based on JSSE, several standard parameters that apply to JSSE will also apply to the Drillbit. However, you typically do not need to configure JSSE parameters.

The following are the SSL configuration options with their descriptions and default values:

drill.exec.security.user.encryption.ssl.enabled	*Hadoop Property Name:* N/A *System Property Name:* N/A

last-updated: Apr 11, 2022

	Description: Enable or disable TLS for Drill client - Drill Server communication. You must set this option in `drill-override.conf`. *Allowed Values:* `true` or `false` *Drill Default:* `false`
drill.exec.ssl.protocol	*Hadoop Property Name:* N/A *System Property Name:* N/A *Description:* The version of the TLS protocol to use. *Allowed Values:* TLS, TLSV1, TLSv1.1, TLSv1.2 *Drill Default:* TLSv1.2 (recommended)
drill.exec.ssl.keyStoreType	*Hadoop Property Name:* ssl.server.keystore.type *System Property Name:* javax.net.ssl.keyStoreType *Description:* Format of the keystore file *Allowed Values:* jks, jceks, pkcs12 *Drill Default:* jks
drill.exec.ssl.keyStorePath	*Hadoop Property Name:* ssl.server.keystore.location *System Property Name:* javax.net.ssl.keyStore *Description:* Location of the Java keystore file containing the Drillbit's own certificate and private key. On Windows, the specified pathname must use forward slashes,`/`, in place of backslashes. *Allowed Values:* Not Applicable *Drill Default:* Not Applicable
drill.exec.ssl.keyStorePassword	*Hadoop Property Name:* ssl.server.keystore.password *System Property Name:* javax.net.ssl.keyStorePassword *Description:* Password to access the private key from the keystore file. This password is used twice: To unlock the keystore file (store password), and to decrypt the private key stored in the keystore (key password) unless a key password is specified separately. *Allowed Values:* Not Applicable *Drill Default:* Not Applicable
drill.exec.ssl.keyPassword	*Hadoop Property Name:* ssl.server.keystore.keypassword *System Property Name:* Not Applicable *Description:* Password to access the private key from the keystore file. May be different from the keystore password. *Allowed Values:* Not Applicable *Drill Default:* Not Applicable
drill.exec.ssl.trustStoreType	*Hadoop Property Name:* ssl.server.truststore.type *System Property Name:* javax.net.ssl.trustStoreType *Description:* Format of the truststore file *Allowed Values:* jks, jceks, pkcs12 *Drill Default:* jks
drill.exec.ssl.trustStorePath	*Hadoop Property Name:* ssl.server.truststore.location

	System Property Name: javax.net.ssl.trustStore *Description:* Location of the Java keystore file containing the collection of CA certificates trusted by the Drill client. On Windows, the specified pathname must use forward slashes, `/`, in place of backslashes. **Note:** If the `trustStorePath` is not provided, Drill ignores the `trustStorePassword` parameter and gets the default Java truststore instead, which causes issues if the Java truststore has a non-default password. The Java APIs to load the default keystore assume the default password. The only way to use the default keystore with a non-default password is to specify both the path and the password to the keystore. To work around this issue, pass the default Java truststore to the `trustStorePath` parameter. *Allowed Values:* Not Applicable *Drill Default:* Not Applicable
drill.exec.ssl.trustStorePassword	*Hadoop Property Name:* ssl.server.truststore.password *System Property Name:* javax.net.ssl.trustStorePassword *Description:* Password to access the private key from the keystore file specified as the truststore. *Allowed Values:* Not Applicable *Drill Default:* Not Applicable
drill.exec.ssl.provider	*Hadoop Property Name:* Not Applicable *System Property Name:* Not Applicable *Description:* Changes the underlying implementation to the chosen value. *Allowed Values:* OpenSSL or JDK *Drill Default:* JDK
drill.exec.ssl.useHadoopConfig	*Hadoop Property Name:* Not Applicable *System Property Name:* Not Applicable *Description:* Use the setting in the Hadoop configuration file. The Hadoop configuration is specified in the file pointed to by the `hadoop.ssl.server.conf` parameter in the `core-site.xml` file. Typically, this parameter points to `$HADOOP_CONF/ssl-server.xml`, which contains the property names to configure TLS. *Allowed Values:* `true` or `false` *Drill Default:* `true`

JDBC Connection Parameters

Use the SSL JDBC connection parameters and fully qualified host name to configure the jdbc connection string in SQLLine and connect to Drill.

The following table lists the parameters that you can include in the `jdbc` connection string using SQLLine:

Note: Examples are provided after the table. For additional instructions, see the Drill JDBC Driver documentation.

last-updated: Apr 11, 2022

Parameter	Value	Required
enableTLS	true/false	[Optional] If true, TLS is enabled. If not set or set to false, TLS is not enabled.
trustStoreType	string	[Optional] Default: JKS The trustStore type. Allowed values are : JKS PKCS12 If the useSystemTrustStore option is set to true (on Windows only), the allowed values are: Windows-MY Windows-ROOT Import the certificate into the "Trusted Root Certificate Authorities" and set trustStoreType=Windows-ROOT. Also import the certificate into "Trusted Root Certificate Authorities" or "Personal" and set trustStoreType=Windows-MY.
trustStorePath	string	[Optional] Path to the truststore. If not provided the default Java truststore will be used. If this is not provided the trustStorePassword parameter will be ignored. Note that the order for looking for the default trustStore java-home/lib/security/jssecacerts then java-home/lib/security/cacerts
trustStorePassword	string	[Optional] Password to the truststore.
disableHostVerification	true/false	[Optional] If true, we will not verify that the host in the certificate is the host we are connecting to. False by default (Hostname verification follows the specification in RFC2818).
disableCertificateVerification	true/false	[Optional] If true we will not validate the certificate against the truststore. False by default.
TLSProtocol	TLS, TLSV1, TLSv1.1, TLSv1.2	[Optional] Default: TLSv1.2 (recommended)

last-updated: Apr 11, 2022

TLSHandshakeTimeout	Time in milliseconds	[Optional] Default: 10 seconds In some cases, the TLS handshake may fail and leave the client hanging. This option sets the time for the client to timeout.
TLSProvider	JDK/OPENSSL	[Optional] Default: JDK Changes the underlying implementation to the chosen value.
useSystemTrustStore	true/false	[Optional, Windows only] Default: false If provided, the client will read certificates from the Windows truststore. In this case, trustStorePath and trustStorePassword, if specified, will be ignored. The user should set the default provider in $JRE_HOME/lib/security/java.security to SunMSCAPI. The trustStoreType should be set to either Windows-MY or Windows-ROOT.

Examples

The following examples show you how to connect to Drill through SQLLine with the `jdbc` connection string when SSL is not enabled and when SSL is enabled with and without a truststore.

No SSL/TLS

```
./sqlline -u
"jdbc:drill:schema=dfs.work;drillbit=l
ocalhost:31010;enableTLS=false"
```

SSL/TLS Enabled - No truststore

The default JSSE truststore will be tried with default password; the provided password will be ignored. If the default truststore password has been changed, this gives an error. To use the default truststore with a different password, pass the path to the default truststore with the password.

```
./sqlline -u
"jdbc:drill:schema=dfs.work;drillbit=l
ocalhost:31010;enableTLS=true;trustSto
rePassword=drill123"
```

SSL/TLS enabled - With truststore

```
./sqlline -u
"jdbc:drill:schema=dfs.work;drillbit=l
ocalhost:31010;enableTLS=true;trustSto
rePath=~/ssl/
truststore.ks;trustStorePassword=drill
123"
```

ODBC Connection Parameters

Use the SSL ODBC connection parameters to configure a connection to Drill through an ODBC tool.

The following table lists the ODBC connection parameters:

Note: The Drill ODBC driver does not support password protected PEM/CRT files or multiple CRT certificates in a single PEM/CRT file. For additional instructions, see the Drill ODBC Driver documentation.

Name	Value	Required	Description
SSL	Clear (0)	No.	This option specifies whether the cliet uses an SSL encrypted connection to communicate with Drill. • Enabled(1):The client communicates with Drill using SSL. • Disabled(0):SS Lis disabled. SSL is configured independently of authentication. When authentication and SSL are both enabled, the driver performs the specified authentication method over an SSL connection.
TLSProtocol	Empty, which defaults to tlsv12.	No	This property specifies the TLS protocol version used. Accepted values are: • tlsv1 • tlsv11 • tlsv12
TrustedCerts	The cacerts.pem file in the \lib subfolder within the Driver's installation directory. The exact file path varies depending on the version of the driver that is installed. For example, the path for the Windows driver is different from the path for the Mac OS driver.	No	The full path of the PEM file containing Trusted CA certificates, for verifying the server. If this option is not set, then the driver defaults to using the trusted CA certificates PEM file installed by the driver.

last-updated: Apr 11, 2022

UseSystemTrustStore	Clear (0)	No	This option specifies whether to use a CA certificate from the system truststore, or from a specified PEM file. • Enabled (1): The driver verifies the connection using a certificate in the system truststore • Disabled (0): The driver verifies the connection using a specified PEM file. **Note:** This option is only available on Windows. If using this option, import the certificate into the "Trusted Root Certificate Authorities" certificate store.
DisableCertificateVerification	0	No	This property specifies that the driver verifies the host certificate against the truststore. Accepted values are: • 0: The driver verifies the certificate against the truststore. • 1: The driver does not verify the certificate against the truststore.
DisableHostVerification	0	No	This property specifies if the driver verifies that the host in the certificate is the host being connected to. Accepted values are: • 0: The driver verifies the certificate against the host being connected to. • 1: The driver does not verify the certificate against the host.

Avoiding Common SSL Issues
The following sections provide insight to some common error messages that you may encounter with SSL.

ERROR: No Cipher suites in common.

This is a general purpose error message that may have many reasons. The most common reason is that in order to use certain cipher suites, JSSE needs to use the private key stored in the Keystore. If this key is not accessible, JSSE filters out all cipher suites that need a private key. This effectively prunes out all available cipher suites so that no cipher suites match between the client and the server.

The private key from the keystore may be inaccessible for the following reasons:

- Missing Keystore file

- Invalid Keystore password
- Empty key password or a key password that is different from the keystore password

JSSE does not allow a key password that is null or an empty string even though it is possible to create a keystore with such a key password. Also, JSSE does not provide a system property to specify the key password. Drill provides a way to set the key password, but if you are using only system properties to configure JSSE, Drill will use the *keystore* password. If the keystore password is not the same as the key password, the key will again be inaccessible.

- Corrupt keystore

You can validate the keystore using keytool.

ERROR: SSL is enabled, but cannot be initialized due to the 'Cannot recover key' exception.

The key is protected with a password and the provided password is not correct.

ERROR: Client connection timeout.

A client connection can timeout because of networking issues or if there is a mismatch between the TLS/SSL configuration on the client and server.

Before trying to debug the TLS/SSL configuration, check if the server is reachable from the client.

If there is a mismatch between the TLS/SSL configuration, the TLS/SSL handshake between the client and server will fail. The server will silently drop the connection and the client will eventually time out. The handshake may fail due to many reasons, including:

1. The server is configured to enableTLS and the client is not (and vice versa).
   a. If the client is not configured to use TLS and the server is, the error message will be similar to the following:

```
Error: Failure in connecting to Drill:
org.apache.drill.exec.rpc.RpcException: HANDSHAKE_COMMUNICATION :
Channel closed /10.10.10.11:49907 <--> hostname/10.10.10.11:31010.
(state=,code=0)
java.sql.SQLNonTransientConnectionException: Failure in
connecting to Drill: org.apache.drill.exec.rpc.RpcException:
HANDSHAKE_COMMUNICATION : Channel closed /10.10.10.11:49907
<-->hostname/10.10.10.11:31010.
```

   b. If the server is not configured to use TLS and the client tries to connect using TLS, the error message will be similar to the following:

```
Error: Failure in connecting to Drill:
org.apache.drill.exec.rpc.NonTransientRpcException: Connecting to the
server timed out. This is sometimes due to a mismatch in the SSL
configuration between client and server. [ Exception: Timeout waiting
for task.] (state=,code=0)
```

2. The server presents a certificate to the client containing a hostname that is not valid. When the client connects to a server, the hostname the client used to connect to the server must match the name of the host the certificate was assigned to. Certificates can contain wildcards for the hostname, so if you're connecting to a Drill cluster via ZooKeeper, it would be best to have a certificate that contains wildcards that cover all the hosts on which Drill might be running. It is also important to ensure that the DNS and the hostnames of the machines in the cluster are set up consistently so that the Drillbits are registered with ZooKeeper using the same name as the name assigned in the certificate. The error message in this case is the same as the previous case:

   ```
   Error: Failure in connecting to Drill:
   org.apache.drill.exec.rpc.NonTransientRpcException: Connecting to the
   server timed out. This is sometimes due to a mismatch in the SSL
   configuration between client and server. [ Exception: Timeout waiting
   for task.] (state=,code=0)
   ```

   Hostname verification can be turned off if there is no way to change the host configuration or the certificate. This is generally not recommended.

Security Between ZooKeeper and Drillbits

When Drill is installed on clusters with the default security enabled, authentication is enabled between the Drillbits and ZooKeeper. The ZooKeeper znode information is secured automatically through authentication and znode ACLs. Communication between the Drillbits and Zookeeper is not encrypted.

Note: If you installed Drill on a cluster that does not have the default security configuration, and you are configuring custom security, you must enable authentication and manually set ACLs on the znodes.

Drill uses ZooKeeper to store certain cluster-level configuration and query profile information in znodes. A znode is an internal data tree in ZooKeeper that stores coordination and execution related information. If information in the znodes is not properly secured, cluster privacy and/or security is compromised.

ZooKeeper uses ACLs to control access to znodes and secure the information they store. Starting in Drill 1.15, you can create a custom ACL (Access Control List) on the znodes to secure data. ACLs specify sets of ids and permissions that are associated with the ids.

Prior to Drill 1.15, ZooKeeper ACLs in secure and insecure clusters were set to `[world:all]`, meaning that all users had create, delete, read, write, and administrator access to the zknodes. Starting in Drill 1.15, ACLs in insecure clusters are set to `[world:all]`. ACLs in secure clusters are set to `[authid: all]`, which provides full access to the authenticated user that created the znode only. Discovery znodes (znodes with the list of Drillbits) have an additional ACL set to `[world:read]` making the list of Drillbits readable by any user.

Note: View the drill-override-example.conf file to see example ACL configurations.

Securing znodes

Complete the following steps to create a custom ACL and secure znodes:

1. Write a class that implements the `ZKACLProvider` interface. This class will contain the ACLs that need to be set on the znodes. You can use the ZKSecureACLProvider class as a sample reference.

2. Add the following dependency to the `pom` file of the project module created:

   ```
   <groupId>org.apache.drill.exec</groupId>
   <artifactId>drill-java-exec</artifactId>
   ```

3. Refer to the steps listed at https://drill.apache.org/docs/manually-adding-custom-functions-to-drill/ to create a JAR and then add the JAR to Drill's classpath.

4. In `/opt/mapr/drill/drill-<version>/conf/drill-override.conf`, set `zk.acl_provider` to the `ZKACLProviderTemplate` type.

5. Restart Drill. When you restart Drill, the ACL, as mentioned in your custom class, is applied to the znode created when Drill starts.

Note: Existing ACLs for persistent znodes will not be affected if a Drillbit is restarted with a different ACL setting. ACLs are applied only at znode creation time. Drill does not recreate any znode that is already present. If you want to change an ACL for existing znodes, connect to the ZooKeeper server using zkCli and then use option a or b, as described:

- a) Shutdown Drillbits, delete the persistent znodes, change the ACL settings, and restart the Drillbit.
- b) Manually change the ACLs on the existing znodes to reflect the new ACL settings, using the `setAcl` command in the zkCli.

For either option to work, an authenticated connection between the zkCli and ZooKeeper Server must be established.

For additional information, refer to:

- ZooKeeper access control using ACLs
- ZooKeeper and SASL

Configuring Drill Web UI and Web API Security

The Drill web client and web API communicate with web browsers or web tools, like curl, through the HTTP or HTTPS. Drill uses HTTP by default.

Drill supports form-based (similar to Plain authentication) and SPNEGO authentication mechanisms to authenticate the communication between the web client and web browser or web tools. Drill supports SSL/TLS for encryption with form-based and SPNEGO authentication.

An administrator can configure security mechanisms and set up Drill Web UI administrators and administrator-user groups to control access to the Drill Web UI and Web API client applications. For example, limiting user access to Drill Web UI functionality, such as viewing or canceling queries submitted by other users.

Note:

- The Drill web server does not support MAPR-SASL (tickets).
- With Drill Web UI security in place, users without administrator privileges must execute the SHOW SCHEMAS command in the Drill Web UI Query page to see storage plugin configuration information.

Form-Based Authentication

In EEP 5.0 and later, Drill supports form-based authentication between the web client and Drillbit. Form-based authentication is like Plain Authentication in that a user is presented with a web form where s/he enters a username and password to access restricted web pages. Form-based authentication also uses the Linux PAM (Pluggable Authentication Module).

Configuring Drill to Use libpam4j provides configuration details. When using form-based authentication, you can also configure Drill to use SPNEGO for HTTP Authentication and SSL/TLS for encryption.

last-updated: Apr 11, 2022

HTTPS Support

The Drill Web UI supports the HTTPS protocol for encryption. With the default security configuration, HTTPS is enabled for Drill and it uses SSL trust- and keystore, which comes with cluster installation.

To use custom certificates, see SSL Certificates in Secure and Unsecure MapR Clusters.

The following example shows the default HTTPS configuration in `<DRILL_INSTALL_HOME>/conf/drill-distrib.conf` for a secure cluster installation:

```
drill.exec: {
             http.ssl_enabled: true,
             ssl.useHadoopConfig: true
}
```

You can configure additional parameters:

Drill Property Name	Hadoop Property Name	System Property Name	Description
drill.exec.http.ssl_enabled:			Enable or disable communication. Y
drill.exec.ssl.keyStorePath	ssl.server.keystore.location	javax.net.ssl.keyStore	Location of the Ja own certificate ar pathname must u
drill.exec.ssl.keyStorePassword	ssl.server.keystore.password	javax.net.ssl.keyStorePassword	Password to acce This password is password), and to (key password) u
drill.exec.ssl.keyPassword	ssl.server.keystore.keypassword		Password to acce be different from
drill.exec.ssl.trustStorePath	ssl.server.truststore.location	javax.net.ssl.trustStore	Location of the Ja certificates truste pathname must u
drill.exec.ssl.trustStorePassword	ssl.server.truststore.password	javax.net.ssl.trustStorePassword	Password to acce specified as the t
drill.exec.ssl.useHadoopConfig			Use the setting in hadoop configura the hadoop.ssl.se file. Typically, this ssl-server.xml wh TLS.  **Note:** Verify is located in symbolic lin for example `<DRILL_` .

Setting up Web UI Administrators and Administrator-User Groups
The `security.admin.user_groups` and `security.admin.users` options set the administrative users when authentication is enabled.

last-updated: Apr 11, 2022

Users listed in `security.admin.users` and the users belonging to the groups listed in `security.admin.user_groups` get administrative privileges. By default, these options are set to the username and groups of the user who started the Drill process. You can modify these options from the Drill Web UI or using the ALTER SYSTEM command.

An administrative user that authenticates to a Drillbit through MapR-SASL, Kerberos, Plain mechanism or through the HTTPS web interface can modify the `security.admin.user_groups` and `security.admin.users` options. The administrator can add or remove user names or user groups.

The `security.admin.user_groups` and `security.admin.users` options allow a single user/group name or a comma separated list of user/group names. When you view these options in the Drill Web UI, dummy default strings appear until the user explicitly changes the values.

Authenticated administrative users can view the current administrative user and administrative user groups on the Drill Web UI landing page (https://<node-ip-address>:8047) in the *Encryption Info* section.

Drill REST API and Web UI

This topic provides information about the Drill REST API and Web UI, including permission requirements.

If Drill has authentication enabled, you must supply credentials when using the Drill REST API.

Although Drill (in HPE Ezmeral Data Fabric) does not support HTTP basic authentication, you can work around this if your HTTP client saves cookies between requests. As a workaround, save the authenticated cookie to a file and then use the cookie in subsequent requests, as shown in the following example:

```
//Log in and save the authenticated cookie to a file:

curl -X POST \
    -H "Content-Type: application/x-www-form-urlencoded" \
    -k -c cookies.txt -s \
    -d "j_username=DRILL_USER" \
    -d "j_password=DRILL_PASSWORD" \
    https://HOSTNAME:8047/j_security_check

//In subsequent requests, use the cookie from that request:

curl -kv \
       -b cookies.txt  \
       -X POST \
       -H "Content-Type: application/json" \
       -d @/tmp/hive-storage-plugin.json
       https://HOSTNAME:8047/storage/hive.json
```

Note: The session remains active for one hour. You can increase the session time through the `drill.exec.http.session_max_idle_secs` option in `drill-override.conf`:

```
drill.exec: {
  http: {
    session_max_idle_secs: 86400, # 24hr
  }
}
```

REST API Methods and Web UI Functions

The following table and subsections describe requests and privilege levels for accessing the REST API methods and corresponding Drill Web UI functions. Privileges in the table are listed as AMDIN, USER, and ALL. ALL indicates privileges given to both the user and administrator.

Path	Request Type	Output Type	Functionality	Privileges

last-updated: Apr 11, 2022

/	GET	text/html	Returns Drillbit stats in a table in HTML format.	ALL
/stats.json	GET	application/json	Returns Drillbit stats such as ports and max direct memory in json format.	ALL
/status	GET	text/html	Returns Running!	ALL
/options.json	GET	application/json	Returns a list of options. Each option consists of name-value-type-kind (for example: (boot system datatype).	ALL
/options	GET	text/html	Returns an HTML table where each row is a form containing the option details and ability to modify the option values.	ALL
/option/{optionName}	POST	text/html	Updates the options and calls getSystemOptions to display list of options.	ADMIN
/storage.json	GET	application/json	Returns a list of storage plugin wrappers each containing name-config (instance of StoragePluginConfig) and enables the storage plugin configuration.	ADMIN
/storage	GET	text/html	Returns an HTML page with sections that contain: • a table where each row is a form containing the plugin button for update page link and a button to disable the plugin. • a table where each row is a form containing the plugin button for update page and a button to enable the plugin.	ADMIN
/storage/{name}.json	GET	application/json	Returns a plugin config wrapper for the requested web page.	ADMIN
/storage/{name}	GET	text/html	Returns an HTML page that has an editable text box for configuration editing, followed by buttons for creating,updating, and deleting. Each of the buttons make calls that generate the new page again.	ADMIN
/storage/{name}/enable/{val}	GET	application/json	Updates the storage plugin status. Returns success or failure.	ADMIN
/storage/{name}.json	DELETE	application/json	Deletes the storage plugin. Returns success or failure.	ADMIN
/storage/{name}/delete	GET	application/json	Same as deletePluginJSON but a GET instead of a DELETE request.	ADMIN
/storage/{name}.json	POST	application/json	Creates or updates the storage plugin. Returns success or failure. Expects JSON input.	ADMIN

/storage/{name}	POST	application/json	Same as createOrUpdatePluginJSON expects JSON or FORM input.	ADMIN
/profiles.json	GET	application/json	Returns currently running and completed profiles from PStore. For each profile a queryId, startTime, foremanAddress, query, user, and state is returned. Each list (running and completed) is organized in reverse chronological order.	ADMIN, USER
/profiles	GET	text/html	Generates an HTML page from the data returned by getProfilesJSON with a hyperlink to a detailed query page.	ADMIN, USER
/profiles/{queryid}.json	GET	application/json	Returns the entire profile in JSON.	ADMIN, USER
/profiles/{queryid}	GET	text/html	Returns a complex profile page.	ADMIN, USER
/profiles/cancel/{queryid}	GET	text/html	Cancels the given query and sends a message.	ADMIN, USER
/query	GET	text/html	Gets the query input page.	ALL
/query.json	POST	application/json	Submits a query and waits until it is completed and then returns the results as one big JSON object.	ALL
/query	POST	text/html	Returns the results of submitQueryJSON in an HTML table.	ALL
/status/metrics	GET	application/json	Returns a page that fetches metric info from resource, status, and metrics.	ALL
/status/threads	GET	text/html	Returns a page that fetches metric information from resource, status, and threads.	ALL
/login	GET	text/html	Returns an HTML log in page. If the user is already logged in, returns the home page. If the URL contains a redirect, sets the redirect URI for the session and forwards the user to the redirect page after the user is successfully logged in.	ALL
/login	POST	text/html	Returns a validation error for incorrect credentials.	ALL
/logout	GET	text/html	Ends a session.	ALL

GET /profiles.json

- ADMIN - gets all profiles on the system.
- USER - only the profiles of the queries the user has launched.

GET /profiles

- ADMIN - gets all profiles on the system.

last-updated: Apr 11, 2022

- USER - only the profiles of the queries the user has launched.

GET /profiles/{queryid}.json

- ADMIN - return the profile.
- USER - if the query is launched the by the requesting user return it. Otherwise, return an error saying no such profile exists.

GET /profiles/{queryid}

- ADMIN - return the profile.
- USER - if the query is launched the by the requesting user return it. Otherwise, return an error saying no such profile exists

GET /profiles/cancel/{queryid}

- ADMIN - can cancel the query.
- USER - cancel the query only if the query is launched by the user requesting the cancellation.

Related concepts

Securing Drill on page 3275
An administrator can install Drill with the default security configuration or manually configure custom security for Drill.

Related information

https://drill.apache.org/docs/rest-api/

SPNEGO for HTTP Authentication

Drill 1.13 and later supports the Simple and Protected GSS-API Negotiation mechanism (SPNEGO) to extend the Kerberos-based single sign-on authentication mechanism to HTTP. An administrator configures the web server (Drillbit) to use SPNEGO for authentication. Depending on the system, either the administrator or the user configures the client (web browser or web client tool) to use SPNEGO for authentication.

An administrator can configure both FORM (username and password) and SPNEGO authentication together, which provides the ability for clients with different security preferences to connect to the same Drill cluster. When a client (a web browser or a web client tool, such as curl) requests access to a secured page from the web server (Drillbit), the SPNEGO mechanism uses tokens to perform a handshake that authenticates the client browser and the web server.

The Drill Web UI provides two possible log in options for a user depending on the configuration. If a user selects FORM, s/he must enter their username and password to access restricted pages in the Drill Web UI. The user is authenticated through PAM. If the user selects SPNEGO, the user is automatically logged in if they are an authenticated Kerberos user. If accessing a protected page directly, the user is redirected to the authentication log in page. If the client fails to authenticate using SPNEGO, an error page displays with an option to use FORM authentication, assuming FORM authentication is configured on the server side.

Browser Support

The following browsers were tested with Drill configured to use SPNEGO authentication:

- Firefox
- Chrome
- Safari
- Internet Explorer

- Web client tool, such as curl

Prerequisites

SPNEGO authentication for Drill requires the following:

- Drill 1.13 or later installed on each node.
- A working Kerberos infrastructure, which Drill does not provide.
- A Linux-based or Windows Active Directory (AD) Kerberos environment with secure clusters and a Drill server configured for Kerberos.
- Kerberos principal and keytab on each web server (Drillbit) that will use SPNEGO for authentication.
- Kerberos Ticket Granting Ticket on the client machine for the user accessing the Drillbit (web server).
- Drill web server configured for SPNEGO.

Configuring SPNEGO on the Web Server and Web Client

The following sections provide the steps that an administrator can follow to configure SPNEGO on the web server (Drillbit). An administrator or a user can follow the steps for configuring the web browser or client tool.

Configuring SPNEGO on the Drillbit (Web Server)
To configure SPNEGO on the web server, complete the following steps:

1. Generate a Kerberos principal on each web server that will receive inbound SPNEGO traffic. Each principal must have a corresponding keytab. The principal must have the following form:

```
"HTTP/<client-known-server-hostname@realm>"

   Example: "HTTP/example.QA.LAB@QA.LAB"
   //In this example, the client known server hostname is example.QA.LAB.
```

   Note: If HTTPS is enabled on the Drillbit (web server), the SPNEGO principal should also start with "HTTP/", not "HTTPS/" even though the URL includes HTTPS.

2. Update the `/opt/mapr/drill/drill-<version>/conf/drill-override.conf` file on each Drillbit with the following server-side SPNEGO configurations:
   - To enable SPNEGO, add the following configuration to `drill-override.conf`:

```
impersonation: {
                    enabled: true,
                    max_chained_user_hops: 3
             },
             drill.exec.http: {
                    spnego.auth.principal:"HTTP/hostname@realm",
                    spnego.auth.keytab:"path/to/keytab",
                    auth.mechanisms: ["SPNEGO"]
             }
      }
      //The default authentication mechanism is "FORM".
```


last-updated: Apr 11, 2022

- To enable SPNEGO and FORM authentication, add the following configuration to `drill-override.conf`:

```
impersonation: {
                enabled: true,
                max_chained_user_hops: 3
              },
              security.user.auth: {
                      enabled: true,
                      packages +=
"org.apache.drill.exec.rpc.user.security",
                      impl: "pam4j",
                      pam_profiles: [ "sudo", "login" ]
                },
             drill.exec.http: {
                      spnego.auth.principal:"HTTP/hostname@realm",
                      spnego.auth.keytab:"path/to/keytab",
                      auth.mechanisms: ["SPNEGO", "FORM"]
             }
      }
```

3. (Optional) To configure the mapping from a Kerberos principal to a user account used by Drill, update the `drill.exec.security.auth.auth_to_local` property in the `drill-override.conf` file with custom rules, as described in Mapping from Kerberos Principal to OS user account.

   Note: Drill uses a Hadoop Kerberos name and rules to transform the client Kerberos principal to the principal Drill uses internally as the client's identity. By default, this mapping rule extracts the first portion from the provided principal. For example, if the principal format is Name1/Name2@realm, the default rule extracts only Name1 from the principal and stores Name1 as the client's identity on server side. Drill uses the short name, for example Name1, as the user account known to Drill. This user account name is used to determine if the authenticated user has administrative privileges.

Configuring SPNEGO on the Client

An administrator or user can configure SPNEGO on the client (web browser or client tools, such as curl). To configure SPNEGO on the client, a Kerberos Ticket Granting Ticket must exist for the user accessing the web server. The Kerberos Ticket Granting Ticket generated on the client side is used by the web client to get a service ticket from the KDC. This service ticket is used to generate a SPNEGO token, which is presented to the web server for authentication.

The client should use the same web server hostname (as configured in the server-side principal) to access the Drill Web Console. If the server hostname differs, SPNEGO authentication will fail. For example, if the server principal is `"HTTP/example.QA.LAB@QA.LAB"`, the client should use `http://example.QA.LAB:8047` as the Drill Web Console URL.

The following sections provide instructions for configuring the supported client-side browsers:

Firefox

To configure Firefox to use a negotiation dialog, such as SPNEGO to authenticate, complete the following steps:

1. Go to **About > Config**, and accept the warnings.
2. Navigate to the network settings.
3. Set `network.negotiate-auth.delegation-uris to "http://,https://"`.
4. Set `network.negotiate-auth.trusted-uris` to “http://,https://”.

Chrome

For MacOS or Linux, add the `--auth-server-whitelist` parameter to the `google-chrome` command. For example, to run Chrome from a Linux prompt, run the `google-chrome` command, as shown:

```
google-chrome --auth-server-whitelist = "hostname/domain"
  Example: google-chrome --auth-server-whitelist = "example.QA.LAB"
```

Safari

No configuration is required for Safari. Safari automatically authenticates using SPNEGO when requested by the server.

Internet Explorer

To configure Internet Explorer to use a negotiation dialog, such as SPNEGO to authenticate, complete the following steps:

1. Go to **Tools > Options > Security > Local Intranet > Sites**, and select all options.
2. Select **Advanced**, and add one or both of the following URLs to server:
   - http://
   - https://

   Note: Make sure you use the hostname of the Drillbit in the URL.
3. Close the **Advanced** tab, and click **OK**.
4. Go to **Tools > Options > Advanced > Security** (in the checkbox list), and enable the **Integrated Windows Authentication** option.
5. Click **OK**.
6. Close and reopen IE. You can browse to your Spengo protected resource.

REST API

You can use CURL commands to authenticate using SPNEGO and access secure web resources over REST.

Issue the following `curl` command to log in using SPNEGO, and save the authenticated session cookie to a file, such as `cookie.txt`, as shown:

```
curl -v --negotiate -c cookie.txt -u : http://<hostname>:8047/spnegoLogin
```

Use the authenticated session cookie stored in the file, for example `cookie.txt`, to access the Drill Web Console pages, as shown in the following example:

```
curl -v --negotiate -b cookie.txt -u : http://<hostname>:8047/query
   Example: curl -v --negotiate -b cookie.txt -u : http://
example.QA.LAB:8047/query
```

Using ACEs on Views to Limit Data Access

Describes how to use access control expressions to limit data access for Views.

Apache Drill on page 3185 is a distributed SQL query layer that runs on the data platform. You can enable user impersonation and create views in Drill to control user access to data stored in the data platform at the

last-updated: Apr 11, 2022

row and column levels. Access to data is based on file permissions set on the data (source files) and on the view definition files.

In addition to standard POSIX permissions, ACEs (access control expressions) are supported to secure data in the distributed filesystem. ACEs are a flexible access control mechanism that applies to files, tables, and streams. Setting an ACE (access control expression) on a file modifies the file permission to honor the *ACE* setting. Drill honors *ACE* set on Drill view files and on the source files that views access.

Each Drill view created has an associated view definition file, with a `.view.drill` extension, on which you can set ACEs to secure the view.

Example

Frank creates a workspace in the dfs storage plugin configuration in Drill that points to his home directory in the distributed filesystem. He then uses Drill to create a table named “employees” that he and the HR group can access:

```
-rwxr----- frank:hr /user/frank/employees
```

Joe, a member of the HR and MGR groups, creates a view named `emp_mgr_view` in his home directory to share a subset of the employees data with managers that belong to the MGR group:

```
-rwxr----- joe:mgr /user/joe/emp_mgr_view.drill.view
```

Managers in the MGR group have read permission on the `emp_mgr_view.drill.view` file so they can query the `emp_mgr_view` that Joe created and they can create new views from his view.

Setting *ACE* on the underlying data source (the “employees” table) or on the view file (`emp_mgr_view.drill.view`) that accesses the underlying data source resets the POSIX mode bits to match the permissions granted through *ACE* settings.

For example, if Frank issues the following command to apply an *ACE* to the “employees” table, a user must be a member of the EXEC group to read data in the “employees” table:

```
hadoop mfs -setace -R -readfile 'g:exec' employees
```

Anyone in the HR group that previously had access to the table can no longer access the table data unless they also belong to the EXEC group.

Running the `-getace` command on the table lists the *ACE* settings on the table:

```
hadoop mfs -getace /user/frank/employees
```

```
Path: /user/frank/employees
 readfile: g:exec
 writefile:
 executefile:
 readdir:
 addchild:
 deletechild:
 lookupdir:
 inherit: true
 mode: ---------
```

Similarly, if Joe issues the following command on the on the `emp_mgr_view.drill.view` file, only members of the HR group can read the file. Users that belong to the MGR group can no longer access the data through the view, unless they also belong to the HR group.

```
hadoop mfs -setace -R -readfile 'g:hr' emp_mgr_view.drill.view
```

Running the `-getace` command on view file shows the *ACE* settings on the file:

```
hadoop mfs -getace /user/joe/emp_mgr_view.drill.view
```

```
Path: /user/joe/emp_mgr_view.drill.view
 readfile: g:hr
 writefile:
  executefile:
  readdir:
  addchild:
  deletechild:
  lookupdir:
  inherit: true
  mode: ---------
```

You may also want to view another File ACE Example on page 1455.

Drill Drivers
HPE Ezmeral Data Fabric provides Drill ODBC and JDBC drivers that you can download and use to connect Drill to BI tools. The drivers are updated periodically to include support for new functionality in Drill.

The following table provides links to driver download sites and documentation:

Driver	Driver Download Site	Driver Documentation
Drill ODBC Driver	All versions of the Drill ODBC driver are located at https://package.mapr.hpe.com/tools/MapR-ODBC/MapR_Drill/.	• Information about the driver, including Drill driver version compatibility and important messages, is located at Drill ODBC Driver on page 3345. • Driver documentation, including installation and configuration instructions, is located in the Drill ODBC Driver PDF file.
Drill JDBC Driver	All versions of the Drill JDBC driver are located at https://package.mapr.hpe.com/tools/MapR-JDBC/MapR_Drill/.	• Information about the driver, including Drill driver version compatibility and important messages, is located at Drill JDBC Drivers on page 3333. • Driver documentation, including installation and configuration instructions, is located in the Drill JDBC Driver PDF file.

Drill JDBC Drivers
Download the Drill JDBC driver and use it on all platforms to connect BI tools, such as SQuirreL and Spotfire, to Drill. Drill also includes an embedded, open-source JDBC driver.

The downloadable Drill JDBC driver provides read-only access to Drill data sources and supports the security features described in Securing Drill.

Alternatively, you can use the open-source JDBC driver embedded in Drill; however, *the open-source JDBC driver is not tested on the MapR Data Platform*. The open-source driver supports Kerberos and Plain authentication mechanisms, but does not support the -SASL authentication mechanism. After you install Drill from the `mapr-drill` package, you can find the open-source JDBC driver files in the following directories:

- `$DRILL_HOME/jars/jdbc-driver/drill-jdbc-all-<drill-version>.jar`
- `$DRILL_HOME/jars/drill-jdbc-<drill-version>.jar`

Drill JDBC Driver Download

Use the version of the driver that correlates with the version of the installed Drill server. Although older versions of the driver may connect to an upgraded version of Drill, the older drivers do not include all the server features available in the newer drivers.

The following table provides links to the download locations for the Drill JDBC drivers that correlate with each of the Drill versions listed:

Drill Version	JDBC Version
1.16.0.x	1.6.7.1010 **Attention:** This driver supports JRE 8 only and includes updated driver classes. See Driver Class on page 3334.
1.16.0.x	1.6.6.1009
1.16.1.x	1.6.6.1008
1.16.0	1.6.0.1001
1.15.0	1.6.0.1001
1.14.0	1.6.0.1001
1.13.0	1.5.9.1018
1.12.0	1.5.8.1017
1.11.0	1.5.6.1012
1.10.0	1.5.3.1006

Driver Class

Important: The Drill JDBC Driver installation and configuration PDF document does not include the information provided in the following sections:

Driver Class

The *Registering the Driver Class* section of the Drill JDBC Driver documentation incorrectly lists the driver classes as `com.simba.drill.jdbc41.Driver` and `com.simba.drill.jdbc41.DataSource`.

- For driver version **1.6.6.1009 and earlier**, the correct driver classes are:
  - `com.mapr.drill.jdbc41.Driver`
  - `com.mapr.drill.jdbc41.DataSource`
- For driver version **1.6.7.1010**, the correct driver classes are:
  - `com.mapr.drill.jdbc.Driver`
  - `com.simba.drill.jdbc.DataSource`

JDBC Connection String

You can indicate the `schema` parameter in the connection string, as shown in the following example:

```
jdbc:drill:zk=10.10.100.30:5181,10.10.100.31:5181,10.10.100.32:5181/drill/
drillbits1;schema=hive
```

You can also include the authentication mechanism in the connection string using the `AuthMech` or `auth` parameter. For MapR-SASL, use `auth=MAPRSASL`.

- If using the MapR-SASL or Plain authentication mechanism, you must add the Drill JDBC JAR files and `/opt/mapr/lib/*` to the classpath of the third-party client tool, as shown in the following example for SQuirreL when the path to the driver is `C:\driver\MapRDrillJDBC41-1.5.6.1012:`

  ```
  -cp
  "%SQUIRREL_CP%;C:\driver\MapRDrillJDBC41-1.5.6.1012\*;C:\opt\mapr\lib\*"
  ```

  The driver JAR files should appear before `/opt/mapr/lib/*` in the classpath.

Using MapR-SASL for Authentication on Windows

Drill is automatically configured with MapR security when you install Drill on a MapR cluster configured with default security. To successfully connect to Drill from a Windows JDBC client, a user ticket must exist on the Windows client in the %TEMP% directory or in the location specified by the $MAPR_TICKETFILE_LOCATION environment variable.

The JDBC driver locates user tickets for the current Windows user in the default ticket location, %TEMP%, or in the location specified by the environment variable, $MAPR_TICKETFILE_LOCATION. See Tickets and Generating a MapR User Ticket for more information.

You can either copy a user ticket that was generated on the MapR cluster into the default location (%TEMP%), or you can install the MapR client on the Windows client and then run the maprlogin command to generate the ticket on the Windows client.

Note: The JDBC user must be the same as the Windows user that created the ticket.

Example

If you want to connect to Drill as the mapr user, you must create a ticket for the mapr user, as shown:

```
$ maprlogin password -user mapr
[Password for user 'mapr' at cluster 'Cluster1':]
```

The credentials for the mapr user in Cluster1 are written to /tmp/maprticket_1000.

Next, place the ticket in the %TEMP% directory on the Windows client. For example, the default location for a Windows 10 user named Tabetha Stephens is shown:

```
'C:\Users\TABETH~1\AppData\Local\Temp/maprticket_Tabetha Stephens'
```

To override this location, set the "MAPR_TICKETFILE_LOCATION" global variable for the Windows user.

Note: Using the MAPR_TICKETFILE_LOCATION is recommended because the %TEMP% directory differs between Windows versions. You may also want to set the MAPR_TICKETFILE_LOCATION per user on the operating system to prevent all users from using the same user ticket on the client.

Avoiding Driver Conflicts

If you download and use the Drill JDBC driver, rename the embedded JDBC driver files to avoid any conflict between the downloaded driver and the open-source driver. The embedded JDBC driver files are in the following directories after you install Drill:

```
$DRILL_HOME/jars/jdbc-driver/drill-jdbc-all-1.10.0.jar
$DRILL_HOME/jars/drill-jdbc-1.10.0.jar
```


last-updated: Apr 11, 2022

Changing the file extension to rename these files, as shown in the following example, prevents Drill or any other application, such as SQLLine, from picking up the embedded driver:

```
$DRILL_HOME/jars/jdbc-driver/drill-jdbc-all-1.10.0.jar.original
$DRILL_HOME/jars/drill-jdbc-1.10.0.jar.original
```

Connecting to Drill via the Drill Shell (SQLLine)

See Connecting to Drill via the Drill Shell (SQLLine) on page 3336.

Driver Limitations

When using MapR-SASL with JDBC or ODBC drivers, there is no way to specify the target cluster name as part of the connection parameters. MapR-SASL reads the first entry in the `/opt/mapr/conf/mapr-clusters.conf` file and assumes it is the target cluster name.

For example, if the `mapr-clusters.conf` file has an entry for '`cluster1`' followed by an entry for '`cluster2`' and you want to connect to a node in '`cluster2`', authentication fails. As a workaround, manually switch the order of entries in the `mapr-clusters.conf` file.

Connecting to Drill via the Drill Shell (SQLLine)
SQLLine is a JDBC application that is packaged with Drill and serves as the Drill shell. When you issue queries from the SQLLine client, SQLLine passes the queries to the connected Drillbit (Drill node).

You can connect to Drill through Sqlline directly or through a connection-property file. To avoid exposing credentials, connect through the connection-property file.

A JDBC connection string supplies the connection information to a Drill node or ZooKeeper cluster. When you connect to a ZooKeeper cluster, ZooKeeper selects the Drillbit for SQLLine to connect to.

JDBC Connection String Example

Here is an example of a JDBC connection string that connects SQLLine to drillnode1:

```
jdbc:drill:drillbit=drillnode1:31010
```

The default port on a Drill node is 31010.

Connection Parameters

You can include SQLLine connection parameters in the connection string and run various shell commands, as described in Configuring the Drill Shell.

In the following example, `-u` is the connection parameter for the JDBC connection string, `-n` is the parameter for the username, and `-p` is the parameter for the password:

```
/opt/mapr/drill/drill-<version>/bin/sqlline -u
"jdbc:drill:drillbit=drillnode1:31010" -n mapr -p mapr
```

Starting SQLLine

Start SQLLine from the Drill installation directory, as shown:

```
/opt/mapr/drill/drill-<version>/bin/sqlline -u
jdbc:drill:drillbit=drillnode1:31010
```

Configuration Options

You can also include configuration options, such as `schema`:

```
/opt/mapr/drill/drill-<version>/bin/sqlline -u "jdbc:drill:drillbit
drillnode1:31010;schema=dfs" -n <username> -p <password>
```

Schema	The `schema` is the name of a storage plugin configuration to use as the default for queries. If you indicate the schema in the connection URL, you do not have to run the `USE <schema>;` query to switch to the schema you want to use. All queries run against the schema indicated in the JDBC connection string.
Authentication	If authentication (Plain, MAPRSASL, or Kerberos) is enabled, include the `auth` option in the connection string. If Drill is installed on a cluster secured by default security, set `auth=MAPRSASL`. For additional configuration options, refer to the *Driver Configuration Options* section in the JDBC Installation and Configuration Guide.

Connecting to a Specific Drill Node

Indicate which Drill node you want SQLLine to connect to in the JDBC connection string by using the following JDBC connection string format:

```
jdbc:drill:drillbit=<host>:<port>
```

Note that properties are case-sensitive. The `host` is the DNS or IP address of the server (Drill node). By default, the driver connects to port 31010.

Example	The following example shows how to run SQLLine with the JDBC connection string and includes the username, password, and auth parameters to authenticate to the server with Plain authentication:
	
	`/opt/mapr/drill/drill-<version>/bin/sqlline -u "jdbc:drill:drillbit=<ip-address>:<port>;auth=PLAIN" -n <username> -p <password>`
	
	If you installed Drill on a cluster with default security enabled, set the `auth` type to `maprsasl`:
	
	`/opt/mapr/drill/drill-<version>/bin/sqlline -u "jdbc:drill:drillbit=<ip-address>:<port>;auth=MAPRSASL" -n <username> -p <password>`

Connecting to ZooKeeper

When you include the ZooKeeper nodes in the JDBC connection string, ZooKeeper selects an available Drill node for SQLLine to use.

last-updated: Apr 11, 2022

Indicate the ZooKeeper cluster you want SQLLine to connect to in the JDBC connection string, using the following JDBC connection string format:

```
jdbc:drill:zk=<zk-server-list>/drill/<clustername>
```

The `zk-server-list` is a comma-separated list of the ZooKeeper nodes in the cluster. The `clustername` is the unique name of the Drillbit cluster that you want to connect to.

Important: You can locate the name of the Drillbit cluster in `/opt/mapr/drill/drill-<version>/conf/drill-distrib.conf`. The default name of the Drillbit cluster is `drillbits1`. The name is set by the `cluster-id` property. If you have multiple Drill clusters, you might want to override the Drillbit cluster name in `drill-override.conf`. However, first back-up your storage plugin configurations, as they might reset to the defaults when you change the cluster name. Restart Drill after you edit `drill-override.conf`.

Note that properties are case-sensitive. The `host` is the DNS or IP address of the server (ZooKeeper node).

Example

The following example shows you how to configure the JDBC connection string to connect SQLLine to the ZooKeeper cluster:

```
/opt/mapr/drill/drill-<version>/bin/
sqlline
jdbc:drill:zk=<node-ip>:<port>,<node-i
p>:<port>,=<node-ip>:<port>/drill/
drillbits1;auth=PLAIN
```

Note: The default port for ZooKeeper nodes in a data-fabric cluster is 5181.

If you installed Drill on a secure cluster, set the `auth` type to `maprsasl`:

```
/opt/mapr/drill/drill-<version>/bin/
sqlline
jdbc:drill:zk=<node-ip>:<port>,<node-i
p>:<port>,=<node-ip>:<port>/drill/
drillbits1;auth=MAPRSASL
```

Using a Connection-Property File with SQLLine

If you use a connection-property file, make sure you restrict user permission on the file to only those users you want to have access.

Complete the following steps to create a connection-property file and connect to Drill:

1. Create a connection-property file named `login.properties` with the following information:

```
url:<jdbc-connection-url>
user:<username>
password:<password>

//Example
cat login.properties
url:jdbc:drill:schema=dfs;drillbit=drill-lab-node01
user:drilluser
password:letsdrill
```

2. To connect to Drill, run SQLLine, as shown:

```
sqlline <sqlline args> <path/to/login.properties file>
```

The following examples show you how to connect to Drill through the connection-property file and how to verify that log in details are safe:

Example 1: Connecting to Drill via the connection-property file

```
sqlline login.properties

Java HotSpot(TM) 64-Bit Server VM
warning: ignoring option
MaxPermSize=512M; support was removed
in 8.0
apache drill 1.16.0
"drill baby drill"
0: jdbc:drill:schema=dfs> !list
1 active connection:
 #0  open
jdbc:drill:schema=dfs;drillbit=drill-l
ab-node01
0: jdbc:drill:schema=dfs>!q
```

Example 2 : Submitting a query when connecting to Drill via the connection-property file

```
sqlline -q "SELECT version FROM
sys.version" login.properties
Java HotSpot(TM) 64-Bit Server VM
warning: ignoring option
MaxPermSize=512M; support was removed
in 8.0
apache drill 1.16.0
"the only truly happy people are
children, the creative minority and
drill users"
0: jdbc:drill:schema=dfs> select
version from sys.version
. . . . . . . . . . . . . > +----------+
| version  |
+----------+
| 1.16.0   |
+----------+
1 row selected (0.295 seconds)
0: jdbc:drill:schema=dfs> Closing:
org.apache.drill.jdbc.impl.DrillConnec
tionImpl
$
```

Example 3: Use the properties command to connect to Drill via the connection-property file

```
Run sqlline from /opt/mapr/drill/
drill-<version>/bin sqlline

Java HotSpot(TM) 64-Bit Server VM
warning: ignoring option
MaxPermSize=512M; support was removed
in 8.0
apache drill 1.16.0
"a little sql for your nosql"

sqlline> !properties /home/drilluser/
login.properties
0: jdbc:drill:schema=dfs>
```

```
0: jdbc:drill:schema=dfs> !list
1 active connection:
 #0  open
jdbc:drill:schema=dfs;drillbit=drill-l
ab-node01
0: jdbc:drill:schema=dfs>
```

Example 4: Verify that Login Details are Safe

You can verify sqlline process information to confirm login details are not exposed to other users.

```
ps -ef | grep sqlline
drilluser     18938 21924 99 14:14
pts/0    00:00:03 /opt/
jdk1.8.0_141/bin/
java -XX:MaxPermSize=512M -Djava.secur
ity.auth.login.config=/opt/mapr/conf/
mapr.login.conf \
-Dzookeeper.sasl.client=false -Dhadoop
.login=simple -Dlog.path=/opt/mapr/
drill/drill-1.10.0/logs/
sqlline.log -Dlog.query.path=/opt/
mapr/drill/drill-1.16.0/logs/
sqlline_queries.json \
-cp /opt/mapr/drill/drill-1.10.0/
conf:/opt/mapr/drill/drill-1.16.0/
jars/*:/opt/mapr/drill/drill-1.16.0/
jars/ext/*:/opt/mapr/drill/
drill-1.16.0/jars/3rdparty/*:/opt/
mapr/drill/drill-1.16.0/jars/classb/*
sqlline.SqlLine -d
org.apache.drill.jdbc.Driver --maxWidt
h=10000 --color=true login.properties
drilluser     20119  1691  0 14:14
pts/1    00:00:00 grep sqlline
```

How to Protect the Password

Use the `!connect` command to mask and protect the password, as shown in the following example:

```
sqlline> !connect
jdbc:drill:drillbit=ip-10-0-0-33.eu-west-2.compute.internal:31010

Enter username for
jdbc:drill:drillbit=ip-10-0-0-33.eu-west-2.compute.internal:31010: alice
Enter password for
jdbc:drill:drillbit=ip-10-0-0-33.eu-west-2.compute.internal:31010: ********
```

Start|Stop the Drill Service

You can start|stop|restart the Drillbit service on one or more nodes by using the Control System or the following command:

```
maprcli node services -name drill-bits -action start|restart|stop -nodes
<node host names separated by a space>
```

Use the host name if possible. Using host names instead of IP addresses is a best practice.

Drill Log Files

You can access the Drill log files in `/opt/mapr/drill/drill-<version>/logs/drillbit.log`.

Using the Drill JDBC Driver with SQuirreL
You can use the Drill JDBC driver with SQuirreL to connect to Drill and query the data sources configured in Drill.

To use the Drill JDBC Driver with SQuirreL, verify that your system meets the prerequisites and then download and configure the driver.

Prerequisites

Verify that the system meets the following prerequisites:

- Java Runtime Environment (JRE), version 7.0 or later, installed on each machine where you plan to use the JDBC driver.
- Drill installed in distributed mode on one or multiple nodes in a cluster with data sources configured. See Connecting Drill to Data Sources.
- Verify that the system can resolve the hostnames of the ZooKeeper nodes of the Drill cluster. You can do this by configuring DNS for all of the systems. Alternatively, you can edit the hosts file to include the hostnames and IP addresses of all the ZooKeeper nodes used with the Drill cluster.
  - For Windows, create the entry in the `%WINDIR%\system32\drivers\etc\hosts`.
  - For Linux and Mac, create the entry in `/etc/hosts`.

    Example: `127.0.1.1 maprdemo`

Downloading and Configuring the Driver

This topic provides instructions for downloading and configuring the Drill JDBC driver for SQuirreL.

When you configure the driver, you define the driver and create an alias. The alias is a specific instance of the driver configuration. SQuirreL uses the driver definition and alias to connect to Drill so you can access data sources that you have registered with Drill. When you create the alias, you provide a connection URL that includes the name of the Drill directory stored in ZooKeeper and the cluster ID. The URL has the following format:

```
jdbc:drill:zk=<zookeeper_quorum>/<drill_directory_in_zookeeper>/<cluster_ID>
```

The following example shows a URL for Drill installed on a single node:

```
jdbc:drill:zk=10.10.100.56:5181/drill/demo_mapr_com-drillbits
jdbc:drill:zk=10.10.100.24:2181/drill/drillbits1
```

The following example shows a URL for Drill installed in distributed mode with a connection to a ZooKeeper quorum:

```
jdbc:drill:zk=10.10.100.30:5181,10.10.100.31:5181,10.10.100.32:5181/drill/
drillbits1
```


last-updated: Apr 11, 2022

Note:

- The ZooKeeper port is 2181. In a MapR cluster, the ZooKeeper port is 5181.
- The Drill directory stored in ZooKeeper is `/drill`.
- The Drill default cluster ID is `drillbits1`. To determine the cluster ID, check the following file:

  ```
  <drill-installation>/conf/drill-override.conf
  ```

  For example:

  ```
  ... drill.exec: { cluster-id: "docs41cluster-drillbits", zk.connect:
  "centos23.lab:5181,centos28.lab:5181,centos29.lab:5181" } ...
  ```

To use the Drill JDBC driver with SQuirreL, complete the following steps:

1. Download the latest Drill JDBC Driver and then unzip the file. The Drill JDBC Driver JAR files must exist in a directory on your machine before you can configure the driver in the SQuirreL client.
2. *If using the MapR-SASL or Plain authentication mechanism*, add the Drill JDBC JAR files and /opt/mapr/lib/* to Squirrel's classpath, as shown in the following example when the path to the driver is C:\driver\MapRDrillJDBC41-1.5.6.1012:

   ```
   -cp
   "%SQUIRREL_CP%;C:\driver\MapRDrillJDBC41-1.5.6.1012\*;C:\opt\mapr\lib\*"
   ```

   Note: The driver JAR files should appear before /opt/mapr/lib/* in the classpath.
3. Define the driver.
   a) Open the SQuirreL client.
   b) In the SQuirreL toolbar, select **Drivers > New Driver**. The Add Driver dialog appears.
   c) Enter the following information:
      - **Name** - Name for the Drill JDBC Driver
      - **Example URL** - `jdbc:drill:zk=<zookeeper_quorum>`

        Example: `jdbc:drill:zk=maprdemo:5181`
      - **Website URL** - `jdbc:drill:zk=<zookeeper_quorum>`

        Example: `jdbc:drill:zk=maprdemo:5181`

        Example: `jdbc:drill:zk=10.10.100.113:5181,10.10.100.115:5181`
   d) Select **Extra Class Path**, and click **Add**.
   e) Navigate to the directory that contains the JDBC JAR files.
   f) Select all of the files in the directory, and click **Choose**.
   g) In the Class Name drop-down field, select the driver class. For driver version 1.6.6.1009 and earlier, select **com.mapr.drill.jdbc41.Driver** or type **com.mapr.drill.jdbc41.Driver** in the field if the option does not appear. For driver version 1.6.7.1010, select **com.mapr.drill.jdbc.Driver** or type **com.mapr.drill.jdbc.Driver** in the field if the option does not appear.

h) Click **Ok**. The SQuirreL client displays a message stating that the driver registration is successful, and you can see the driver in the Drivers panel.

4. Create a database alias.
   a) Select the **Aliases** tab.
   b) In the SQuirreL toolbar, select **Aliases > New Alias**. The Add Alias dialog box appears.
   c) Enter the following information and click **Ok**.
      - **Alias Name** - A unique name for the Drill JDBC Driver alias
      - **Driver** - Select the Drill JDBC Driver
      - **URL** - Enter the connection URL with the name of the Drill directory stored in ZooKeeper and the cluster ID.
      - **User Name** - admin
      - **Password** - admin

      The *Connect to:* dialog appears.
   d) Click **Connect**. SQuirreL displays a message stating that the connection is successful.
   e) Click **Ok**. SQuirrel is connected to Drill through the Drill JDBC driver. You can run your queries.

Running a Drill Query from SQuirreL

Query sample data in Drill to verify that the SQuirreL client is successfully connected to the cluster through the Drill JDBC driver.

Run a test query on sample data to test the Drill connection.

To query sample data with Squirrel, complete the following steps:

1. Click the **SQL** tab.
2. Enter the following query in the query box: `SELECT * FROM cp.`employee.json`;`
3. Press **Ctrl+Enter** to run the query. The query results display.

You have successfully run a Drill query from the SQuirreL client!

Java Sample Code

To use the Drill JDBC driver in an application, you must include all of the JAR files from the ZIP archive in the classpath for the Java project.

The following Java code demonstrates how to use the JDBC API to:

- Register the driver for Drill
- Establish a connection to a Drill server
- Query the database
- Parse a result set
- Handle exceptions
- Clean up to avoid memory leakage

```
// java.sql packages are required
import java.sql.*;
```


last-updated: Apr 11, 2022

```
class DrillJDBCExample {
     // Define a string as the fully qualified class name
     // (FQCN) of the desired JDBC driver
       private static final String JDBC_DRIVER =
       "com.mapr.drill.jdbc.Driver";
     // Define a string as the connection URL
        private static final String CONNECTION_URL =
        "jdbc:drill:drillbit=192.168.1.1:31010";

            public static void main(String[] args) {
                Connection con = null;
                Statement stmt = null;
                ResultSet rs = null;
    // Define a plain query
       String query = "SELECT first_name, last_name, emp_id
       FROM `hive`.`default`.`emp`";

                try {

                    // Register the driver using the class name
                    Class.forName(JDBC_DRIVER);
                    // Establish a connection using the connection
                    // URL
                    con = DriverManager.getConnection(CONNECTION_
                    URL);
                    // Create a Statement object for sending SQL
                    // statements to the database
                    stmt = con.createStatement();

                    // Execute the SQL statement
                    rs = stmt.executeQuery(query);
                    // Display a header line for output appearing in
                    // the Console View
                    System.out.printf("%20s%20s%20s\r\n", "FIRST
                    NAME", "LAST NAME" , "EMPLOYEE ID");

                    // Step through each row in the result set
                    // returned from the database
                    while(rs.next()) {
                        // Retrieve values from the row where the

                        // cursor is currently positioned using
                        // column names
                        String FirstName = rs.getString("first_
                        name");
                        String LastName = rs.getString("last_name");
                        String EmployeeID = rs.getString("emp_id");

                    // Display values in columns 20 characters
                    // wide in the Console View using the
                    // Formatter
                    System.out.printf("%20s%20s%20s\r\n",
                    FirstName, LastName, EmployeeID);
                    }
                } catch (SQLException se) {
                    // Handle errors encountered during interaction
                    // with the data source
                } catch (Exception e) {
                    // Handle other errors
                } finally {
                    // Perform clean up
                try {
                    if (rs != null) {
                        rs.close();
```

```
                    }
                } catch (SQLException se1) {
                    // Log this
                }
                try {
                    if (stmt != null) {
                        stmt.close();
                }
                } catch (SQLException se2) {
                    // Log this
                }
                try {
                    if (con != null) {
                        con.close();
                    }
                } catch (SQLException se3) {
                    // Log this
                } // End try
            } // End try
        } // End main
    } // End DrillJDBCExample
```

Drill ODBC Driver

HPE Ezmeral Data Fabric provides a Drill ODBC driver that you can download and use on all platforms to connect BI tools, such as Tableau, to Drill.

Use the version of the driver that correlates with the version of the installed Drill server. Although older versions of the driver may be able to connect to an upgraded version of Drill, the older drivers do not include all the server features available in the newer drivers.

The following table provides links to the download locations for the Drill ODBC drivers that correlate with each of the Drill versions listed:

Drill Version	ODBC Version
1.16.0.100, 1.16.1.100	1.5.1.1002
1.16.0, 1.16.1.0	1.3.22.1055
1.15.0	1.3.22.1055
1.14.0	1.3.22.1055
1.13.0	1.3.16.1049
1.12.0	1.3.15.1048
1.11.0	1.3.15.1046
1.10.0	1.3.8.1030

Important:

- Detailed documentation for the Drill ODBC driver is available at Drill ODBC Driver.
- The 32-bit version of the Drill ODBC driver does not support MapR-SASL. MapR-SASL is only supported in the 64-bit Drill ODBC driver.
- If you plan to use MapR-SASL for authentication on Windows, review the following section for additional information and instructions.

last-updated: Apr 11, 2022

Using MapR-SASL for Authentication on Windows

Drill is automatically configured with default security when you install Drill 1.11 and later on a secure (version 6.x or later) cluster configured with the default security. To successfully connect to Drill from a Windows ODBC client, the MapR client must be installed and a `mapr` user ticket must exist on the Windows client in the %TEMP% directory or in the location specified by the $MAPR_TICKETFILE_LOCATION environment variable.

The ODBC driver locates user tickets for the current Windows user in the default ticket location, %TEMP%, or in the location specified by the environment variable, $MAPR_TICKETFILE_LOCATION. See Tickets and Generating a MapR User Ticket for more information.

You can either copy a user ticket that was generated on the cluster into the default location (%TEMP%), or you can run the `maprlogin` command to generate the ticket on the Windows client.

If you copy a user ticket that was generated on the cluster, you must copy the `mapr-clusters.conf` file to the client machine. Copy the file from `/opt/mapr/conf/mapr-clusters.conf` to `C:/opt/mapr/conf/mapr-clusters.conf` on the client machine. Verify that the cluster to which the client is connecting is listed as the first entry in the `mapr-clusters.conf` file. Also, if the cluster is secure, verify that `secure=true` for the cluster entry in the file.

Note: The ODBC user must be the same as the Windows user that created the ticket.

Example

If you want to connect to Drill as the `mapr` user, you must create a ticket for the `mapr` user, as shown:

```
$ maprlogin password -user mapr
[Password for user 'mapr' at cluster 'Cluster1':]
```

The credentials for the `mapr` user in Cluster1 are written to `/tmp/maprticket_1000`.

Next, place the ticket in the %TEMP% directory on the Windows client. For example, the default location for a Windows 10 user named Tabetha Stephens is shown:

```
'C:\Users\TABETH~1\AppData\Local\Temp/maprticket_Tabetha Stephens'
```

To override this location, set the "MAPR_TICKETFILE_LOCATION" global variable for the Windows user.

Note: Using the MAPR_TICKETFILE_LOCATION is recommended because the %TEMP% directory differs between Windows versions. You may also want to set the MAPR_TICKETFILE_LOCATION per user on the operating system to prevent all users from using the same user ticket on the client.

Driver Limitations

When using MapR-SASL with JDBC or ODBC drivers, there is no way to specify the target cluster name as part of the connection parameters. MapR-SASL reads the first entry in the `/opt/mapr/conf/mapr-clusters.conf` file and assumes it is the target cluster name.

For example, if the `mapr-clusters.conf` file has an entry for '`cluster1`' followed by an entry for '`cluster2`' and you want to connect to a node in '`cluster2`', authentication fails. As a workaround, manually switch the order of entries in the `mapr-clusters.conf` file.

Drill Configuration Files

The Drill installation includes configuration files with start-up options that you can modify prior to starting Drill.

The configuration files reside in a HOCON configuration file format, which is a hybrid between a properties file and a JSON file. The files have a nested relationship and a hierarchical structure, where one file overrides another. You can locate the files in the /opt/mapr/drill/drill-<version>/conf directory.

last-updated: Apr 11, 2022

The configuration files are listed below in their hierarchical order. The drill-distrib.conf file overrides the drill-module.conf file, and the drill-override.conf file overrides the drill-distrib.conf file.

- drill-override.conf
- drill-distrib.conf
- drill-module.conf

Environment variables are also overridden in the same way, in the order listed below:

- drill-env.sh (or explicitly defined in environment)
- distrib-env.sh
- drill-config.sh

The following table lists the configuration files with their descriptions:

File Name	Description	Default Configuration with Secure Installation
drill-distrib.conf	Contains distribution-specific configurations for Drill. Automatically updated by configure.sh when you configure the cluster.	• Enables authentication, impersonation, and encryption with MapR -SASL as the default mechanism. • Enables TLS for the HTTPS channel.  **Note:** By default, HTTPS uses the SSL certificate provided by the cluster installation; however, an administrator can specify a certificate in a keystore. • Enables the inbound impersonation policy for administrators to impersonate any other user.
distrib-env.sh	Contains distribution-specific defaults for various environment variables.	• Enables authentication between the Drillbits and ZooKeeper. • Configures the location of the default security configuration file, mapr.login.conf, used by Drill.
drill-env.sh	The drill-env.sh file contains the cluster admininstrator-specific environment variables that can differ from the defaults. You can modify this file to override the default values of system properties defined in the distrib-env.sh file or to define a new system property. For example, you can configure the amount of heap and direct memory allocated to Drill. See Configuring Drill Memory on page 3239.	Empty upon installation.

last-updated: Apr 11, 2022

drill-override.conf	Use the drill-override.conf file to override the default values obtained from drill-module.conf and drill-distrib.conf. A cluster administrator can update this file to configure a Drillbit as required (different from default installation)	When you first install Drill, drill-override.conf contains ZooKeeper and Drillbit configuration information; however, after you run configure.sh -R, the entries are removed and the file does not contain any configurations.

Monitoring Drill Metrics

You can monitor Drill metrics and logs using the Kibana and Grafana interfaces that are available through MapR Monitoring. The Kibana interface is a log monitoring tool. The Grafana interface is a metrics monitoring tool where you can view system-level metrics for Drill.

Drill uses JMX (Java Management Extensions) to monitor queries at runtime. JMX provides the architecture to dynamically manage and monitor applications. JMX collects Drill system-level metrics that you can access through Grafana or through the Metrics page in the Drill Web Console.

You must install a specific set of services on cluster nodes to use the Kibana and Grafana monitoring tools. You can install the services using the MapR installer, or you can install these services manually. If you install the monitoring services in a cluster running Drill, you must restart Drill in order for Drill to communicate with JMX. However, if you install Drill after the monitoring services are installed, you must run the `configure.sh` command and restart the Drillbit service in order for the monitoring services to recognize that a new application is running in the cluster.

The following table lists the predefined Drill system-level metrics that you can view in Grafana:

Metric	Description
mapr.drill.allocator_root_used	The amount of memory used by the internal memory allocator. Measured in bytes.
mapr.drill.queries_running	The number of queries running for which the Drillbit is the foreman.
mapr.drill.queries_completed	The number of completed, cancelled, or failed queries for which the Drillbit was the foreman.
mapr.drill.fragments_running	The number of query fragments currently running in the Drillbit.
mapr.drill.allocator_root_peak	The peak amount of memory used by the internal memory allocator. Measured in bytes.
mapr.drill.heap_used	The amount of heap memory used by the JVM. Measured in bytes.
mapr.drill.non_heap_used	The amount of non-heap memory used by the JVM. Measured in bytes.
mapr.drill.count	The number of live daemon and non-daemon threads.
mapr.drill.fd_usage	The ratio of used file descriptors to total file descriptors.
mapr.drill.runnable_count	The number of threads executing in the JVM. This metric is useful for debugging Drill issues.
mapr.drill.waiting_count	The number of threads waiting to be executed. This may occur when a thread waits on another thread to perform an action before proceeding. This metric is useful for debugging Drill issues.
mapr.drill.blocked_count	The number of blocked threads waiting for a monitor lock. This metric is useful for debugging Drill issues.

Optimizing Queries with Indexes

MapR Database provides a highly scalable key-value database platform on which you can run SQL queries using Drill. As of the 6.0 release of the MapR Data Platform, MapR Database natively supports indexes on secondary fields in JSON tables.

Note: MapR Database does not support indexes on binary tables.

An index is a special table that stores a subset of document fields from a JSON table. The primary field in a JSON table is the `_id` field (unique key field). By default, MapR Database sorts the JSON table by the `_id` field. All other fields in the JSON table are secondary fields. You can create indexes on the secondary fields in a JSON table to eliminate full tables scans and significanlty improve query performance. See MapR Database as a Document Database and Secondary Index Concepts for more information.

Benefits of Indexes

Well-designed indexes can optimize access to data stored in MapR Database JSON tables and improve performance for high read operations, fast integrated analytics, and complex operational analytics. See Secondary Indexes for more information about the benefits of indexes.

Types of Queries that Benefit from Indexes

Indexes primarily benefit queries with filters in the WHERE clause and queries with an ORDER BY clause for sorting, as described in the following table:

Query Type	Description
Equality	Equality queries contain equality conditions, such as a=1 and can also include IN. See Equality Queries.
Range	Range queries contain range conditions, such as <=, >=, and the LIKE pattern matching condition. See Range Queries. **Note:** The LIKE operator only works on fields that have varchar data types. To use the LIKE operator in queries, use the CAST function to explicitly cast fields to varchar. To use indexes for such queries, create indexes on the cast expressions, as explained in Using Casts in Secondary Indexes.
ORDER BY	ORDER BY queries specify a sort order. If the ordering and sorting of the index key list match the ordering specified in a query, the optimizer in Drill does not have to sort the data after the index scan. See ORDER BY Queries.
Multi-index	Multi-index queries contain conditions on multiple fields. Drill can scan multiple indexes and use the intersection of the matching documents to optimize these queries. Multi-index queries are an alternative to using composite key indexes. See Multi-Index Queries.

Drill can create index plans for queries with and without filters in the WHERE clause. For example, Drill can create an index plan for an ORDER BY query that does not have filters.

Drill 1.12 and later also supports the following types of queries without filters :

- GROUP BY
- JOIN
- DISTINCT

last-updated: Apr 11, 2022

See Index Planning in Drill for more information.

Types of Indexes Supported by MapR Database

MapR Database supports several types of indexes on JSON tables including simple, composite, hashed, covering, and indexes with the CAST function.

Note: MapR Database enforces certain restrictions on indexes, such as a limit of 32 KB on the collective size of all indexed keys for each index. See Restrictions on Secondary Indexes for a full list of restrictions and Data Types Supported for Secondary Indexes.

The following table lists the supported index types with brief descriptions and links to topics that provide more information:

Index Type	Description
Simple	Simple indexes are indexes with a single indexed field (or key). See Simple Indexes.
Composite	Composite indexes are indexes that have more than one indexed field (or key). See Composite Indexes.
Hashed	Hashed indexes are indexes that distribute keys across logical partitions to avoid the creation of hot spots when MapR Database updates the index with new keys from the JSON table. See Hashed Indexes.
Covering	A covering index is an index that allows MapR Database to process a query using only the secondary indexes. MapR Database does not have to read data in the JSON table. See Covering Indexes on page 560.
Indexes with the CAST function	Indexes with the CAST function convert the indexed field to the data type specified by the CAST function and store the results. See Using Casts in Secondary Indexes.

Steps Required to Use Indexes

To use the index functionality with Drill, complete the following steps:

1. Install the latest version of the required MapR software on the cluster. See Preparing Clusters for Querying using Secondary Indexes on JSON Tables and Installing Drill.
2. Evaluate your queries and design indexes that support the queries. See Understanding the Secondary Index Workflow and Designing Secondary Indexes.
3. Create indexes on JSON tables in MapR Database. See Adding Secondary Indexes on JSON Tables and Managing Secondary Indexes.

   Note: The user that creates indexes on a JSON table must have created the table or have the `indexperm` permission in addition to `readAce` on the volume and `lookupdir` on directories in the table path. If you do not have these permissions, consult with your system administrator.
4. Issue queries.
5. Verify that Drill uses the available indexes. See Determining Index Use and Troubleshooting Indexes.

Additional Information

- To see how Drill selects a query plan, see Selection and Execution of Secondary Indexes.
- To learn about the index planning and execution configuration options available in Drill, see Index Planning and Execution Configuration Options.

last-updated: Apr 11, 2022

- For information about index architecture, see Implementation of Secondary Indexes.

Index Planning in Drill
Index planning reduces the I/O operation costs associated with full table scans. If an index is available, Drill can use the index to improve query performance.

Drill can use indexes to create query plans for queries that filter on indexed fields or fields included in an index. Fields in COUNT, COUNT DISTINCT, JOIN, GROUP BY, and ORDER BY also determine index use. Drill can create index-based query plans for queries with and without filters (WHERE clause).

Note: In Drill 1.11 and earlier, if a query does not have a filter, the query must have an ORDER BY clause.

Drill can create index plans for queries with an ORDER BY clause whether or not the query contains a filter, as shown in the following example:

```
SELECT L_LINENUMBER FROM lineitem ORDER BY L_LINENUMBER;
```

Note: In this example, L_LINENUMBER is an indexed field in the index selected for the query plan.

In Drill 1.12 and later, Drill can also create index-based query plans for the following types of queries when they do not have filters (WHERE clause):

- **GROUP BY** queries, as shown in the following example where L_COMMITDate is an indexed field in the index selected for the query plan:

  ```
  SELECT L_COMMITDate FROM lineitem GROUP BY L_COMMITDate;
  ```

- **JOIN** queries, as shown in the following example where L_ORDERKEY and O_ORDERKEY are indexed fields and L_LINESTATUS is an included field in the index selected for the query plan:

  ```
  SELECT L.L_LINESTATUS FROM lineitem L, orders O WHERE
  L.L_ORDERKEY=O.O_ORDERKEY;
  ```

  Note: If the planner picks two indexes, one for lineitem and one for orders, a sort merge join is used instead of a hash join.

- Queries with **DISTINCT** projections, as shown in the following examples where L_LINENUMBER is an indexed field in the index selected for the query plan:

  ```
  SELECT DISTINCT L_LINENUMBER FROM lineitem;
  SELECT COUNT(DISTINCT L_LINENUMBER) FROM lineitem;
  ```

Drill can use indexes for queries that GROUP BY or ORDER BY the leading fields in an index. Drill does not use indexes for queries that GROUP BY or ORDER BY the trailing or included fields in an index.

When a query contains GROUP BY and ORDER BY operations on the leading indexed column, Drill can use the sort order of the index to create index-based query plans that use streaming aggregates and merge joins to improve query performance.

You can run the EXPLAIN PLAN FOR command with a query to see the query plan that Drill creates. See Covering and Non-Covering Queries for more information about index planning in Drill.

last-updated: Apr 11, 2022

Index Planning and Execution Configuration Options

The 1.11 release of Drill introduces options that affect how Drill uses indexes when planning and executing queries. You can set the query planning and execution options, at the system or session level, using the ALTER SYSTEM|SESSION SET commands, as shown:

```
ALTER SYSTEM SET `planner.enable_index_planning` = true
ALTER SESSION SET `planner.enable_index_planning` = false
```

Options set at the session level only apply to queries that you run during the current Drill connection. Options set at the system level affect the entire system and persist between restarts. Session level settings override system level settings. Typically, you set the options at the session level unless you want the setting to persist across all sessions.

The following table lists the index planning and execution options that you can enable, disable, or modify:

Note: The planning option names are prefaced by planner, for example `planner.enable_index_planning`. The execution options are prefaced by exec, for example `exec.query.rowkeyjoin_batchsize`.

Option	Description	Default Value	Possible Values
planner.enable_index_planning	Enables or disables index planning	true	true|false
planner.index.force_sort_noncovering	Forces Drill to sort for non-covering indexes. If the query has an ORDER-BY on index columns and a non-covering index is chosen, by default Drill leverages the sortedness of the index columns and does not sort. Fast changing primary table data may produce a partial sort. This option forces a sort within Drill. **Note:** (Drill 1.11 only) You must enable this option for Drill to return the results of a non-covering query in sorted order.	false	true|false
planner.enable_rowkeyjoin_conversion	Introduced in Drill 1.13. Drill can push down the rowkey filter to MapR Database during runtime. For a query to qualify for runtime filter pushdown, the join condition must filter on a rowkey. A rowkey is the value of the _id field in a JSON document, for example: `SELECT t.mscIdentities FROM dfs.root.`/user/mapr/MixTable` t WHERE t.row_key IN (SELECT max(convert_fromutf8(i.KeyA.ENTRY_KEY)) FROM dfs.root.`/user/mapr/TableIMSI` i WHERE i.row_key='460021050005636')` Drill evaluates the results of the subquery at runtime. The subquery yields a list of rowkeys from the TableIMSI table. Drill pushes down the list of rowkeys to MapR Database. MapR Database uses the rowkeys to locate the corresponding documents in the MixTable table and sends the results to Drill. **Note:** Currently, Drill does not support runtime filters for queries with equality conditions. The query planner in Drill converts an equality condition to a left join. As a workaround, use the IN operator instead of the equality (=) operator for queries in which you want Drill to push down the rowkey filter to MapR Database. Drill does not perform runtime filter pushdown for queries that filter on rowkeys in small fact tables when the rowcount is generated from the right side of the join.	true	true|false

last-updated: Apr 11, 2022

planner.rowkeyjoin_conversion_selectivity_threshold	Introduced in Drill 1.13. Sets the selectivity (as a percentage) under which Drill uses a rowkey join for eligible queries.	0.01	Range: 0.0-1.0
planner.rowkeyjoin_conversion_using_hashjoin	Introduced in Drill 1.13. When enabled, Drill uses the hash join operator instead of a rowkey join.	false	true|false
planner.index.covering_selectivity_threshold	For covering indexes, this option specifies the filter selectivity that corresponds to the leading prefix of the index below which the index is considered for planning. For example, for the filter ‘a > 10 AND b < 20’ if an index has indexed fields (a, b, c) and the combined selectivity of the above condition is less than the threshold, the index is considered for the query plan.	0.75	0 - 1.0
planner.index.noncovering_selectivity_threshold	For non-covering indexes, this option specifies the filter selectivity that corresponds to the leading prefix of the index below which the index is considered for planning.	0.025	0 - 1.0
planner.index.max_chosen_indexes_per_table	The maximum number of “chosen” indexes for a table after index costing and ranking.	5	0 - 100
planner.index.rowkeyjoin_cost_factor	The cost factor that provides some control over the I/O cost for non-covering indexes when the rowkey join back to the primary table causes random I/O from the primary table.	0.1	0 - max_double
planner.enable_statistics	Enable or disable statistics for the filter conditions on indexed columns.	true	true|false
exec.query.rowkeyjoin_batchsize	For batch GET operations, this option specifies the batch size in terms of the number of rowkeys. Used for non-covering index plans when doing joins back to primary table.	128	0 - Long.MAX_VALUE
exec.query.progress.update	Enable or disable updating transient query state in ZooKeeper. Disable this option for short running operational queries. When disabled, you do not see the query state , such as STARTING and RUNNING in the Drill Web Console.	true	true|false
exec.udf.use_dynamic	Enable or disable using dynamic UDFs for the queries. Disable this option for operational queries. When disabled, you cannot use dynamic UDFs for queries.	true	true|false
exec.query_profile.save	Enable or disable saving query profiles for the queries. Disable this option for operational queries. When disabled, Drill does not save query profiles and they are not available for analysis or debugging.	true	true|false
planner.use_simple_optimizer	Enable or disable using simple optimizer for queries. Simple optimizer applies fewer rules to reduce planning time and is meant to be used only for simple operational queries that use limit, sort, and filter. This optimizer applies rules for leveraging secondary indexes when index planning is enabled. Enable this option for operational queries.	false	true|false

Index Planning and Execution Options for Operational Queries

The following table lists the index planning and execution options for operational queries that you can enable, disable, or modify:

Option	Description	Default Value	Possible Values

last-updated: Apr 11, 2022

exec.query.progress.update	Enable or disable updating transient query state in ZooKeeper. Disable this option for short running operational queries. When disabled, you do not see the query state , such as STARTING and RUNNING in the Drill Web Console.	true	true|false
exec.udf.use_dynamic	Enable or disable using dynamic UDFs for the queries. Disable this option for operational queries. When disabled, you cannot use dynamic UDFs for queries.	true	true|false
exec.query_profile.save	Enable or disable saving query profiles for the queries. Disable this option for operational queries. When disabled, Drill does not save query profiles and they are not available for analysis or debugging.	true	true|false
planner.use_simple_optimizer	Enable or disable using simple optimizer for queries. Simple optimizer applies fewer rules to reduce planning time and is meant to be used only for simple operational queries that use limit, sort, and filter. This optimizer applies rules for leveraging secondary indexes when index planning is enabled. Enable this option for operational queries.	false	true|false

Covering and Non-Covering Queries

Drill uses a cost-based approach to determine an optimal query plan. When queries are eligible for index planning, the queries are either covering or non-covering.

For covering queries, only the index is needed to process the query. Drill creates an index-based query plan that includes an index scan. Covering queries avoid the overhead of fetching data from the primary table.

For non-covering queries, the index only contains a subset of the data required to process the query. Drill creates a query plan that includes an index scan and a join back to the primary table. In some scenarios, a full table scan is more cost efficient than an index scan and Drill will not create an index plan.

Note: (Drill 1.11 only) You must enable the `planner.index.force_sort_noncovering` option for Drill to return the results of a non-covering query in sorted order. See Index Planning and Execution Configuration Options

.

Indexes for covering and non-covering queries can contain indexed fields, or a combination of indexed and included fields. MapR Database stores included fields in the index. Each field added to the index increases the storage requirement for the index. As the storage size increases, the cost of reading the index also increases. Likewise, for the cost of adding and updating documents. Consider the impact on storage and updates when adding included fields to an index.

last-updated: Apr 11, 2022

- For information about how Drill selects a query plan, see Selection and Execution of Secondary Indexes.
- For information about the types of queries that qualify for index-based plans, see Queries that Benefit from Secondary Indexes.
- For index concepts, see Secondary Index Concepts.

Covering and Non-Covering Query Examples

A query can be covering or non-covering based on the fields referenced in the query and the fields on which an index is created and/or includes.

The following query examples use an index, *l_comp_1*, created on a table, *lineitem*.

The *l_comp_1* index was created using the `maprcli table index add` command, as shown:

```
maprcli table index add -path /drill/testdata/tpch/sf1/
maprdb/json/range/lineitem -index l_comp_1 -indexedfields
L_LINENUMBER,L_ORDERKEY -includedfields L_LINESTATUS,L_QUANTITY
```

Covering Query Example

The following query references the L_LINESTATUS, L_QUANTITY, L_LINENUMBER, and L_ORDERKEY fields in the *lineitem* table:

```
SELECT L_LINESTATUS, L_QUANTITY FROM lineitem WHERE L_LINENUMBER = 1 AND
L_ORDERKEY BETWEEN 40 AND 75;
```

Because the *l_comp_1* index includes all fields referenced in the query, Drill creates a query plan that uses the index only.

Running the EXPLAIN PLAN FOR command with the query shows that Drill created a query plan that only uses the index to process the query:

```
EXPLAIN PLAN FOR SELECT L_LINESTATUS, L_QUANTITY FROM lineitem WHERE
L_LINENUMBER = 1 AND L_ORDERKEY BETWEEN 40 AND 75;

00-00    Screen
00-01      Project(L_LINESTATUS=[$0], L_QUANTITY=[$1])
00-02        Scan(table=[[si, tpch_sf1_maprdb_range,
lineitem]], groupscan=[JsonTableGroupScan [ScanSpec=JsonScanSpec
[tableName=maprfs:///drill/testdata/tpch/sf1/maprdb/json/range/lineitem,
condition=(((L_LINENUMBER = {"$numberLong":1}) and (L_ORDERKEY
>= {"$numberLong":40})) and (L_ORDERKEY <= {"$numberLong":75})),
indexName=l_comp_1], columns=[`L_LINESTATUS`, `L_QUANTITY`]]])
```

Reading the query plan, you can see that the plan includes an index scan, as indicated by *groupscan=[JsonTableGroupScan* and *indexName*. Drill and MapR Database can process this query using only the index.

Non-Covering Query Example

The following query references the L_RETURNFLAG, L_LINESTATUS, L_QUANTITY L_LINENUMBER, and L_ORDERKEY fields in the *lineitem* table:

```
SELECT L_RETURNFLAG, L_LINESTATUS, L_QUANTITY FROM lineitem WHERE
L_LINENUMBER = 1 AND L_ORDERKEY BETWEEN 40 AND 75;
```

Because the *l_comp_1* index does not include the L_RETURNFLAG field, Drill creates a query plan that uses the index, but also includes a join on the primary table.

Running the EXPLAIN PLAN FOR command with the query shows that Drill includes an index scan and a table scan:

```
EXPLAIN PLAN FOR SELECT L_RETURNFLAG, L_LINESTATUS, L_QUANTITY FROM
lineitem WHERE L_LINENUMBER = 1 AND L_ORDERKEY BETWEEN 40 AND 75;

00-00    Screen
00-01      Project(L_RETURNFLAG=[$0], L_LINESTATUS=[$1], L_QUANTITY=[$2])
00-02        Project(L_RETURNFLAG=[$2], L_LINESTATUS=[$3], L_QUANTITY=[$4])
00-03          Project(L_LINENUMBER=[$0], L_ORDERKEY=[$1],
L_RETURNFLAG=[$2], L_LINESTATUS=[$3], L_QUANTITY=[$4])
00-04            RowKeyJoin(condition=[=($5, $6)], joinType=[inner])
00-06              Scan(table=[[si, tpch_sf1_maprdb_range,
lineitem]], groupscan=[RestrictedJsonTableGroupScan [ScanSpec=JsonScanSpec
[tableName=maprfs:///drill/testdata/tpch/sf1/maprdb/json/range/lineitem,
condition=(((L_LINENUMBER = {"$numberLong":1}) and (L_ORDERKEY
>= {"$numberLong":40})) and (L_ORDERKEY <= {"$numberLong":75}))],
columns=[`L_LINENUMBER`, `L_ORDERKEY`, `L_RETURNFLAG`, `L_LINESTATUS`,
`L_QUANTITY`, `_id`], rowcount=60012.15000000001]])
00-05              Scan(table=[[si, tpch_sf1_maprdb_range,
lineitem]], groupscan=[JsonTableGroupScan [ScanSpec=JsonScanSpec
[tableName=maprfs:///drill/testdata/tpch/sf1/maprdb/json/range/lineitem,
condition=(((L_LINENUMBER = {"$numberLong":1}) and (L_ORDERKEY
>= {"$numberLong":40})) and (L_ORDERKEY <= {"$numberLong":75})),
indexName=l_comp_1], columns=[`_id`]]])
```

Reading the query plan, you can see that the plan includes an index scan, as indicated by the *groupscan=[JsonTableGroupScan* and *indexName*, and also a scan on the primary table, as indicated by the *groupscan=[RestrictedJsonTableGroupScan* and the *RowKeyJoin*. To process this query, Drill and MapR Database can use the index, but MapR Database must also use the rowkey to perform a join on the primary table to fetch data in the L_RETURNFLAG field.

If this query ran on a regular basis, you could remove the *l_comp_1* index and create a new index that includes all fields referenced in the query, including the L_RETURNFLAG field, to improve query performance. However, running a query only once or a few times may not justify the overhead of removing the old index and creating a new index.

Non-Hashed and Hashed Indexes
You can create non-hashed and hashed indexes for queries on JSON tables in MapR Database.

Non-hashed indexes support conditional queries with an ORDER BY clause because MapR Database sorts the data in non-hashed indexes. When processing ORDER BY queries, Drill does not have to perform sort operations on the data.

Hashed indexes support the same conditional queries as non-hashed indexes, but they do not have a guaranteed sort order. Hashed indexes enable MapR Database to evenly distribute new writes on an index across logical partitions to avoid hot spotting. Drill must perform a sort for ORDER BY queries that use hashed indexes. Sorting the data can increase the CPU costs and negatively impact performance. See Hashed Indexes for additional information.

If you notice performance issues with ORDER BY queries that use hashed indexes, review the query plans to see if the plans include sort and merge operations. If this is the case, create non-hashed indexes to support the queries and achieve the best performance.

Examples of Hashed and Non-Hashed Index Plans for an ORDER BY Query

The examples here show the difference between a hashed and non-hashed index plan for the following query on the lineitem table that contains the ORDER BY clause:

```
SELECT L_LINESTATUS, L_QUANTITY FROM lineitem WHERE L_LINENUMBER = 1 AND
L_ORDERKEY BETWEEN 40 AND 75 ORDER BY L_LINENUMBER;
```

Hashed Index Plan Example

A hashed index, *l_hash_comp_1*, was created using the *maprcli table index add* command on a table, *lineitem*, as shown:

```
maprcli table index add -path /drill/testdata/tpch/sf1/
maprdb/json/hash/lineitem -index l_hash_comp_1 -indexedfields
L_LINENUMBER,L_ORDERKEY -includedfields L_LINESTATUS,L_QUANTITY -hashed true
```

Running the example query with the EXPLAIN PLAN FOR command shows that Drill produces an index plan with sort and merge operations to process the query when using the hashed index, as follows:

```
EXPLAIN PLAN FOR SELECT L_LINESTATUS, L_QUANTITY FROM lineitem WHERE
L_LINENUMBER = 1 AND L_ORDERKEY BETWEEN 40 AND 75 ORDER BY L_LINENUMBER;

00-00    Screen
00-01      Project(L_LINESTATUS=[$0], L_QUANTITY=[$1])
00-02        SingleMergeExchange(sort0=[2])
01-01          SelectionVectorRemover
01-02            Sort(sort0=[$2], dir0=[ASC])
01-03              Project(L_LINESTATUS=[$2], L_QUANTITY=[$3],
L_LINENUMBER=[$0])
01-04                Scan(table=[[si, tpch_sf1_maprdb_hash,
lineitem]], groupscan=[JsonTableGroupScan [ScanSpec=JsonScanSpec
[tableName=maprfs:///drill/testdata/tpch/sf1/maprdb/json/hash/lineitem,
condition=(((L_LINENUMBER = {"$numberLong":1}) and (L_ORDERKEY
>= {"$numberLong":40})) and (L_ORDERKEY <= {"$numberLong":75})),
indexName=l_hash_comp_1], columns=[`L_LINENUMBER`, `L_ORDERKEY`,
`L_LINESTATUS`, `L_QUANTITY`]]])
```

Reading the query plan, you can see that Drill uses the hashed index in the plan, as indicated by *indexName=l_hash_comp_1*. To process the query, MapR Database can use the index, but Drill must sort and merge the data, as indicated by the *Sort* and *SingleMergeExchange* operations in the query plan.

Using the hashed index plan for this ORDER BY query requires additional processing and negatively impacts performance.

Non-Hashed Index Plan Example

A non-hashed index, *l_comp_1*, was created using the `maprcli table index add` command on a table, *lineitem*, as shown:

```
maprcli table index add -path /drill/testdata/tpch/sf1/
maprdb/json/range/lineitem -index l_comp_1 -indexedfields
L_LINENUMBER,L_ORDERKEY -includedfields L_LINESTATUS,L_QUANTITY
```

Running the example query with the EXPLAIN PLAN FOR command shows that Drill produces an index plan without the additional sort and merge operations when using the non-hashed index to process the query, as follows:

```
EXPLAIN PLAN FOR SELECT L_LINESTATUS, L_QUANTITY FROM lineitem WHERE
L_LINENUMBER = 1 AND L_ORDERKEY BETWEEN 40 AND 75 ORDER BY L_LINENUMBER;

00-00    Screen
00-01      Project(L_LINESTATUS=[$0], L_QUANTITY=[$1])
00-02        Project(L_LINESTATUS=[$2], L_QUANTITY=[$3], L_LINENUMBER=[$0])
00-03          Scan(table=[[si, tpch_sf1_maprdb_range,
lineitem]], groupscan=[JsonTableGroupScan [ScanSpec=JsonScanSpec
[tableName=maprfs:///drill/testdata/tpch/sf1/maprdb/json/range/lineitem,
condition=(((L_LINENUMBER = {"$numberLong":1}) and (L_ORDERKEY
>= {"$numberLong":40})) and (L_ORDERKEY <= {"$numberLong":75})),
```


last-updated: Apr 11, 2022

```
indexName=l_comp_1], columns=[`L_LINENUMBER`, `L_ORDERKEY`, `L_LINESTATUS`,
`L_QUANTITY`]]])
```

Reading the query plan, you can see that Drill uses the non-hashed index plan, as indicated by *indexName=l_comp_1*. To process the query, MapR Database uses the index and Drill does not have to perform sort and merge operations on the data, as indicated by the absence of the Sort and SingleMergeExchange operations in the query plan. MapR Database sorted the data in the index when the index was created.

Writing Drill Queries that Leverage Indexes on Array Fields

Starting in EEP 6.0, the query planner in Drill can leverage indexes created on MapR Database JSON document fields with array data types, such as `"NUMBERS": [1, 2, 3, 4, 5]` and `"ADDRESSES": [{"CITY" : "SAN JOSE"}, {"CITY" : "PALO ALTO"}]`.

See JSON Document Data Types and Data Types and Secondary Index Fields for definitions and detailed examples.

If you want the query planner in Drill to leverage an index created on a field with an array data type, you must write the Drill query such that it includes specific SQL syntax, as shown in bold in the following example:

```
SELECT NAME, PHONE
FROM CUSTOMERS
WHERE _id IN ( SELECT _id
               FROM ( SELECT _id, FLATTEN(ADDRESSES) as f
          FROM CUSTOMERS) as t
               WHERE t.f.CITY = 'SAN JOSE' and t.f.STATE = 'CA')
;
```

The specific SQL syntax indicates (to the query planner in Drill) that the query is eligible for an index-based query plan.

The FLATTEN function separates elements in an array into individual records in a table. For example, if an array consists of five elements, FLATTEN separates each element into a single row, creating a table with five rows.

The IN operator prevents Drill from returning duplicate rows. For example, when an array is flattened into a table, duplicate values may exist for a particular _id (rowkey). Using IN prevents Drill from returning rows with duplicate values.

Example

Suppose a JSON primary table named CUSTOMERS exists in MapR Database with the following data:

```
{ "_id": "001",
 "NAME": "ALICE",
 "PHONE": "408-555-1212",
 "ADDRESSES": [{"CITY" : "SAN JOSE" , "ZIPCODE" : 95124, "STATE" :
"CA", "UNITS" : [{"UNIT_NO":555, "FLOOR": 5}, {"UNIT_NO":777, "FLOOR":
7}]}, {"CITY" : "PALO ALTO", "ZIPCODE" : 94020, "STATE" : "CA",
"UNITS" : [{"UNIT_NO":555, "FLOOR": 5}, {"UNIT_NO":777, "FLOOR": 7}]},
{"CITY" : "SANTA CLARA", "ZIPCODE" : 95050, "STATE" : "CA", "UNITS" :
[{"UNIT_NO":555, "FLOOR": 5}, {"UNIT_NO":777, "FLOOR": 7}]}],
  "QTY": [11, 25, 16, 2, 10, 39, 5, 8, 7, 11]
}
{ "_id": "002",
 "NAME": "BOB",
 "PHONE": "408-555-1313",
 "ADDRESSES": [{"CITY" : "SAN JOSE" , "ZIPCODE" : 95132, "STATE" :
"CA", "UNITS" : [{"UNIT_NO":838, "FLOOR": 8}, {"UNIT_NO":888, "FLOOR":
8}]}, {"CITY" : "SAN JOSE", "ZIPCODE" : 95127, "STATE" : "CA", "UNITS" :
[{"UNIT_NO":555, "FLOOR": 5}, {"UNIT_NO":777, "FLOOR": 7}]}, {"CITY" :
```

```
"SAN RAMON", "ZIPCODE" : 94582, "STATE" : "CA", "UNITS" : [{"UNIT_NO":123,
"FLOOR": 1}, {"UNIT_NO":124, "FLOOR": 1}]}],
 "QTY": [2, 8, 1, 4, 3, 10, 2, 23]
}
{ "_id": "003",
 "NAME": "CHRIS",
 "PHONE": "408-555-1414",
 "ADDRESSES": [{"CITY" : "MOUNTAIN VIEW", "ZIPCODE" : 94043, "STATE" :
"CA", "UNITS" : [{"UNIT_NO":922, "FLOOR": 9}, {"UNIT_NO":958, "FLOOR":
9}]}, {"CITY" : "PALO ALTO", "ZIPCODE" : 94020, "STATE" : "CA", "UNITS" :
[{"UNIT_NO":666, "FLOOR": 6}, {"UNIT_NO":728, "FLOOR": 7}]}, {"CITY" :
"SUNNYVALE", "ZIPCODE" : 94086, "STATE" : "CA", "UNITS" : [{"UNIT_NO":226,
"FLOOR": 2}, {"UNIT_NO":333, "FLOOR": 3}]}],
 "QTY": [56, 19, 45, 25, 4, 77, 110, 3, 2, 1]
}
```

Note: The QTY field is an array. The ADDRESSES field is an array of maps.

The following query on the CUSTOMERS table returns the result of flattening the “ADDRESSES" array field into a column aliased as “f" where each element in the array is flattened into individual rows:

Note: In the results, notice that Bob has two addresses where the “CITY" is “SAN JOSE". Later in this example, you will see that using the IN operator prevents the query from returning duplicate rows.

```
SELECT NAME, PHONE, f FROM (SELECT NAME, PHONE, FLATTEN(ADDRESSES) AS f
FROM CUSTOMERS);

+---------+-------------+-----------------------------------------------------------------

| NAME    | PHONE        |
f
                                        |
+---------+-------------+-----------------------------------------------------------------

| ALICE   | 408-555-1212 | {"CITY":"SAN JOSE","STATE":"CA","UNITS":
[{"FLOOR":5,"UNIT_NO":555},
{"FLOOR":7,"UNIT_NO":777}],"ZIPCODE":95124}
                         |
| ALICE   | 408-555-1212 | {"CITY":"PALO ALTO","STATE":"CA","UNITS":
[{"FLOOR":5,"UNIT_NO":555},
{"FLOOR":7,"UNIT_NO":777}],"ZIPCODE":94020}
         |
| ALICE   | 408-555-1212 | {"CITY":"SANTA CLARA","STATE":"CA","UNITS":
[{"FLOOR":5,"UNIT_NO":555},
{"FLOOR":7,"UNIT_NO":777}],"ZIPCODE":95050}
                            |
| BOB     | 408-555-1313 | {"CITY":"SAN JOSE","STATE":"CA","UNITS":
[{"FLOOR":8,"UNIT_NO":838},
{"FLOOR":8,"UNIT_NO":888}],"ZIPCODE":95132}
                            |
| BOB     | 408-555-1313 | {"CITY":"SAN JOSE","STATE":"CA","UNITS":
[{"FLOOR":5,"UNIT_NO":555},
{"FLOOR":7,"UNIT_NO":777}],"ZIPCODE":95127}
                            |
| BOB     | 408-555-1313 | {"CITY":"SAN RAMON","STATE":"CA","UNITS":
[{"FLOOR":1,"UNIT_NO":123},
{"FLOOR":1,"UNIT_NO":124}],"ZIPCODE":94582}
                            |
| CHRIS   | 408-555-1414 | {"CITY":"MOUNTAIN VIEW","STATE":"CA","UNITS":
[{"FLOOR":9,"UNIT_NO":922},
{"FLOOR":9,"UNIT_NO":958}],"ZIPCODE":94043}
                            |
```

```
| CHRIS   | 408-555-1414 | {"CITY":"PALO ALTO","STATE":"CA","UNITS":
[{"FLOOR":6,"UNIT_NO":666},
{"FLOOR":7,"UNIT_NO":728}],"ZIPCODE":94020}
                                |
| CHRIS   | 408-555-1414 | {"CITY":"SUNNYVALE","STATE":"CA","UNITS":
[{"FLOOR":2,"UNIT_NO":226},
{"FLOOR":3,"UNIT_NO":333}],"ZIPCODE":94086}
                 |
+---------+--------------+-----------------------------------------------------------------
```

The following query returns the results of filter conditions on the fields “CITY" and “STATE" if the CITY is SAN JOSE and STATE is CA.

```
SELECT NAME, PHONE, f FROM (SELECT NAME, PHONE, FLATTEN(ADDRESSES) AS f
FROM CUSTOMERS) AS t WHERE t.f.CITY = 'SAN JOSE' and t.f.STATE = 'CA';

+-------+--------------+-----------------------------------------------------------------

| NAME  | PHONE        |
f

|
+-------+--------------+-----------------------------------------------------------------

| ALICE | 408-555-1212 | {"CITY":"SAN JOSE","STATE":"CA","UNITS":
[{"FLOOR":5,"UNIT_NO":555},
{"FLOOR":7,"UNIT_NO":777}],"ZIPCODE":95124}
                               |
| BOB   | 408-555-1313 | {"CITY":"SAN JOSE","STATE":"CA","UNITS":
[{"FLOOR":8,"UNIT_NO":838},
{"FLOOR":8,"UNIT_NO":888}],"ZIPCODE":95132}
                          |
| BOB   | 408-555-1313 | {"CITY":"SAN JOSE","STATE":"CA","UNITS":
[{"FLOOR":5,"UNIT_NO":555},
{"FLOOR":7,"UNIT_NO":777}],"ZIPCODE":95127}
         |
+-------+--------------+-----------------------------------------------------------------
```

Suppose a composite index exists on `ADDRESSES[].CITY` and `ADDRESSES[].STATE`with "NAME" as an included field. For the query planner to use the index, you must write the query using the specific SQL syntax that indicates that the query is eligible for an index-based query plan, as shown:

```
SELECT NAME, PHONE
FROM CUSTOMERS
WHERE _id IN ( SELECT _id
               FROM ( SELECT _id, FLATTEN(ADDRESSES) as f
          FROM CUSTOMERS) as t
               WHERE t.f.CITY = 'SAN JOSE' and t.f.STATE = 'CA');

//Issuing this query against the data in the CUSTOMERS table returns the
following results:
+--------+--------------+
|  NAME  |     PHONE    |
+--------+--------------+
| ALICE  | 408-555-1212 |
| BOB    | 408-555-1313 |
+--------+--------------+
```

Note: Although Bob has two addresses where the "CITY" is "SAN JOSE", the query returns only one result. The IN operator prevents the query from returning duplicate rows.

The following list summarizes key points about this query:

- The innermost subquery projects on the _id field (rowkey) and includes the FLATTEN function to separate the array elements in the “ADDRESSES” field. The field “ADDRESSES” is flattened into a table aliased as “t”, in a column aliased as “f”.
- The query uses the IN operator to ensure that the results returned contain unique values only; no duplicates. DISTINCT on the subquery to the right of IN is implicit. The SQL query pattern indicates to the query planner that the query is eligible for an index-based query plan.
- The query projects on column “NAME” and “PHONE”. “PHONE” requires a join back to the primary table on the _id field (rowkey) because it is not included in the composite index.
- The query planner recognizes that t.f.CITY references t.ADDRESSES[].CITY and t.f.STATE references t.ADDRESSES[].STATE and creates an index-based query plan.
- The index table in MapR Database is already flattened for the array field, “ADDRESSES”. Flatten is not evaluated in Drill. Drill pushes the filter conditions on the array field into MapR Database.

Filter Conditions on Various Types of Array Fields

The following table shows examples of filter conditions on various types of array fields and includes the MapR Database notation for the array field with the filter condition, as well as the SQL syntax for writing queries against the array fields.

Note: The queries in the table are written against the CUSTOMERS data used in the previous example.

Filter condition on …	Example using MapR Database notation (not SQL notation)	SQL

Array of scalar values	QTY[] < 10	`SELECT NAME, PHONE`
		`FROM CUSTOMERS`
		`WHERE _id IN ( SELECT _id`
		`FROM ( SELECT _id, FLATTEN(Q`
		`FROM CUSTOMERS) as t`
		`WHERE t.f<10);`
		
		This query returns the following results:
		
		`+--------+---------------+`
		`|  NAME  |     PHONE     |`
		`+--------+---------------+`
		`| ALICE  | 408-555-1212  |`
		`| BOB    | 408-555-1313  |`
		`| CHRIS  | 408-555-1414  |`
		`+--------+---------------+`

last-updated: Apr 11, 2022

Map field within an array of maps	ADDRESSES[].ZIPCODE > 94000 and ADDRESSES[].ZIPCODE < 95000	<pre>SELECT NAME, PHONE
		FROM CUSTOMERS
		**WHERE _id IN (SELECT _id
		FROM (SELECT _id, FLATTEN(ADDR
		FROM CUSTOMERS) as t
		WHERE t.f.ZIPCODE BETWEEN 94000**</pre>This query returns the following results:<pre>+--------+---------------+
		| NAME | PHONE |
		+--------+---------------+
		| ALICE | 408-555-1212 |
		| BOB | 408-555-1313 |
		| CHRIS | 408-555-1414 |
		+--------+---------------+</pre>

AND-ed condition on 2 fields of the same array element	elementAND(ADDRESSES[], CITY=SAN JOSE, STATE = CA)	<pre>SELECT NAME, PHONE
		FROM CUSTOMERS
		**WHERE _id IN (SELECT _id
		FROM (SELECT _id, FLATTEN(ADDR
		FROM CUSTOMERS) as t
		WHERE t.f.CITY = 'SAN JOSE' and**</pre>This query returns the following results:<pre>+--------+---------------+
		| NAME | PHONE |
		+--------+---------------+
		| ALICE | 408-555-1212 |
		| BOB | 408-555-1313 |
		+--------+---------------+</pre>

AND-ed condition on 2 fields of different array elements	ADDRESSES[].CITY = SAN JOSE AND ADDRESSES[].ZIPCODE = 94020	<pre>SELECT NAME, PHONE
		FROM CUSTOMERS
		**WHERE _id IN (SELECT _id
		FROM (SELECT _id, FLATTEN(ADDR
		FROM CUSTOMERS) as t
		WHERE t.f1.CITY = 'SAN JOSE' an**</pre>This query returns the following results:<pre>+--------+---------------+
		| NAME | PHONE |
		+--------+---------------+
		| ALICE | 408-555-1212 |
		+--------+---------------+</pre>

<table>
<tr>
<td>AND-ed condition on scalar field and array field</td>
<td>PHONE = 408-555-1212 AND ADDRESSES[].ZIPCODE = 94020</td>
<td>

```
SELECT NAME, PHONE
FROM CUSTOMERS
WHERE _id IN ( SELECT _id
                  FROM ( SELECT _id, FLATTEN(A
                         FROM CUSTOMERS) as t
                  WHERE t.f.ZIPCODE = 94020 AN
```

This query returns the following results:

```
+--------+--------------+
|  NAME  |     PHONE    |
+--------+--------------+
| ALICE  | 408-555-1212 |
+--------+--------------+
```
</td>
</tr>
<tr>
<td>Map field within nested array of maps</td>
<td>ADDRESSES[].UNITS[].FLOOR < 5</td>
<td>

```
SELECT NAME, PHONE
FROM CUSTOMERS
WHERE _id IN ( SELECT _id
               FROM ( SELECT t1._id, flatten(t
                      FROM (SELECT _id, FLATT
                            FROM CUSTOMERS) a
               WHERE t2.u.`FLOOR` <5);
```

This query returns the following results:

```
+--------+--------------+
|  NAME  |     PHONE    |
+--------+--------------+
| BOB    | 408-555-1313 |
| CHRIS  | 408-555-1414 |
+--------+--------------+
```
</td>
</tr>
<tr>
<td>Exact match for lists or maps</td>
<td>col = ADDRESSES[].UNITS[]. {"FLOOR":7,"UNIT_NO":777}</td>
<td>

```
SELECT NAME, PHONE
FROM CUSTOMERS
WHERE _id IN ( SELECT _id
               FROM ( SELECT t1._id, flatten(t
                      FROM (SELECT _id, FLATT
                            FROM CUSTOMERS) a
               WHERE t2.u = CAST('{"FLOOR":7,"
```

This query returns the following results:

```
+--------+--------------+
|  NAME  |     PHONE    |
+--------+--------------+
| ALICE  | 408-555-1212 |
| BOB    | 408-555-1313 |
+--------+--------------+
```
</td>
</tr>
</table>

Performance Considerations

When writing queries that leverage indexes on array fields, consider the following points about performance:

- Query patterns that match those described previously in this document are pushed down to MapR Database. Drill does not evaluate the filter conditions, which adds considerable performance benefits even when the query planner does not select an index-based query plan.

last-updated: Apr 11, 2022

- Deduplication on the _id is an extra operation (compared to regular, non-complex, indexes) that requires the overhead of hash aggregation.
- Try to avoid array columns in included fields within an index table, as they add a significant amount of storage overhead. However, this may result in the query planner selecting non-covering plans.
- Indexes with deeply nested array elements, such as `a[].b[].c[]...x.y`, can add to the MapR Database storage overhead and can potentially make Drill queries longer and more complex.

Limitations

Drill queries that leverage indexes on array fields have the following limitations:

- Only queries with patterns similar to those described previously in this document are eligible for index planning, assuming that the index is defined on an array field.
- The following conditions do not produce a covering index plan:
  - Pushdown conditions on indexed fields and included fields on same array element. For example, if an index has indexed fields a[].b and included fields a[].c, elementAND(a[], b > 10, c > 20) does not produce a covering index plan.
  - Pushdown conditions on scalar indexed fields and included fields containing an array element. For example, an index with indexed field m and included fields a[].b, m = 10 AND a[].b > 20 does not produce a covering index plan.
- For included array fields, the element must be provided without the [] for the query planner to pick covering plans. For example, a and not `a[]`. Note that MapR Database considers both a and a[] syntaxes as equivalent for included fields.
- Index planning is disabled for queries with multi-level flattens and intermediate filters that reference multi-level flattens. A filter can reference the root level flatten, but not the intermediate flattens.

Determining Index Use

Evaluate the query plan to analyze query performance and determine if Drill uses indexes. You can view query plans in the Drill Web Console or through the command line using the EXPLAIN command. You can also disable the indexing option in Drill and compare an index-based plan to a full table scan plan.

Drill leverages indexes during the physical planning phase of the query. Drill estimates the cost of an index-based plan and a plan that includes a full table scan. See Selection and Execution of Secondary Indexes for information about how Drill selects a query plan. In cases where Drill does not select the index-based plan and instead selects a full table scan plan, you may want to remove the indexes to free up storage space and eliminate the overhead of the indexes.

The following example shows you how to determine if Drill selected an index-based plan for a query through the query profile in the Drill Web Console and the EXPLAIN PLAN FOR output.

Example

The subsequent sections assume that an index exists on a table named "lineitem." The index, l_single_c_5, is a single field index created on the L_QUANTITY field. The index also covers the L_SUPPKEY, L_DISCOUNT, L_SHIPDate, and L_SHIPMODE fields. If a query contains fields covered by the index, the query is a covering query. If a query contains fields not covered by the index, the query is non-covering and requires a lookup back into the primary table to retrieve data.

The following list summarizes the assumptions:

- **Table name:** lineitem
- **Index name:** l_single_c_5

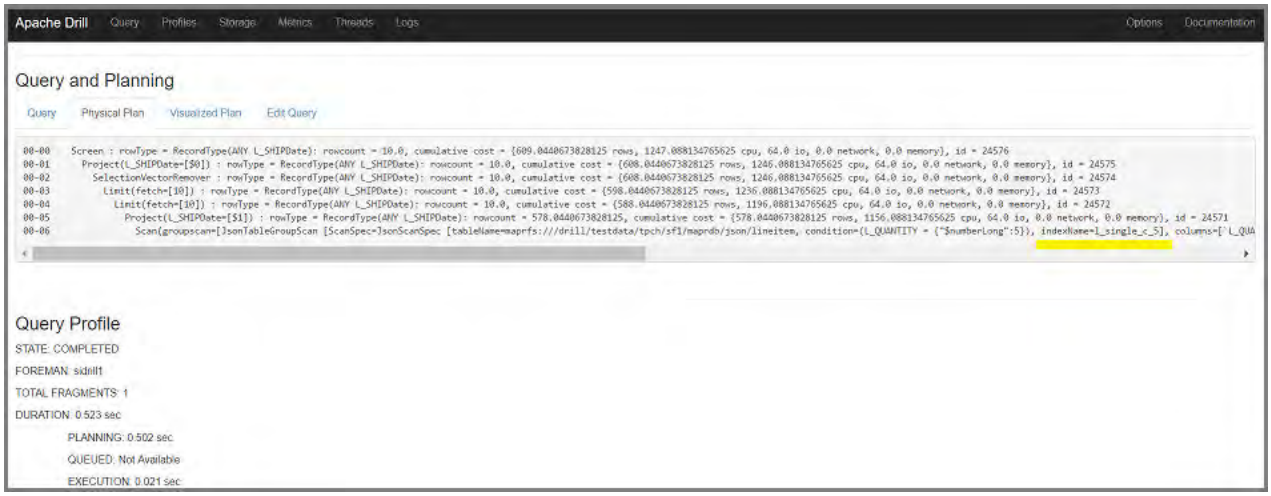
- **Indexed field:** L_QUANTITY
- **Included fields:** L_SUPPKEY, L_DISCOUNT, L_SHIPDate, L_SHIPMODE

Query Profile

View the query plan on the **Profiles** tab in the Drill Web Console. See Starting the Web Console. Select the query you want to evaluate and then select the **Physical Plan** tab. You can see the physical plan that Drill used to execute the query.

The following image shows the physical plan that Drill used to execute this simple equality query:

```
SELECT L_SHIPDate FROM lineitem WHERE L_QUANTITY = 5;
```



In the plan, you can see that Drill scanned the index, l_single_c_5, instead of the primary table. The query was completely covered by the index because the index contains all fields referenced in the query and the query filtered on the indexed field.

EXPLAIN PLAN

Alternatively, you can issue the EXPLAIN command to see how Drill executes a query. To see the chosen physical execution plan for a query without running the query, issue the EXPLAIN PLAN FOR command. This command shows you if Drill plans to use the index when executing the query.

The following image shows the physical plan that Drill plans to use to execute this simple equality query:

```
EXPLAIN PLAN FOR SELECT L_SHIPDate FROM lineitem WHERE L_QUANTITY = 5 LIMIT
10;
+------+------+
| text | json |
+------+------+
| 00-00    Screen
00-01      Project(L_SHIPDate=[$0])
00-02        SelectionVectorRemover
00-03          Limit(fetch=[10])
00-04            Limit(fetch=[10])
00-05              Project(L_SHIPDate=[$1])
00-06                Scan(groupscan=[JsonTableGroupScan
[ScanSpec=JsonScanSpec [tableName=maprfs:///drill/testdata/tpch/sf1/maprdb/
json/lineitem, condition=(L_QUANTITY = {"$numberLong":5}),
indexName=l_single_c_5], columns=[`L_QUANTITY`, `L_SHIPDate`]]])
```


last-updated: Apr 11, 2022

In the plan, you can see that Drill plans to use the index, l_single_c_5, instead of performing a full table scan. The query is completely covered by the index because the index contains all fields referenced in the query and the query filters on the indexed field.

Compare Plans

If you want to compare an index-based plan against a plan with a full table scan, disable the planner.enable_index_planning option in Drill, and run the EXPLAIN PLAN FOR command for the query. Running this command with the planner.enable_index_planning option disabled forces Drill to generate a plan that includes a full table scan. You can compare the full table scan plan against the index-based plan to compare the costs and resource consumption of each plan.

You can see in the following query, with the indexing feature turned on, Drill generated a plan using the index:

```
EXPLAIN PLAN FOR SELECT L_SHIPDate FROM lineitem WHERE L_QUANTITY = 5 LIMIT
10;
+------+------+
| text | json |
+------+------+
| 00-00    Screen
00-01      Project(L_SHIPDate=[$0])
00-02        SelectionVectorRemover
00-03          Limit(fetch=[10])
00-04            Limit(fetch=[10])
00-05              Project(L_SHIPDate=[$1])
00-06                Scan(groupscan=[JsonTableGroupScan
[ScanSpec=JsonScanSpec [tableName=maprfs:///drill/testdata/tpch/sf1/maprdb/
json/lineitem, condition=(L_QUANTITY = {"$numberLong":5}),
indexName=l_single_c_5], columns=[`L_QUANTITY`, `L_SHIPDate`]]])
```

If you turn the option off, as shown:

```
ALTER SESSION SET planner.enable_index_planning = false
```

You can run the EXPLAIN PLAN FOR command again to see the plan with a full table scan included:

```
EXPLAIN PLAN FOR SELECT L_SHIPDate FROM lineitem WHERE L_QUANTITY = 5 LIMIT
10;
+------+------+
| text | json |
+------+------+
| 00-00    Screen
00-01      Project(L_SHIPDate=[$0])
00-02        SelectionVectorRemover
00-03          Limit(fetch=[10])
00-04            UnionExchange
01-01              SelectionVectorRemover
01-02                Limit(fetch=[10])
01-03                  Project(L_SHIPDate=[$1])
01-04                    Scan(groupscan=[JsonTableGroupScan
[ScanSpec=JsonScanSpec [tableName=maprfs:///drill/testdata/tpch/sf1/maprdb/
json/lineitem, condition=(L_QUANTITY = {"$numberLong":5})],
columns=[`L_QUANTITY`, `L_SHIPDate`]]])
 |
```

Note: To see the cost of each plan, go to the Drill Web Console and view the query profile for each EXPLAIN PLAN FOR command that you issue through the command line.

Drill Limitations
Provides information about Drill limitations.

last-updated: Apr 11, 2022

Working with subqueries

- The SELECT list in a scalar subquery can only contain one item/column.
- Correlated subqueries should return exactly one row.
- A WHERE clause of a subquery should not refer to more than one column of the table in the outer query.

Flume

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Apache Flume™ is a distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of log data. It has a simple and flexible architecture based on streaming data

last-updated: Apr 11, 2022

flows. It is robust and fault tolerant with tunable reliability mechanisms and many failover and recovery mechanisms. It uses a simple extensible data model that allows for online analytic application.

This section provides information about installing, configuring, and using Flume with MapR, but does not duplicate Apache documentation. You can refer also to documentation available from the Apache Flume Project and the Flume Release Notes for MapR.

Configuring Flume

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

This section contains information about configuring Flume with MapR. Additional documentation is available on the Apache Flume website.

Configure a Secure HBase Sink

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Configure Flume agents to use Kerberos when you want to write sinks to secure HBase.

1. Create a keytab file called `flume.keytab` that contains a principal that matches the Kerberos identity of the user that will be running `flume-ng`. For example:

   ```
   # kadmin
   : addprinc -randkey username/<FQDN@REALM>
   : ktadd -k /opt/mapr/conf/flume.keytab username/<FQDN@REALM>
   ```

   The `flume.keytab` file must be owned and readable only by the `mapr` user.

2. In the `flume.conf` file, configure the following properties:

Property	Value	Description
`<agent>.sinks.<hbaseSink>.kerberosPrincipal`	username/FQDN@REALM.COM	The Kerberos identity of the user running `flume-ng`.
`<agent>.sinks.<hbaseSink>.kerberosKeytab`	path_to_keytab	The path to a valid keytab file (flume.keytab) for the user running `flume-ng`

For additional properties that you may want to configure, see the Apache Flume documentation.

Note: Once Kerberos is enabled, the maprlogin ticket generation is performed implicitly.

Configure a Secure File System Sink

When writing to the file system on a secure MapR cluster, you must configure Flume agents to use either a user ticket or a Kerberos ticket.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

A secure MapR cluster may use either SASL or Kerberos to provide authentication. Therefore, the user that launches the flume-ng JVM agent on a secure cluster can authenticate with the MapR file system using a MapR user ticket or a Kerberos ticket. When you authenticate with Kerberos, the user does not need to run the `maprlogin` utility to authenticate with the cluster as long a a valid kerberos ticket is present. When you authenticate with a mapr user ticket, you must run the maprlogin utility to generate a ticket before you launch the flume-ng JVM agent.

last-updated: Apr 11, 2022

Configure Flume agents to use MapR user tickets when writing to MapR file system

If you use MapR SASL (MapR user ticket) to authenticate, configure a dummy value for the Kerberos principal and keytab file in the `flume.conf`. Example:

```
agent1.sinks.sink1.hdfs.kerberosPrincipal = mapr
agent1.sinks.sink1.hdfs.kerberosKeytab = /opt/mapr/conf/cldb.conf
```

These dummy Kerberos principal and keytab files are not used with the HDFSSink operations. However, when the dummy Kerberos properties are not configured, Flume agent error logs display the following error messages:

```
Dec 2013 13:01:42,448 ERROR [conf-file-poller-0]
        (org.apache.flume.sink.hdfs.HDFSEventSink.authenticate:510)  -
Hadoop running in secure
        mode, but Flume config doesn't specify a principal to use for
Kerberos auth.
10 Dec 2013 13:01:42,448 ERROR [conf-file-poller-0]
        (org.apache.flume.sink.hdfs.HDFSEventSink.configure:241)  - Failed
to authenticate!
```

These errors relate to Kerberos authentication prerequisite failures and can be ignored when you are not using Kerberos. Secure Flume operations with `maprlogin`-mediated tickets continue to be available.

Configure Flume agents to use a Kerberos ticket when writing to MapR file system

1. Create a keytab file called `flume.keytab` which contains a principal that matches the Kerberos identity of the user that will be running `flume-ng`. Example:

   ```
   # kadmin
   : addprinc -randkey username/<FQDN@REALM>
   : ktadd -k /opt/mapr/conf/flume.keytab username/<FQDN@REALM>
   ```

   The `flume.keytab` file must be owned and readable only by the mapr user.

2. In the `flume.conf` file, configure the following properties:

Property	Value	Comment
`<agent>.sinks.<sink>.type`	HDFS	
`<agent>.sinks.<sink>.hdfs.proxyUser`	weblogs	
`<agent>.sinks.<sink>.hdfs.kerberosPrincipal`	username/FQDN@REALM.COM	The user component of the principal must be the username of the user running `flume-ng`.
`<agent>.sinks.<sink>.hdfs.kerberosKeytab`	*path to file*	Provide the path to your `flume.keytab` file.

Note: Flume does not support any authentication mechanism for an Avro client.

For additional properties that you may want to configure, see the Apache Flume documentation.

Flume Thrift Security Parameters
On a secure cluster, SSL for Flume Thrift Source and Sink is automatically configured.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

last-updated: Apr 11, 2022

In your MapR secure cluster, if the `flume-agent` configuration file does not specify SSL parameters, they are automatically supplied as follows:

- In EEP 6.0.0, SSL for Flume Thrift Source and Sink is automatically configured.
- In EEP 6.0.1, SSL for Flume Thrift and Avro Source and Sink is automatically configured.

Flume Thrift clients have the following configuration parameters for wire-level security:

Parameter for SSL

`ssl`	Enables SSL. If set to `true`, the `keystore` and `keystore-password` parameters must also be specified. Default: `false`. If the `ssl` parameter is not specified, it is defaulted to `true` for Thrift clients if cluster security is enabled.

Parameters for Thrift Source

`keystore`	Specifies the path to the Java keystore. The `ssl_keystore` uses the same `ssl_keystore` specified in the `ssl.server.keystore.location` section of `/opt/mapr/conf/ssl-client.xml`, `/opt/mapr/conf/ssl_keystore`.
`keystone-password`	Specifies the password for the Java keystore. The `keystore-password` uses the same password specified in the `ssl.client.keystore.password` of `/opt/mapr/conf/ssl-client.xml`, `<ssl-keystore-password>`.
`keystore-type`	Specifies the type of the Java keystore: `JKS` or `PKCS12`. The `keystore-type` uses the same `ssl_keystore` specified in the `ssl.client.keystore.type` section of `/opt/mapr/conf/ssl-client.xml`.

Parameters for Thrift Sink

`truststore`	Specifies the path to the Java truststore. The `truststore` uses the same `ssl_keystore` specified in the `ssl.client.truststore.location` section of `/opt/mapr/conf/ssl-client.xml`.
`truststore-password`	Specifies the password for the Java truststore. The `truststore-password` uses the same `ssl_keystore` specified in the `ssl.client.truststore.password` section of `/opt/mapr/conf/ssl-client.xml`.
`truststore-type`	Specifies the type of the Java keystore. This can be `JKS` or `PKCS12`. The `truststore-type` uses the same `ssl_keystore` specified in the `ssl.client.truststore.type` section of `/opt/mapr/conf/ssl-client.xml`.

Configure User Impersonation for Flume

Configure impersonation for Flume agents to allow one user (the mapr super user) to access data and submit jobs on behalf of another user.

last-updated: Apr 11, 2022

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

1. Complete the steps to enable Impersonation for the mapr Superuser.
2. Configure the `flume.conf` file with the following properties:

Property	Value	Comment
`<agent>.sinks.<sink>.type`	HDFS	
`<agent>.sinks.<sink>.hdfs.proxyUser`	*user ID of target user*	Can be any valid MapR user identity.
`<agent>.sinks.<sink>.hdfs.kerberosPrincipal`	mapr/FQDN@REALM.COM	The user component of the principal must be the username of the user running `flume-ng`. Flume agents that use impersonation must run as the user mapr on the cluster.
`<agent>.sinks.<sink>.hdfs.kerberosKeytab`	*path to file*	If you are not using Kerberos to authenticate, this can the path to any valid file. If you are using Kerberos to authenticate, provide the path to your `flume.keytab` file.

3. In the `flume-ng` file (`/opt/mapr/flume/flume-<version>/bin/flume-ng`), add export `MAPR_IMPERSONATION_ENABLED=1` after the `HADOOP_HOME` setting:

Example for Flume 1.6 and Later

```
if [ $HADOOP_VERSION == 1.0.3 ];
then
   export HADOOP_HOME=${BASEMAPR}/
hadoop/hadoop-<current_version>/
else
   export HADOOP_HOME=${BASEMAPR}/
hadoop/${HADOOP_VERSION}/
fi
export MAPR_IMPERSONATION_ENABLED=1
```

Example for Flume 1.5 and Earlier

```
HADOOP_VERSION=`readlink \`which
hadoop\` | awk -F "/" '{print$5}'`
export HADOOP_HOME=${BASEMAPR}/
hadoop/${HADOOP_VERSION}/
export MAPR_IMPERSONATION_ENABLED=1
```

Integrate Flume with MapR Event Store For Apache Kafka

Integrate Flume 1.6 with MapR Event Store For Apache Kafka when you want to write Flume source data to a MapR Event Store For Apache Kafka topic or when you want to write MapR Event Store For Apache Kafka topics to a Flume sink.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You can use Flume and MapR Event Store For Apache Kafka for many different use cases; however, here are a few examples:

Write MapR Event Store For Apache Kafka topics to file on the MapR filesystem	To write MapR Event Store For Apache Kafka topics to the MapR filesystem, create an agent with a Kafka source and a HDFS sinks.

last-updated: Apr 11, 2022

Write MapR Event Store For Apache Kafka topics to MapR Database	To write MapR Event Store For Apache Kafka topics to MapR Database, create an agent with a Kafka source and a MapR Database sink.
Write data from any supported Flume source to MapR Event Store For Apache Kafka topics	To write data from any supported Flume source to MapR Event Store For Apache Kafka topics, create an agent with the appropriate Flume source type and a Kafka sink or Kafka channel. For example, you can use a syslogs source to turn logs into a MapR stream topic.

Note: Before you integrate Flume with MapR Event Store For Apache Kafka, verify that the Streams Client is installed on all Flume nodes. For more information, see Installing Flume on page 182.

Read and Write MapR Event Store For Apache Kafka Messages With Flume

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Flume uses agent files to define the data flow. To read or write MapR Event Store For Apache Kafka data, you configure Flume agent files with Flume sources, channels, and sinks.

When you read MapR Event Store For Apache Kafka topics, you use a Kafka source or a Kafka channel. When you write to a MapR Event Store For Apache Kafka topic, you can use a Kafka channel or a Kafka sink.

See the following topics for information on how to configure the agent file:

- Read MapR Event Store For Apache Kafka Using a Kafka Source on page 3372
- Read or Write to MapR Event Store For Apache Kafka using a Kafka Channel on page 3373
- Write to MapR Event Store For Apache Kafka using a Kafka Sink on page 3374

See Example Agents Files: Flume 1.6 or Lower and MapR Event Store For Apache Kafka Integration on page 3376 and Example Agents Files: Flume 1.7 or Above and MapR Event Store For Apache Kafka Integration on page 3375 for example agent files.

Read MapR Event Store For Apache Kafka Using a Kafka Source

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following parameters are required when configuring Kafka source to read data from MapR Streams topics:

Property Name	Description
channels	If the source writes events to a channel, this is the name of the channel that the source writes events to.
type	This property must be set to `org.apache.flume.source.kafka.KafkaSource`
kafka.consumer.group.id	A unique identifier of the consumer group. This defaults to `flume`. Setting the same id in multiple sources or agents indicates that they are part of the same consumer group; when these sources or agents are running at the same time, they will each read a unique set of partitions for the topics in the group.

last-updated: Apr 11, 2022

Property Name	Description
kafka.topics	A comma separated list of MapR Event Store For Apache Kafka topics where each topic is specified with the volume path and stream name. For example: `/volume_path/stream_name:topic_name1, /volume_path/stream_name:topic_name2` **Note:** It is critical that the path to the topic starts with a slash(/), as the slash is what distinguishes the topic as a MapR Event Store For Apache Kafka topic.
batchDurationMillis	Maximum number of milliseconds to wait before closing a batch. The default value is 1000.

For additional properties that you may want to configure, see the Flume documentation. However, note that `kafka.bootstrap.servers` is not required for reading MapR Event Store For Apache Kafka.

Tip: To increase throughput, set the `batchSize` to a higher value. The `batchSize` is the maximum number of messages written to Channel in one batch. By default, it is set to 1000.

Read or Write to MapR Event Store For Apache Kafka using a Kafka Channel

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Using a Kafka channel without a Flume sink or Flume source is more reliable; it also involves less code. A Kafka channel is also useful when you want to use one agent to store messages in a topic and then copy the messages to another type of sink.

Note: Do not use a Kafka Channel when you have the following requirements:

- Maintain of the order of each event.
- Read or write data from more than one topic.
- Good performance. Writing messages to a topic with a Kafka sink can be twice as fast as a Kafka channel.

The following parameters are required when configuring Kafka channel to read or write data in MapR Event Store For Apache Kafka:

Property Name	Description
type	This property must be set to `org.apache.flume.channel.kafka.KafkaChannel`
kafka.topic	A topic is specified with the volume path and stream name. For example: `/volume_path/stream_name:topic_name1`

last-updated: Apr 11, 2022

Property Name	Description
	Note: It is critical that the path to the topic starts with a slash (/), as the slash is what distinguishes the topic as a MapR Event Store For Apache Kafka topic as opposed to a Kafka topic.
kafka.pollTimeout	The maximum amount of time in milliseconds the channel will wait for events if they are not available. The default is 500.
parseAsFlumeEvent	This must be set to `false` if other channels or sinks write to the same topic.
producer.linger.ms	These properties are used to configure the Kafka Producer. Any producer property supported by Kafka can be used. The only requirement is to prepend the property name with the prefix `kafka.producer`. For example: `kafka.producer.linger.ms=0`
capacity	The maximum number of events stored in the channel. The default value is 100.
transactionCapacity	The maximum number of events the channel will take from a source or give to a sink per transaction. The default value is 100.

For additional properties that you may want to configure for the Kafka channel, see the Flume documentation. Note that `brokerList` is not required when writing to MapR Event Store For Apache Kafka.

Write to MapR Event Store For Apache Kafka using a Kafka Sink

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following parameters are required when configuring Kafka sink to write to a MapR Stream:

Property Name	Description
type	This property must be set to `org.apache.flume.sink.kafka.KafkaSink`
kafka.topic	A topic specified with the volume path and stream name. For example: `/streaming_data/flume_stream:topic1` **Note:** It is critical that the path to the topic starts with a slash(/), as the slash is what distinguishes the topic as a MapR Event Store For Apache Kafka topic. The default is "default-flume-topic". **Note:** If the header for a given message contains a "topic" field, the message will be published to that topic instead of the topic that you configure here.

last-updated: Apr 11, 2022

For additional properties that you may want to configure for the Kafka sink, see the Flume documentation.

Note: `kafka.flumeBatchSize` sets the number of messages to process in a batch and the default is 100. To ensure that the events are written in the original order, set `kafka.flumeBatchSize=1` and specify a topic with a single partition. If maintaining the order is not required, you can increase the value. However, note that larger batches improve throughput while also adding latency.

Example Agents Files: Flume 1.7 or Above and MapR Event Store For Apache Kafka Integration

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following examples can be used to understand how you might want to configure Flume agent files in Flume version 1.7 and above.

Example: Read MapR Event Store For Apache Kafka topics and Write to MapR Filesystem

In this example, the agent reads two topics (`log_topic1 and log_topic2`), stores the event data in memory channel, and then writes the event data to a file on the filesystem (`maprfs:///flume/log_data`).

```
agent1.sources = source1
agent1.channels = channel1
agent1.sinks = sink1
agent1.sources.source1.channels = channel1
agent1.sinks.sink1.channel = channel1
agent1.sources.source1.type = org.apache.flume.source.kafka.KafkaSource
agent1.sources.source1.kafka.topics = /streaming_data/
flume_stream:log_topic1,
/streaming_data/flume_stream:log_topic2
agent1.sources.source1.kafka.consumer.group.id = flume
agent1.sources.source1.batchSize = 20
agent1.sources.source1.batchDurationMillis = 1000
agent1.sinks.sink1.type = hdfs
agent1.sinks.sink1.hdfs.path = maprfs:///flume/log_data
agent1.sinks.sink1.hdfs.filePrefix = source
agent1.sinks.sink1.hdfs.rollCount = 0
agent1.sinks.sink1.hdfs.rollInterval = 0
agent1.sinks.sink1.hdfs.rollSize = 10485760
agent1.sinks.sink1.hdfs.fileType = DataStream
agent1.channels.channel1.type = memory
agent1.channels.channel1.capacity = 10000
agent1.channels.channel1.transactionCapacity = 1000
```

Example: Read Log File and Write Log File to a Streams Topic

In this example, the agent uses an exec source to read messages from a local error log file, stores data in a channel, and then publishes the data as messages in a MapR Event Store For Apache Kafka topic (`/streaming_data/error_stream:error_log_topic`).

```
agent1.sources = source1
agent1.channels = channel1
agent1.sinks = sink1
agent1.sources.source1.channels = channel1
agent1.sinks.sink1.channel = channel1
agent1.sources.source1.type = exec
agent1.sources.source1.command = tail -f /opt/app/logs/error_log_file
agent1.channels.channel1.type = memory
agent1.channels.channel1.capacity = 10000
agent1.channels.channel1.transactionCapacity = 1000
agent1.sinks.sink1.type = org.apache.flume.sink.kafka.KafkaSink
```


last-updated: Apr 11, 2022

```
agent1.sinks.sink1.kafka.topic = /streaming_data/
error_stream:error_log_topic
agent1.sinks.sink1.flumeBatchSize = 5
```

Example: Read Log Events and Write to a MapR Filesystem File

In this example, the agent reads events from syslogtcp server, uses a Kafka channel to store events in a streams topic (`/streaming_data/flume_stream:syslogtcp_topic`), and then writes the data to a file on the filesystem (`maprfs:///flume/analytics`).

```
agent1.sources = source1
agent1.channels = channel1
agent1.sinks = sink1
agent1.sources.source1.channels = channel1
agent1.sinks.sink1.channel = channel1
agent1.sources.source1.type = syslogtcp
agent1.sources.source1.host=syslog_host
agent1.sources.source1.port=5140
agent1.channels.channel1.type = org.apache.flume.channel.kafka.KafkaChannel
agent1.channels.channel1.kafka.pollTimeout = 500
agent1.channels.channel1.kafka.topic = /streaming_data/
flume_stream:syslogtcp_topic
agent1.channels.channel1.transactionCapacity = 1000
agent1.channels.channel1.capacity = 1000
agent1.channels.channel1.producer.linger.ms=0
agent1.sinks.sink1.type = hdfs
agent1.sinks.sink1.hdfs.path = maprfs:///flume/analytics
agent1.sinks.sink1.hdfs.rollInterval = 5
agent1.sinks.sink1.hdfs.rollSize = 0
agent1.sinks.sink1.hdfs.rollCount = 0
agent1.sinks.sink1.hdfs.fileType = DataStream
```

Example Agents Files: Flume 1.6 or Lower and MapR Event Store For Apache Kafka Integration

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following examples can be used to understand how you might want to configure Flume agent files in Flume version 1.6 or lower.

Example: Read MapR Event Store For Apache Kafka topics and Write to MapR Filesystem

In this example, the agent reads two topics (`log_topic1 and log_topic2`), stores the event data in memory channel, and then writes the event data to a file on the filesystem (`maprfs:///flume/log_data`).

```
agent1.sources = source1
agent1.channels = channel1
agent1.sinks = sink1
agent1.sources.source1.channels = channel1
agent1.sinks.sink1.channel = channel1
agent1.sources.source1.type = org.apache.flume.source.kafka.v09.KafkaSource
agent1.sources.source1.kafka.topics = /streaming_data/
flume_stream:log_topic1,
/streaming_data/flume_stream:log_topic2
agent1.sources.source1.kafka.consumer.group.id = flume
agent1.sources.source1.batchSize = 20
agent1.sources.source1.batchDurationMillis = 1000
agent1.sinks.sink1.type = hdfs
agent1.sinks.sink1.hdfs.path = maprfs:///flume/log_data
agent1.sinks.sink1.hdfs.filePrefix = source
```


last-updated: Apr 11, 2022

```
agent1.sinks.sink1.hdfs.rollCount = 0
agent1.sinks.sink1.hdfs.rollInterval = 0
agent1.sinks.sink1.hdfs.rollSize = 10485760
agent1.sinks.sink1.hdfs.fileType = DataStream
agent1.channels.channel1.type = memory
agent1.channels.channel1.capacity = 10000
agent1.channels.channel1.transactionCapacity = 1000
```

Example: Read Log File and Write Log File to a Streams Topic

In this example, the agent uses an exec source to read messages from a local error log file, stores data in a channel, and then publishes the data as messages in a MapR Event Store For Apache Kafka topic (`/streaming_data/error_stream:error_log_topic`).

```
agent1.sources = source1
agent1.channels = channel1
agent1.sinks = sink1
agent1.sources.source1.channels = channel1
agent1.sinks.sink1.channel = channel1
agent1.sources.source1.type = exec
agent1.sources.source1.command = tail -f /opt/app/logs/error_log_file
agent1.channels.channel1.type = memory
agent1.channels.channel1.capacity = 10000
agent1.channels.channel1.transactionCapacity = 1000
agent1.sinks.sink1.type = org.apache.flume.sink.kafka.v09.KafkaSink
agent1.sinks.sink1.kafka.topic = /streaming_data/
error_stream:error_log_topic
agent1.sinks.sink1.flumeBatchSize = 5
```

Example: Read Log Events and Write to a MapR Filesystem File

In this example, the agent reads events from syslogtcp server, uses a Kafka channel to store events in a MapR Event Store For Apache Kafka topic (`/streaming_data/flume_stream:syslogtcp_topic`), and then writes the data to a file on the filesystem (`maprfs:///flume/analytics`).

```
agent1.sources = source1
agent1.channels = channel1
agent1.sinks = sink1
agent1.sources.source1.channels = channel1
agent1.sinks.sink1.channel = channel1
agent1.sources.source1.type = syslogtcp
agent1.sources.source1.host=syslog_host
agent1.sources.source1.port=5140
agent1.channels.channel1.type =
org.apache.flume.channel.kafka.v09.KafkaChannel
agent1.channels.channel1.kafka.pollTimeout = 500
agent1.channels.channel1.kafka.topic = /streaming_data/
flume_stream:syslogtcp_topic
agent1.channels.channel1.transactionCapacity = 1000
agent1.channels.channel1.capacity = 1000
agent1.channels.channel1.producer.linger.ms=0
agent1.sinks.sink1.type = hdfs
agent1.sinks.sink1.hdfs.path = maprfs:///flume/analytics
agent1.sinks.sink1.hdfs.rollInterval = 5
agent1.sinks.sink1.hdfs.rollSize = 0
agent1.sinks.sink1.hdfs.rollCount = 0
agent1.sinks.sink1.hdfs.fileType = DataStream
```


last-updated: Apr 11, 2022

Using Flume

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Start a Flume Agent

You can start a Flume agent with the following command:

```
 /opt/mapr/flume/flume-<version>/bin/flume-ng agent --conf-file <path to
conf file> -name <agent name>
```

Note: To send debugging output to the console, add the flag `-Dflume.root.logger=DEBUG,console`. To send debugging output to a log file, add the flag `-Dflume.root.logger=DEBUG,LOGFILE`. You can configure the location of this logfile in the `flume-<version>/conf/log4j.properties` file.

Additional Documentation

You can also refer to the following documents available on the Apache Flume website:

- Flume User Guide
- Flume Developer Guide

Related Links

- Apache Flume project
- MapR Forum posts related to Flume
- Search the MapR Blog for Flume topics

Flume 1.7.0 API

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

This section contains the following:

New API in Flume 1.7.0

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Flume 1.7.0 includes the following new classes and interfaces.

New Classes

Class	Description
org.apache.flume.source.taildir.ReliableTaildirEventReader	Creates a ReliableTaildirEventReader to watch the directory.
org.apache.flume.source.taildir.TaildirMatcher	Identifies and caches the files matched by single file pattern for TAILDIR source.
org.apache.flume.source.taildir.TaildirSource	Class is responsible for starting, stoping and configurating taildir source
org.apache.flume.source.taildir.TaildirSourceConfigurationConstants	Constants used for configuration TaildirSource

last-updated: Apr 11, 2022

org.apache.flume.source.taildir.TailFile	Class is responsible for reading and writing positions of event
org.apache.flume.auth.KerberosUser	Simple Pair class used to define a unique (principal, keyTab) combination.
org.apache.flume.conf.LogPrivacyUtil	Utility class to help any Flume component determine whether logging potentially sensitive information is allowed or not.
org.apache.flume.formatter.output.DefaultPathManager	Creates the files used by the RollingFileSink.
org.apache.flume.formatter.output.DefaultPathManager.Builder	Constructs path manager
org.apache.flume.formatter.output.PathManagerFactory	Creates PathManager instances.
org.apache.flume.formatter.output.PathManagerType	Enum that specify Builder class
org.apache.flume.formatter.output.RollTimePathManager	Appends time to end of file
org.apache.flume.formatter.output.RollTimePathManager.RollTimePathManager.Builder	Constructs path manager from given context
org.apache.flume.source.http.BLOBHandler	BLOBHandler for HTTPSource that accepts any binary stream of data as event.
org.apache.flume.source.kafka.KafkaSource.Subscriber	This class is a helper to subscribe for topics by using different strategies

New Interfaces

Interface	Description
org.apache.flume.formatter.output.PathManager.Builder	Knows how to construct this path manager. Note: Implementations MUST provide a public a no-arg constructor.

Removed API in Flume 1.7.0

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following API classes have been removed in Flume 1.7.0

Class
org.apache.flume.sink.kafka.KafkaSinkUtil
org.apache.flume.source.kafka.KafkaSourceUtil

Changed API in Flume 1.7.0

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Flume 1.7.0 includes the following changed classes.

Changed Classes

Class	Description and notes
org.apache.flume.channel.kafka.KafkaChannel	Class responsible for starting, stoping and configurating kafka channel

last-updated: Apr 11, 2022

org.apache.flume.channel.kafka.KafkaChannelConfiguration	Configuration for Kafka Channel
org.apache.flume.client.avro.ReliableSpoolingFileEventReader.Builder	Special builder class for ReliableSpoolingFileEventReader
org.apache.flume.formatter.output.BucketPath	These are useful to other classes which might want to search for tags in strings.
org.apache.flume.formatter.output.PathManager	Creates the files used by the RollingFileSink
org.apache.flume.serialization.AvroEventSerializerConfigurationConstants	Configuration constants
org.apache.flume.serialization.ResettableFileInputStream	This class makes the following assumption: The underlying file is not changing while it is being read
org.apache.flume.sink.hbase.HBaseSinkConfigurationConstants	Constants used for configuration of HBaseSink and AsyncHBaseSink
org.apache.flume.sink.kafka.KafkaSink	A Flume Sink that can publish messages to Kafka
org.apache.flume.sink.kafka.KafkaSinkConstants	Kafka sink constants
org.apache.flume.source.NetcatSourceConfigurationConstants	Configuration constants
org.apache.flume.source.SpoolDirectorySource	Track of which files have been converted into Flume events and which still need to be processed
org.apache.flume.source.SpoolDirectorySourceConfigurationConstants	Configuration constants
org.apache.flume.source.SyslogUDPSource	Process Suyslog from given source
org.apache.flume.source.kafka.KafkaSourceConstants	Configuration constants
org.apache.flume.tools.TimestampRoundDownUtil	Utility class for timestamps

Deprecated API in Flume 1.7.0

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following interfaces, classes, fields, constructors, and methods have been deprecated in Flume 1.7.0.

Deprecated Interfaces

Interface
org.apache.flume.formatter.output.EventFormatter

Deprecated Classes

Class	Use this instead
org.apache.flume.source.SyslogTcpSource	MultiportSyslogTCPSource

Deprecated Fields

Deprecated Field	Use this instead
org.apache.flume.source.SpoolDirectorySourceConfigurationConstants.BUFFER_MAX_LINE_LENGTH	

org.apache.flume.source.SpoolDirectorySourceConfigurationConstants.BUFFER_MAX_LINES	
org.apache.flume.source.SpoolDirectorySourceConfigurationConstants.DEFAULT_BUFFER_MAX_LINE_LENGTH	
org.apache.flume.source.SpoolDirectorySourceConfigurationConstants.DEFAULT_BUFFER_MAX_LINES	
org.apache.flume.channel.jdbc.ConfigurationConstants.OLD_CONFIG_CREATE_FK	ConfigurationConstants.CONFIG_CREATE_FK
org.apache.flume.channel.jdbc.ConfigurationConstants.OLD_CONFIG_CREATE_INDEX	ConfigurationConstants.CONFIG_CREATE_INDEX
org.apache.flume.channel.jdbc.ConfigurationConstants.OLD_CONFIG_CREATE_SCHEMA	ConfigurationConstants.CONFIG_CREATE_SCHEMA
org.apache.flume.channel.jdbc.ConfigurationConstants.OLD_CONFIG_DATABASE_TYPE	ConfigurationConstants.CONFIG_DATABASE_TYPE
org.apache.flume.channel.jdbc.ConfigurationConstants.OLD_CONFIG_JDBC_DRIVER_CLASS	ConfigurationConstants.CONFIG_JDBC_DRIVER_CLASS
org.apache.flume.channel.jdbc.ConfigurationConstants.OLD_CONFIG_JDBC_PROPS_FILE	ConfigurationConstants.CONFIG_JDBC_PROPS_FILE
org.apache.flume.channel.jdbc.ConfigurationConstants.OLD_CONFIG_JDBC_SYSPROP_PREFIX	ConfigurationConstants.CONFIG_JDBC_SYSPROP_PREFIX
org.apache.flume.channel.jdbc.ConfigurationConstants.OLD_CONFIG_MAX_CAPACITY	ConfigurationConstants.CONFIG_MAX_CAPACITY
org.apache.flume.channel.jdbc.ConfigurationConstants.OLD_CONFIG_MAX_CONNECTIONS	ConfigurationConstants.CONFIG_MAX_CONNECTIONS
org.apache.flume.channel.jdbc.ConfigurationConstants.OLD_CONFIG_PASSWORD	ConfigurationConstants.CONFIG_PASSWORD
org.apache.flume.channel.jdbc.ConfigurationConstants.OLD_CONFIG_TX_ISOLATION_LEVEL	ConfigurationConstants.CONFIG_TX_ISOLATION_LEVEL
org.apache.flume.channel.jdbc.ConfigurationConstants.OLD_CONFIG_URL	ConfigurationConstants.CONFIG_URL
org.apache.flume.channel.jdbc.ConfigurationConstants.OLD_CONFIG_USERNAME	ConfigurationConstants.CONFIG_USERNAME

Deprecated Constructors

Constructor
org.apache.flume.conf.FlumeConfiguration(Properties)

last-updated: Apr 11, 2022

Deprecated Methods

Deprecated Method
org.apache.flume.formatter.output.BucketPath.containsTag(string)
org.apache.flume.formatter.output.BucketPath.expandShorthand(char)
org.apache.flume.formatter.output.BucketPath.getEscapeMapping(string, Map)
org.apache.flume.formatter.output.BucketPath.getEscapeMapping(String, Map, Boolean, int, int)
org.apache.flume.api.RpcClientFactory.getInstance(string, integer)
org.apache.flume.api.RpcClientFactory.getInstance (string, integer, integer)
org.apache.flume.formatter.output.BucketPath.replaceShorthand(char, Map)
org.apache.flume.formatter.output.BucketPath.replaceShorthand(char, Map, Boolean, int, int)
org.apache.flume.formatter.output.BucketPath.replaceShorthand(char, Map, TimeZone, Boolean, int, int, Boolean)

HBase

Apache HBase™ is the Hadoop database, a distributed, scalable, big data store. You can use Apache HBase when you need random, realtime read-write access to your Big Data. This section describes how to use HBase with the MapR Platform, but does not duplicate Apache documentation.

The goal of Apache HBase is to host very large tables – billions of rows with millions of columns – atop clusters of commodity hardware. Apache HBase is an open-source, distributed, versioned, column-oriented store modeled after Google's Bigtable: A Distributed Storage System for Structured Data by Chang et al. Just as Bigtable leverages the distributed data storage provided by the Google File System, Apache HBase provides Bigtable-like capabilities on top of Hadoop and Hadoop-compatible filesystems, such as the MapR File System.

Installing Apache HBase on a MapR cluster involves storing all HBase components in a single volume mapped to directory `/hbase` in the cluster. Tables are stored in a flat namespace, not grouped logically with related files. Because all Apache HBase data resides in one volume, only one set of storage policies can be applied to the entire Apache HBase datastore. Mirrors and snapshots of the HBase volume do not provide functional replication of the datastore. Despite this limitation, mirrors can be used to back up HLogs and HFiles in order to provide a recovery point for Apache HBase data.

This section documents how to work with HBase on the MapR Converged Data Platform. You can refer also to documentation available from the Apache HBase project.

Note: The MapR Database provides native storage for table data, compatible with the HBase API. For new applications, consider using MapR Database binary tables for increased performance, more versatile table operations, and easier cluster administration.

Configuring HBase
Configure MapR-SASL Security (Authentication and Encryption) for HBase
This section describes the manual method for configuring security in HBase.

Starting with EEP 6.3.0, HBase services are secured by default with MapR-SASL. After installing HBase, you configure it by running the `$MAPR_HOME/server/configure.sh` script with the `-R` option. There are two methods to configure HBase to be secure by default:

- Automatic Method
- Manual Method

Automatic Method

If you installed HBase by using the MapR Installer, the MapR Installer configures HBase daemons during installation. Additional configuration is not required.

Manual Method

After a new manual installation, to generate a valid default ecosystem configuration, run:

```
$MAPR_HOME/server/configure.sh -R
```

Four HBase services require configuration:

- HBase Master
- HBase RegionServer
- HBase Thrift
- HBase REST

Each service can be configured for authentication and encryption, as shown later on this page:

HBase Master and RegionServer

The Master and RegionServer services require the same configuration for security.

Authentication

To enable MapR-SASL authentication, include the following property in the `hbase-site.xml` file:

```
<property>
  <name>hbase.security.authentication</name>
  <value>maprsasl</value>
</property>
```

Encryption

To enable MapR-SASL encryption, include the following property in the `hbase-site.xml` file:

```
<property>
  <name>hbase.rpc.protection</name>
  <value>privacy</value>
</property>
```

Possible values for the `hbase.rpc.protection` property are:

- `authentication (auth)`
- `integrity (auth-int)`
- `privacy (auth-conf)`

The best practice is to spell out the values (authentication/integrity/privacy). The abbreviated values (in parentheses) can work, but using them is not recommended. Encryption is enabled only for the highest level of security (`privacy`).

HBase Thrift

It is possible to configure the HBase Thrift service to work over sockets or over the HTTP protocol. For authentication purposes, configuration is the same for both cases. For encryption, configuration is different for each case. Note that starting with the EEP6.3.0 property,

last-updated: Apr 11, 2022

`hbase.thrift.security.authentication` is no longer used to configure HBase Thrift for authentication.

Authentication

HBase Thrift relies on the same property used for Master and RegionServer. To enable authentication, include the following property in the `hbase-site.xml` file:

```
<property>
  <name>hbase.security.authentication</name>
  <value>maprsasl</value>
</property>
```

Encryption for Thrift over Sockets

To enable encryption with MapR-SASL for Thrift over sockets, make sure that the `hbase.regionserver.thrift.http` property is set to `false` and the following property is present in the `hbase-site.xml` file:

```
<property>
  <name>hbase.thrift.security.qop</name>
  <value>auth-conf</value>
</property>
```

Possible values for `hbase.thrift.security.qop` are:

- `auth`
- `auth-int`
- `auth-conf`

Encryption is enabled only for the highest level of security (`auth-conf`).

Encryption for Thrift over HTTP

To enable Thrift to work over the HTTP protocol, include the following property in the `hbase-site.xml` file:

```
<property>
  <name>hbase.regionserver.thrift.http</name>
  <value>true</value>
</property>
```

To enable Thrift over HTTP encryption through SSL, include the following property in the `hbase-site.xml` file:

```
<property>
  <name>hbase.thrift.ssl.enabled</name>
  <value>true</value>
</property>
```

HBase REST

Authentication

To enable HBase REST authentication, include the following property in the `hbase-site.xml` file:

```
<property>
  <name>hbase.rest.authentication.type</name>
```

```
<value>org.apache.hadoop.security.authentication.server.MultiMechsAuthentica
tionHandler</value>
</property>
```

With the MultiMechsAuthenticationHandler, MapR-SASL, Kerberos, and PAM authentication headers are supported. A custom AuthenticationHandler could be implemented and specified with the full class name in this property.

Encryption

To enable HBase REST SSL encryption, include the following property in the `hbase-site.xml` file:

```
<property>
  <name>hbase.rest.ssl.enabled</name>
  <value>true</value>
</property>
```

HBase Services Web UIs

Web UIs are available for each HBase service. The Web UIs run simultaneously with the service and within the same process. Security for these UIs must be configured too.

Authentication

To enable HBase Web UI authentication, include the following property in the `hbase-site.xml` file:

```
<property>
  <name>hbase.security.authentication</name>
  <value>maprsasl</value>
</property>
```

Authentication is implemented through the MultiMechsAuthenticationHandler and therefore supports MapR-SASL, Kerberos, and PAM authentication headers.

Encryption

To enable HBase Web UI SSL encryption, include the following property in the `hbase-site.xml` file:

```
<property>
  <name>hbase.ssl.enabled</name>
  <value>true</value>
</property>
```

Configure HBase to use Kerberos

HBase supports MapR-SASL and Kerberos security, and can run securely independently of the security status of your MapR cluster.

To configure HBase to use Kerberos, perform the following steps:

1. Install the `mapr-hbase-master` and `mapr-hbase-regionserver` packages on the cluster.
2. On all HBase nodes, perform the following steps:
   a) Install the `krb5` packages and configure the Kerberos client as per the configuration for your environment.
   b) Set up the HBase Kerberos principal `mapr/<fqdn>@<realm>`. Each node requires a unique keytab file and Kerberos identity.
   c) Create an `hbase.keytab` file with the HBase Kerberos principal with the same process used to generate the CLDB keytab.
   d) Copy the `hbase.keytab` file to the `/opt/mapr/conf` directory.

last-updated: Apr 11, 2022

e) Use the `chown` command to change the keytab file's ownership to `mapr:mapr`.

f) Use the `chmod` command to set the file's permissions to `600`.

g) Update the `hbase-site.xml` file by adding the following section:

```
<property>
   <name>hbase.security.authentication</name>
   <value>kerberos</value>
 </property>
 <property>
  <name>hbase.security.authorization</name>
   <value>true</value>
 </property>
 <property>
   <name>hbase.regionserver.kerberos.principal</name>
   <value>mapr/_HOST@<KERBEROS_REALM></value>
 </property>
 <property>
   <name>hbase.master.kerberos.principal</name>
  <value>mapr/_HOST@<KERBEROS_REALM></value>
 </property>
```

h) On a MapR cluster with security features enabled, replace the `${SIMPLE_LOGIN_OPTS}` value of the `MAPR_HBASE_SERVER_OPTS` property with `${KERBEROS_LOGIN_OPTS}` and the value of the `MAPR_HBASE_CLIENT_OPTS` property with `${HYBRID_LOGIN_OPTS}`. Also remove the `-Dzookeeper.sasl.client=false` option from the definition of `MAPR_HBASE_CLIENT_OPTS`.

These properties are located in the `/opt/mapr/conf/env.sh` file.

i) On a MapR cluster with security features disabled, replace the `${SIMPLE_LOGIN_OPTS}` value of the `MAPR_HBASE_SERVER_OPTS` and `MAPR_HBASE_CLIENT_OPTS` properties in the `/opt/mapr/conf/env.sh` file with `${KERBEROS_LOGIN_OPTS}`.

3. On all HBase regionserver nodes, update the `hbase-site.xml` file by adding the following section:

```
<property>
   <name>hbase.regionserver.keytab.file</name>
   <value>/opt/mapr/conf/hbase.keytab</value>
 </property>
 <property>
   <name>hbase.coprocessor.region.classes</name>
   <value>
org.apache.hadoop.hbase.security.token.TokenProvider,org.apache.hadoop.hb
ase.security.access.AccessController</value>
 </property>
```

4. On the HBase master node, update the `hbase-site.xml` file by adding the following section:

```
<property>
  <name>hbase.master.keytab.file</name>
  <value>/opt/mapr/conf/hbase.keytab</value>
</property>
<property>
  <name>hbase.coprocessor.master.classes</name>
  <value>org.apache.hadoop.hbase.security.access.AccessController</value>
</property>
```

5. Restart the HBase master and regionserver nodes.

Enable Impersonation for HBase

HBase can be configured to offer impersonation, with or without Kerberos. This means that users can send commands to HBase through Hue without losing the fact that they will be run under their own credentials, instead of the `hue` user.

For instructions, see Enable Impersonation for HBase Thrift1 Gateway.

Configure HBase ACLs

HBase supports Access Control Lists (ACLs) to limit the privileges of users on the system. Before you can use ACLs, you need to perform the steps to enable ACLs.

HBase ACLs support the following privileges:

- `Read`
- `Write`
- `Execute`
- `Create tables`
- `Administrator`

The possible scopes are:

- `Superuser`
- `Global`
- `Namespace`
- `Table`
- `ColumnFamily`
- `Cell`

For information about each scope, see Understanding Access Levels.

Once you enable the use of ACLs, you can grant and remove privileges from users by using the `grant` and `revoke` commands from the HBase shell. The following example grants user `jfoo` read privileges from column family `cf1` of table `mytable`:

```
hbase(main):001:0> grant 'jfoo' 'R' 'mytable','cf1'
```

This example removes user `kbar`'s administrative privileges on the cluster:

```
hbase(main):001:0> revoke 'kbar' 'A'
```


last-updated: Apr 11, 2022

Enable HBase Access Control
The following steps explain how to enable HBase ACLs.

1. On the HBase Region Server, edit the `/opt/mapr/hbase/hbase-<version>/conf/hbase-site.xml` file, and add the following section:

```
<property>
    <name>hbase.coprocessor.region.classes</name>

<value>org.apache.hadoop.hbase.security.token.TokenProvider,org.apache.hadoop.hbase.security.access.AccessController</value>
</property>
<property>
    <name>hbase.superuser</name>
    <value><admin1>,<admin2>,@<group1>,...</value> <!-- group names are prefixed with '@' -->
  </property>
```

2. On the HBase Master, edit the `/opt/mapr/hbase/hbase-<version>/conf/hbase-site.xml` file, and add the following section:

```
<property>
    <name>hbase.coprocessor.master.classes</name>
    <value>org.apache.hadoop.hbase.security.access.AccessController</value>
</property>
<property>
    <name>hbase.superuser</name>
    <value><admin1>,<admin2>,@<group1>,...</value> <!-- group names are prefixed with '@' -->
  </property>
```

3. Restart HBase on every node.

Set Up Compression with HBase
Using compression with HBase reduces the number of bytes transmitted over the network and stored on disk. These benefits often outweigh the performance cost of compressing the data on every write and uncompressing it on every read.

GZip Compression

GZip compression is included with most Linux distributions and works natively with HBase. To use GZip compression, specify it in the per-column family compression flag while creating tables in HBase shell. For example:

```
create 'mytable', {NAME=>'colfam', COMPRESSION=>'gz'}
```

LZ4 Compression

The LZ4 algorithm gives a slightly worse compression ratio than the LZO algorithm – which in turn is worse than algorithms like DEFLATE. However, compression speeds are similar to LZO and several times faster than DEFLATE, while decompression speeds can be significantly higher than LZO. Here is an example of configuring LZ4 compression:

```
create 'mytable1', {NAME=>'colfam', COMPRESSION=>'lz4'}
```

Snappy Compression

The Snappy compression algorithm is optimized for speed over compression. Snappy compression is included in the core MapR installation, and no additional configuration is required.

Configure the Default Database for HBase Clients

For HBase version 1.1 and later, a default database configuration determines whether clients connect to HBase tables or MapR Database tables. You can change the default setting for all HBase clients, or you can set the database for a particular job. This setting is ignored for HBase 0.98.12 client connections.

Set the Default Database using configure.sh

`configure.sh` automatically sets the default database for a node based on the presence of the following `mapr` packages:

- `mapr-hbase-master`
- `mapr-hbase-regionserver`

`configure.sh` also provides a parameter that you can use to set the default database. To explicitly set the default database to either `maprdb` or `hbase`, run `configure.sh` with the `-defaultdb` parameter. For example:

```
configure.sh -R -defaultdb maprdb
```

The following table describes the effect of various `configure.sh` commands on the default database setting:

Table

If ...	And you run `configure.sh` with the following `-defaultdb` parameter ...		
	No `-defaultdb` parameter specified	`-defaultdb hbase`	`-defaultdb maprdb`
`hbasemaster` or `hbaseregionserver` is installed on a node	The default database is set to `hbase`.	The default database is set to `hbase`	The default database is set to `hbase`, and the `maprdb` setting is ignored.
`hbasemaster` and `hbaseregionserver` are NOT installed on a node	The default database is set to `maprdb`.	The default database is set to `hbase`. However, this configuration does not work because HBase is not running.	The default database is set to `maprdb`.

For more information about `configure.sh`, see configure.sh on page 2053.

Set the Default Database using hbase-site.xml

You can configure the `mapr.hbase.default.db` property in the `hbase-site.xml` to override the default database that is set for the cluster:

1. In the `hbase-site.xml`, edit the default value of `mapr.hbase.default.db`, and set it to either `hbase` or `maprdb`.

   For example:

   ```
   <property>
       <name>mapr.hbase.default.db</name>
       <value>hbase</value>
   </property>
   ```

2. Copy the property to the `hbase-site.xml` on each node that runs HBase, including any HBase client nodes.

last-updated: Apr 11, 2022

Set the Database Type in the Job Configuration

To set the database type in the job configuration you can add the following code:

- To connect to MapR Database tables:

```
Configuration conf = HBaseConfiguration.create();
conf.set("mapr.hbase.default.db", "maprdb");
Connection connection = ConnectionFactory.createConnection(conf);
Table table = connection.getTable(<TABLE_NAME>);
```

- To connect to HBase tables:

```
Configuration conf = HBaseConfiguration.create();
conf.set("mapr.hbase.default.db","hbase");
Connection connection = ConnectionFactory.createConnection(conf);
Table table = connection.getTable(<TABLE_NAME>);
```

HBase Configuration Properties
This section describes and shows examples of the configuration properties used in the `hbase-site.xml` file.

Table

Category	Property	Description
Basic Properties	`Hbase.rootdir`	Specifies where the HBase data is stored. If not specified, by default HBase uses the `/tmp/` local folder. It is possible to use the local filesystem and MapR File System or a remote MapR File System instance.
	<property> <name>hbase.rootdir</name> <value>maprfs:///hbase</value> </property>	
	`HBase.cluster.distributed`	The mode the cluster will be in. Possible values are `false` for standalone mode and `true` for distributed mode. If `false`, startup runs all HBase and ZooKeeper daemons together in the one JVM. **Default:** `false`.
	<property> <name>hbase.cluster.distributed</name> <value>true</value> </property>	
	`Hbase.zookeeper.quorum`	Comma-separated list of servers in the ZooKeeper ensemble. For example, "host1.mydomain.com,host2.mydomain.com,host3.mydomain.com". By default this property is set to localhost for local and pseudo-distributed modes of operation. For a fully-distributed setup, this property should be set to a full list of ZooKeeper ensemble servers. If HBASE_MANAGES_ZK is set in `hbase-env.sh`, this is the list of servers that HBase will start or stop ZooKeeper on as part of cluster start or stop. Client-side, we will take this list of ensemble members and put it together with the `hbase.zookeeper.property.clientPort config.` and pass it into the Zookeeper constructor as the `connectString` parameter. Port could be specified together with hosts. In this case, the `hbase.zookeeper.property.clientPort` configuration is useless.
	<property> <name>hbase.zookeeper.quorum</name> <value>node11.cluster.com:5181</value> </property>	
	`Dfs.support.append`	Specifies whether DFS allows appends to files.
	<property> <name>dfs.support.append</name> <value>true</value> </property>	
	`Hbase.fsutil.maprfs.impl`	Specifies the FSUtil class (the utility methods for interacting with the underlying filesystem) used in HBase.
	<property> <name>hbase.fsutil.maprfs.impl</name> <value>org.apache.hadoop.hbase.util.FSMapRUtils</value> </property>	
	`Hbase.regionserver.handler.count`	Sets the count of RPC Listener instances spun up on RegionServers. The Same property is used by the Master for a count of master handlers. Too many handlers can be counter-productive. Make it a multiple of the CPU count. If mostly read-only, handlers count close to CPU count does well. Start with

last-updated: Apr 11, 2022

Table (Continued)

Category	Property	Description
Security Properties*	`Hbase.security.authorization`	Specifies whether authorization is enabled or not.
	<property> <name>hbase.security.authorization</name> <value>true</value> </property>	
	`Hbase.security.exec.permission.checks`	Without this option, all users continue to have access to execute endpoint coprocessors. This option is not enabled when you enable HBase Secure Authorization for backward compatibility.
	<property> <name>hbase.security.exec.permission.checks</name> <value>true</value> </property>	
	`hbase.coprocessor.master.classes`	A comma-separated list of coprocessors that are loaded by the master (MasterObserver coprocessors). The AccessController has to be active to support authorization.
	<property> <name>hbase.coprocessor.master.classes</name> <value>org.apache.hadoop.hbase.security.access. AccessController</value> </property>	
	`Hbase.coprocessor.region.classes`	A comma-separated list of RegionObserver and Endpoint coprocessors. TokenProvider and AccessController must be active to support authorization.
	<property> <name>hbase.coprocessor.region.classes</name> <value>org.apache.hadoop.hbase.security.token.TokenProvider. org.apache.hadoop.hbase.security.access.AccessController</value> </property>	

Table (Continued)

Category	Property	Description
Authentication and Encryption Properties	`hbase.security.authentication`	Defines whether to use SASL mechanisms in HBase to authenticate RPC connections from clients to HBase Master and RegionServer. Also defines whether to support authentication for HBaseThrift. Specifying `maprsasl` enables authentication for HBaseThrift over http.
	<property> <name>hbase.security.authentication</name> <value>maprsasl</value> </property>	
	`hbase.rpc.protection`	Enables or disables transport security encryption. To support encryption, the `auth-conf` (privacy) value must be specified. Possible values are: • `auth` or `authentication` • `auth-int` or `integrity` • `auth-conf` or `privacy`
	<property> <name>hbase.rpc.protection</name> <value>auth-conf</value> </property>	
	`hbase.ssl.enabled`	Enables or disables SSL encryption for HBase WebUIs.
	<property> <name>hbase.ssl.enabled</name> <value>true</value> </property>	
	`hbase.thrift.ssl.enabled`	Enables or disables SSL encryption for HBaseThrift. Works only for HBaseThrift over http (the `hbase.regionserver.thrift.http` property must be set to `true`).
	<property> <name>hbase.thrift.ssl.enabled</name> <value>true</value> </property>	
	`Hbase.thrift.security.qop`	Enables or disables transport security encryption for HBaseThrift. Use the `Auth-conf` value to support encryption. This property works only for HBaseThrift over sockets (the `hbase.regionserver.thrift.http` property must be set to `false`). Possible values are: • `auth` • `auth-int` • `auth-conf`
	<property> <name>hbase.thrift.security.qop</name> <value>auth-conf</value> </property>	

last-updated: Apr 11, 2022

Table (Continued)

Category	Property	Description
Impersonation Properties	`hbase.thrift.support.proxyuser`	Enables or disables impersonation for HBaseThrift. Works only for thrift over http (the `hbase.regionserver.thrift.http` property must be set to `true`).
	<property> <name>hbase.thrift.support.proxyuser</name> <value>true</value> </property>	
	`hbase.rest.support.proxyuser`	Enables or disables impersonation for HBaseRest.
	<property> <name>hbase.rest.support.proxyuser</name> <value>true</value> </property>	
	`hbase.regionserver.thrift.http`	Defines whether to use HBaseThrift over http (if `true` is specified) or over sockets. Used to support impersonation for thrift over http.
	<property> <name>hbase.regionserver.thrift.http</name> <value>true</value> </property>	

*To support authorization, four properties must be enabled:

- `hbase.security.authorization`
- `hbase.security.exec.permission.checks`
- `hbase.coprocessor.master.classes`
- `hbase.coprocessor.region.classes`

If any of them is missing, authorization will not be fully supported.

Using HBase

Related Links

- Apache HBase Reference Guide
- Apache HBase project
- Search the MapR Blog for HBase topics

This section includes the following topics about working with HBase:

Getting Started in HBase

In this section, we'll create an HBase table on the cluster, enter some data, query the table, then clean up the data and exit.

HBase tables are organized by column, rather than by row. Furthermore, the columns are organized in groups called *column families*. When creating an HBase table, you must define the column families before inserting any data. Column families should not be changed often, nor should there be too many of them, so it is important to think carefully about what column families will be useful for your particular data. Each column family, however, can contain a very large number of columns. Columns are named using the format `family:qualifier`.

Unlike columns in a relational database, which reserve empty space for columns with no values, HBase columns simply don't exist for rows where they have no values. This not only saves space, but means that different rows need not have the same columns; you can use whatever columns you need for your data on a per-row basis.

1. Start the HBase shell by typing the following command:

```
hbase shell
```

2. Create a table called `weblog` with one column family named `stats`:

```
create 'weblog', 'stats'
```

3. Verify the table creation by listing everything:

```
list
```

4. Add a test value to the `daily` column in the `stats` column family for row 1:

```
put 'weblog', 'row1', 'stats:daily', 'test-daily-value'
```

5. Add a test value to the `weekly` column in the `stats` column family for row 1:

```
put 'weblog', 'row1', 'stats:weekly', 'test-weekly-value'
```

6. Add a test value to the `weekly` column in the `stats` column family for row 2:

```
put 'weblog', 'row2', 'stats:weekly', 'test-weekly-value'
```

7. Type `scan 'weblog'` to display the contents of the table. Sample output:

```
ROW                  COLUMN+CELL
 row1                column=stats:daily, timestamp=1321296699190,
value=test-daily-value
 row1                column=stats:weekly, timestamp=1321296715892,
value=test-weekly-value
 row2                column=stats:weekly, timestamp=1321296787444,
value=test-weekly-value
2 row(s) in 0.0440 seconds
```

8. Type `get 'weblog', 'row1'` to display the contents of row 1. Sample output:

```
COLUMN               CELL
 stats:daily         timestamp=1321296699190, value=test-daily-value
 stats:weekly        timestamp=1321296715892, value=test-weekly-value
2 row(s) in 0.0330 seconds
```


last-updated: Apr 11, 2022

9. Type `disable 'weblog'` to disable the table.

10. Type `drop 'weblog'` to drop the table and delete all data.

11. Type `exit` to exit the HBase shell.

Running MapReduce Jobs with HBase

To run MapReduce applications with data stored in HBase, use a command such as the following to export table data to the MapR filesystem:

```
$ hadoop jar /opt/mapr/hbase/hbase-1.1.13/lib/
hbase-server-1.1.13.0-mapr-1912.jar export t1 /user/mapr/t1
```

or

```
$  hbase org.apache.hadoop.hbase.mapreduce.Export t1 /user/mapr/t4
```

The result is the same because of the tools included in the hbase-server.jar file:

```
$ hadoop fs -ls /user/mapr/t1/
Found 2 items
-rwxr-xr-x   3 mapr mapr          0 2019-11-11 15:00 /user/mapr/t1/_SUCCESS
-rw-r--r--   3 mapr mapr        249 2019-11-11 15:00 /user/mapr/t1/
part-m-00000
$ hadoop fs -ls /user/mapr/t4/
Found 2 items
-rwxr-xr-x   3 mapr mapr          0 2019-11-11 15:09 /user/mapr/t4/_SUCCESS
-rw-r--r--   3 mapr mapr        249 2019-11-11 15:09 /user/mapr/t4/
part-m-00000
$
```

Following is an example of the full output:

```
$ hadoop jar /opt/mapr/hbase/hbase-1.1.13/lib/
hbase-server-1.1.13.0-mapr-1912.jar export t1 /user/mapr/t1
19/11/11 14:59:41 INFO mapreduce.Export: versions=1, starttime=0,
endtime=9223372036854775807, keepDeletedCells=false
19/11/11 14:59:42 INFO mapreduce.TableMapReduceUtil: Configured
mapr.hbase.default.db hbase
19/11/11 14:59:42 INFO client.ConnectionFactory: ConnectionFactory receives
mapr.hbase.default.db(hbase), set clusterType(HBASE_ONLY), user(mapr),
hbase_admin_connect_at_construction(false)
19/11/11 14:59:42 INFO zookeeper.RecoverableZooKeeper: Process
identifier=TokenUtil-getAuthToken connecting to ZooKeeper
ensemble=node5.cluster.com:5181
19/11/11 14:59:43 INFO zookeeper.RecoverableZooKeeper: Process
identifier=hconnection-0x2c306a57 connecting to ZooKeeper
ensemble=node5.cluster.com:5181
19/11/11 14:59:43 INFO client.ConnectionManager$HConnectionImplementation:
Closing zookeeper sessionid=0x100044f486eff26
19/11/11 14:59:45 INFO impl.TimelineClientImpl: Timeline service address:
https://node5.cluster.com:8190/ws/v1/timeline/
19/11/11 14:59:45 INFO client.MapRZKBasedRMFailoverProxyProvider: Updated
RM address to node5.cluster.com/192.168.33.15:8032
19/11/11 14:59:47 INFO client.ConnectionFactory: mapr.hbase.default.db
unsetDB is neither MapRDB or HBase, set HBASE_MAPR mode since mapr client
is installed.
19/11/11 14:59:47 INFO client.ConnectionFactory: ConnectionFactory receives
mapr.hbase.default.db(unsetDB), set clusterType(HBASE_MAPR), user(mapr),
hbase_admin_connect_at_construction(false)
19/11/11 14:59:47 INFO zookeeper.RecoverableZooKeeper: Process
```

```
identifier=hconnection-0x6b63e6ad connecting to ZooKeeper
ensemble=node5.cluster.com:5181
19/11/11 14:59:48 INFO client.ConnectionManager$HConnectionImplementation:
Closing master protocol: MasterService
19/11/11 14:59:48 INFO client.ConnectionManager$HConnectionImplementation:
Closing zookeeper sessionid=0x100044f486eff2a
19/11/11 14:59:48 INFO mapreduce.JobSubmitter: number of splits:1
19/11/11 14:59:48 INFO mapreduce.JobSubmitter: Submitting tokens for job:
job_1572957695341_0001
19/11/11 14:59:48 INFO mapreduce.JobSubmitter: Kind:
HBASE_AUTH_TOKEN, Service: 9161aa11-2f19-4b20-82f8-9678db86e0a7, Ident:
(org.apache.hadoop.hbase.security.token.AuthenticationTokenIdentifier@0)
19/11/11 14:59:49 INFO security.ExternalTokenManagerFactory:
Initialized external token manager class -
com.mapr.hadoop.yarn.security.MapRTicketManager
19/11/11 14:59:51 INFO impl.YarnClientImpl: Submitted application
application_1572957695341_0001
19/11/11 14:59:51 INFO mapreduce.Job: The url to track the job: https://
node5.cluster.com:8090/proxy/application_1572957695341_0001/
19/11/11 14:59:51 INFO mapreduce.Job: Running job: job_1572957695341_0001
19/11/11 15:00:05 INFO mapreduce.Job: Job job_1572957695341_0001 running in
uber mode : false
19/11/11 15:00:05 INFO mapreduce.Job:  map 0% reduce 0%
19/11/11 15:00:13 INFO mapreduce.Job:  map 100% reduce 0%
19/11/11 15:00:15 INFO mapreduce.Job: Job job_1572957695341_0001 completed
successfully
19/11/11 15:00:15 INFO mapreduce.Job: Counters: 42
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=136674
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                MAPRFS: Number of bytes read=59
                MAPRFS: Number of bytes written=249
                MAPRFS: Number of read operations=11
                MAPRFS: Number of large read operations=0
                MAPRFS: Number of write operations=39
        Job Counters
                Launched map tasks=1
                Rack-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=6111
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=6111
                Total vcore-seconds taken by all map tasks=6111
                Total megabyte-seconds taken by all map tasks=6257664
                DISK_MILLIS_MAPS=3056
        Map-Reduce Framework
                Map input records=3
                Map output records=3
                Input split bytes=59
                Spilled Records=0
Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=68
                CPU time spent (ms)=1620
                Physical memory (bytes) snapshot=246943744
                Virtual memory (bytes) snapshot=3582681088
                Total committed heap usage (bytes)=287309824
        HBase Counters
                BYTES_IN_REMOTE_RESULTS=0
                BYTES_IN_RESULTS=93
                MILLIS_BETWEEN_NEXTS=518
                NOT_SERVING_REGION_EXCEPTION=0
```


last-updated: Apr 11, 2022

```
                NUM_SCANNER_RESTARTS=0
                NUM_SCAN_RESULTS_STALE=0
                REGIONS_SCANNED=1
                REMOTE_RPC_CALLS=0
                REMOTE_RPC_RETRIES=0
                RPC_CALLS=3
                RPC_RETRIES=0
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=249
```

The following table shows the tools included in the hbase-server.jar:

Name[1]	Class[2]	Description
rowcounter	RowCounter	Count rows in HBase table
CellCounter	CellCounter	Count cells in HBase table
export	Export	Write table data to MapR filesystem
import	Import	Import data written by Export
importtsv	ImportTsv	Import data in TSV format
completebulkload	LoadIncrementalHFiles	Complete a bulk data load
copytable	CopyTable	Export a table from local cluster to peer cluster
verifyrep	VerifyReplication	Compare the data from tables in two different clusters  **Note:** This function does not work for incrementColumnValues cells since the timestamp is changed after being appended to the log.
WALPlayer	WALPlayer	Replay WAL files
exportsnapshot	ExportSnapshot	Export the specific snapshot to a given FileSystem

[1] Class is used for `hbase.org.apache.hadoop.hbase.mapreduce.<class>....`

[2] Name is used for `hadoop jar /opt/mapr/hbase/hbase-1.1.13/lib/hbase-server-1.1.13.0-mapr-1912.jar <name>...`

Using the libhbase Library

libhbase is a JNI-based, thread-safe C library that implements a native HBase client. You can use libhbase to build applications that access HBase.

This page contains the following topics:

- Installing libhbase
- Upgrading libhbase
- Building applications with libhbase
- Configuring the application environment
- Running a libhbase performance test

For examples that show how to use the APIs, see the sample source file.

Installing libhbase

Install libhbase on the nodes from which you will build and run the application.

last-updated: Apr 11, 2022

Complete the following steps to install libhbase from a repository:

1. Configure the repository to point to http://package.mapr.hpe.com/releases/MEP/MEP-6.3.0/.
2. Based on your operating system, run one of the following commands to install the package:
   - On Red Hat /Centos: `yum install mapr-libhbase`
   - On SLES: `zypper install mapr-libhbase`
   - On Ubuntu: `apt-get install mapr-libhbase`

Once the installation completes, the libhbase installation includes the following directories under `/opt/mapr/libhbase/libhbase-<version>`:

```
/
+---bin/
+---conf/
+---include/
|   +--hbase/
+---lib/
|   +---native/
+---src
    +---examples/
    |   +---async/
    +---test/
        +---native/
            +---common/
```

Note: The `include` folder contains the headers required to build applications. The `lib/native` directory contains shared libraries.

Upgrading libhbase

To upgrade to a more recent version of libhbase:

1. Install the new version.
2. Re-configure the application environment to refer to the new libraries.

Building Applications with libhbase

libhbase should be installed on each node that builds the application.

Note the following items when you build applications with libhbase:

- The headers required to build applications are located under `/opt/mapr/libhbase/libhbase-<version>/include`.
- libhbase shared library is located in the following directory: `/opt/mapr/libhbase/libhbase-<version>/lib/native`.
- Since libhbase uses JNI, you must also link your application against libjvm. In general, the libjvm library is located within the JDK/JRE installation directory.

For example, the following command builds the `hello_hbase` application with the `hello_hbase.c` source code:

```
gcc -o hello_hbase hello_hbase.c -I/opt/mapr/
libhbase/libhbase-0.98.7/include -L/opt/mapr/libhbase/libhbase-0.98.7/lib/
native -lhbase -L/usr/lib/jvm/java-7-sun/jre/lib/amd64/server -ljvm
```


last-updated: Apr 11, 2022

Configuring the Application Environment

Complete the following steps to configure the node from which you run the application:

- Verify that libhbase is installed on the node.
- Verify that both the libhbase and libjvm shared libraries are in the application's library search path. The libhbase shared library is located under `/opt/mapr/libhbase/libhbase-<version>/lib/native`. In general, the libjvm library is located within the JDK/JRE installation directory.
- Specify any JARs required by the application with one of the following environment variables: `CLASSPATH` or `HBASE_LIB_DIR`.
- Specify custom JVM options, such as `-Xmx`, using the environment variable `LIBHBASE_OPTS`.

Running a libhbase Performance Test

libhbase 0.98.7 includes a performance test that supports sequential/random gets and puts. In libhbase 0.98.9, the performance test utility also includes support for Zipfian, support for uniform random key generation, and it test for scans. You can run the test using this shell script.

HBase Client and MapR Database Binary Tables

MapR 6.0.x and 6.1 provide Apache HBase-compatible APIs and client interfaces but do not support HBase as an ecosystem component. MapR Database binary tables provide native storage for table data and include high performance and availability, versatile table operations, and streamlined cluster administration. The following APIs and tools are available for MapR Database binary tables:

HBase Client	• After installing the HBase Client, you can use HBase Shell commands to manipulate MapR Database binary tables on a remote machine. See Installing HBase on a Client Node on page 185 and MapR Database HBase Shell (Binary Tables) on page 5325. • MapR Database supports binary tables through the libMapRClient (a library of C APIs) and Apache HBase Java APIs. See Developing Applications for Binary Tables on page 2452.
HBase REST Gateway	• The HBase REST Gateway allows users to manipulate MapR Database binary tables through the HBase REST API. See Installing the HBase REST Gateway on page 187.
HBase Thrift Gateway	• The HBase Thrift Gateway allows users to manipulate MapR Database binary tables through the HBase Thrift API. See Installing the HBase Thrift Gateway on page 186.

Using the HBase Thrift Gateway

HBase Thrift Gateway includes an API and a service that accepts Thrift requests to connect to MapR Database and HBase tables. The HBase Thrift Gateway is installed as a service that is managed by Warden. When `mapr-hbasethrift` is installed, the `warden.hbasethrift.conf` file is added to the `/opt/mapr/conf/conf.d` directory.

Note: MapR SASL authentication, encryption, and impersonation for HBase Thrift Gateway are enabled by default on secure clusters.

last-updated: Apr 11, 2022

Starting the HBase Thrift Service

To start the HBase thrift service, enter the following command with the name of the host where hbasethrift is running:

```
maprcli node services -name hbasethrift -action start -nodes <node_hostname>
```

Configure Kerberos for HBase Thrift Gateway

1. Add the following to the `hbase-site.xml` file for every Thrift gateway:

```
<property>
    <name>hbase.thrift.keytab.file</name>
    <value>$KEYTAB</value>
</property>
<property>
    <name>hbase.thrift.kerberos.principal</name>
    <value>$USER/_HOST@HADOOP.LOCALDOMAIN</value>
    <!-- This may need to be HTTP/_HOST@<REALM> and _HOST may not work.
You may have to put the concrete full hostname. -->
</property>
<property>
   <name>hbase.thrift.security.qop</name>
   <value>auth-conf</value>
</property>
<!-- Add these if you need to configure a different DNS interface from
the default -->
<property>
    <name>hbase.thrift.dns.interface</name>
    <value>default</value>
</property>
<property>
    <name>hbase.thrift.dns.nameserver</name>
    <value>default</value>
</property>
```

   Substitute the appropriate credential and keytab for $USER and $KEYTAB respectively.

2. If you are running HBase Thrift in HTTP mode, you must add additional properties to the `hbase-site.xml` to enable HTTP connections through Kerberos. This is required if you enabled the following property in the `hbase-site.xml`:

```
<property>
  <name>hbase.regionserver.thrift.http</name>
  <value>true</value>
</property>
```

   Add the following properties to enable HTTP connections through Kerberos:

```
<property>
    <name>hbase.thrift.spnego.principal</name>
    <value>HTTP/_HOST@HADOOP.LOCALDOMAIN</value>
</property>
<property>
    <name>hbase.thrift.spnego.keytab.file</name>
    <value>$KEYTAB</value>
</property>
```


last-updated: Apr 11, 2022

3. To use MapR Database tables without the full path, add the following property to the `core-site.xml` file:

```
<property>
    <name>hbase.table.namespace.mappings</name>
    <value>*:/</value>
</property>
```

   Add this property ONLY if you are working with MapR Database tables. Working with HBase tables is not possible when this property is present. For more information, see Considerations for Upgrading to HBase 1.1.13 on page 337. For more information about mapping tables, see Mapping to HBase Table Namespaces on page 430.

The Thrift gateway authenticates with HBase using the supplied credential. No authentication is performed by the Thrift gateway itself. All client access via the Thrift gateway uses the Thrift gateway's credential and has its privilege.

Enable Impersonation for HBase Thrift Gateway

To configure the Thrift gateway to authenticate to HBase on the client's behalf, and to access HBase using a proxy user:

1. To allow proxy users, add the following to the `hbase-site.xml` file for every HBase node:

```
<property>
    <name>hadoop.proxyuser.$USER.groups</name>
    <value>$GROUPS</value>
</property>
<property>
    <name>hadoop.proxyuser.$USER.hosts</name>
    <value>$GROUPS</value>
</property>
```

2. To enable the `doAs` feature, add the following to the `hbase-site.xml` file for every Thrift gateway:

```
<property>
    <name>hbase.regionserver.thrift.http</name>
    <value>true</value>
</property>
<property>
    <name>hbase.thrift.support.proxyuser</name>
    <value>true/value>
</property>
```

3. Restart the Thrift gateway processes for the changes to take effect. If a node is running Thrift, the output of the `jps` command will list a `ThriftServer` process.
   - To restart Thrift on a node, use the following `maprcli` command:

```
maprcli node services -name hbasethrift -action restart -nodes
<node_hostname>
```

Using the HBase REST Gateway

HBase REST Gateway includes an API and a service that accepts REST requests to connect to MapR Database and HBase tables. Starting in version 0.98.9, the HBase REST Gateway is installed as a service that is managed by Warden. When mapr-hbase-rest is installed, the `warden.hbase-rest.conf` file is added to the `/opt/mapr/conf/conf.d` directory.

last-updated: Apr 11, 2022

Note: PAM authentication, encryption, and impersonation for HBase REST are enabled by default on secure clusters.

Starting the HBase REST Service

To start the HBase REST service, enter the following command with the name of the host where hbaserest is running:

```
maprcli node services -name hbaserest -action start -nodes <node_hostname>
```

Configure Kerberos for HBase REST Gateway

1. Add the following to the `hbase-site.xml` file for every REST Gateway:

   ```
   <property>
       <name>hbase.rest.keytab.file</name>
       <value>$KEYTAB</value>
   </property>
   <property>
       <name>hbase.rest.kerberos.principal</name>
       <value>$USER/_HOST@HADOOP.LOCALDOMAIN</value>
   </property>
   ```

   Substitute the appropriate credential and keytab for $USER and $KEYTAB respectively.

   The REST Gateway will authenticate with HBase using the supplied credential.

2. To enable REST Gateway Kerberos authentication for client access, add the following to the `hbase-site.xml` file for every REST Gateway:

   ```
   <property>
       <name>hbase.rest.authentication.type</name>
       <value>kerberos</value>
   </property>
   <property>
       <name>hbase.rest.authentication.kerberos.principal</name>
       <value>HTTP/_HOST@HADOOP.LOCALDOMAIN</value>
   </property>
   <property>
       <name>hbase.rest.authentication.kerberos.keytab</name>
       <value>$KEYTAB</value>
   </property>
   <!-- Add these if you need to configure a different DNS interface from
   the default -->
   <property>
       <name>hbase.rest.dns.interface</name>
       <value>default</value>
   </property>
   <property>
       <name>hbase.rest.dns.nameserver</name>
       <value>default</value>
   </property>
   ```

   Substitute the keytab for HTTP for $KEYTAB.

Enable Impersonation for HBase REST Gateway

To enable HBase REST Gateway impersonation, configure all HBase servers to allow proxy users, then configure every REST Gateway to enable impersonation.

last-updated: Apr 11, 2022

- To enable REST Gateway impersonation, add the following to the `hbase-site.xml` file for every REST gateway:

```
<property>
    <name>hbase.rest.support.proxyuser</name>
    <value>true</value>
</property>
```

HBase REST Gateway and HBase Thrift Gateway Secured By Default to Use SSL
Starting in EEP 6.0.0, HBase REST and HBase Thrift use SSL by default on secured clusters.

1. On a secure cluster, by default, HBase REST and HBase Thrift read the `ssl-client.xml` file and configure SSL using this file.
2. To enable HBase REST and Thrift encryption, use the following properties. Note that SSL for Thrift is enabled only when the `hbase.regionserver.thrift.http` property is `true`:

   Enabling HBase REST encryption

```
<property>
  <name>hbase.rest.ssl.enabled</name>
  <value>true</value>
</property>
```

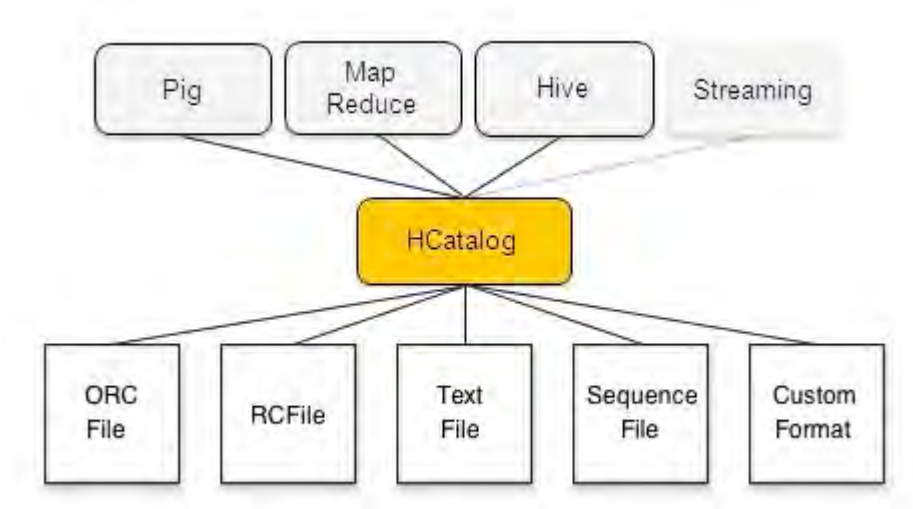
   Enabling HBase Thrift encryption

```
<property>
  <name>hbase.thrift.ssl.enabled</name>
  <value>true</value>
</property>
```

HCatalog

HCatalog is a table and storage management layer for Hadoop that enables users with different data processing tools — Pig, MapReduce — to more easily read and write data on the grid. HCatalog's table abstraction presents users with a relational view of data in the Hadoop distributed filesystem (HDFS) and ensures that users need not worry about where or in what format their data is stored — RCFile format, text files, SequenceFiles, or ORC files.

HCatalog supports reading and writing files in any format for which a SerDe (serializer-deserializer) can be written. By default, HCatalog supports RCFile, CSV, JSON, and SequenceFile, and ORC file formats. To use a custom format, you must provide the InputFormat, OutputFormat, and SerDe.



HCatalog is also automatically installed and upgraded along with Hive. For information about using HCatalog with Hive, see Hive and WebHCat Integration and Hive and HCatalog Integration.

Hive

Apache Hive™ is a data warehouse system for Hadoop that facilitates easy data summarization, ad-hoc queries, and the analysis of large datasets stored in Hadoop-compatible file systems, such as the MapR Converged Data Platform. Hive provides a mechanism to project structure onto this data and query the data using a SQL-like language called HiveQL. At the same time this language also allows traditional map/reduce programmers to plug in their custom mappers and reducers when it is inconvenient or inefficient to express this logic in HiveQL.

last-updated: Apr 11, 2022

You can refer also to documentation available from the Apache Hive project.

Hive components include the following:

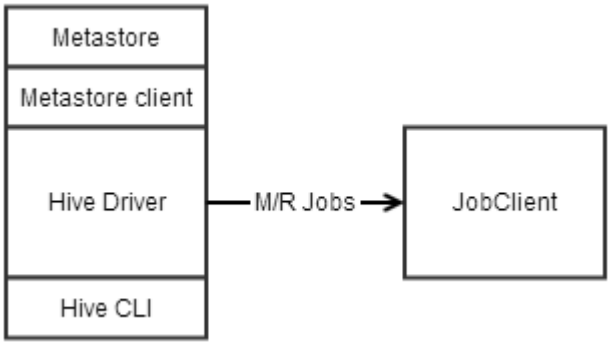
- Hive Metastore
- HiveServer2
- HCatalog
- WebHCat
- Hive CLI
- Beeline

Note: If you installed Hive using the installer with the "Enable Security" check box selected, Hive is secured and no further configuration is required. However, if you installed Hive manually and wish to enable security for Hive, see Hive Security Configuration Options on page 3427 for information.

The following examples show how these components communicate with each other and when you might want to configure security features such as authentication and encryption:

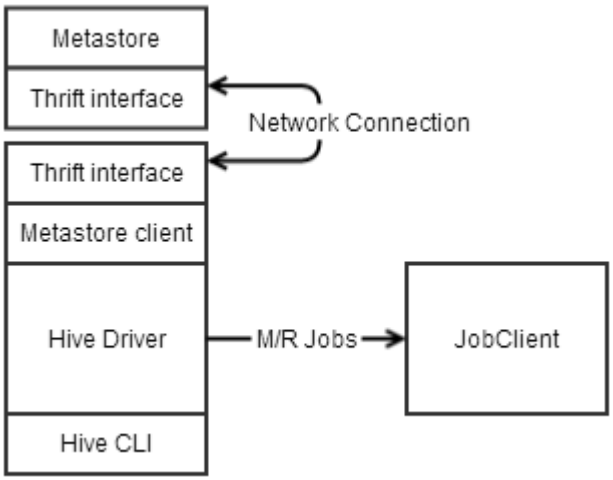
Case 1: Jobs Submitted by the Hive CLI, Embedded Metastore

In this case, all the information needed by Hive is contained within a single process, and no security is needed beyond that already provided by the JobClient's communications.



Case 2: Jobs Submitted by the Hive CLI, Remote Metastore

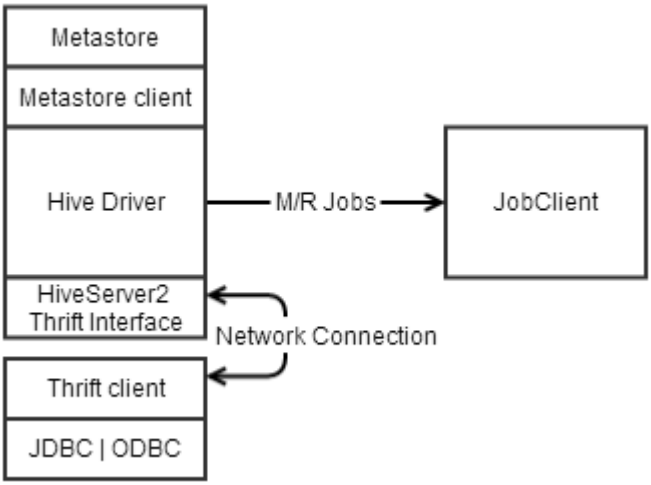
In this case, Hive needs to access a metastore remote to Hive's process using a Thrift interface. This communication can be left non-secured, secured with Kerberos, or secured with MapR-SASL.



Case 3: Jobs Submitted by HiveServer2, Embedded Metastore

In this case, JDBC or ODBC on a user's machine sends queries to HiveServer2, which submits the queries to the driver for parsing. The communication between JDBC and HiveServer2 can be secured with username and password with SSL, MapR-SASL, or with Kerberos. Either approach offers authentication and encryption. JDBC/ODBC can also be configured to use username and password without SSL, which offers authentication *only*.

To use SSL from a client machine the `ssl_truststore` file must be copied from the cluster to the client.



Case 4: Jobs Submitted by HiveServer2, Remote Metastore

In this case, JDBC or ODBC on a user's machine sends queries to HiveServer2, which submits the queries to the driver, which runs the query and returns the results. The metastore is remote. In this case, there are two communications links to secure: the Thrift interface between the metastore client and server, and the Thrift interface between the client JDBC/ODBC and HiveServer2. The security arrangements for these links are identical to Cases 2 and 3.

Getting Started with Hive

In this tutorial, you'll create a Hive table, load data from a tab-delimited text file, and run a couple of basic queries against the table. For details on setting up HiveServer2 and starting BeeLine, see Using JDBC or Beeline to Connect to HiveServer2 on page 3516.

last-updated: Apr 11, 2022

Note: If you are using HiveServer2, you will use the BeeLine CLI instead of the Hive shell, as shown below.

Take a look at the source data

First, take a look at the contents of the file using the terminal:

1. Save the following data to a text file named `sample-table.txt`:

   The `sample-table.txt` table columns are delimited by tabs:

   ```
   1320352532    1001    http://www.mapr.com/doc    http://www.mapr.com
   192.168.10.1
   1320352533    1002    http://www.mapr.com    http://www.example.com
   192.168.10.10
   1320352546    1001    http://www.mapr.com    http://www.mapr.com/doc
   192.168.10.1
   ```

   If you are working on the MapR Virtual Machine, we'll be loading the file from the MapR Virtual Machine's local filesystem (not the cluster storage layer), so save the file in the MapR Home directory (for example, `/home/mapr`).

2. Make sure you are in the Home directory where you saved `sample-table.txt` (type `cd ~` if you are not sure).

3. Type `cat sample-table.txt` to display the following output.

```
mapr@mapr-desktop:~$ cat sample-table.txt
1320352532    1001    http://www.mapr.com/doc    http://www.mapr.com
192.168.10.1
1320352533    1002    http://www.mapr.com    http://www.example.com
192.168.10.10
1320352546    1001    http://www.mapr.com    http://www.mapr.com/doc
192.168.10.1
```

Notice that the file consists of only three lines, each of which contains a row of data fields separated by the TAB character. The data in the file represents a web log.

Create a table in Hive and load the source data:

1. Type the following command to start the Hive shell, using tab-completion to expand the `<version>`:

   ```
   /opt/mapr/hive/hive-<version>/bin/hive
   ```

2. At the `hive>` prompt, type the following command to create the table:

   ```
   CREATE TABLE web_log(viewTime INT, userid BIGINT, url STRING, referrer
   STRING, ip STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';
   ```

3. Type the following command to load the data from `sample-table.txt` into the table:

   ```
   LOAD DATA LOCAL INPATH '/home/mapr/sample-table.txt' INTO TABLE web_log;
   ```

Run basic queries against the table

- Try the simplest query, one that displays all the data in the table:

  ```
  SELECT web_log.* FROM web_log;
  ```

  This query would be inadvisable with a large table, but with the small sample table it returns very quickly.

- Try a simple SELECT to extract only data that matches a desired string:

  ```
  SELECT web_log.* FROM web_log WHERE web_log.url LIKE '%doc';
  ```

  This query launches a MapReduce application to filter the data.

Configuring Hive

This section contains the following topics:

Configure Hive Directories

You can configure the following Hive directories:

- Hive Scratch Directory
- Hive Warehouse Directory
- Hive Error Logs Directory

Hive Scratch Directory

In Hive 1.0, MapR configures the Hive scratch directory to be `/user/<user.name>/tmp/hive/<user.name>`. In Hive 0.13, the default scratch directory is /user/<user.name>/tmp/hive. The hive user must have write access to the /user folder.

To modify this parameter, perform one of the following operations:

- Set this parameter in the `hive-site.xml`. Copy the hive.exec.scratchdir property elements from the $HIVE_HOME/conf/hive-default.xml.template file and paste them into an XML configuration element in the $HIVE_HOME/conf/hive-site.xml file. Then, modify the value elements for these directories in the hive-site.xml file.
- Set this parameter from the Hive shell. Example:

  ```
  hive> set hive.exec.scratchdir=/myvolume/tmp
  ```

Note: You will see better performance when queries import data from a table that is in the same MapR volume as Hive scratch directory.

How Hive Handles Scratch Directories on MapR

When a query requires Hive to query existing tables and create data for new tables, Hive uses the following workflow:

1. Create the query scratch directory `hive_<timestamp>_<randomnumber>` under the Hive scratch directory.
2. Create the following directories as subdirectories of the scratch directory:
   a. Final query output directory. This directory's name takes the form `-ext-<number>`.

last-updated: Apr 11, 2022

   b. An output directory for each MapReduce application. These directories' names take the form `-mr-<number>`.

3. Hive executes the tasks, including MapReduce applications and loading data to the query output directory.
4. Hive loads the data from output directory into a table. By default, the table's directory is in the `/user/hive/warehouse` directory. You can configure this location with the `hive.metastore.warehouse.dir` parameter in `hive-site.xml`, unless the table DDL specifies a custom location. Hive renames the output directory to the table directory in order to load the output data to the table.
5. The scratch directories are automatically deleted after the query completes successfully.

MapR uses Administering Volumes on page 856, which are logical units that enable you to apply policies to a set of files, directories, and sub-volumes. When the output directory and the table directory are in different volumes, this workflow involves moving data across volumes. Moving data across volumes is slower than moving data within a volume. Therefore, MapR sets `hive.optimize.insert.dest.volume` to `true` to automatically create a scratch directory in the same volume as the target table.

Hive Warehouse Directory

Hive tables are stored in the Hive warehouse directory. By default, MapR configures the Hive warehouse directory to be `/user/hive/warehouse` under the root volume. This default is defined in the `$HIVE_HOME/conf/hive-default.xml.template` file.

To modify this parameter, perform one of the following operations:

- Set this parameter in the `hive-site.xml`. Copy the hive.metastore.warehouse.dir property elements from the $HIVE_HOME/conf/hive-default.xml.template file and paste them into an XML configuration element in the $HIVE_HOME/conf/hive-site.xml file. Then, modify the value elements for these directories in the hive-site.xml file.
- Set this parameter from the Hive shell. Example:

```
hive> set hive.metastore.warehouse.dir=/myvolume/mydirectory
```

Note: You will see better performance when queries move data between tables in the same volume.

Hive Error Logs Directory

The log files are stored in `/opt/mapr/hive/hive-<version>/logs/<user>` by default.

To modify the log location:

1. Configure `hive.log.dir` in `$HIVE_HOME/conf/hive-log4j.properties` file. Example:

```
hive.log.dir=<other_location>
```

2. Set the sticky bit on the new directory. Example:

```
chmod 1777 <other_location>
```

Configuring Hive Client on MapR Client Node

This topic describes how to configure the Hive client on a MapR client node.

last-updated: Apr 11, 2022

Configuring Hive Client for Previous EEP 8.1.0

To use the Hive client with secure and non-secure clusters for previous EEP 8.1.0, perform the following steps:

1. Install MapR client. See Installing the MapR Client on CentOS, RedHat, Oracle Linux on page 390.
2. Copy the `daemon.conf` file from the `MAPR_HOME/conf` directory on the cluster to the `MAPR_HOME/conf` directory on the MapR client.

Configuring Hive Client for EEP 8.1.0 and Later

To use the Hive client with secure and non-secure clusters for EEP 8.1.0 and later, perform the following steps:

1. Install MapR client. See Installing the MapR Client on CentOS, RedHat, Oracle Linux on page 390.
2. Copy the `daemon.conf` file from the `MAPR_HOME/conf` directory on the cluster to the `MAPR_HOME/conf` directory on the MapR client.
3. Copy `maprtrustcreds.jceks` from the `MAPR_HOME/conf` directory on the cluster to the `MAPR_HOME/conf` directory on the MapR client.

Configuring `MSCK REPAIR TABLE`

This section guides you through configuring `MSCK REPAIR TABLE` command to compare and update the partitions in Hive Metastore and file systems.

Use the `MSCK REPAIR TABLE` command to manually update (ADD, DROP, SYNC) the partitions on Hive metastore with respect to file systems like HDFS, Amazon S3, filesystem, and others.

For example: You specify the location of filesystem when you create a Hive table. When you add or delete the partitions to or from the filesystem, the partitions in filesystem and Hive metastore becomes inconsistent.

Run `MSCK REPAIR TABLE` command to compare the partitions in filesystem and the partitions in Hive metastore and update the partitions in Hive metastore.

```
MSCK [REPAIR] TABLE <table name> [ADD/DROP/SYNC PARTITIONS];
```

Configure the Hive Metastore with the following Hive property:

Property	Default	Description
hive.msck.repair.batch.max.retries	0	Maximum number of retries for the msck repair command when adding unknown partitions. If the value is greater than zero it will retry adding unknown partitions until the maximum number of attempts is reached or batch size is reduced to 0, whichever is earlier. In each retry attempt, it will reduce the batch size by a factor of 2 until it reaches zero. If the value is set to zero it will retry until the batch size becomes zero as described above.

Configuring Database for Hive Metastore

The metadata for Hive tables and partitions are stored in the Hive Metastore. By default, the Hive Metastore stores all Hive metadata in an embedded Apache Derby database in the MapR filesystem. The following sections describe how to configure other DBs for Hive Metastore.

last-updated: Apr 11, 2022

CAUTION: Do not use `datanucleus.schema.autoCreateAll` for populating underlying databases. For more details, see prohibited usage of `datanucleus.schema.autoCreateAll` property .

Use MySQL for the Hive Metastore
The metadata for Hive tables and partitions are stored in the Hive Metastore. By default, the Hive Metastore stores all Hive metadata in an embedded Apache Derby database in the MapR filesystem. Derby only allows one connection at a time; if you want multiple concurrent Hive sessions, you can use MySQL for the Hive Metastore.

Review the following prerequisites before you begin:

- Verify that MySQL (version 5.6.17 or later) is installed on the machine that will host the Hive metastore, and also verify that you can connect to the MySQL server from the Hive machine. You can run the Hive metastore on any machine that is accessible from Hive. You can test this with the following command:

  ```
  mysql -h <hostname> -u <user>
  ```

- The database administrator must create a database for the Hive metastore data, and the username specified in `javax.jdo.option.ConnectionUserName` must have permissions to access it. The database can be specified using the `ConnectionURL` parameter. The tables and schemas are created automatically when the metastore is first started.

  Tip: In MapR 6.1.0 and earlier releases, the following steps can be used interchangeably for MariaDB.

Complete the following steps to configure Hive to use MySQL for the Hive Metastore:

Important: For MySQL 8, set the `javax.jdo.option.ConnectionDriverName` property to `com.mysql.cj.jdbc.Driver`. The `com.mysql.jdbc.Driver` is deprecated. The new driver class is `com.mysql.cj.jdbc.Driver`. However, the driver is automatically registered via the Service Provider Interface, so manual loading of the driver class is generally unnecessary.

1. Update the `hive-site.xml` in the Hive configuration directory (`/opt/mapr/hive/hive-<version>/conf`) with the following contents:

```
<configuration>

 <property>
    <name>javax.jdo.option.ConnectionURL</name>
    <value>jdbc:mysql://localhost:3306/hive?
createDatabaseIfNotExist=true</value>
    <description>JDBC connect string for a JDBC metastore</description>
</property>

 <property>
    <name>javax.jdo.option.ConnectionDriverName</name>
    <value>com.mysql.jdbc.Driver</value>
    <description>Driver class name for a JDBC metastore</description>
 </property>

 <property>
    <name>javax.jdo.option.ConnectionUserName</name>
    <value>root</value>
    <description>username to use against metastore database</description>
 </property>

 <property>
    <name>javax.jdo.option.ConnectionPassword</name>
    <value><fill in with password></value>
    <description>password to use against metastore database</description>
 </property>

 <property>
    <name>hive.metastore.uris</name>
    <value>thrift://localhost:9083</value>
 </property>

</configuration>
```

2. Run the `schematool` command as an initialization step.

```
/opt/mapr/hive/hive-<version>/bin/schematool -dbType mysql -initSchema
```

3. To connect to an existing MySQL metastore, make sure the `ConnectionURL` parameter and the `Thrift URIs` parameters in `hive-site.xml` point to the metastore's host and port.

4. To set a specific port for Thrift URIs, add the command `export METASTORE_PORT=<port>` into the file `hive-env.sh` (if `hive-env.sh` does not exist, create it in the Hive configuration directory). Example:

```
export METASTORE_PORT=9083
```

5. Start the Hive Metastore service using one of the following commands:

   If you want the Hive Metastore to be managed by Warden, the maprcli, and the Control System:

   ```
   maprcli node services -name hivemeta -action start -nodes <space
   delimited list of nodes>
   ```

   If you want the Hive Metastore to be managed with standard hive commands:

   ```
   /opt/mapr/hive/hive-<version>/bin/hive --service metastore --start
   ```

   You can use also use `nohup hive --service metastore` to run the Metastore in the background.

   Warning: If you have not configured a MySQL Metastore, do not run the Hive shell from an NFS mount location. If you try to do this, Hive will fail. The same problem will occur if you use the `hive-site.xml` file to configure the Metastore on an NFS mount location. Avoid both of these configurations.

Configuring a Remote MySQL Database for the Hive Metastore

After installing MySQL, perform the following steps to configure Hive Metastore on MySQL

1. Install the MySQL connector. To install:

   - **MySQL connector on a RHEL 6+ system**

     On the Hive Metastore server host, install `mysql-connector-java` and symbolically link the file to the `/opt/mapr/hive/hive-<version>/lib/` directory.

     ```
     $ sudo yum install
     mysql-connector-java
     $ ln -s /usr/share/
     java/mysql-connector-java.jar /opt/
     mapr/hive/hive-<version>/lib/
     mysql-connector-java.jar
     ```

   - **MySQL connector on a SLES system**

     On the Hive Metastore server host, install `mysql-connector-java` and symbolically link the file to the `/opt/mapr/hive/hive-<version>/lib/` directory.

     ```
     $ sudo zypper install
     mysql-connector-java
     $ ln -s /usr/share/
     java/mysql-connector-java.jar /opt/
     mapr/hive/hive-<version>/lib/
     mysql-connector-java.jar
     ```

   - **MySQL connector on a Debian/Ubuntu system**

     On the Hive Metastore server host, install `mysql-connector-java` and symbolically link the file into the `/opt/mapr/hive/hive-<version>/lib/` directory.

     ```
     $ sudo apt-get install
     libmysql-java
     $ ln -s /usr/
     share/java/libmysql-java.jar /opt/
     ```


last-updated: Apr 11, 2022

```
mapr/hive/hive-<version>/lib/
mysql-connector-java.jar
```

2. Create the database and an associated user. The following commands are for a Hive Metastore with hostname `metastorehost` to create a MySQL user with name `hive` and password `mypassword`:

```
$ mysql -u root -p

mysql> CREATE DATABASE metastore;
mysql> CREATE USER 'hive'@'metastorehost' IDENTIFIED BY 'mypassword';
...
mysql> REVOKE ALL PRIVILEGES, GRANT OPTION FROM 'hive'@'metastorehost';
mysql> GRANT ALL PRIVILEGES ON metastore.* TO 'hive'@'metastorehost';
mysql> FLUSH PRIVILEGES;
mysql> quit;
```

3. Configure the Metastore service to communicate with the MySQL database by setting the necessary properties (shown below) in the `/opt/mapr/hive//hive-<version>/conf/hive-site.xml` file.

   Suppose a MySQL database running on `myhost` and the user account `hive` with the password `mypassword`, set the following properties (overwriting any existing values) in the `hive-site.xml` file:

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:mysql://myhost/metastore</value>
  <description>the URL of the MySQL database</description>
</property>

<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>com.mysql.jdbc.Driver</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionUserName</name>
  <value>hive</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionPassword</name>
  <value>mypassword</value>
</property>

<property>
  <name>hive.metastore.uris</name>
  <value>thrift://<n.n.n.n>:9083</value>
  <description>IP address (or fully-qualified domain name) and port of
the metastore host</description>
</property>
```

   Note: Though you can set the same `hive-site.xml` properties on all the hosts (client, Metastore, HiveServer), `hive.metastore.uris` is the only property that must be configured on all the hosts; the other properties are only needed on the Metastore host.

4. Run `schemaTool` to create the initial DB structure.

```
/opt/mapr/hive/hive-<version>/bin/schematool -dbType mysql -initSchema
```

Configuring a Remote PostgreSQL Database for the Hive Metastore
Before you can run the Hive metastore with a remote PostgreSQL database, you must configure a JDBC driver to the remote PostgreSQL database, set up the initial database schema, and configure the PostgreSQL user account for the Hive user.

After installing PostgreSQL, perform the following steps to configure Hive Metastore on PostgreSQL.

Installing and Configuring PostgreSQL for the Hive Metastore

1. Download the PostgreSQL JDBC driver.

   Refer to the official PostgreSQL JDBC Driver website to download the JDBC driver and get information about the latest updates. Determine the appropriate database version and get the released drivers and JAR file.

2. Run the following commands using `sudo`:

   a. Move the JAR into the Java `share` directory:

   ```
   sudo mv <postgresql-jdbc.jar> /usr/share/java/postgresql-jdbc.jar
   ```

   b. Change the access mode of the JAR file to 644:

   ```
   sudo chmod 644 /usr/share/java/postgresql-jdbc.jar
   ```

   c. Create symbolic link to the `/usr/lib/hive/lib/ directory`, for example:

   ```
   sudo ln -s /usr/share/java/postgresql-jdbc.jar /opt/mapr/hive/
   hive-<version>/lib/postgresql-jdbc.jar
   ```

3. Create the Metastore database and user accounts:

   ```
   $ sudo -u postgres psql

   postgres=# CREATE USER hiveuser WITH PASSWORD 'mypassword';
   postgres=# CREATE DATABASE metastore;
   ```

   To verify the connection from the Metastore service host, run the following command:

   ```
   psql -h myhost -U hiveuser -d metastore
   metastore=#
   ```

4. Configure the Metastore service to communicate with the PostgreSQL database by setting the necessary properties (shown below) in the `/opt/mapr/hive//hive-<version>/conf/hive-site.xml` file. Suppose a PostgreSQL database running on host `myhost` under the user account `hive` with the password `mypassword`, set the following configuration properties in the `hive-site.xml` file:

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:postgresql://myhost/metastore</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>org.postgresql.Driver</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionUserName</name>
  <value>hiveuser</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionPassword</name>
  <value>mypassword</value>
</property>

<property>
  <name>hive.metastore.uris</name>
  <value>thrift://<n.n.n.n>:9083</value>
  <description>IP address (or fully-qualified domain name) and port of
the metastore host</description>
  </property>
```

Note: Though you can use the same `hive-site.xml` properties on all the hosts (client, metastore, HiveServer), `hive.metastore.uris` is the only property that must be configured on all of the hosts; the other properties are only needed on the Metastore host.

5. Run `schemaTool` to create the initial DB structure:

```
/opt/mapr/hive/hive-<version>/bin/schematool -dbType postgres -initSchema
```

Configuring a Remote Oracle Database for the Hive Metastore

After installing Oracle, perform the following steps to configure Hive Metastore on Oracle.

1. Install the Oracle JDBC Driver.
   a) Download the Oracle JDBC Driver (`ojdbc6.jar`) from the Oracle website.
   b) Move the `ojdbc6.jar` file to `/opt/mapr/hive/hive-<version>/lib/` directory

last-updated: Apr 11, 2022

2. Create the Metastore database and user account.

   Connect to your Oracle database as administrator, create the user that will use the Hive Metastore, and create the Metastore schema. For example:

   ```
   $ sqlplus "sys as sysdba"
   SQL> create user hiveuser identified by mypassword;
   SQL> grant connect to hiveuser;
   SQL> grant all privileges to hiveuser;
   SQL>CREATE DATABASE metastore
   ```

3. Configure the Metastore service to communicate with the Oracle database by setting the necessary properties (shown below) in the `/opt/mapr/hive//hive-<version>/conf/hive-site.xml` file.

   Suppose an Oracle database running on `myhost` and the user account `hiveuser` with the password `mypassword`, set the following properties (overwriting any existing values) in the `hive-site.xml` file:

   ```
   <property>
     <name>javax.jdo.option.ConnectionURL</name>
     <value>jdbc:oracle:thin:@//myhost/metastore</value>
   </property>

   <property>
     <name>javax.jdo.option.ConnectionDriverName</name>
     <value>oracle.jdbc.OracleDriver</value>
   </property>

   <property>
     <name>javax.jdo.option.ConnectionUserName</name>
     <value>hiveuser</value>
   </property>

   <property>
     <name>javax.jdo.option.ConnectionPassword</name>
     <value>mypassword</value>
   </property>

   <property>
     <name>hive.metastore.uris</name>
     <value>thrift://<n.n.n.n>:9083</value>
     <description>IP address (or fully-qualified domain name) and port of
   the metastore host</description>
   </property>
   ```

   Note: Though you can set the same `hive-site.xml` properties on all the hosts (client, Metastore, HiveServer), `hive.metastore.uris` is the only property that must be configured on all the hosts; the other properties are only needed on the Metastore host.

4. Run `schemaTool` to create the initial DB structure.

   ```
   /opt/mapr/hive/hive-<version>/bin/schematool -dbType oracle -initSchema
   ```

Configuring an Oracle Schema

You must create schemas for Oracle databases manually.

Use `schematool` to view and create relational database management system (RDBMS) schemas.

See Apache Hive documentation for the detailed steps.

Configuring a Remote MS SQL SERVER Database for the Hive Metastore

last-updated: Apr 11, 2022

After installing MS SQL, perform the following steps to configure Hive Metastore on MS SQL.

1. Create `hiveuser` and Metastore schema.

```
1>CREATE DATABASE metastore;
2>GO
1>CREATE LOGIN <hiveuser> with password='<mypassword>;
2>CREATE USER <hiveuser> for login <hiveuser>;
3>GRANT <PRIVILEGES> to <hiveuser>;
4>GO
```

2. Download JDBC Driver from here, untar the file, and follow instructions in the `install.txt` file to install the driver.

3. Copy the JAR file to `/opt/mapr/hive/hive-version>/lib/` directory.

   - For Java 7

```
cp ~/sqljdbc_6.0/enu/jre7/sqljdbc41.jar /opt/mapr/hive/
hive-<version>/lib/
```

   - For Java 8

```
cp ~/sqljdbc_6.0/enu/jre8/sqljdbc42.jar /opt/mapr/hive/
hive-<version>/lib/
```


last-updated: Apr 11, 2022

4. Configure the Metastore service to communicate with the MS SQL database by setting the necessary properties (shown below) in the `/opt/mapr/hive//hive-<version>/conf/hive-site.xml` file.

   Suppose an MS SQL database running on `myhost` and the user account `hiveuser` with the password `mypassword`, set the following properties (overwriting any existing values) in the `hive-site.xml` file:

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:sqlserver://<SERVER_NAME>:1433;DatabaseName=metastore;</
value>
</property>

<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>com.microsoft.sqlserver.jdbc.SQLServerDriver</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionUserName</name>
  <value>hiveuser</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionPassword</name>
  <value>mypassword</value>
</property>

<property>
  <name>hive.metastore.uris</name>
  <value>thrift://<n.n.n.n>:9083</value>
  <description>IP address (or fully-qualified domain name) and port of
the metastore host</description>
  </property>
```

   Note: Though you can set the same `hive-site.xml` properties on all the hosts (client, Metastore, HiveServer), `hive.metastore.uris` is the only property that must be configured on all the hosts; the other properties are only needed on the Metastore host.

5. Run `schemaTool` to create the initial DB structure.

```
/opt/mapr/hive/hive-<version>/bin/schematool -dbType mssql -initSchema
```

Configuring MariaDB for the Hive Metastore

Installing MariaDB

To install MariaDB, use the MariaDB Repository Configuration Tool. See MariaDB Downloads.

Configuring Repositories

The following steps describe how to configure a repository and install the latest available stable version of MariaDB for different operating systems.

1. Configure a repository for MariaDB:

- **Red Hat / CentOS and SLES**

  Copy and paste the following custom MariaDB repository entry into a file under `/etc/yum.repos.d/`. You can name the file `MariaDB.repo` or something similar:

  ```
  # MariaDB 10.4 RedHat repository list
  # http://downloads.mariadb.org/mariadb/repositories/
  [mariadb]
  name = MariaDB
  baseurl = http://yum.mariadb.org/10.4/rhel7-amd64
  gpgkey=https://yum.mariadb.org/RPM-GPG-KEY-MariaDB
  gpgcheck=1
  ```

- **Ubuntu**

  You can also create a custom MariaDB `sources.list` file. To do so, after importing the signing key as outlined above, copy and paste the following into a file under `/etc/apt/sources.list.d/`. You can name the file `MariaDB.list` or something similar. Or you can add it to the bottom of your `/etc/apt/sources.list` file:

  ```
  # MariaDB 10.4 repository list - created 2020-04-17 08:34 UTC
  # http://downloads.mariadb.org/mariadb/repositories/
  deb [arch=amd64,arm64,ppc64el] http://mirror.mephi.ru/mariadb/repo/
  10.4/ubuntu bionic main
  deb-src http://mirror.mephi.ru/mariadb/repo/10.4/ubuntu bionic main
  ```

2. After the `sources.list` file is in place, install MariaDB:

   - **Red Hat / CentOS**

     ```
     sudo yum clean all && sudo yum install MariaDB-server MariaDB-client
     ```

   - **SLES**

     ```
     sudo zypper update && sudo zypper install mariadb
     ```

   - **Ubuntu**

     ```
     sudo apt update && sudo apt install mariadb-server
     ```

3. Start the MariaDB server:

   - **Red Hat / CentOS and Ubuntu**

     ```
     sudo service mariadb start
     ```

   - **SLES**

     ```
     sudo systemctl start mariadb
     ```

4. In the command line, run the `mysql_secure_installation` shell script:

```
sudo mysql_secure_installation
Enter current password for root (enter for none): press Enter
Set root password? Y
New password: Type new root password
Re-enter new password: Confirm the password
Remove anonymous users? Y
Disallow root login remotely? Y
Remove test database and access to it? Y
Reload privilege tables now? Y
```

Configuring a JDBC Driver for MariaDB

Before you can run the Hive Metastore with a MariaDB database, you must:

- Configure a JDBC driver for the MariaDB database.
- Set up the initial database schema.
- Configure the MariaDB user account for the Hive user.

Use the following steps:

1. Install the MariaDB Connector/J manually with a `.jar` file. The MariaDB Connector/J can also be installed by manually installing a `.jar` file to a directory in your CLASSPATH. Download the MariaDB Connector/J `.jar` files from the following URL: https://downloads.mariadb.com/Connectors/java/connector-java-2.5.4/.

2. Copy the `.jar` files to the `/opt/mapr/hive/hive-<version>/lib/` directory:

```
cp mariadb-java-client-2.5.4-sources.jar /opt/mapr/hive-<version>/lib/
cp mariadb-java-client-2.5.4.jar /opt/mapr/hive/hive-<version>/lib/
cp mariadb-java-client-2.5.4-javadoc.jar /opt/mapr/hive/
hive-<version>/lib/
```

3. Restart Hive services:

```
maprcli node services -name hivemeta -action restart -nodes 'hostname -f'
maprcli node services -name hs2 -action restart -nodes 'hostname -f'
```

4. Create the Hive Metastore database and user accounts:

```
$ mysql -u root -p <password>

MariaDB [(none)]> CREATE USER hiveuser IDENTIFIED BY PASSWORD 'password';
MariaDB [(none)]> CREATE DATABASE metastore;
MariaDB [(none)]> REVOKE ALL PRIVILEGES, GRANT OPTION FROM
'hiveuser'@'metastorehost';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON metastore.* TO
'hiveuser'@'metastorehost';
MariaDB [(none)]> FLUSH PRIVILEGES;
MariaDB [(none)]> quit;
```

Configuring the Hive Metastore on MariaDB

Use these steps:

1. In the Hive configuration directory (`/opt/mapr/hive/hive-<version>/conf`), update the `hive-site.xml` file with the following properties.

```
<property>
    <description>the URL of the MariaDB database</description>
    <name>javax.jdo.option.ConnectionURL</name>
    <value>jdbc:mysql://<hostname>:3306/metastore</value>
</property>
<property>
    <name>javax.jdo.option.ConnectionDriverName</name>
    <value>org.mariadb.jdbc.Driver</value>
</property>
<property>
    <name>javax.jdo.option.ConnectionUserName</name>
    <value><hiveuser></value>
</property>
<property>
    <name>javax.jdo.option.ConnectionPassword</name>
    <value><fill in with password></value>
</property>
<property>
    <description>IP address (or FQDN) and port of the metastore host</
description>
    <name>hive.metastore.uris</name>
    <value>thrift://<hostname>:9083</value>
</property>
```

2. Run the `schematool` command as an initialization step:

```
/opt/mapr/hive/hive-<version>/bin/schematool -dbType mysql -initSchema
```

User Impersonation for Hive

User impersonation enables Hive to submit jobs as a particular user. Without impersonation, Hive submits queries and hadoop commands as the user that started HiveServer2 and Hive Metastore. On a MapR cluster, this user is typically the `mapr` user or the user specified in the `MAPR_USER` environment variable.

Note: Impersonation is enabled by default.

last-updated: Apr 11, 2022

Enable User Impersonation

On non-secure clusters

1. Set the following properties in the `/opt/mapr/hive/<version>/conf/hive-site.xml` file on the nodes where HiveServer2 is installed:

```
<property>
  <name>hive.server2.enable.doAs</name>
  <value>true</value>
  <description>Set this property to enable impersonation in Hive Server
2</description>
</property>
<property>
  <name>hive.metastore.execute.setugi</name>
  <value>true</value>
  <description>Set this property to enable Hive Metastore service
impersonation in non-secure mode. In non-secure mode, setting this
property to true will cause the metastore to execute DFS operations
using the client's reported user and group permissions. Note that this
property must be set on both the client and server sides. If the client
sets it to true and the server sets it to false, the client setting will
be ignored.</description>
</property>
```

2. Set the following property `/opt/mapr/hive/<version>/conf/hive-site.xml` file on the nodes where Hive Metastore is installed:

```
<property>
  <name>hive.metastore.execute.setugi</name>
  <value>true</value>
  <description>Set this property to enable Hive Metastore service
impersonation in non-secure mode. In non-secure mode, setting this
property to true will cause the metastore to execute DFS operations
using the client's reported user and group permissions. Note that this
property must be set on both the client and server sides. If the client
sets it to true and the server sets it to false, the client setting will
be ignored.</description>
</property>
```

On secure (MAPR-SASL and Kerberos) clusters

1. Set the following properties in the `/opt/mapr/hive/<version>/conf/hive-site.xml` file on the nodes where HiveServer2 is installed:

```
<property>
  <name>hive.server2.enable.doAs</name>
  <value>true</value>
  <description>Set this property to enable impersonation in Hive Server
2</description>
</property>
<property>
  <name>hive.metastore.execute.setugi</name>
  <value>false</value>
  <description>Set this property to enable Hive Metastore service
impersonation in non-secure mode. In non-secure mode, setting this
property to true will cause the metastore to execute DFS operations
using the client's reported user and group permissions. Note that this
property must be set on both the client and server sides. If the client
sets it to true and the server sets it to false, the client setting will
be ignored.</description>
</property>
```

2. Set the following property `/opt/mapr/hive/<version>/conf/hive-site.xml` file on the nodes where Hive Metastore is installed:

```
<property>
  <name>hive.metastore.execute.setugi</name>
  <value>false</value>
  <description>Set this property to enable Hive Metastore service
impersonation in non-secure mode. In non-secure mode, setting this
property to true will cause the metastore to execute DFS operations
using the client's reported user and group permissions. Note that this
property must be set on both the client and server sides. If the client
sets it to true and the server sets it to false, the client setting will
be ignored.</description>
</property>
```

Note: The `hive.metastore.execute.setugi` property is set to false automatically after `/opt/mapr/server/configure.sh -R` is running.

On both secure and non-secure clusters

On nodes where the **Resource Manager** and the **Node Manager** are installed, set the following properties in the `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/core-site.xml` file:

```
<property>
  <name>hadoop.proxyuser.mapr.groups</name>
  <value>*</value>
  <description>Allow the superuser mapr to impersonate any member of any
group</description>
</property>
<property>
  <name>hadoop.proxyuser.mapr.hosts</name>
  <value>*</value>
  <description>The superuser can connect from any host to impersonate a
user</description>
</property>
```


last-updated: Apr 11, 2022

Warning: The impersonated user must have write permissions to `/user/hive/warehouse` and `/user/mapr-user/tmp/hive` directories.

Verify that User Impersonation is Enabled

To verify that Hive queries do not run as the `mapr` user, connect to HiveServer2 as a user other than `mapr`. Then run queries and verify that queries were run as the user that connected to HiveServer2.

To verify that hadoop commands submitted by Hive do not run as the `mapr` user, start the shell or connect to HiveServer2 as a user other than `mapr`. Then create some tables, and verify that the tables in `/user/hive/warehouse` are created under the user that started the shell or the user connected to HiveServer2.

Example: Hive Impersonation

The following examples illustrate Hive impersonation:

Example 1

1. Log in as a non-`mapr` user and generate a MapR ticket:

```
$ su mapruser1
$ maprlogin password
```

2. Connect to HiveServer2:

```
$ hive --service beeline
Beeline version 2.3.3-mapr-SNAPSHOT by Apache Hive
beeline> !connect jdbc:hive2://node4.cluster.com:10000/
default;ssl=true;auth=maprsasl
Connecting to jdbc:hive2://node4.cluster.com:10000/
default;ssl=true;auth=maprsasl
Connected to: Apache Hive (version 2.3.3-mapr-SNAPSHOT)
Driver: Hive JDBC (version 2.3.3-mapr-SNAPSHOT)
Transaction isolation: TRANSACTION_REPEATABLE_READ
```

3. Create a table, and upload data:

```
0: jdbc:hive2://node4.cluster.com:10000/defau> create table
impersonation_example_first (id int, username string);
0: jdbc:hive2://node4.cluster.com:10000/defau> insert into
impersonation_example_first values (1, 'mapruser1');
```

4. To check that impersonation works, use the following commands to check the `/warehouse` directory of the MapR file system:

```
$ hadoop fs -ls /user/hive/warehouse
Found 1 items
drwxr-xr-x   - mapruser1 mapruser1          1 2019-05-22 14:40
/user/hive/warehouse/impersonation_example_first

$ hadoop fs -ls /user/hive/warehouse/impersonation_example_first
Found 1 items
-rwxrwxrwx   3 mapruser1 mapruser1         12 2019-10-15 07:21
/user/hive/warehouse/impersonation_example_first/000000_0
```

Example 2

1. Generate a MapR ticket for a non-`mapr` user.
2. Connect through JDBC using the `hive.server2.proxy.user` option with a non-`mapr` user name as an argument:

```
$ hive --service beeline
beeline> !connect
jdbc:hive2://node4.cluster.com:10000/
default;auth=maprsasl;ssl=true;hive.server2.proxy.user=mapruser1
Connecting to
jdbc:hive2://node4.cluster.com:10000/
default;auth=maprsasl;ssl=true;hive.server2.proxy.user=mapruser1
Client: auth-conf,auth-int,auth.Using Server one
Connected to: Apache Hive (version 2.3.3-mapr-SNAPSHOT)
Driver: Hive JDBC (version 2.3.3-mapr-SNAPSHOT)
Transaction isolation: TRANSACTION_REPEATABLE_READ
```

3. Create a table and upload data:

```
0: jdbc:hive2://node4.cluster.com:10000/default> create table
impersonation_example_second (id int);
0: jdbc:hive2://node4.cluster.com:10000/default> insert into table
impersonation_example_second values (1), (2), (3), (5);
```

4. Check the owner of the table and data:

```
$ hadoop fs -ls /user/hive/warehouse/impersonation_example_second
Found 1 items
drwxrwxrwx   - mapruser1 mapruser1          1 2019-05-23 12:29
/user/hive/warehouse/impersonation_example_second

$ hadoop fs -ls /user/hive/warehouse/impersonation_example_second
Found 1 items
-rwxrwxrwx   3 mapruser1 mapruser1          8 2019-05-23 12:29
/user/hive/warehouse/impersonation_example_second/000000_0
```

Hive Security

You can configure the following features for Hive security:

Hive Security Configuration Options
This section describes changes made in Hive default configuration. It shows how to configure Hive after manual installation.

Unlike the previous releases, starting in EEP 4.0, Hive should be configured by running the `$MAPR_HOME/server/configure.sh` script with the `-R` option after installing Hive. Hive demons will not start automatically if Hive is not configured correctly. The security configuration are described in the following sections:

1. Automatic
2. Manual
3. Custom

Note: Do not use ecosystem `$HIVE_HOME/bin/configure.sh` script for Hive configuration. Every configuration of Hive should be done via the `$MAPR_HOME/server/configure.sh` utility by running it with the `-R` option. The core `$MAPR_HOME/server/configure.sh` utility invokes the ecosystem `configure.sh` script automatically with appropriate security option.

Automatic

If you installed Hive using the MapR Installer, the MapR Installer configures Hive daemons during installation. Additional configuration is not required.

Manual

After a new manual installation, to generate a valid default ecosystem configuration, run:

```
$MAPR_HOME/server/configure.sh -R
```

Table

Node, Package	Hive	HiveServer2	Hive Metastore	WebHCat
Node 1	X	X		
Node 2	X		X	
Node 3	X			X

- After a manual installation, run the following command on Node 1, Node 2, and Node 3:

  ```
  $MAPR_HOME/server/configure.sh -R
  ```

As a result:

- All Hive daemons are configured to support MapR-SASL. If the `hive.metastore.sasl.enabled` property is enabled in the `hive-site.xml` file, its value is set to `true`. If the property is not present, it is added in the `configuration` section as follows:

  ```
  <property>
     <name>hive.metastore.sasl.enabled</name>
     <value>true</value>
  </property>
  ```

- HiveServer2 is configured to support encryption between Hiveserver2 and Hive clients. If the `hive.server2.thrift.sasl.qop` property is available in the `hive-site.xml` file, its value is set to `auth-conf`. If the property is not present, it is added in the configuration section as follows:

  ```
  <property>
     <name>hive.server2.thrift.sasl.qop</name>
     <value>auth-conf</value>
  </property>
  ```

- The `configure.sh` script creates a backup folder for the current Hive configuration before it changes the configuration. All configuration properties including `*.conf`, `*.properties`, and `*.xml` are saved in the backup folder.

  ```
  $HIVE_HOME/conf.YYYYMMDD_HHMMS
  ```


last-updated: Apr 11, 2022

- 644 Unix permissions are applied to all configuration files. Each run of `configure.sh` with the `-R` option overwrites permissions of the configuration files to 644.
- The Hive default ports listed below are verified as available. If a port is not available, the `configure.sh` script generates an error message during configuration.

Hive default ports are as follows:

Role	Default Port
Hive Metastore	9083
HiveServer2	10000
HiveWebHCat	50111

Custom

For PAM, LDAP, and Kerberos custom configurations, run `configure.sh` with the `-R` option:

```
$MAPR_HOME/server/configure.sh -R
```

The `hive-site.xml` file is not changed. However, Warden files are copied and a `HIVE_HOME/conf` backup folder is created.

Preventing a Non-Administrative User from Installing Hooks

For a fresh install of EEP 6.1, a non-administrative user is prevented from installing hooks by default. For a minor version update (for example, EEP 6.0.0 to EEP 6.1.0 or EEP 5.0.1 to EEP 5.0.2), you need to modify the Hive configuration to prevent a malicious user from using Hive hooks to install malware on your MapR cluster.

In general, a hook is a mechanism for intercepting events, messages, or function calls during processing. Hive hooks are a mechanism to tie into the internal workings of Hive without the need of re-compiling Hive. Hive hooks, in this sense, provide the ability to extend and integrate external functionality with Hive.

Any user using beeline can install Java code as a Hive hook. On the MapR platform, these hooks run as the `mapr` user, which could represent a security vulnerability. To prevent a malicious user from using Hive hooks to install malware on a MapR cluster, the cluster admin should add the following properties to the default value of `hive.conf.restricted.list` in the `hive-site.xml` file, and then restart HiveServer 2 (HS2):

- `hive.exec.pre.hooks`
- `hive.exec.post.hooks`
- `hive.exec.failure.hooks`
- `hive.exec.query.redactor.hooks`

Adding the properties prevents a non-admin user from installing hooks into Hive.

1. Add all hook-related properties to the default value of `hive.conf.restricted.list` in the `hive-site.xml` file:

   Hive 2.3

   - `hive.exec.pre.hooks`
   - `hive.exec.post.hooks`
   - `hive.exec.failure.hooks`

- `hive.exec.query.redactor.hooks`
- `hive.semantic.analyzer.hook`
- `hive.query.lifetime.hooks`
- `hive.exec.driver.run.hooks`
- `hive.server2.session.hook`

Hive 2.1

- `hive.exec.pre.hooks`
- `hive.exec.post.hooks`
- `hive.exec.failure.hooks`
- `hive.exec.query.redactor.hooks`
- `hive.semantic.analyzer.hook`
- `hive.exec.driver.run.hooks`
- `hive.server2.session.hook`

2. Make sure `hive.conf.restricted.list` configuration parameter already has a default value which contains:

Hive 2.3

```
hive.security.authenticator.manager
hive.security.authorization.manager
Hive.security.metastore.authorizatio
n.manager
hive.security.metastore.authenticato
r.manager
Hive.users.in.admin.role,hive.server
2.xsrf.filter.enabled
hive.security.authorization.enabled
hive.server2.authentication.ldap.bas
eDN
hive.server2.authentication.ldap.url
hive.server2.authentication.ldap.Dom
ain
hive.server2.authentication.ldap.gro
upDNPattern
hive.server2.authentication.ldap.gro
upFilter
hive.server2.authentication.ldap.use
rDNPattern
hive.server2.authentication.ldap.use
rFilter
hive.server2.authentication.ldap.gro
upMembershipKey
hive.server2.authentication.ldap.use
rMembershipKey
hive.server2.authentication.ldap.gro
upClassKey
hive.server2.authentication.ldap.cus
tomLDAPQuery
```

Hive 2.1

```
hive.security.authenticator.manager
hive.security.authorization.manager
hive.users.in.admin.role
hive.server2.xsrf.filter.enabled
```

3. Add the default values already present in `hive.conf.restricted.list` to the `hive-site.xml` file:

Hive 2.3

```
<property>
  <name>hive.conf.restricted.list</
name>
  <value>

hive.security.authenticator.manager,

hive.security.authorization.manager,

hive.security.metastore.authorizatio
n.manager,

hive.security.metastore.authenticato
r.manager,

hive.users.in.admin.role,hive.server
2.xsrf.filter.enabled,

hive.security.authorization.enabled,

hive.server2.authentication.ldap.bas
eDN,

hive.server2.authentication.ldap.url
,

hive.server2.authentication.ldap.Dom
ain,

hive.server2.authentication.ldap.gro
upDNPattern,

hive.server2.authentication.ldap.gro
upFilter,

hive.server2.authentication.ldap.use
rDNPattern,

hive.server2.authentication.ldap.use
rFilter,

hive.server2.authentication.ldap.gro
upMembershipKey,

hive.server2.authentication.ldap.use
rMembershipKey,

hive.server2.authentication.ldap.gro
upClassKey,

hive.server2.authentication.ldap.cus
```

```
tomLDAPQuery,
     hive.exec.pre.hooks,
     hive.exec.post.hooks,
     hive.exec.failure.hooks,
     hive.exec.query.redactor.hooks,
     hive.semantic.analyzer.hook,
     hive.query.lifetime.hooks,
     hive.exec.driver.run.hooks,
     hive.server2.session.hook,
  </value>
</property>
```

Hive 2.1

```
<property>
  <name>hive.conf.restricted.list</
name>
  <value>

hive.security.authenticator.manager,

hive.security.authorization.manager,
     hive.users.in.admin.role,

hive.server2.xsrf.filter.enabled,
     hive.exec.pre.hooks,
     hive.exec.post.hooks,
     hive.exec.failure.hooks,
     hive.exec.query.redactor.hooks,
     hive.semantic.analyzer.hook,
     hive.exec.driver.run.hooks,
     hive.server2.session.hook,
  </value>
</property>
```

Note: Values of the `hive.conf.restricted.list` are split into separate lines for better readability. In the actual `hive-site.xml` file, no spaces or newlines exist between the commas.

Configuring Security Headers for Web Servers

This section describes how to configure response headers for REST API servers used in Hive WebHCat and the HiveServer2 web UI.

About the Headers File

The XML file with security headers is located at:

```
/opt/mapr/hive/hive-<version>/conf/headres.xml
```

The `headres.xml` file contains the following headers:

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>
  <entry key="X-Content-Type-Options">nosniff</entry>
  <entry key="X-XSS-Protection">1; mode=block</entry>
  <entry key="Strict-Transport-Security">max-age=31536000;
includeSubDomains</entry>
  <entry key="Content-Security-Policy">default-src https:</entry>
</properties>
```

This table describes each header:

Header	Description	Default Value
X-XSS-Protection	Stops pages from loading when reflected cross-site scripting (XSS) is detected. Supported by IE, Chrome, and Safari.	`1: mode=block`
X-Content-Type-Options	Indicates that the MIME types advertised in the `Content-Type` headers should not be changed and should be followed.	`nosniff`
Strict-Transport-Security	Tells all browsers that the website should only be accessed using HTTPS instead of using HTTP.	`max-age=31536000;includeSubDomains`
Content-Security-Policy	Allows web-site administrators to control resources the user agent is allowed to load for a given page. This helps guard against cross-site scripting attacks (XSS).	`default-src https:`

Configuring Security Headers for WebHCat

To enable security headers for WebHCat, add the following to the `webhcat-site.xml` file, and replace `<version>` with your Hive version:

```
<property>
<name>templeton.jetty.response.headers.file</name>
<value>/opt/mapr/hive/hive-<version>/conf/headers.xml</value>
</property>
```

After configuring and restarting WebHCat, you should see security headers in the server response. For example:

```
< HTTP/1.1 200 OK
< Date: Thu, 03 Oct 2019 11:35:39 GMT
< Set-Cookie:
hadoop.auth="u=mapr&p=mapr&t=multiauth&e=1570138539451&s=CpX+tI7sScnnSUZpA1K
df+7hamM="; Path=/; Domain=.cluster.com; Expires=Thu, 03-Oct-2019 21:35:39
GMT; Secure; HttpOnly
< Content-Security-Policy: default-src https:
< X-XSS-Protection: 1; mode=block
< X-Content-Type-Options: nosniff
< Strict-Transport-Security: max-age=31536000
< Content-Type: application/json
< Transfer-Encoding: chunked
< Server: Jetty(9.4.19.v20190610)
```

Configuring Security Headers for HiveServer2 Web UI

To enable security headers for the Hiveserver2 Web UI, add the following to the `hive-site.xml` file, replacing `<version>` with your Hive version:

```
<property>
    <name>hive.server2.webui.jetty.response.headers.file</name>
    <value>/opt/mapr/hive/hive-<version>/conf/headers.xml</value>
</property>
```

Then restart HiveServer2.

last-updated: Apr 11, 2022

Configuring Custom Headers

To configure custom headers for web servers, edit the `headers.xml` file, and add `Custom-header` as follows:

```
<entry key="Custom-header">custom-value</entry>
```

Security Headers Auto-Configuration

If you install Hive on a secure cluster (MapR SASL or Kerberos) and run the following command after Hive installation, Hive automatically configures itself to enable security headers, and no additional action is needed:

```
/opt/mapr/server/configure.sh -R
```

Hive Authentication
The authentication method that you configure for the Hive Metastore, HiveServer2, and WebHcat determines how these Hive components access and connect to each other.

Clients of these components may require additional configuration and specific connection strings based on the selected authentication method.

To enable and use authentication for Hive, complete the following steps:

1. Determine which authentication methods are supported for each component and its clients.
2. Configure authentication for Hive components and their clients. See the following topics:
   - Authentication for Hive Metastore
   - Authentication for HiveServer2
   - Authentication for WebHCat
3. Determine how clients connect to each component. See Connecting to Hive.

Hive Metastore Authentication Support

The following table describes the different supported authentication methods for Hive Metastore and how it impacts the authentication options for its clients:

MapR Cluster	Hive Metastore (Remote) Authentication	HiveServer 2 Authentication Options	WebHCat Authentication Options
Secure	NONE	• NONE • KERBEROS • LDAP • PAM • CUSTOM • MAPRSASL • NOSASL	PAM
Secure	KERBEROS	KERBEROS	KERBEROS with SPNEGO
Secure	MAPRSASL (default)*	MAPRSASL (default)*	PAM

last-updated: Apr 11, 2022

MapR Cluster	Hive Metastore (Remote) Authentication	HiveServer 2 Authentication Options	WebHCat Authentication Options
Not Secure	NONE	NONE	Simple authentication with <user.name> only
*As of Hive 0.13-1504 and Hive 1.0-1504, Hive Metastore supports MapR-SASL and MapR-SASL is enabled by default when the MapR cluster is secure.			

HiveServer2 Authentication Support

The following table describes the different supported authentication option for HiveServer2 based on the authentication method configured for Hive Metastore:

MapR Cluster	Hive Metastore (Remote) Authentication	HiveServer 2 Authentication Options
Secure	NONE	NONE
Secure	NONE	KERBEROS
Secure	NONE	LDAP
Secure	NONE	PAM (default)*
Secure	NONE	CUSTOM
Secure	NONE	MAPRSASL*
Secure	KERBEROS	KERBEROS
Secure	MAPRSASL (default)*	MAPRSASL*
Not Secure	NONE	NONE
*As of Hive 0.13-1510, Hive 1.0-1510, and Hive 1.2.1-1510, PAM and MapR-SASL are enabled by default when the cluster is secure. In Hive 0.13-1508 and Hive 1.0-1508, PAM is enabled by default when the cluster is secure. In Hive 0.13-1504 and Hive 1.0-1504, MapR-SASL is supported and enabled by default when the MapR cluster is secure.		

Clients of HiveServer2 authenticate with the same authentication method that is configured for HiveServer2. Clients of HiveServer 2 include ODBC, JDBC, and Beeline.

Note: Connections to HiveServer2 using ODBC do not support MapR-SASL.

WebHCat Authentication Support

The following table describes the different authentication options for WebHCat based on the authentication method configured for Hive Metastore :

MapR Cluster	Hive Metastore (Remote) Authentication	WebHCat Authentication
Secure	KERBEROS	KERBEROS with SPNEGO
Secure	KERBEROS	PAM
Secure	MAPRSASL (default)*	PAM
Not Secure	NONE	Simple authentication with user.name only

last-updated: Apr 11, 2022

MapR Cluster	Hive Metastore (Remote) Authentication	WebHCat Authentication
*As of Hive 0.13-1504 and Hive 1.0-1504, Hive Metastore supports MapR-SASL and MapR-SASL is enabled by default when the MapR cluster is secure.		

Clients of WebHCat authenticate with the same authentication method that is configured for WebHCat. Web browsers are clients of WebHCat.

Description of Security Values

The following table describes the different security values:

Authentication Options	Description
NONE	No authentication check
LDAP	LDAP/AD based authentication
KERBEROS	Kerberos/GSSAPI authentication
CUSTOM	Custom authentication provider (use with property `hive.server2.custom.authentication.class`)
PAM	Pluggable authentication module
NOSASL	Raw transport
MAPRSASL	MapR SASL security

Authentication for Hive Metastore

You can configure authentication for in-bound client connections to the Hive Metastore when the metastore is remote, not embedded. Clients of Hive Metastore include the HiveCLI, HCatalog, HiveServer2, and WebHCat.

Hive Metastore supports the following authentication methods:

- MapR-SASL authentication
- Kerberos Authentication

MapR-SASL Authentication

MapR-SASL is available starting with the 1504 release of Hive 0.13 and Hive 1.0 and it is the default authentication method when the cluster is secure.

Kerberos Authentication

When the cluster is secure, you can configure Hive Metastore to use Kerberos authentication. You must also configure Hive Metastore clients to use Kerberos when authenticating with Hive Metastore.

Configuring Hive Metastore Authentication

This section describes how to configure the `hive.metastore.authentication` property for secured and unsecured clusters. It describes cases when the property must be configured explicitly and when it can be omitted from `hive-site.xml`.

Hive Metastore supports two types of authentication: `MAPRSASL` and `KERBEROS`. At startup, Hive Metastore reads the system property `metastore.auth`. If `metastore.auth` is equal to null, then the authentication type is `NONE`. Otherwise, Hive Metastore takes the value of the

system property `metastore.auth` and assigns it to the Hive Metastore configuration property `hive.metastore.authentication`.

You do not need to set up the `metastore.auth` system property manually. If a cluster is secured, Hive assigns the `MAPRSASL` value to the `metastore.auth` property. If a cluster is not secured, Hive assigns the `NONE` value to the `metastore.auth` property.

To enable Kerberos authentication, set the value of `hive.metastore.authentication` directly in `hive-site.xml`, as shown in the following table:

Table

Security	Value of `hive.metastore.authentication`	Notes
No security	`NONE`	The value is set automatically. You do not need to make an entry in `hive-site.xml`.
MapR SASL security	`MAPRSASL`	The value is set automatically. You do not need to make an entry in `hive-site.xml`.

Kerberos security	`KERBEROS`	You must make the following entry in `hive-site.xml`:
		`<property>`
		`<name>hive.metastore.authentication</name>`
		`<value>KERBEROS</value>`
		`</property>`

Configure Hive Metastore to use MapR-SASL

Edit the `/opt/mapr/conf/env.sh` file and set the following properties:

```
MAPR_HIVE_LOGIN_OPTS="-Dhadoop.login=maprsasl"
MAPR_HIVE_SERVER_LOGIN_OPTS="-Dhadoop.login=maprsasl_keytab"
```

Configuring Hive Metastore Clients to use MapR-SASL

The Hive metastore clients are configured to use MapR-SASL when authenticating with Hive Metastore.

Note: Hive Metastore clients must provide a valid MapR ticket to connect to the Hive Metastore. See Connecting to Hive on page 3515 for details.

1. Ensure that the cluster is secure.
2. Edit the `/opt/mapr/conf/env.sh` file and set the following property:

```
MAPR_HIVE_LOGIN _OPTS="-Dhadoop.login=maprsasl"
```

Configure Hive Metastore to use Kerberos

Enabling Hive Metastore to use Kerberos authentication requires a kerberos principal, kerberos keytab, and the following configurations.

last-updated: Apr 11, 2022

Complete the following steps on each node where a Hive Metastore is installed:

1. Create a Kerberos server identity and add it to a keytab file. You can use the following commands in a Linux-based Kerberos environment to set up the identity and update the keytab file:

   Note: MapR clusters do not provide Kerberos infrastructure. The tips in this step assume a Linux-based Kerberos environment, and the specific commands for your environment may vary. Consult with your Kerberos administrator for assistance.

   ```
   # kadmin
       : addprinc -randkey username/<FQDN@REALM>
       : ktadd -k /opt/mapr/conf/hive.keytab username/<FQDN@REALM>
   ```

   The `hive.keytab` file must be owned and readable only by the `mapr` user.

2. Configure the following properties in the following file:

   ```
   /opt/mapr/hive/hive-<version>/conf/hive-site.xml
   ```

Property	Value
`hive.metastore.kerberos.keytab.file`	The Keytab file that contains the HiveMetastore principal.
`hive.metastore.kerberos.principal`	<The HiveMetastore principal. For example, mapr/<FQDN@REALM>.>

```
<property>
  <name>hive.metastore.kerberos.keytab.file</name>
  <value>/opt/mapr/conf/metastore.keytab</value>
  <description>The path to the Kerberos Keytab file
containing the metastore thrift server's service principal.</
description>
</property>
<property>
  <name>hive.metastore.kerberos.principal</name>
  <value>mapr/<FQDN@REALM></value>
  <description>The service principal for the metastore thrift server.
The special string _HOST will be replaced automatically with the correct
hostname.</description>
</property>
```

3. Configure the following properties in `/opt/mapr/conf/env.sh` on each node where the Hive Metastore is installed:

   - Set MAPR_HIVE_LOGIN _OPTS to

     ```
     "-Dhadoop.login=hybrid"
     ```

   - Set MAPR_HIVE_SERVER_LOGIN_OPTS to

     ```
     "-Dhadoop.login=hybrid"
     ```

Configure Hive Metastore Clients to use Kerberos

When the Hive Metastore is configured to use Kerberos authentication, you must also configure Hive Metastore Clients to use Kerberos when authenticating with Hive Metastore.

last-updated: Apr 11, 2022

Complete the following steps on each node where a Hive Metastore client is installed:

1. Configure MAPR_HIVE_LOGIN _OPTS to `" -Dhadoop.login=hybrid"` in `/opt/mapr/conf/env.sh`.
2. Configure the following property in `hive-site.xml`:

Property	Value
`hive.metastore.kerberos.principal`	The HiveMetastore principal. For example, mapr/<FQDN@REALM>.

```
<property>
  <name>hive.metastore.kerberos.principal</name>
  <value>mapr/<FQDN@REALM></value>
  <description>The service principal for the metastore thrift server.
The special string _HOST will be replaced automatically with the correct
hostname.</description>
</property>
```

See Connecting to Hive on page 3515 for details on how to connect to HiveMetastore once the server and client node are configured to use Kerberos.

Note: The MAPR_HIVE_LOGIN_OPTS and MAPR_HIVE_SERVER_LOGIN_OPTS were added in 1504 release of Hive 0.13 and Hive 1.0. If you have Hive 0.13 from a prior release, you do not need to configure these properties. Instead, set MAPR_ECOSYSTEM_LOGIN_OPTS and MAPR_ECOSYSTEM_SERVER_LOGIN_OPTS to `"-Dhadoop.login=hybrid"` in `/opt/mapr/conf/env.sh`.

Authentication for HiveServer2

You can configure authentication for in-bound client connection to HiveServer2. Clients of HiveServer 2 include beeline and odbc/jdbc client applications.

Credentials are submitted from the HiveServer2 clients to HiveServer2 as plain text. To secure the credential transmission, MapR supports SSL encryption for HiveServer2. For information about how to configure encryption, see Hive Encryption.

HiveServer2 supports the following authentication methods:

Configure MapR-SASL Authentication for HiveServer 2

MapR-SASL is available starting with the 1504 release of Hive 0.13 and Hive 1.0. However, the configuration requirements for MapR-SASL differ based on the version of Hive that you have installed:

- As of Hive 0.13-1501, Hive 1.0-1510, and Hive 1.2-1510, MapR-SASL and PAM are enabled by default on a secure cluster; no configuration is required. Complete the steps below if you want HiveServer2 to only accept MapR-SASL authentication.
- In Hive 0.13-1508 and Hive 1.0-1508, MapR-SASL is not the default and must be configured.

last-updated: Apr 11, 2022

- In Hive 0.13-1504 and Hive 1.0-1504, MapR-SASL is the default authentication method when the cluster is secure. No configuration is required.

1. Configure the following property in hive-site.xml on each node where HiveServer2 is installed:

Property	Value
hive.server2.authentication	MAPRSASL

```
<property> <name>hive.server2.authentication</name> <value>MAPRSASL</
value></property>
```

2. Restart HiveServer2 to apply these changes.

```
maprcli node services -name hs2 -action restart -nodes <comma separated
list of
    nodes>
```

Configure HiveServer2 to use LDAP Authentication

1. Configure the following properties in the `hive-site.xml` file on each node where HiveServer2 is installed:

Property	Value
hive.server2.authentication	LDAP
hive.server2.authentication.ldap.url	The access URL for your LDAP server
hive.server2.authentication.ldap.baseDN	The base LDAP DN for your LDAP server. For example, `ou=People,dc=mycompany,dc=com`.
hive.server2.authentication.ldap.userDNPattern	User DN Pattern - A DN pattern that can be used to directly login users to the LDAP database. This pattern is used for creating a DN string for "direct" user authentication, where the pattern is relative to the base DN in `ldapUrl`.

```
<property>
  <name>hive.server2.authentication</name>
  <value>LDAP</value>
</property>
<property>
  <name>hive.server2.authentication.ldap.url</name>
  <value><LDAP URL></value>
</property>
<property>
  <name>hive.server2.authentication.ldap.baseDN</name>
  <value><LDAP Base DN></value>
</property>
```

For generic LDAP servers, you must use:

a. `hive.server2.authentication.ldap.baseDN`

b. `hive.server2.authentication.ldap.userDNPattern`

However, Active Directory (AD) does not require the above two options, they can be replaced by the following property:

- hive.server2.authentication.ldap.Domain

last-updated: Apr 11, 2022

Property	Value
hive.server2.authentication.ldap.Domain	The active directory domain for your environment.

```
<property>
  <name>hive.server2.authentication.ldap.Domain</name>
  <value><AD Domain Name></value>
</property>
```

2. Restart HiveServer2 to apply these changes.

```
maprcli node services -name hs2 -action restart -nodes <comma separated
list of nodes>
```

Configure HiveServer2 to use PAM Authentication

You can configure HiveServer2 to use Pluggable Access Modules (PAM). The configuration requirements for PAM differ based on the version of Hive that you have installed.

Hive Version		Default Configuration	Configuration Requirement
Hive 2.3	1904	MapR-SASL and PAM are enabled by default on a secure cluster.	No configuration is required.
	1901	MapR-SASL and PAM are enabled by default on a secure cluster.	No configuration is required.
	1808	MapR-SASL and PAM are enabled by default on a secure cluster.	No configuration is required.
Hive 2.1	1904	MapR-SASL and PAM are enabled by default on a secure cluster.	No configuration is required.
	1901	MapR-SASL and PAM are enabled by default on a secure cluster.	No configuration is required.
	1808	MapR-SASL and PAM are enabled by default on a secure cluster.	No configuration is required.
	1803	MapR-SASL and PAM are enabled by default on a secure cluster.	No configuration is required.

Configure HiveServer2 to explicitly use PAM Authentication

1. In the `hive-site.xml` on each HiveServer2 node, set the `hive.server2.authentication` property to PAM:

```
<property>
  <name>hive.server2.authentication</name>
  <value>PAM</value>
</property>
```

2. Restart HiveServer2 to apply these changes:

```
maprcli node services -name hs2 -action restart -nodes <comma-separated
list of nodes>
```

Configure HiveServer 2 to use Custom Authentication

1. Create a custom Authenticator class derived from the following interface:

```
public interface PasswdAuthenticationProvider {
  /**
   * The Authenticate method is called by the HiveServer2 authentication
layer
   * to authenticate users for their requests.
   * If a user is to be granted, return nothing/throw nothing.
   * When a user is to be disallowed, throw an appropriate {@link
AuthenticationException}.
   *
   * For an example implementation, see {@link
LdapAuthenticationProviderImpl}.
   *
   * @param user - The username received over the connection request
   * @param password - The password received over the connection request
   * @throws AuthenticationException - When a user is found to be
   * invalid by the implementation
   */
  void Authenticate(String user, String password) throws
AuthenticationException;
}
```

   The SampleAuthenticator.java on page 3442 code has an example implementation that has stored usernames and passwords.

2. Configure the following properties in the `hive-site.xml` file on each node where HiveServer2 is installed:

Property	Value
hive.server2.authentication	CUSTOM
hive.server2.custom.authentication.class	The authentication class name. For example, hive.server2.custom.authentication.class

```
<property>
<name>hive.server2.authentication</name>
<value>CUSTOM</value>
</property>

<property>
<name>hive.server2.custom.authentication.class</name>
<value>hive.test.SampleAuthenticator</value>
</property>
```

3. Restart Hiveserver2 to apply the changes:

```
maprcli node services -name hs2 -action restart -nodes <comma separated
list of nodes>
```

SampleAuthenticator.java

```
package hive.test;

import java.util.Hashtable;
import javax.security.sasl.AuthenticationException;
import org.apache.hive.service.auth.PasswdAuthenticationProvider;
```

```
/*
 javac -cp $HIVE_HOME/lib/hive-service-0.11-mapr.jar
SampleAuthenticator.java -d .
 jar cf sampleauth.jar hive
 cp sampleauth.jar $HIVE_HOME/lib/.
*/


public class SampleAuthenticator implements PasswdAuthenticationProvider {

  Hashtable<String, String> store = null;

  public SampleAuthenticator () {
    store = new Hashtable<String, String>();
    store.put("user1", "passwd1");
    store.put("user2", "passwd2");
  }

  @Override
  public void Authenticate(String user, String  password)
      throws AuthenticationException {

    String storedPasswd = store.get(user);

    if (storedPasswd != null && storedPasswd.equals(password))
      return;

    throw new AuthenticationException("SampleAuthenticator: Error
validating user");
  }

}
```

Configure HiveServer 2 to use Kerberos

Note: You can configure HiveServer2 to use Kerberos authentication. MapR clusters do not provide Kerberos infrastructure. The tips in this section assume a Linux-based Kerberos environment, and the specific commands for your environment may vary. Consult with your Kerberos administrator for assistance.

Enabling HiveServer to use Kerberos authentication requires following steps on each node where HiveServer 2 is installed:

1. Create a Kerberos Identity and keytab. You can use the following commands in a Linux-based Kerberos environment to set up the identity and update the keytab file: The `hive.keytab` file must be owned and readable only by the `mapr` user.

   ```
   # kadmin
       : addprinc -randkey username/<FQDN@REALM>
       : ktadd -k /opt/mapr/conf/hive.keytab username/<FQDN@REALM>
   ```

2. Configure the following properties in hive-site.xml on each node where hiveserver2 is installed:

Property	Value
hive.server2.authentication	KERBEROS
hive.server2.authentication.kerberos.principal	<HiveServer2 Principle. For example, mapr/FQDN@REALM>

last-updated: Apr 11, 2022

Property	Value
hive.server2.authentication.kerberos.keytab	<The keytab file for the HiverServer2 principle. For example, /opt/mapr/conf/hive.keytab>

```
<property>
  <name>hive.server2.authentication</name>
  <value>KERBEROS</value>
  <description>authenticationtype</description>
</property>
<property>
  <name>hive.server2.authentication.kerberos.principal</name>
  <value>mapr/FQDN@REALM</value>
  <description>HiveServer2 principal. If _HOST is used as the FQDN
portion, it will be replaced with the actual hostname of the running
instance.</description>
</property>
<property>
  <name>hive.server2.authentication.kerberos.keytab</name>
  <value>/opt/mapr/conf/hive.keytab</value>
  <description>Keytab file for HiveServer2 principal</description>
</property>
```

3. Reconfigure the following options in `env.sh` (`/opt/mapr/conf/env.sh`) on each node where hiveserver2 is installed:

   Note: These configurations are listed in the portion of the file that begins with `if [ "$MAPR_SECURITY_STATUS" = "true" ];`. However, you should make the changes in the `/opt/mapr/conf/env_override.sh` file. For more information, see About env_override.sh on page 2290.

Existing Configuration	Required Configuration
`MAPR_HIVE_SERVER_LOGIN_OPTS="-Dhadoop.login=maprsasl_keytab"` `MAPR_HIVE_LOGIN_OPTS="-Dhadoop.login=maprsasl"`	`MAPR_HIVE_SERVER_LOGIN_OPTS="-Dhadoop.login=hybrid"` `MAPR_HIVE_LOGIN_OPTS="-Dhadoop.login=hybrid"`

4. Restart HiveServer2 to apply these changes.

```
maprcli node services -name hs2 -action restart -nodes <comma separated
list of nodes>
```

Configure HiveServer 2 Clients to use Kerberos

When HiveServer 2 is configured to use Kerberos authentication, you must also configure HiveServer2 clients to use Kerberos.

On each node where HiveServer2 clients (not including Beeline) are installed, reconfigure the following option in `env.sh` (`/opt/mapr/conf/env.sh`) file:

Existing Configuration	Required Configuration
`MAPR_HIVE_LOGIN_OPTS="-Dhadoop.login=maprsasl"`	`MAPR_HIVE_LOGIN_OPTS="-Dhadoop.login=hybrid"`

last-updated: Apr 11, 2022

Note: This configuration is listed in the portion of the file that begins with `if [ "$MAPR_SECURITY_STATUS" = "true" ];`. However, you should make the change in the `/opt/mapr/conf/env_override.sh` file. For more information, see About env_override.sh on page 2290.

On each node where Beeline is installed, reconfigure the following option in `beeline.sh` (`$hive_home/bin/ext/beeline.sh`) file:

Existing Configuration	Required Configuration
`HADOOP_OPTS="$HADOOP_OPTS${MAPR_HIVE_LOGIN_OPTS}"`	`HADOOP_OPTS="$HADOOP_OPTS${KERBEROS_LOGIN_OPTS}"`

For more information, see Connecting to Hive on page 3515.

Note: The MAPR_HIVE_LOGIN_OPTS and MAPR_HIVE_SERVER_LOGIN_OPTS were added in 1504 release of Hive 0.13 and Hive 1.0. If you have Hive 0.13 from a prior release, you do not need to configure these properties. Instead, set MAPR_ECOSYSTEM_LOGIN_OPTS and MAPR_ECOSYSTEM_SERVER_LOGIN_OPTS to `"-Dhadoop.login=hybrid"` in /opt/mapr/conf/env.sh.

Configure HiveServer2 Web UI to use PAM Authentication

You can configure HiveServer2 web UI to use Pluggable Access Modules (PAM) authentication. The following Hive properties are added to enable PAM authentication for the HiveServer2 web UI:

```
hive.server2.webui.use.pam
Default value: false
Description: If true, the HiveServer2 WebUI will be secured with PAM
```

```
hive.server2.webui.pam.authenticator
Default value: org.apache.hive.http.security.PamAuthenticator
Description: Class for PAM authentication
```

Modifying the hive-site.xml file:

Configure the following properties in the hive-site.xml file to enable authentication on each node where HiveServer2 is installed:

```
<!-- HS2 web UI PAM -->
<property>
  <name>hive.server2.webui.use.pam</name>
  <value>true</value>
</property>

<!-- HS2 web UI SSL -->
<property>
  <name>hive.server2.webui.use.ssl</name>
  <value>true</value>
</property>

<property>
  <name>hive.server2.webui.keystore.path</name>
  <value>/opt/mapr/conf/ssl_keystore</value>
</property>

<property>
  <name>hive.server2.webui.keystore.password</name>
```


last-updated: Apr 11, 2022

```
  <value><ssl-keystore-password></value>
</property>
```

Note: After running `/opt/mapr/server/configure.sh -R`, all properties needed for HiveServer2 Web UI to use PAM authentication is added automatically to hive-site.xml on the Mapr-SASL secure cluster. Connections to HiveServer2 using ODBC do not support MapR-SASL.

Authentication for WebHCat

You can configure authentication for in-bound client connections to WebHCat. Clients of WebHCat include web browsers. WebHCat is a client of Hive Metastore.

WebHCat supports the following authentication methods:

Configure Kerberos Authentication for WebHCat

When security features are enabled on your MapR cluster and Kerberos is in use, communications between WebHCat and its clients can use Kerberos with SPNEGO.

To enable WebHCat to use Kerberos, complete the following steps on the node where WebHCat is installed.

1. Create the principal `HTTP/<FQDN@REALM>` for WebHCat and add the principal to the keytab file. For example:

   ```
   kadmin: addprinc -randkey HTTP/<FQDN@REALM>
   kadmin: xst -k /opt/mapr/HTTP.keytab HTTP/<FQDN>
   ```

2. Verify the following:
   - The principal was added to the `/opt/mapr/conf/HTTP.keytab` file and that the file is only readable by the `mapr` user. For example: chown mapr /opt/mapr/conf/HTTP.keytab
   - The node where the WebHCat server is running has an HTTP user with a valid `maprlogin` password.

3. Add the following section to the `/opt/mapr/hive/hive-<version>/hcatalog/etc/webhcat/webhcat-site.xml` file:

   ```
   <property>
       <name>templeton.kerberos.secret</name>
       <value>secret value</value>
   </property>
   <property>
       <name>templeton.kerberos.principal</name>
       <value>HTTP/<FQDN@REALM></value>
   </property>
   <property>
       <name>templeton.kerberos.keytab</name>
       <value>/opt/mapr/conf/HTTP.keytab</value>
   </property>
   ```


last-updated: Apr 11, 2022

4. Add the following section to the `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/core-site.xml` file:

```
<property>
      <name>hadoop.proxyuser.HTTP.groups</name>
      <value>*</value>
      <description>Allow the superuser mapr to impersonate any member of
any group</description>
</property>
<property>
      <name>hadoop.proxyuser.HTTP.hosts</name>
      <value>*</value>
      <description>The superuser can connect from any host to
impersonate a user</description>
</property>
```

5. Start WebHCat. See Managing the WebHCat Server.

6. To test if the connection is working, generate a Kerberos ticket with the `kinit` utility and then run the following command:

```
curl --negotiate -i -u : 'http://<FQDN>:50111/templeton/v1/ddl/
database/'
```

Configure Simple Authentication for WebHCat

When the MapR cluster is not secure, simple authentication is enabled for WebHCat. No configuration is required.

Configure PAM Authentication for WebHCat

When the MapR cluster is secure, username and password authentication is enabled for WebHCat. No configuration is required.

Hive Encryption

When you configure encryption, the thrift messages sent between the Hive Metastore, HiveServer 2, and HiveServer2 clients are encrypted.

When you configure encryption, the thrift messages sent between the Hive Metastore, HiveServer 2, and HiveServer2 clients are encrypted.

Encryption is supported when HiveServer2 has no authentication or when it is configured to use MapR-SASL or Kerberos authentication.

This section contains the following topics:

Configure Encryption with MapR-SASL or Kerberos Authentication

Complete the following steps on each node where HiveServer2 is installed:

1. In `hive-site.xml` file, set the following property:

Property	Value
`hive.server2.thrift.sasl.qop`	`auth-conf`

Note: As of Hive 0.13-1504 and Hive 1.0-1504, `hive.server2.thrift.sasl.qop` is set to `auth-conf` by default on secure clusters.

```
<property>
  <name>hive.server2.thrift.sasl.qop</name>
  <value>auth-conf</value>
  <description>Sasl QOP value; one of 'auth', 'auth-int' and
```


last-updated: Apr 11, 2022

```
'auth-conf'</description>
</property>
```

2. Restart HiveServer2 to apply these changes.

```
maprcli node services -name hs2 -action restart -nodes <comma separated
list of nodes>
```

Configure Encryption without Authentication

Complete the following steps on each node where HiveServer2 is installed:

1. In `hive-site.xml` file, set the following properties:

Property	Value
`hive.server2.use.SSL`	`true`
`hive.server2.ssl.keystore`	<path to keystore file>
`hive.server2.ssl.keystore.password`	<password>

Warning: If you specify the password in the `hive-site.xml` file, protect the file with the appropriate file permissions. HiveServer2 automatically prompts for the keystore password during startup when no password is stored in the `hive-site.xml` file.

```
<property>
  <name>hive.server2.use.ssl</name>
  <value>true</value>
  <description>enable/disable SSL communication</description>
</property>
<property>
  <name>hive.server2.ssl.keystore</name>
  <value><path-to-keystore-file></value>
  <description>path to keystore file</description>
</property>

<property>
  <name>hive.server2.ssl.keystore.password</name>
  <value><password></value>
  <description>keystore password</description>
</property>
```

2. Restart HiveServer2 to apply these changes.

```
maprcli node services -name hs2 -action restart -nodes <comma separated
list of nodes>
```

Configure HiveServer2 Clients to use Encryption

Based on the encryption method, the requirements for clients to connect to HiveServer2 differ.

- When HiveServer2 uses encryption with MapR-SASL or Kerberos authentication, the client must specify the same sasl qop value that is set for HiveServer2 (auth-conf is the default, recommended option).
- When HiveServer2 uses SSL encryption without authentication, the client must specify a truststore. The `ssl_truststore` file must be copied from the cluster to the client. Specifying a truststore password is optional.

last-updated: Apr 11, 2022

For details, see Connecting to Hive on page 3515.

Configure the TLS (SSL) Protocol Version in Hive

Beginning with EEPs 6.3.1 and 7.0.0, the default protocol version for TLS (SSL) is `TLSv1.2`, but you can use the `hive.ssl.protocol.version` property to set a custom value for TLS (SSL).

Setting the TLS (SSL) Protocol Version

To enable the direct configuration of the TLS (SSL) version, Hive provides the following property:

Property	Type	Default Value	Description
`hive.ssl.protocol.version`	`String`	`TLSv1.2`	SSL protocol versions for all Hive servers.

To set a custom value for the TLS (SSL) protocol version in Hive:

1. Add the following to the `hive-site.xml` file:

```
<property>
<name>hive.ssl.protocol.version</name>
<value><custom_value></value>
</property>
```

In this example, `<custom_value>` can be one of the following:

- `SSLv2`
- `SSLv3`
- `SSLv2Hello`
- `TLSv1`
- `TLSv1.1`
- `TLSv1.2`

For more information, see the following table:

#	Algorithm Name (TLS/SSL Version)	Description
1	`Default`	Use the default algorithm.
2	`SSL`	Supports some versions of SSL; may support other versions.
3	`SSLv2`	Supports SSL version 2 or later; may support other versions.
4	`SSLv3`	Supports SSL version 3; may support other versions.
5	`TLS`	Supports some versions of TLS; may support other versions.
6	`TLSv1`	Supports RFC 2246: TLS version 1.0; may support other versions.
7	`TLSv1.1`	Supports RFC 4346: TLS version 1.1; may support other versions.
8	`TLSv1.2`	Supports RFC 5246: TLS version 1.2; may support other versions.

2. Restart all Hive services.

last-updated: Apr 11, 2022

Special Considerations for Protocol Versions

Note these special considerations for the protocol versions:

- When `hive.ssl.protocol.version` is set to `TLSv1.2`, the protocol supports TLS 1.2. When `hive.ssl.protocol.version` is set to `TLSv1`, the protocol supports TLS versions up to TLS 1.0 (but not TLS 1.1 and 1.2). When `hive.ssl.protocol.version` is set to `TLSv1.1`, the protocol supports versions up to TLS 1.1 (but not TLS 1.2).
- `SSLv2Hello` is not a real encryption protocol. It merely enables clients to find out which encryption protocols are supported by the server to which they connect. As long as `SSLv2Hello` is used only by clients and servers to negotiate a safe protocol, such as `TLSv1.1` or `TLSv1.2`, it does not pose a security risk.
- Hive has a property called `hive.ssl.protocol.blacklist` with a default value of `SSLv2,SSLv3,SSLv2Hello,TLSv1,TLSv1.1`. If you want to enable `TLSv1.1`, for example, you must remove it from the blacklist above. For example:

```
<property>
<name>hive.ssl.protocol.blacklist</name>
<value>SSLv2,SSLv3,SSLv2Hello,TLSv1</value>
</property>

<property>
<name>hive.ssl.protocol.version</name>
<value>TLSv1.1</value>
</property>
```

- If you use the TLS (SSL) protocol version from the blacklist, you will get the following exception when connecting to Hiveserver2 via JDBC:

```
Unknown HS2 problem when communicating with Thrift server.
Error: Could not open client transport with JDBC
Uri: jdbc:hive2://<hostname>:10000/default;auth=maprsasl;ssl=true:
javax.net.ssl.SSLHandshakeException: Received fatal alert:
handshake_failure (state=08S01,code=0)
```

- Empty values are allowed for `hive.ssl.protocol.version`. Hive uses the default value in that case. The same is true for `hive.ssl.protocol.blacklist`.
- The `hive.ssl.protocol.version` property is out of scope for a secure-by-default configuration. This means that it will not appear in the `hive-site.xml` after you use the Hive `configure.sh` script. Nevertheless, the default value of `hive.ssl.protocol.version` is still `TLSv1.2`, and you do not need to set it explicitly.

Hive Password Encryption
EEP 4.0 introduces default configuration for Hive Metastore password encryption using the MapR Installer. The password is stored in the `hive-site.xml` file.

EEP 4.0 introduces default configuration for Hive Metastore password encryption using the MapR Installer. The password is stored in the `hive-site.xml` file.

Note: For Hive-2.1 (EEP-5.0.0 and later) and Hive-2.3 (EEP-6.0.0 and later) installed using the MapR Installer, `javax.jdo.option.ConnectionPassword` is automatically encrypted.

```
<property>
  <name>javax.jdo.option.ConnectionPassword<name>
```

```
  <value>{password}<value>
<property>
```

The `hadoop.security.credential.provider.path` configuration property replaces the `javax.jdo.option.ConnectionPassword` property in the `hive-site.xml` file that contains the path to the keystore file created by the Hadoop Crediential Provider. Credential providers store and protect passwords out of clear text for the underlying database. By default, the MapR Installer creates the keystore file in MapR filesystem. `/user/${MAPR_USER}/hivemetastore.jceks`.

Note: Starting from Hive-2.3 EEP 6.0.0, SSL keystore passwords, `hive.server2.webui.keystore.password`, `hive.server2.keystore.password`, and `templeton.keystore.password`, are automatically read from the `/opt/mapr/conf/ssl-client.xml` file without any additional steps from your side. But you can still encrypt them manually and store them in the `*jceks` files.

Reset MapR Installer Default Configuration

To remove changes made by the MapR Installer and reset Hive to its default setting:

1. Open the `hive-site.xml` file.
2. Delete the `hadoop.security.credential.provider.path` property.
3. Add the `javax.jdo.option.ConnectionPassword` property.
4. Save and close the `hive-site.xml` file.

Manual Password Encryption

Note: For any user to use Hive, the keystore file requires read permission (644). To limit keystore file access to a smaller number of Hive users, modify permissions as necessary.

To encrypt a password manually:

1. Create the keystore file using the Hadoop Credential Provider as follows:

   ```
   hadoop credential create javax.jdo.option.ConnectionPassword -provider
   <path-to-keystore>
   ```

   Where `<path-to-keystore>` is `jceks://<file-system-name>/<path-to-keystore>`.

   For example, `jceks://maprfs/user/mapr/hivemetastore.jceks`.

2. Delete the `javax.jdo.option.ConnectionPassword` property in the `hive-site.xml` file:

   ```
   <property>
     <name>javax.jdo.option.ConnectionPassword</name>
     <value>{yourpassword}</value>
   </property>
   ```


last-updated: Apr 11, 2022

3. Add the `hadoop.security.credential.provider.path` property to the `/opt/mapr/hive/hive-2.3/conf/hive-site.xml` file:

```
<property>
  <name>hadoop.security.credential.provider.path</name>
  <value>jceks://maprfs/user/mapr/hivemetastore.jceks</value>
  <description>password to use against metastore database</description>
</property>
```

4. Restart the Hive services to update the configuration:

```
maprcli node services -name hivemeta -action restart -nodes `hostname -f`
maprcli node services -name hs2 -action restart -nodes `hostname -f`
maprcli node services -name hcat -action restart -nodes `hostname -f`
```

Hive Authorization

MapR Data Platform has built-in platform authorization that protects all data regardless of the execution engine. This topic describes alternative authorization modes you can choose to implement.

For more information, refer to Authorization in MapR on page 687.

In addition to the centralized authorization provided by the MapR Data Platform, you can use several authorization modes for Hive. The use cases and trade-offs for these authorization modes are described in the sections below.

Understanding Hive Authorization Use Cases

Table Storage Layer and SQL Query Engine are the two primary use cases for client-based authorization protection, delivered as part of the open source project.

Use Case 1: Table Storage Layer

This is the use case for Hive HCatalog API users.

In this case, Hive provides a table abstraction and metadata for files on storage (typically MapR filesystem). You have direct access to MapR filesystem and the metastore server (which provides an API for metadata access).

MapR filesystem access is authorized through the use of MapR filesystem permissions. You need to authorize metadata access using Hive configuration.

Use Case 2: SQL Query Engine

This is one of the most common use cases of Hive. This is the "Hive view" of SQL users and BI tools. This use case has the following two subcategories:

- Hive command line users - You have direct access to MapR filesystem and the Hive metastore, which makes this use case similar to use case 1.
- ODBC/JDBC and other HiveServer2 API users (Beeline CLI is an example) - You have all data or metadata access through HiveServer2. You do not have direct access to MapR filesystem or the metastore.

Understanding Hive Authorization Modes

Different modes of Hive authorization are available to satisfy different use cases.

Secure by Default Configuration (Storage Based Authorization in the Metastore Server)

Hive default security configuration is the storage based authorization in the Metastore server. Managed by mapr-tickets and impersonation level, Hive configurations control the data access and MapR filesystem permissions act as one source of truth for the table storage access. By enabling storage based

authorization in the metastore server, you can use this single source for truth and have a consistent data and metadata authorization policy.

For use cases where the users have direct access to the data, Hive configurations do not control the data access. The MapR filesystem permissions act as the one source of truth for table storage access. To control metadata access on the metadata objects such as databases, tables, and partitions, MapR filesystem checks if you have permission to access the corresponding directories on the filesystem.

You can also protect access through HiveServer2 (use case 2.2) by ensuring that the queries run as the end user. The `hive.server2.enable.doAs` option should be `true` in the HiveServer2 configuration, this is a default value.

For more information, see Hive Security Configuration Options on page 3427.

SQL Standards Based Authorization in HiveServer2

Although storage based authorization provides access control at the level of databases, tables, and partitions, it can only control authorization at finer levels such as columns and views for MapR Database tables and not for files because the access control provided by the filesystem is at the level of directory and files. SQL standards authorization makes authorization possible for files BUT at the expense of not being able to enforce that access from any other tool.

For enabling SQL standards based authorization, refer to SQL Standards-Based Hive Authorization on page 3453.

Legacy Hive Authorization

Old default authorization is the authorization mode that has been available in earlier versions of Hive. However, this mode does not have a complete access control model, leaving many security gaps unaddressed.

For example, the permissions needed to grant privileges for a user are not defined, and any user can grant themselves access to a table or database.

This model is similar to the SQL standards based authorization mode, in that it provides grant or revoke statement-based access control. However, the access control policy is different from SQL standards based authorization, and they are not compatible. Use of this mode is also supported for Hive command line users. However, for reasons mentioned under the discussion of SQL standards based authorization, it is not a secure mode of authorization for the Hive command line.

Related Links

For information related to Hive authorization modes, see:

- Storage Based Authorization in the Metastore Server
- HCatalog Authorization
- SQL Standard Based Hive Authorization
- Hive deprecated authorization mode / Legacy Mode
- Hive security design document
- Hive security document

SQL Standards-Based Hive Authorization

Using EEP 6.0.0 and later, you can configure SQL standards-based authorization to enable fine grained access control with SQL commands.

The SQL standards-based authorization mode can be used in conjunction with storage-based authorization on the Metastore server. Similar to the current default authorization in Hive, SQL

standards-based authorization is also enforced at query compilation time. To provide security through this option, the client will have to be secured. You can do this by allowing users access only through HiveServer2, and by restricting the user code and non-SQL commands that can be run. The checks will happen against the user who submits the request, but the query will run as the Hive server user. The directories and files for input data would have read access for this Hive server user. For users who do not have the need to protect against malicious users, this could potentially be supported through the Hive command line as well.

1. Add the following properties to `hive-site.xml`:

```
<!-- SQL standard based authorization -->
<property>
  <name>hive.server2.enable.doAs</name>
  <value>false</value>
</property>
<property>
  <name>hive.users.in.admin.role</name>
  <value>mapr</value>
</property>
<property>
  <name>hive.security.metastore.authorization.manager</name>

<value>org.apache.hadoop.hive.ql.security.authorization.MetaStoreAuthzAPI
AuthorizerEmbedOnly</value>
</property>
<property>
  <name>hive.security.authorization.manager</name>

<value>org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.SQL
StdConfOnlyAuthorizerFactory</value>
</property>
```

2. Create a `hiveserver2-site.xml` configuration file:

```
touch /opt/mapr/hive/hive-<version>/conf/hiveserver2-site.xml
```

   Add the following properties to the `hiveserver2-site.xml` file:

```
<configuration>
<property>
  <name>hive.security.authorization.manager</name>

<value>org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.SQL
StdHiveAuthorizerFactory</value>
</property>
<property>
  <name>hive.security.authorization.enabled</name>
  <value>true</value>
</property>
<property>
  <name>hive.security.authenticator.manager</name>

<value>org.apache.hadoop.hive.ql.security.SessionStateUserAuthenticator</
value>
</property>
<property>
  <name>hive.metastore.uris</name>
  <value></value>
</property>
</configuration>
```

3. Change owner of the `hiveserver2-site.xml` file to `mapr`, and restart Hive services:

```
chown mapr:mapr /opt/mapr/hive/hive-<version>/conf/hiveserver2-site.xml

maprcli node services -name hs2 -action restart -nodes `hostname -f`
maprcli node services -name hivemeta -action restart -nodes `hostname -f`
```

If you are a database administrator and want to run commands such as `create role` and `drop role` or to access objects without being given explicit access, you must run the `set role` command.

1. Create a test role:

```
hive> set role admin;
OK
Time taken: 0.02 seconds

hive> create role example_role;
OK
Time taken: 0.099 seconds

hive> show roles;
OK
admin
public
role1
example_role
Time taken: 0.02 seconds, Fetched: 3 row(s)
```

2. Grant access:

```
hive> GRANT example_role to USER testuser;
OK
Time taken: 0.058 seconds

hive> GRANT SELECT  on table eg_test to role example_role;
OK
Time taken: 0.146 seconds
```

3. Using the test role, check access:

```
sudo -u mapruser1 hive
```

If there is an access violation, correct it. The following is an example of an access violation error:

```
hive> insert into table eg_test values (4), (5), (6);
FAILED: RuntimeException Cannot create staging directory
'maprfs:///user/hive/warehouse/
eg_test/.hive-staging_hive_2018-06-08_10-24-11_566_5325052587659005252-1'
:
User mapruser1(user id 5001) has been denied access to
create .hive-staging_hive_2018-06-08_10-24-11_566_5325052587659005252-1
```

You can apply access restrictions to all actions except for READ access.

The following are examples of permitted access operations:

last-updated: Apr 11, 2022

- Select:

```
hive> select count (*) from eg_test;
OK
3
Time taken: 2.491 seconds, Fetched: 1 row(s)
```

- Describe:

```
hive> describe extended eg_test;
OK
id                      int

Detailed Table Information
Table(tableName:eg_test, dbName:default, owner:mapr,
createTime:1528453013, lastAccessTime:0, retention:0,
sd:StorageDescriptor(cols:[FieldSchema(name:id, type:int,
comment:null)],
location:maprfs:/user/hive/warehouse/eg_test,
inputFormat:org.apache.hadoop.mapred.TextInputFormat,
outputFormat:org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
,
compressed:false, numBuckets:-1, serdeInfo:SerDeInfo(name:null,
serializationLib:org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe,
parameters:{serialization.format=1}), bucketCols:[], sortCols:[],
parameters:{},
skewedInfo:SkewedInfo(skewedColNames:[], skewedColValues:[],
skewedColValueLocationMaps:{}),
storedAsSubDirectories:false), partitionKeys:[], parameters:
{totalSize=6, numRows=3, rawDataSize=3,
COLUMN_STATS_ACCURATE={"BASIC_STATS":"true"}, numFiles=1,
transient_lastDdlTime=1528453046}, viewOriginalText:null,
viewExpandedText:null, tableType:MANAGED_TABLE, rewriteEnabled:false)
Time taken: 0.12 seconds, Fetched: 3 row(s)
```

- Show columns:

```
hive> SHOW COLUMNS from eg_test;
OK
id
Time taken: 0.049 seconds, Fetched: 1 row(s)
```

Note: For more information about privileges required for Hive operations, see the open source documentation.

Configure Hive Metastore to use Storage-Based Authorization

You can enable storage-based authorization (SBA) for the Hive Metastore server.

Storage-based authorization controls access to the data using HDFS permissions (HDFS ACL). To control metadata access on the metadata objects, such as Databases, Tables, and Partitions, it checks if you have permission on the corresponding directories on the filesystem.

To enable storage-based authorization for the Hive Metastore server, set these properties in `hive-site.xml`:

Property	Value	Description
`hive.metastore.pre.event.listeners`	`org.apache.hadoop.hive.ql.security.authorization.AuthorizationPreEventListener*`	Turns on Metastore security. A MetaStorePreEventListener that performs authorization or authentication checks on the metastore side. Note that this can only perform authorizaiton checks on defined metastore PreEventContexts, such as the adding, dropping, and altering of databases, tables, and partitions.
`hive.security.metastore.authorization.manager`	`org.apache.hadoop.hive.ql.security.authorization.StorageBasedAuthorizationProvider*` **Note:** The `StorageBasedAuthorizationProvider` setting first appeared, on the Metastore side only, in Hive 0.10.0. With Hive 0.12.0 and later, it can also run on the client side. **Note:** Starting from EEP 6.1.0, `hive.security.metastore.authorization.manager` is set to the `StorageBasedAuthorizationProvider` value by default.	`StorageBasedAuthorizationProvider` - Specifies use of an HDFS permission-based model (recommended) for the Metastore-side authorization provider. `DefaultHiveMetastoreAuthorizationProvider` - This default implements the standard Hive grant/revoke model.
`hive.security.metastore.authenticator.manager`	`org.apache.hadoop.hive.ql.security.HadoopDefaultMetastoreAuthenticator` (default)	Authentication manager class name to be used in the metastore for authentication. The user-defined authenticator should implement the `org.apache.hadoop.hive.ql.security.HiveAuthenticationProvider` interface.
`hive.security.metastore.authorization.auth.reads`	true (default)	Default value (does not appear in `hive-site.xml` file). Set to `true`, Metastore authorization also performs a read authorization check (first supported in Hive 0.14.0).
`hive.server2.enable.doAs`	`true` (default)	Use for protected access through HiveServer2.

* In secure clusters, the MapR "Secure-by-Default" configuration implicitly configures these properties in the `hive-site.xml` file.

SBA configuration example in hive-site.xml File

```
<property>
    <name>hive.security.metastore.authorization.manager</name>

<value>org.apache.hadoop.hive.ql.security.authorization.StorageBasedAuthorizationProvider</value>
 </property>

<property>
    <name>hive.security.metastore.authenticator.manager</name>
```


last-updated: Apr 11, 2022

```
<value>org.apache.hadoop.hive.ql.security.HadoopDefaultMetastoreAuthenticato
r
    </value>
</property>

<property>
    <name>hive.security.metastore.authorization.auth.reads</name>
    <value>true</value>
</property>

<property>
    <name>hive.server2.enable.doAs</name>
    <value>true</value>
</property>

<property>
    <name>hive.metastore.pre.event.listeners</name>

<value>org.apache.hadoop.hive.ql.security.authorization.AuthorizationPreEven
tListener</value>
</property>
```

If you use storage-based authorization, you still need to use one of the following authorization models to protect actions within the HiveServer2:

- Configuring Fallback Hive Authorizer on page 3458
- Configure Hive to use Sentry Authorization on page 3830 (supported only for Impala users)
- SQL Standards-Based Hive Authorization on page 3453

Fallback Hive Authorizer
Fallback Hive Authorizer is used by Hive DDL (Data Definition Language) tasks for access control and for checking authorization from `Driver.doAuthorization()`.

It is designed to prevent CVE-2018-11777.

In addition to the centralized authorization provided by the MapR Data Platform, you can use several authorization modes for Hive. The use cases and trade-offs for these authorization modes are described in the sections below.

Configuring Fallback Hive Authorizer

You can enable protection of actions within the HiveServer2 instance by using the Fallback Authorizer.

Use the Fallback Authorizer when you want to protect access for Hive clients (JDBC/ODBC, Beeline CLI, and other HiveServer2 API users).

To enable Fallback Authorization for Hive clients, set these properties in the `hive-site.xml` file:

Property	Value	Description
`hive.security.authorization.enabled`	`true*`	Enable or disable the Hive client authorization.
`hive.security.authorization.manager`	`org.apache.hadoop.hive.ql.security.authorization.plugin.fallback.FallbackHiveAuthorizerFactory*`	Class name for the Hive client authorization manager.

Property	Value	Description
`hive.users.in.admin.role`	`mapr*`	Comma-separated list of users who need to be added to the `admin` role. Note that a user who belongs to the `admin` role needs to run the `set role` command before getting the privileges of the `admin` role, as the `admin` role is not in `current roles` by default.

* In secure clusters, the MapR "Secure-by-Default" configuration implicitly configures the Fallback Authorizer in the `hive-site.xml` file.

Fall Back Authorizer applies the following restrictions:

- Allows `set` only for selected whitelist parameters.
- Disallows dfs commands except for `admin`.
- Disallows local file location in SQL statements except for `admin`.
- Disallows `ADD JAR`, `COMPILE`, and `TRANSFORM` statements.

Fallback Authorization Configuration Example in hive-site.xml File

```
<property>
    <name>hive.security.authorization.enabled</name>
    <value>true</value>
</property>

<property>
    <name>hive.security.authorization.manager</name>

<value>org.apache.hadoop.hive.ql.security.authorization.plugin.fallback.Fall
backHiveAuthorizerFactory</value>
</property>

<property>
    <name>hive.users.in.admin.role</name>
    <value>mapr</value>
 </property>
```

Action Restrictions with Fallback Hive Authorizer

After enabling Fallback Hive Authorizer, you can perform action restriction operations.

- Disallow local file location in SQL statements for all except the administrator.
- Allow `set` for selected white list parameters.
- Disallow `dfs` commands for all except the administrator.
- Disallow `ADD JAR` statements for all except the administrator.
- Disallow `COMPILE` statements for all except the administrator.
- Disallow `TRANSFORM` statements.

Using a White List with Fallback Hive Authorizer

You can add an exception to Fallback Hive Authorizer restrictions using the `hive.security.authorization.sqlstd.confwhitelist.append` property.

last-updated: Apr 11, 2022

The `hive.security.authorization.sqlstd.confwhitelist` property is list of comma-separated Java regexes that you can append to. Appending to this list instead of updating the original list means that you can append to the default set by SQL-standard authorization instead of replacing it entirely.

You can modify the configurations parameters that match these regexes when SQL-standard authorization is enabled.

To get the default value, use the `set <param>` command. The `hive.conf.restricted.list` checks are still enforced after the white-list check.

An example of a white-list configuration is as follows:

```
<property>
  <name>hive.security.authorization.sqlstd.confwhitelist.append</name>
  <value>hive.reloadable.aux.jars.path</value>
</property>
```

- After adding this configuration to the `hive-site.xml` file, execute the following command:

  ```
  set hive.reloadable.aux.jars.path=/path/to/jar
  ```

Integrating Hive

Hive and MapR Database Integration

You can create MapR Database binary tables from Hive that can be accessed by both Hive and MapR Database. You can run Hive queries on MapR Database binary tables, convert existing MapR Database binary tables into Hive-MapR Database tables, and run Hive queries on those tables as well.

Install and Configure Hive

1. Install and configure Hive if it is not already installed. See Installing Hive on page 187 for details.
2. Execute the `jps` command and ensure that all relevant Hadoop, MapR, and Zookeeper processes are running. Example:

   ```
   $ jps
   1549 jenkins.war
   15051 QuorumPeerMain
   30935 Jps
   15551 CommandServer
   15293 ResourceManager
   15328 NodeManager
   15131 WardenMain
   ```

3. Open the `hive-site.xml` file with your favorite editor, or create a `hive-site.xml` file if it doesn't already exist:

   ```
   $ cd $HIVE_HOME
   $ vi conf/hive-site.xml
   ```

4. Copy the following XML code and paste it into the `hive-site.xml` file.

   Note: If you already have an existing `hive-site.xml` file with a `configuration` element block, just copy the `property` element block code below and paste it inside the `configuration` element block in the `hive-site.xml` file. Be sure to use the correct values for the paths to your auxiliary JARs and ZooKeeper IP numbers.

   Example configuration:

```
configuration>

<property>
  <name>hive.aux.jars.path</name>
  <value>file:///opt/mapr/hive/hive-<version>/lib/
hive-hbase-handler-<version>-mapr.jar,
file:///opt/mapr/hbase/hbase-<version>/lib/
hbase-client-<version>-mapr.jar, file:///opt/mapr/hbase/
hbase-<version>/lib/hbase-server-<version>-mapr.jar,file:///opt/mapr/
hbase/hbase-<version>/lib/hbase-protocol-<version>-mapr.jar,file:///opt/
mapr/zookeeper/zookeeper-<version>/zookeeper-<version>.jar</value>
  <description>A comma separated list (with no spaces)
of the jar files required for Hive-HBase integration</description>
</property>

<property>
  <name>hbase.zookeeper.quorum</name>
  <value>xx.xx.x.xxx,xx.xx.x.xxx,xx.xx.x.xxx</value>
  <description>A comma separated list (with no spaces) of
the IP addresses of all ZooKeeper servers in the cluster.</description>
</property>

<property>
  <name>hbase.zookeeper.property.clientPort</name>
  <value>5181</value>
  <description>The Zookeeper
client port. The MapR default clientPort is 5181.</description>
</property>

</configuration>
```

5. Save and close the `hive-site.xml` file.

If you have successfully completed all of the steps in this section, you're ready to begin the tutorial in the next section.

Getting Started with Hive and MapR Database Binary Integration

In this tutorial we will:

- Create a Hive table
- Populate the Hive table with data from a text file
- Query the Hive table
- Create a Hive-MapR Database table
- Introspect the Hive-MapR Database table from the HBase shell
- Populate the Hive-MapR Database table with data from the Hive table
- Query the Hive-MapR Database table from Hive

last-updated: Apr 11, 2022

- Convert an existing MapR Database table into a Hive-MapR table

Be sure that you have successfully completed all of the steps in Installing Hive on page 187 and review the MapR Database topics before beginning this Getting Started tutorial.

This Getting Started tutorial is based on the Hive-HBase Integration section of the Apache Hive Wiki. However, please note that there are some significant differences.

Create a Hive table with two columns

Change to your Hive installation directory if you're not already there and start Hive:

```
$ cd $HIVE_HOME
$ bin/hive
```

Execute the CREATE TABLE command to create the Hive pokes table

```
hive> CREATE TABLE pokes (foo INT, bar STRING);
```

To see if the pokes table has been created successfully, execute the `SHOW TABLES` command

```
hive> SHOW TABLES;
OK
pokes
Time taken: 0.74 seconds
```

The `pokes` table appears in the list of tables. **Populate the Hive pokes table with data:**

The `kv1.txt` file is provided in the `$HIVE_HOME/examples/files` directory. Execute the LOAD DATA LOCAL INPATH command to populate the Hive `pokes` table with data from the `kv1.txt` file.

```
hive> LOAD DATA LOCAL INPATH './examples/files/kv1.txt' OVERWRITE INTO
TABLE pokes;
```

A message appears confirming that the table was created successfully, and the Hive prompt reappears:

```
Copying data from file:
...
OK
Time taken: 0.278 seconds
hive>
```

Execute a SELECT query on the Hive pokes table

```
hive> SELECT * FROM pokes WHERE foo = 98;
```

The SELECT statement executes, runs a MapReduce application, and prints the application output:

```
OK
98      val_98
98      val_98
Time taken: 18.059 seconds
```

The output of the SELECT command displays two identical rows because there are two identical rows in the Hive `pokes` table with a key of 98.

Warning:

Hive tables can have multiple identical keys. As we will see shortly, MapR Database tables cannot have multiple identical keys, only unique keys.

Create a Hive-MapR Database table

Enter these four lines of code at the Hive prompt:

```
hive> CREATE TABLE mapr_table_1(key int, value string)
    > STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
    > WITH SERDEPROPERTIES ("hbase.columns.mapping" = ":key,cf1:val")
    > TBLPROPERTIES ("hbase.table.name" = "/user/mapr/xyz");
```

After a brief delay, a message appears confirming that the table was created successfully:

```
OK
Time taken: 5.195 seconds
```

Note: The TBLPROPERTIES command is not required, but those new to Hive-MapR Database integration may find it easier to understand what's going on if Hive and MapR Database use different names for the same table.

In this example, Hive will recognize this table as "mapr_table_1" and MapR Database will recognize this table as "xyz".

Start the HBase shell

Keeping the Hive terminal session open, start a new terminal session for HBase, then start the HBase shell:

```
$ cd $HBASE_HOME
$ bin/hbase shell
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 0.90.4, rUnknown, Wed Nov  9 17:35:00 PST 2011

hbase(main):001:0>
```

Execute the `list` command to see a list of HBase tables

```
hbase(main):001:0> list
TABLE
/user/mapr/xyz
1 row(s) in 0.8260 seconds
```

HBase recognizes the Hive-MapR Database table named `xyz` in directory `/user/mapr`. This is the same table known to Hive as `mapr_table_1`.

Display the description of the `/user/mapr/xyz` table in the HBase shell

```
hbase(main):004:0> describe "/user/mapr/xyz"
DESCRIPTION                                          ENABLED
 {NAME => '/user/mapr/xyz', FAMILIES => [{NAME => 'cf1', DATA_B true
 LOCK_ENCODING => 'NONE', BLOOMFILTER => 'NONE', REP
 LICATION_SCOPE => '0', VERSIONS => '3', MIN_VERSION
 S => '0', TTL => '2147483647', KEEP_DELETED_CELLS =
 > 'false', BLOCKSIZE => '65536', IN_MEMORY => 'fals
 e', ENCODE_ON_DISK => 'true', BLOCKCACHE => 'true'}
 ]}
1 row(s) in 0.0240 seconds
```

From the Hive prompt, insert data from the Hive table pokes into the Hive-MapR Database table `mapr_table_1`

```
hive> INSERT OVERWRITE TABLE mapr_table_1 SELECT * FROM pokes WHERE foo=98;
...
```


last-updated: Apr 11, 2022

```
2 Rows loaded to mapr_table_1
OK
Time taken: 13.384 seconds
```

Query `mapr_table_1` to see the data we have inserted into the Hive-MapR Database table

```
hive> SELECT * FROM mapr_table_1;
OK
98      val_98
Time taken: 0.56 seconds
```

Even though we loaded two rows from the Hive `pokes` table that had the same key of 98, only one row was actually inserted into `mapr_table_1`. This is because `mapr_table_1` is a MapR Database table, and although Hive tables support duplicate keys, MapR Database tables only support unique keys. MapR Database tables arbitrarily retain only one key, and silently discard all of the data associated with duplicate keys.

Convert a pre-existing MapR Database table to a Hive-MapR Database table

To convert a pre-existing MapR Database table to a Hive-MapR Database table, enter the following four commands at the Hive prompt.

Note that in this example the existing MapR Database table is `my_mapr_table` in directory `/user/mapr`.

```
hive> CREATE EXTERNAL TABLE mapr_table_2(key int, value string)
    > STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
    > WITH SERDEPROPERTIES ("hbase.columns.mapping" = "cf1:val")
    > TBLPROPERTIES("hbase.table.name" = "/user/mapr/my_mapr_table");
```

Now we can run a Hive query against the pre-existing MapR Database table `/user/mapr/my_mapr_table` that Hive sees as `mapr_table_2`:

```
hive> SELECT * FROM mapr_table_2 WHERE key > 400 AND key < 410;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
...
OK
401     val_401
402     val_402
403     val_403
404     val_404
406     val_406
407     val_407
409     val_409
Time taken: 9.452 seconds
```

Optimizing MapR Database Tables Search by ID

Starting from the 1904 release (EEP 6.0.2, EEP 6.1.1, and EEP 6.2.0), search by ID is supported with Hive MapR Database JSON tables.

Property of Optimization

The property name is `hive.mapr.db.json.fetch.by.id.task.conversion` and the value has a boolean type and by default is set to `true`, which means it is enabled.

- To disable optimization, set `hive.mapr.db.json.fetch.by.id.task.conversion` to `false`.

Conditions for Optimization

- This optimizer is designed for queries such as:

```
SELECT *
FROM <mapr_db_json_table>
WHERE _id = <constant_string_value>;
```

or:

```
SELECT *
FROM <mapr_db_json_table>
WHERE _id = <constant_string_value> AND (<condition_1>) AND
(<condition_2>) ... AND (<condition_N>);
```

or:

```
SELECT *
FROM <mapr_db_json_table>
WHERE <Constant false operator>
```

where `_id` is a key column of MapR Database JSON table. It provides usage of the `findById()` method of the MapR Database JSON table. The following functionality is not supported:

- joins
- group by
- distinct
- lateral view
- subquery
- create table as select (CTAS) or insert
- analyze
- single source

The predicate is not actually a part of the filter, so it is ignored by push down:

```
SELECT * FROM t WHERE (CASE WHEN _id = 'value_a' THEN 2 ELSE 4 END) > 3;
```

Using Optimization

1. Consider the following MapR Database JSON table:

```
CREATE TABLE t(doc_id string, col1 string, col2 string)
STORED BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'
TBLPROPERTIES("maprdb.table.name" = "/user/mapr/
db_json_table","maprdb.column.id" = "doc_id");
```

2. Run the `EXPLAIN` command:

```
EXPLAIN SELECT col1 FROM t WHERE doc_id='id_004';
```

3. The following output is produced:

```
STAGE DEPENDENCIES:
 Stage-0 is a root stage

STAGE PLANS:
 Stage: Stage-0
  MapR DB JSON Fetch By Id Operator
  limit: -1
  Processor Tree:
  TableScan
  alias: t_small
  filterExpr: (doc_id = 'id_004') (type: boolean)
  Statistics: Num rows: 1 Data size: 0 Basic stats: PARTIAL Column
stats: NONE
  Filter Operator
   predicate: (doc_id = 'id_004') (type: boolean)
   Statistics: Num rows: 1 Data size: 0 Basic stats: PARTIAL Column
stats: NONE
   Select Operator
   expressions: col1 (type: string)
   outputColumnNames: _col0
   Statistics: Num rows: 1 Data size: 0 Basic stats: PARTIAL Column
stats: NONE
   ListSink
```

An important part of a query plan is that it shows if optimization is available for the query:

```
STAGE PLANS:
Stage: Stage-0
MapR DB JSON Fetch By Id Operator
```

Connecting Using Hive MapR Database JSON Connector

This section describes the Hive connector for MapR Database JSON table.

The Hive connector supports the creation of MapR Database based Hive tables. You can create a JSON table on MapR Database and load CSV data and/or JSON files to MapR Database using the connector. MapR Database based Hive tables can be:

- Queried just like MapR File System based Hive tables.
- Combined with MapR File System based Hive tables in joins and sub-queries.

Note: If you use Drill to query Hive tables based on MapR Database tables, you can enable the native Drill reader, which can improve query performance.

The following table lists the Hive data type and the corresponding (supported) MapR Database OJAI type:

Hive Type	MapR Database OJAI Type
BOOLEAN	BOOLEAN
BINARY	BINARY
TINYINT	BYTE
DATE	DATE
DOUBLE	DOUBLE
FLOAT	FLOAT
INT	INT

last-updated: Apr 11, 2022

Hive Type	MapR Database OJAI Type
BIGINT	LONG
SMALLINT	SHORT
STRING	STRING
TIMESTAMP	TIMESTAMP

The Hive connector for MapR Database JSON table also supports the use of the following complex data types:

- map
- array
- struct

Note: The MapR Database JSON tables do not support ACID transactions, bucketing, and alteration.

Creating a MapR Database JSON Table and Hive Table Using Hive

- To create a table, run the command similar to the following:

  Note: The required properties are shown in bold.

```
CREATE TABLE primitive_types (
 id string,
 bo boolean,
 d double,
 da date,
 f double,
 i int,
 s string,
 ts timestamp)
STORED BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'
TBLPROPERTIES("maprdb.table.name" = "/tbl","maprdb.column.id" = "id");
```

  Here:

  - The `maprdb.table.name`, `maprdb.column.id` and `STORED BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'` are mandatory properties.
  - The value for `maprdb.column.id` column should be of type string or binary.

  To create a Hive table that exists on MapR Database, specify `EXTERNAL` in the table DDL. If the table created is EXTERNAL, when the table is dropped, only its metadata is deleted; the underlying MapR Database data remains intact. On the other hand, if the table is not EXTERNAL, dropping the table deletes both the metadata associated with the table and the underlying MapR Database data.

  For example, suppose a JSON table named `/apps/my_users` with the following values:

```
{"_id":"001","first_name":"John","last_name":"Doe","age":34}
{"_id":"002","first_name":"Jack","last_name":"Smith","age":26}
```

  To create a Hive table over existing MapR Database JSON table:

```
CREATE EXTERNAL TABLE primitive_types (
 user_id string,
```


last-updated: Apr 11, 2022

```
 first_name string,
 last_name string,
 age int)
STORED BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'
TBLPROPERTIES("maprdb.table.name" = "/apps/my_users","maprdb.column.id" =
"user_id");
```

Now, because table `primitive_types` points to MapR Database table, you can perform ETL query similar to MapR File System based Hive tables:

```
SELECT COUNT(*) FROM test_external;
SELECT MAX(age) AS label FROM test_external;
...
```

Loading CSV Data to MapR Database JSON Table

1. Create intermediate table.
   For example:

   ```
   CREATE TABLE stage(id STRING, name STRING, age INT) ROW FORMAT DELIMITED
   FIELDS TERMINATED BY ',';
   ```

2. Load data to table.
   For example:

   ```
   LOAD DATA INPATH '/data' into table stage;
   ```

3. Create MapR Database table in Hive.
   For example:

   ```
   CREATE TABLE users(id STRING, name STRING, age INT)
   STORED BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'
   TBLPROPERTIES("maprdb.table.name" = "/users","maprdb.column.id" = "id");
   ```

4. Insert data through stage table.
   For example:

   ```
   INSERT INTO TABLE users select id, name, age from stage;
   ```

Loading JSON Files to MapR Database JSON Table

1. Add SerDe JAR for JSON.
   For example:

   ```
   add jar /opt/mapr/hive/hive-<version>/hcatalog/share/hcatalog/
   hive-hcatalog-core-<version>-mapr.jar
   ```

2. Create intermediate table.
   For example:

   ```
   CREATE EXTERNAL TABLE stage(id string, name string, age int)
   ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe'
   STORED AS TEXTFILE;
   ```

3. Load data in stage table.
   For example:

```
LOAD DATA INPATH '/data' into table stage;
```

   Note: If there is a key in the JSON file that starts with "_" (for example, "_id"), then treat the names as literals upon creating the schema and query using the same literal syntax. For example, specify `` `_id` string `` without any special serde properties. Then in the query, use `` select `_id` from sometable; ``. Alternatively, you can use 'org.openx.data.jsonserde.JsonSerDe' and add WITH SERDEPROPERTIES ("mapping.id" = "_id") to your table definition.

4. Create MapR Database table in Hive.
   For example:

```
CREATE TABLE users(id STRING, name STRING, age INT)
STORED BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'
TBLPROPERTIES("maprdb.table.name" = "/users","maprdb.column.id" = "id");
```

5. Insert data through stage table.
   For example:

```
INSERT INTO TABLE users select id, name, age from stage;
```

   If there is a key in your JSON file that starts with "_" (for example, "`_id`"), treat the names as literals upon creating the schema and also query using the same literal syntax. In the above example, it would look like `` `_id` string `` without any special serde properties for it. Then, use again in query as shown below:

```
select `_id`  from sometable;
```

   Alternatively, use `org.openx.data.jsonserde.JsonSerDe` and add `WITH SERDEPROPERTIES ("mapping.id" = "_id" )` to your table definition.

Refer to Hive MapR Database JSON Connector Tutorial for a connector example.

Understanding the UPDATE Statement

Starting with EEP 6.0.0 (Hive 2.3), EEP 5.0.1 (Hive 2.1), EEP 4.1.2, and EEP 3.0.4, the UPDATE statement is supported with Hive MapR Database JSON tables.

You can use the `UPDATE` statement to update primitive, complex, and complex nested data types in MapR Database JSON tables, using the Hive connector.

Updating Primitive Data Types

last-updated: Apr 11, 2022

This section describes how to use the UPDATE statement to update primitive data types in MapR Database JSON tables, using the Hive connector.

1. Create a MapR Database JSON table and a Hive table:

```
CREATE TABLE simple_types_update (
>>>>>>> Incorporated edit comments
  doc_id string,
  bo boolean,
  d double,
  da date,
  f float,
  i int,
  s string,
  ts timestamp,
  ti tinyint,
  bi bigint,
  si smallint,
  bin binary)
STORED BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'
TBLPROPERTIES("maprdb.table.name" = "/
simple_types_update","maprdb.column.id" = "doc_id");
```

2. Insert data into the table:

```
INSERT INTO TABLE simple_types_update VALUES ('1', true, 124.14,
'2017-11-29', 9192.12,
214566190, 'text', '2017-03-17 00:14:13', 125, 9223372036854775806,
23434, "binary string");
```

3. Run the UPDATE command on the table:

```
UPDATE simple_types_update
SET da = '2018-12-11',
bo = FALSE,
f = 91.777
WHERE doc_id = '1';
```

4. Verify that the data is inserted in both Hive and MapR Database JSON tables.
   - Verifying Hive table data:

```
hive> SELECT * FROM simple_types_update;

1    false    124.14    2018-12-11    91.777    214566190    text
2017-03-17 00:14:13    125    9223372036854775806    23434    binary
string
```

   - Verifying MapR Database JSON table data:

```
find '/simple_types_update'

{"_id":"1","bi":{"$numberLong":9223372036854775806},"bin":
{"$binary":"YmluYXJ5IHN0cmluZwAAAAAAAA=="},
 "bo":false,"d":124.14,"da":{"$dateDay":"2018-12-11"},
 "f":{"$numberFloat":91.777},"i":{"$numberInt":214566190},
 "s":"text","si":{"$numberShort":23434},"ti":{"$numberByte":125},"ts":
{"$date":"2017-03-17T00:14:13.000Z"}}
```


last-updated: Apr 11, 2022

Updating Complex Data Types

This section describes how to use the UPDATE statement to update complex data types in MapR Database JSON tables, using the Hive connector.

1. Create a MapR Database JSON table and a Hive table:

```
CREATE TABLE complex_types_update (
  doc_id string,
  info MAP<STRING, INT>,
  pets ARRAY<STRING>,
  user_info STRUCT<name:STRING, surname:STRING, age:INT, gender:STRING>)
STORED BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'
TBLPROPERTIES("maprdb.table.name" = "/
complex_types_update","maprdb.column.id" = "doc_id");
```

2. Insert data into the table:

```
INSERT INTO TABLE complex_types_update SELECT '1', map('age', 28),
array('Cat', 'Cat', 'Cat'),
named_struct('name', 'Santa', 'surname', 'Claus','age', 1000,'gender',
'MALE');
```

3. Run the UPDATE command on the table:

```
UPDATE complex_types_update SET
info = map('year', 32),
pets = array('Dog', 'Cat', 'Pig'),
user_info = named_struct('name', 'Vasco', 'surname', 'da Gama','age',
558,'gender', 'MALE')
WHERE doc_id = '1';
```

4. Verify that the data is inserted in both Hive and MapR Database JSON tables.
   - Verifying Hive table data:

```
hive> SELECT * FROM complex_types_update;

1    {"year":32}    ["Dog","Cat","Pig"]
{"name":"Vasco","surname":"da Gama","age":558,"gender":"MALE"}
```

   - Verifying MapR Database JSON table data:

```
find '/complex_types_update'

{"_id":"1","info":{"year":{"$numberInt":32}},"pets":
["Dog","Cat","Pig"],"user_info":{"age":{"$numberInt":558},
 "gender":"MALE","name":"Vasco","surname":"da Gama"}}
```

Updating Complex Nested Data Types

This section describes how to use the UPDATE statement to update complex nested data types in MapR Database JSON tables, using the Hive connector.

1. Create a MapR Database JSON table and a Hive table using Hive:

```
CREATE TABLE complex_nested_data_type_update
(
          entry STRING,
          num INT,
          postal_addresses MAP <STRING,
          struct
<USER_ID:STRING,ADDRESS:STRING,ZIP:STRING,COUNTRY:STRING>>
)
stored BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'
tblproperties
(
           "maprdb.table.name" = "/complex_nested_data_type_update",
           "maprdb.column.id" = "entry"
);
```

2. Insert data into the table:

```
INSERT INTO TABLE complex_nested_data_type_update
SELECT '001', '1',
MAP ( 'Bill',
Named_struct ('user_id', '1', 'address', '3205 Woodlake ct', 'zip',
'45040', 'country', 'USA'));
```

3. Run the UPDATE command on the table by updating the COUNTRY value in `map(struct)`:

```
UPDATE complex_nested_data_type_update
SET postal_addresses = MAP ('Bill',
Named_struct ('user_id', '1', 'address', '3205 Woodlake ct', 'zip',
'45040', 'country', 'Hun'))
WHERE entry = '001';
```

4. Verify that the data is inserted in both Hive and MapR Database JSON tables.
   - Verifying Hive table data:

```
hive> SELECT * FROM complex_nested_data_type_update;

001    1    {"Bill":{"user_id":"1","address":"3205 Woodlake
ct","zip":"45040","country":"Hun"}}
```

   - Verifying MapR Database JSON table data:

```
find '/complex_nested_data_type_update'
{"_id":"001","num":{"$numberInt":1},"postal_addresses":{"Bill":
{"address":"3205 Woodlake
ct","country":"Hun","user_id":"1","zip":"45040"}}}
```

UPDATE Statement Limitations

This section describes the features that the UPDATE statement does not support.

The UPDATE statement has the following known limitations:

- The UPDATE statement is fully supported only for primitive data types (see Connecting to MapR Database).
- The UPDATE statement is partly supported for complex data types; you can replace only the whole value of a complex type with new a value.
- You cannot update the `maprdb.column.id` value.

Understanding the INSERT INTO Statement

This section describes how to use the `INSERT INTO` statement to insert or overwrite rows in nested MapR Database JSON tables, using the Hive connector.

- Single-row insert on page 3473
- Multiple-row insert on page 3475
- Overwriting data on page 3478

Note: The output shown in these examples is for illustration only; actual Hive CLI output varies, depending on your specific situation.

Single-row insert

You can use the `INSERT INTO` statement to insert a single table row into a nested MapR Database table using one of two methods.

For example, imagine that you have the following Hive MapR Database JSON table, `nested_data_insert`:

```
 CREATE TABLE nested_data_insert
 (
    entry STRING,
    num INT,
    postal_addresses MAP <STRING,
    struct <USER_ID:STRING,ADDRESS:STRING,ZIP:STRING,COUNTRY:STRING>>
 )
 stored BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'
tblproperties
 (
    "maprdb.table.name" = "/nested_data_insert",
    "maprdb.column.id" = "entry"
 );
```

- You can insert the new row into your table by using a dummy table:

```
WITH dummy_table AS
  (SELECT '001' AS KEY,
          '1' AS num,
          MAP ('Adam',
               Named_struct ('user_id', '1', 'address', '3205 Woodlake
ct', 'zip', '45040', 'country', 'Usa'),
               'Wilfred',
               Named_struct ('user_id', '2', 'address', '777 Brockton
Avenue', 'zip', '34000', 'country', 'Ita')) AS postal_addresses)
INSERT INTO nested_data_insert
SELECT *
FROM dummy_table;
```


last-updated: Apr 11, 2022

- Alternatively, you can insert the new row into your table by using a `SELECT` statement:

```
INSERT INTO TABLE nested_data_insert
SELECT '002',
       '2',
       MAP ('Bill',
            Named_struct ('user_id', '1', 'address', '328 Virginia Ave',
'zip', '54956', 'country', 'Bol'),
            'Stiv',
            Named_struct ('user_id', '2', 'address', 'Schererville',
'zip', '46375', 'country', 'Efi'));
```

After you insert data, you should verify that the data is inserted in both Hive and MapR Database JSON tables:

- Verify the insertion into the Hive table by using the `SELECT * FROM` syntax.

```
SELECT * FROM nested_data_insert;
```

Sample output:

Table

		postal_address				
entry	**num**		**USER_ID**	**ADDRESS**	**ZIP**	**COUNTRY**
001	1	Adam	1	3205 Woodlake ct	45040	Usa
		Wilfred	2	777 Brockton Avenue	34000	Ita
002	2	Bill	1	328 Virginia Ave	54956	Bol
		Stiv	2	Schererville	46375	Efi

- Verify the insertion into the MapR Database JSON table data using the `find` statement:

```
find '/nested_data_insert'

{
  "Adam": {
      "user_id": "1",
      "address": "3205 Woodlake ct",
      "zip": "45040",
      "country": "Usa"
          },
  "Wilfred": {
      "user_id": "2",
      "address": "777 Brockton Avenue",
      "zip": "34000",
      "country": "Ita"
          }
       }
{
   "Bill": {
      "user_id": "1",
      "address": "328 Virginia Ave",
      "zip": "54956",
      "country": "Bol"
       },
   "Stiv": {
      "user_id": "2",
      "address": "Schererville",
      "zip": "46375",
      "country": "Efi"
       }
}
```

Multiple-row insert

Now imagine that you want to insert three rows of data into `nested_data_insert`.

- You can insert the new rows into your table by using a dummy table:

```
WITH dummy_table AS
  (SELECT '003' AS KEY,
          '3' AS num,
          MAP ('Rony',
               Named_struct ('user_id', '1', 'address', '4333 Backer
str', 'zip', '12311', 'country', 'Hun')) AS postal_addresses
   UNION ALL SELECT '004' AS KEY,
                    '4' AS num,
                    MAP ('Ivan',
                         Named_struct ('user_id', '1', 'address', '833
Bridle Avenue', 'zip', '95111', 'country', 'CA')) AS postal_addresses
   UNION ALL SELECT '005' AS KEY,
                    '5' AS num,
                    MAP ('Ivan',
                         Named_struct ('user_id', '1', 'address', '664
Devon Ave', 'zip', '92021', 'country', 'Tog')) AS postal_addresses)
INSERT INTO nested_data_insert
SELECT *
FROM dummy_table;
```


last-updated: Apr 11, 2022

- Alternatively, you can insert the new rows into your table by using a `SELECT` statement:

```
INSERT INTO TABLE nested_data_insert
SELECT '006',
       '6',
       MAP ('Rony',
            Named_struct ('user_id', '1', 'address', '150 National City',
'zip', '91950', 'country', 'Hun'))
UNION ALL
SELECT '007',
       '7',
       MAP ('Tomason',
            Named_struct ('user_id', '1', 'address', '272 Ocean Circle' ,
'zip', '92801', 'country', 'CA'))
UNION ALL
SELECT '008',
       '8',
       MAP ('Davin',
            Named_struct ('user_id', '1', 'address', '81 Augusta Ave',
'zip', '93905', 'country', 'CA'));
```

After you insert data, you should verify that the data is inserted in both Hive and MapR Database JSON tables:

- Verify the insertion into the Hive table by using the `SELECT * FROM` syntax.

```
SELECT * FROM nested_data_insert WHERE entry > '002' ;
```

Sample output:

Table

entry	num	postal_address				
			USER_ID	ADDRESS	ZIP	COUNTRY
003	3	Rony	1	4333 Backer str	12311	Hun
004	4	Ivan	1	833 Bridle Avenue	95111	CA
005	5	Ivan	1	664 Devon Ave.	92021	Tog
006	6	Rony	1	150 National City	91950	Hun
007	7	Tomason	1	272 Ocean Circle	92801	CA
008	8	Davin	1	81 Augusta Ave	93905	CA

- Verify the insertion into the MapR Database JSON table data using the `find` statement:

```
find '/nested_data_insert'

{
   "_id": "003",
   "num": {
     "$numberInt": 3
          },
   "postal_addresses": {
        "Rony": {
             "address": "4333 Backer str",
             "country": "Hun",
             "user_id": "1",
             "zip": "12311"
             }
         }
      }
{
   "_id": "004",
   "num": {
       "$numberInt": 4
          },
   "postal_addresses": {
          "Ivan": {
              "address": "833 Bridle Avenue",
              "country": "CA",
              "user_id": "1",
              "zip": "95111"
            }
       }
 }
 {
   "_id": "005",
   "num": {
           "$numberInt": 5
          },
   "postal_addresses": {
           "Ivan": {
               "address": "664 Devon Ave",
               "country": "Tog",
               "user_id": "1",
                "zip": "92021"
          }
       }
    }
{
     "_id": "006",
     "num": {
           "$numberInt": 6
            },
     "postal_addresses": {
          "Rony": {
             "address": "150 National City",
             "country": "Hun",
             "user_id": "1",
             "zip": "91950"
           }
        }
    }
{
    "_id": "007",
    "num": {
```

```
          "$numberInt": 7
            },
    "postal_addresses": {
          "Tomason": {
              "address": "272 Ocean Circle",
              "country": "CA",
              "user_id": "1",
              "zip": "92801"
           }
       }
    }
{
    "_id": "008",
    "num": {
          "$numberInt": 8
            },
    "postal_addresses": {
           "Davin": {
                "address": "81 Augusta Ave",
                "country": "CA",
                "user_id": "1",
                "zip": "93905"
            }
          }
}
```

Overwriting data

Still using sample table `nested_data_insert`, you can use the INSERT statement on a dummy table to overwrite one or more complete rows.

For example, to overwrite the first row in `nested_data_insert (001)` with new values, use the following syntax:

```
WITH dummy_table AS
(SELECT '001' AS KEY,
'1' AS num,
MAP ('newAdam',
Named_struct ('user_id', '1', 'address', 'newAdress', 'zip', 'newZip',
'country', 'newCountry')) AS postal_addresses)
INSERT INTO nested_data_insert
SELECT *
FROM dummy_table;
```

After you overwrite data, you should verify that the data is changed in both Hive and MapR Database JSON tables:

- Verify the data into the Hive table by using the `SELECT * FROM` syntax.

  ```
  hive> SELECT * FROM nested_data_insert WHERE entry = '001';
  ```

  Sample output:

Table

entry	num	postal_address				
			USER_ID	ADDRESS	ZIP	COUNTRY
001	1	newAdam	1	newAddress	newZip	newCountry

- Verify the data in the MapR Database JSON table data using the `findbyid` statement:

```
findbyid '/nested_data_insert' --id 001

{
   "_id": "001",
   "num": {
   "$numberInt": 1
           },
   "postal_addresses": {
        "newAdam": {
              "address": "newAdress",
              "country": "newCountry",
              "user_id": "1",
              "zip": "newZip"
           }
       }
}
```

For another example, imagine that you want to overwrite `003` and `004` rows in `nested_data_insert` with new values:

```
WITH dummy_table AS (
SELECT '003' AS KEY,
'3' AS num,
MAP ('newName1',
Named_struct ('user_id', '1', 'address', 'newAdress1', 'zip', 'newZip1',
'country', 'newCountry1')) AS postal_addresses
UNION ALL
SELECT '004' AS KEY,
'4' AS num,
MAP ('newName2',
Named_struct ('user_id', '1', 'address', 'newAdress2', 'zip', 'newZip2',
'country', 'newCountry2')) AS postal_addresses)
INSERT INTO nested_data_insert
SELECT * FROM dummy_table;
```

After you overwrite the data, you should verify that the data is changed in both Hive and MapR Database JSON tables.

- Verify the data in the Hive table by using the `SELECT * FROM` syntax.

```
hive> SELECT * FROM nested_data_insert WHERE entry IN ('003', '004');
```

Sample output:

Table

		postal_address				
entry	**num**		**USER_ID**	**ADDRESS**	**ZIP**	**COUNTRY**
003	3	newName1	1	newAddress1	newZip1	newCountry1
004	4	newName2	1	newAddress2	newZip2	newCountry2

Verify the data in the MapR Database JSON table data using the `findbyid` statement:

```
findbyid '/nested_data_insert' --id 003
{
  "_id": "003",
  "num": {
```

```
       "$numberInt": 3
          },
  "postal_addresses": {
  "newName1": {
       "address": "newAdress1",
       "country": "newCountry1",
       "user_id": "1",
       "zip": "newZip1"
          }
        }
    }
findbyid '/nested_data_insert' --id 004
     {
       "_id": "004",
       "num": {
          "$numberInt": 4
          },
       "postal_addresses": {
           "newName2": {
              "address": "newAdress2",
              "country": "newCountry2",
              "user_id": "1",
              "zip": "newZip2"
           }
        }
    }
```

Warning: If you exclude columns both from the SELECT statement in your INSERT statement and from the table schema, the value of this column changes to NULL.

Finally, imagine that you want to overwrite the first row in `nested_data_insert (001)` with new values and overwrite the `num` column to `NULL`:

```
WITH dummy_table AS
(SELECT '001' AS KEY,
MAP ('newAdam',
Named_struct ('user_id', '1', 'address', 'newAdress', 'zip', 'newZip',
'country', 'newCountry')) AS postal_addresses)
INSERT INTO nested_data_insert (entry, postal_addresses)
SELECT * FROM dummy_table;
```

After you overwrite data, you should verify that the data is changed in both Hive and MapR Database JSON tables.

- Verify the data in the Hive table by using the `SELECT * FROM` syntax.

  ```
  hive> SELECT * FROM nested_data_insert WHERE entry = '001';
  ```

  Sample output:

Table

entry	num	postal_address				
			USER_ID	ADDRESS	ZIP	COUNTRY
001	NULL	newAdam	1	newAddress	newZip	newCountry

last-updated: Apr 11, 2022

- Verify the data in the MapR Database JSON table (num row is not present):

```
findbyid '/nested_data_insert' --id 001

{
   "_id": "001",
   "postal_addresses": {
   "newAdam": {
         "address": "newAdress",
         "country": "newCountry",
         "user_id": "1",
         "zip": "newZip"
        }
     }
}
```

Understanding the MERGE Statement

You can use the MERGE statement to perform record-level INSERT and UPDATE operations efficiently within Hive tables.

The MERGE statement can be a key tool of MapR-cluster data management. It is based on ANSI-standard SQL.

The following scenarios can help you understand how to use the MERGE statement:

Simple merge.maprdb.column.id is the join key

Consider merging the following example source and target tables:

Table

id	first_name	last_name	age
001	Dorothi	Hogward	7777
002	Alex	Bowee	7777
088	Robert	Dowson	25

Table

id	first_name	last_name	age
001	John	Smith	45

Table (Continued)

id	first_name	last_name	age
002	Michael	Watson	27
003	Den	Brown	33

You can use the following SQL-standard MERGE statement:

```
MERGE into customer_db_json_target trg
USING customer_source src
ON src.id = trg.id
WHEN MATCHED THEN UPDATE SET age = src.age
WHEN NOT MATCHED THEN
INSERT VALUES (src.id, src.first_name, src.last_name, src.age);
```

The result is:

id	first_name	last_name	age
001	John	Smith	7777
002	Michael	Watson	7777
003	Den	Brown	33
088	Robert	Dowson	25

Note: The age column is updated and a new `id` column is inserted.

Simple `merge.maprdb.column.id` is not the join key

Merging when `merge.maprdb.column` is not the join key is not recommended.

Deleting while merging (EEP 6.3.0 and previous)

Deletions are not supported in a `MERGE` statement. This example:

```
MERGE INTO  customer_db_json old
USING customer_new new ON new.id = old.id
WHEN MATCHED AND old.age > 10 THEN DELETE;
```

Raises the following exception:

```
Error: Error while compiling statement: FAILED: SemanticException Deletes
are not supported for MapR DB JSON tables (state=42000,code=40000)
```

DELETE syntax in the MERGE statement (EEP 6.3.1 and later)

This section describes how to use the `DELETE` syntax in the `MERGE` statement for MapR Database JSON tables. Included are examples of usage and limitations.

Consider two tables: `tgt` which is the target table of the `MERGE` statement, and `src`, which is the source table from which data will be taken. Both tables use `MapRDBJsonStorageHandler` to store data. The following table shows the initial contents of the `tgt` table:

Table

id	Value
1	AAA

Table (Continued)

2	BBB
3	CCC
4	DDD
5	EEE

The following table shows the initial contents of the `src` table:

Table

id	Value
1	AAA
222	BBB---
3	CCC
444	DDD---
5	EEE

The following merge statement contains a `WHEN MATCHED THEN DELETE` clause. It means that if the `id` from the `tgt` table equals the `id` from the `src` table, the row is removed from the `tgt` table. When the value of `id` does not match, a new row is inserted into the `tgt` table:

```
MERGE INTO tgt
USING src ON tgt.id=src.id
WHEN MATCHED THEN DELETE
WHEN NOT MATCHED THEN INSERT VALUES (src.id, src.value);
```

The following table shows the result of the merge:

Table

id	Value
2	BBB
222	BBB---
4	DDD
444	DDD---

Here we removed rows with `id` 1, 3, and 5 from the `tgt` table because they existed in the `src` table, and they matched values from the `tgt` table. We did not touch rows with `id` 2 and 4, because there were no such values in the `src` table. We inserted new rows with `id` values 222 and 444 because they existed in the `src` table and did not exist in the `tgt` table.

Limitations

The preceding solution has three limitations:

1. Subqueries are not supported as a source when `DELETE` is used.
2. The source table should be a MapR Database JSON table when deletion is used in a `MERGE` operator.
3. The `DELETE` operator is not supported with additional conditions after `WHEN MATCHED`. Use either a single `UPDATE` or `DELETE`.

last-updated: Apr 11, 2022

Limitation #3 means that queries like the following are not supported:

```
MERGE INTO tgt
USING src
ON tgt._id = src._id
WHEN MATCHED AND [boolean expression1] THEN DELETE
WHEN MATCHED AND [boolean expression2] THEN UPDATE
WHEN NOT MATCHED THEN INSERT
```

Multiple source rows match a given target row (cardinality violation)

Consider merging the two tables `customer_db_json` and `customer_new`:

Table

id	first_name	last_name	age
001	John	Smith	45
002	Michael	Watson	27
003	Den	Brown	33

And:

Table

id	first_name	last_name	age
001	Dorothi	Hogward	77
001	Dorothi	Hogward	77
088	Robert	Dowson	25

To `MERGE customer_new` and `customer_db_json`:

```
MERGE INTO customer_db_json trg
USING customer_new src ON src.id = trg.id
WHEN MATCHED THEN UPDATE
SET first_name = src.first_name,
last_name = src.last_name
WHEN NOT MATCHED THEN INSERT VALUES
(src.id, src.first_name, src.last_name, src.age);
```

This example causes an exception because of duplicate values in the `id` column in the `customer_new` table:

```
Caused by: org.apache.hadoop.hive.ql.metadata.HiveException: Error
evaluating cardinality_violation(_col0)
```

To avoid cardinality violation, set `hive.merge.cardinality.check=false`, but in this case the result is unpredictable because there is no rule that defines the order of duplicated data that will be inserted by using the `MERGE` statement.

Merge on mixed data types

The merge operation also supports mixed data types, such as arrays, maps, and structures.

Consider two tables `mixed_types_source` and `mixed_types_target`:

last-updated: Apr 11, 2022

Table

doc_id	user_info
1	{"name":"Brandon","surname":"Lee","age":31,"gender":"MALE"}
2	{"name":"Johnson","surname":"Fall","age":23,"gender":"MALE"}
3	{"name":"Mary","surname":"Dowson","age":11,"gender":"FEMALE"}
4	{"name":"Paul","surname":"Rodgers","age":41,"gender":"MALE"}

And:

Table

id	user_info
1	{"name":"Lexx","surname":"Comfuzer","age":31,"gender":"MALE"}

To merge `mixed_types_source` and `mixed_types_target`:

```
MERGE INTO mixed_types_target trg
USING mixed_types_source src
ON src.doc_id = old.doc_id
WHEN MATCHED THEN UPDATE
SET user_info = src.user_info
WHEN NOT MATCHED THEN INSERT VALUES
(src.doc_id, src.user_info);
```

The result is:

Table

id	first_name
1	`{"name":"Brandon","surname":"Lee","age":31,"gender":"MALE"}`
2	`{"name":"Johnson","surname":"Fall","age":23,"gender":"MALE"}`
3	`{"name":"Mary","surname":"Dowson","age":11,"gender":"FEMALE"}`
4	`{"name":"Paul","surname":"Rodgers","age":41,"gender":"MALE"}`

Note that you cannot update only a part of a complex structure field. For example, suppose you have a structure stored as one field in a Hive table:

```
{"name":"Johnson","surname":"Fall","age":23,"gender":"MALE"}
```

You cannot update only the `age` field in the structure. You can only replace all values of the structure with new ones. For details, see Understanding the UPDATE Statement on page 3469.

Merge into external MapR Database JSON tables

The `MERGE` operator is also available for external MapR Database JSON tables. You can use the MERGE statement to insert and update values in external MapR database JSON table targets.

Merge into partitioned MapR Database JSON tables

Partitioned MapR Database JSON tables are not supported.

Merge into temporary MapR Database JSON tables

The `MERGE` operator is also available for temporary MapR Database JSON tables. Use temporary tables as target tables for merge. No additional syntax is needed.

Understanding the DELETE FROM Operation

In EEP 6.3.1 and later, you can use the `DELETE FROM` operation with MapR Database JSON tables.

Delete All Data from a Table

To delete all data from a MapR DatabaseJSON table use the following operator:

```
DELETE FROM <table_name>;
```

Example. In this example we create a table, insert data, and delete all rows:

```
DROP TABLE IF EXISTS customer;
CREATE TABLE customer(doc_id STRING, first_name STRING, last_name STRING)
STORED BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'
TBLPROPERTIES("maprdb.table.name" = "/customer","maprdb.column.id" =
"doc_id");
INSERT INTO TABLE customer VALUES ("001", "Max", "Born"), ("002", "Demmy",
"John"), ("003", "Robby", "Smart");
SELECT doc_id, first_name, last_name FROM customer;
```

Table

doc_id	first_name	last_name
001	Max	Born
002	Demmy	John
003	Robby	Smart

The following query gives an empty set. Deletions are supported only for MapR Database JSON tables and transactional tables.

```
DELETE FROM customer;
SELECT doc_id, first_name, last_name FROM customer;
```

Example. In this example, we try to delete data from a non-transactional and non-MapR Database JSON table:

```
DROP TABLE IF EXISTS simple_data;
CREATE TABLE simple_data (id INT);
INSERT INTO TABLE simple_data VALUES (1), (2), (3);
DELETE FROM simple_data;
```

The result is:

```
FAILED: SemanticException Operation is not supported. Table is nor ACID
neither MapRDbJSON
```

Delete a Single Row from a Table

To delete a single row from a MapR Database JSON table, use the following syntax:

```
DELETE FROM <table_name> WHERE <id> = <value>;
```

Where:

`<table_name>` is the MapR Database JSON table.

last-updated: Apr 11, 2022

<id> is a key column of the MapR Database JSON table. It corresponds to the maprdb.column.id property.

<value> is the value to be deleted.

Example. In this example, we create a table, insert data, and delete a single row:

```
DROP TABLE IF EXISTS customer;
CREATE TABLE customer(doc_id STRING, first_name STRING, last_name STRING)
STORED BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'
TBLPROPERTIES("maprdb.table.name" = "/customer","maprdb.column.id" =
"doc_id");
INSERT INTO customer VALUES ("001", "Max", "Born"), ("002", "Demmy",
"John"), ("003", "Robby", "Smart");
SELECT doc_id, first_name, last_name FROM customer;
```

Table

doc_id	first_name	last_name
001	Max	Born
002	Demmy	John
003	Robby	Smart

The following query deletes a single row using the WHERE clause:

```
DELETE FROM customer WHERE doc_id = "002";
SELECT doc_id, first_name, last_name FROM customer;
```

Table

doc_id	first_name	last_name
001	Max	Born
003	Robby	Smart

Note. Deletions are supported only for key columns of MapR Database JSON tables.

Example. In this example, we try to use a column other than a key column of the MapR Database JSON table in deletion.

```
DROP TABLE IF EXISTS customer;
CREATE TABLE customer(doc_id STRING, first_name STRING, last_name STRING)
STORED BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'
TBLPROPERTIES("maprdb.table.name" = "/customer","maprdb.column.id" =
"doc_id");
INSERT INTO customer VALUES ("001", "Max", "Born"), ("002", "Demmy",
"John"), ("003", "Robby", "Smart");
SELECT doc_id, first_name, last_name FROM customer;
```

Column first_name is not the key column of the table.

```
DELETE FROM customer WHERE first_name = "Max";
```

The result is:

```
FAILED: SemanticException Deletion over column first_name is forbidden. Use
only key column of MapR Db Json table: doc_id
```

Delete Several Rows from a Table

To delete several rows from a table, use the following syntax:

```
DELETE FROM <table_name> WHERE <id> IN (<value1>, <value2>, ...);
```

Where:

`<table_name>` is the MapR Database JSON table.

`<id>` is a key column of the MapR Database JSON table. It corresponds to the `maprdb.column.id` property.

`<value1>, <value2>,` are values to be deleted.

Example. In this example, we create a table, insert data, and delete several rows:

```
DROP TABLE IF EXISTS customer;
CREATE TABLE customer(doc_id STRING, first_name STRING, last_name STRING)
STORED BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'
TBLPROPERTIES("maprdb.table.name" = "/customer","maprdb.column.id" =
"doc_id");
INSERT INTO TABLE customer VALUES ("001", "Max", "Born"), ("002", "Demmy",
"John"), ("003", "Robby", "Smart");
SELECT doc_id, first_name, last_name FROM customer;
```

Table

doc_id	first_name	last_name
001	Max	Born
002	Demmy	John
003	Robby	Smart

The following query deletes several rows using the `WHERE ... IN` clause:

```
DELETE FROM customer WHERE doc_id IN ("001", "002");
SELECT doc_id, first_name, last_name FROM customer;
```

Table

doc_id	first_name	last_name
003	Robby	Smart

Delete All Rows in a Table Except Listed Rows

To delete all rows from a table except a listed row, use the following syntax:

```
DELETE FROM <table_name> WHERE <id> NOT IN (<value1>, <value2>, ...);
```

Where:

`<table_name>` is the MapR Database JSON table.

`<id>` is a key column of the MapR Database JSON table. It corresponds to the `maprdb.column.id` property.

`<value1>, <value2>,` are values to be preserved.

Example. In this example, we create a table, insert data, and delete all rows except the listed rows:

```
DROP TABLE IF EXISTS customer;
CREATE TABLE customer(doc_id STRING, first_name STRING, last_name STRING)
```

```
STORED BY 'org.apache.hadoop.hive.maprdb.json.MapRDBJsonStorageHandler'
TBLPROPERTIES("maprdb.table.name" = "/customer","maprdb.column.id" =
"doc_id");
INSERT INTO TABLE customer VALUES ("001", "Max", "Born"), ("002", "Demmy",
"John"), ("003", "Robby", "Smart");
SELECT doc_id, first_name, last_name FROM customer;
```

Table

doc_id	first_name	last_name
001	Max	Born
002	Demmy	John
003	Robby	Smart

The following query deletes all rows except the listed rows:

```
DELETE FROM customer WHERE doc_id NOT IN ("003");
SELECT doc_id, first_name, last_name FROM customer;
```

Table

doc_id	first_name	last_name
003	Robby	Smart

Limitations of the DELETE FROM Operation

The following are three limitations of the current implementation:

- The current implementation does not support arbitrary conditions in the `WHERE` clause of the `DELETE` statement even if a key column is used.

  Example. In this example, `DELETE FROM` is used with an arbitrary condition:

  ```
  DELETE FROM customer WHERE doc_id == "003" OR doc_id <> "005";
  ```

  The result is:

  ```
  FAILED:
          SemanticException This condition is not supported for MapR Db
  Json deletions. Supported
          WHERE clauses are: <id> = value, <id> IN (value1, value2, ...),
  <id> NOT IN (value1, value2,
          ...)
  ```

- The current implementation does not support subqueries in the `WHERE` clause.
- The current implementation does not support deletions in the `MERGE` statement.

Hive and HBase Integration

You can create HBase tables from Hive that can be accessed by both Hive and HBase. This allows you to run Hive queries on HBase tables. You can also convert existing HBase tables into Hive-HBase tables and run Hive queries on those tables as well.

Install and Configure Hive and HBase

1. Install and configure Hive if it is not already installed. See Installing Hive on page 187.

last-updated: Apr 11, 2022

2. Install and configure HBase if it is not already installed. See Installing HBase on page 183.
3. Run the `jps` command, and ensure that all relevant Hadoop, HBase and Zookeeper processes are running:

```
$ jps
21985 HRegionServer
1549 jenkins.war
15051 QuorumPeerMain
30935 Jps
15551 CommandServer
15698 HMaster
15293 ResourceManager
15328 NodeManager
15131 WardenMain
```

Getting Started with Hive-HBase Integration

In this tutorial you will:

- Create a Hive table
- Populate the Hive table with data from a text file
- Query the Hive table
- Create a Hive-HBase table
- Introspect the Hive-HBase table from HBase
- Populate the Hive-Hbase table with data from the Hive table
- Query the Hive-HBase table from Hive
- Convert an existing HBase table into a Hive-HBase table

Be sure that you have successfully completed all the steps in the Install and Configure Hive and HBase section before beginning this Getting Started tutorial. This Getting Started tutorial closely parallels the Hive-HBase Integration section of the Apache Hive Wiki, and thanks to Samuel Guo and other contributors to that effort.

Create a Hive table with two columns:

Change to your Hive installation directory if you're not already there and start Hive:

```
$ cd $HIVE_HOME
$ bin/hive
```

Execute the CREATE TABLE command to create the Hive pokes table:

```
hive> CREATE TABLE pokes (foo INT, bar STRING);
```

To see if the pokes table has been created successfully, execute the SHOW TABLES command:

```
hive> SHOW TABLES;
OK
pokes
Time taken: 0.74 seconds
```

The `pokes` table appears in the list of tables.

Populate the Hive pokes table with data

Execute the LOAD DATA LOCAL INPATH command to populate the Hive `pokes` table with data from the `kv1.txt` file.

The `kv1.txt` file is provided in the `$HIVE_HOME/examples` directory.

```
hive> LOAD DATA LOCAL INPATH './examples/files/kv1.txt' OVERWRITE INTO
TABLE pokes;
```

A message appears confirming that the table was created successfully, and the Hive prompt reappears:

```
Copying data from file:
...
OK
Time taken: 0.278 seconds
hive>
```

Execute a SELECT query on the Hive pokes table:

```
hive> SELECT * FROM pokes WHERE foo = 98;
```

The SELECT statement executes, runs a MapReduce application, and prints the job output:

```
OK
98      val_98
98      val_98
Time taken: 18.059 seconds
```

The output of the SELECT command displays two identical rows because there are two identical rows in the Hive `pokes` table with a key of 98. Note: This is a good illustration of the concept that Hive tables can have multiple identical keys. As we will see shortly, HBase tables cannot have multiple identical keys, only unique keys.

To create a Hive-HBase table, enter these four lines of code at the Hive prompt:

```
hive> CREATE TABLE hbase_table_1(key int, value string)
    > STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
    > WITH SERDEPROPERTIES ("hbase.columns.mapping" = ":key,cf1:val")
    > TBLPROPERTIES ("hbase.table.name" = "xyz");
```

After a brief delay, a message appears confirming that the table was created successfully:

```
OK
Time taken: 5.195 seconds
```

Note: The TBLPROPERTIES command is not required, but those new to Hive-HBase integration may find it easier to understand what's going on if Hive and HBase use different names for the same table.

In this example, Hive will recognize this table as "hbase_table_1" and HBase will recognize this table as "xyz".

Start the HBase shell:

Keeping the Hive terminal session open, start a new terminal session for HBase, then start the HBase shell:

```
$ cd $HBASE_HOME
$ bin/hbase shell
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
```

```
Version 0.90.4, rUnknown, Wed Nov  9 17:35:00 PST 2011

hbase(main):001:0>
```

Execute the list command to see a list of HBase tables:

```
hbase(main):001:0> list
TABLE
xyz
1 row(s) in 0.8260 seconds
```

HBase recognizes the Hive-HBase table named `xyz`. This is the same table known to Hive as `hbase_table_1`.

Display the description of the xyz table in the HBase shell:

```
hbase(main):004:0> describe "xyz"
DESCRIPTION
      ENABLED
 {NAME => 'xyz', FAMILIES => [{NAME => 'cf1', BLOOMFILTER => 'NONE',
REPLICATI true
 ON_SCOPE => '0', COMPRESSION => 'NONE', VERSIONS => '3', TTL =>
'2147483647', BL
 OCKSIZE => '65536', IN_MEMORY => 'false', BLOCKCACHE => 'true'}]}
1 row(s) in 0.0190 seconds
```

From the Hive prompt, insert data from the Hive table pokes into the Hive-HBase table hbase_table_1

```
hive> INSERT OVERWRITE TABLE hbase_table_1 SELECT * FROM pokes WHERE foo=98;
...
2 Rows loaded to hbase_table_1
OK
Time taken: 13.384 seconds
```

Query hbase_table_1 to see the data we have inserted into the Hive-HBase table:

```
hive> SELECT * FROM hbase_table_1;
OK
98      val_98
Time taken: 0.56 seconds
```

Even though we loaded two rows from the Hive `pokes` table that had the same key of 98, only one row was actually inserted into `hbase_table_1`. This is because `hbase_table_1` is an HBASE table, and although Hive tables support duplicate keys, HBase tables only support unique keys. HBase tables arbitrarily retain only one key, and will silently discard all the data associated with duplicate keys.

Convert a pre-existing HBase table to a Hive-HBase table

To convert a pre-existing HBase table to a Hive-HBase table, enter the following four commands at the Hive prompt.

Note that in this example the existing HBase table is `my_hbase_table`.

```
hive> CREATE EXTERNAL TABLE hbase_table_2(key int, value string)
    > STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
    > WITH SERDEPROPERTIES ("hbase.columns.mapping" = "cf1:val")
    > TBLPROPERTIES("hbase.table.name" = "my_hbase_table");
```

Now we can run a Hive query against the pre-existing HBase table `my_hbase_table` that Hive sees as `hbase_table_2`:

```
hive> SELECT * FROM hbase_table_2 WHERE key > 400 AND key < 410;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
...
OK
401     val_401
402     val_402
403     val_403
404     val_404
406     val_406
407     val_407
409     val_409
Time taken: 9.452 seconds
```

Hive and HPL/SQL Integration

Note: This feature is presented as a developer preview. Developer previews are not tested for production environments, and should be used with caution.

HPL/SQL includes a Hive UDF function that allows you to execute HPL/SQL scripts (user-defined functions written in HPL/SQL language) in Hive queries.

HPL/SQL uses the `hplsql_locals.sql` file to parse a prepared procedure that can be used in the Hive query. If you want to add and use multiple functions, you should add each function to the `hplsql_locals.sql` file.

For example, to call the `hello` function from a Hive query, you can add a `hello` function to the `hplsql_locals.sql` file:

```
CREATE FUNCTION hello(text STRING)
 RETURNS STRING
BEGIN
 RETURN 'Hello, ' || text || '!';
END;
```

There are two possible ways to run the HPL/SQL `hello` function:

Running HPL/SQL from Hive CLI/Hive Beeline

The `hplsql_locals.sql` file must be located in the directory where the Hive CLI is started or in the `/opt/mapr/hive/hive-<version>/bin` directory if you are using Beeline. After adding the `hello` function to the `hplsql_locals.sql` file, register the HPL/SQL UDF in Hive as follows:

```
CREATE TEMPORARY FUNCTION hplsql AS 'org.apache.hive.hplsql.Udf';
```

To use the `hello` function written in HPL/SQL language in Hive, use a query such as the following:

```
SELECT hplsql('hello(:1)', name) FROM users;
```

Running HPL/SQL from the HPL/SQL CLI

When you run HPL/SQL scripts using the HPL/SQL CLI, you can use user-defined functions the same way you use built-in functions:

```
hplsql -e "SELECT hello(name) FROM users;"
```


last-updated: Apr 11, 2022

The HPL/SQL CLI automatically connects to HiveServer2 using the configuration from the `hplsql-site.xml` file, registers the Hive UDF, and modifies the function call in the SQL statements. But you must ensure that the `hplsql_locals.sql` file containing the user-defined functions is located in the `/opt/mapr/hive/hive-<version>/bin` directory, where HiveServer2 can parse it.

For more information, see User-Defined Functions and Stored Procedures.

Hive and HCatalog Integration

The HCatalog on page 3404 library provides applications with a table view of the MapR File System layer in your cluster, expanding your application's options from read/write data streams to add table operations such as get row and store row. The HCatalog library stores the metadata required for its operations in the Hive Metastore.

The `hcat` utility can execute any of the data definition language (DDL) commands available in Hive that do not involve launching a MapReduce application. Internally, the `hcat` utility passes DDL commands to the `hive` program. Data stored in the MapR filesystem is serialized and deserialized through `InputStorageFormats` and `OutputStorageFormats` objects for records. Fields within a record are parsed with `SerDes`.

Warning:

The `hive-json-serde-0.2.jar` JSON serializer/deserializer has not implemented a `serialize()` method and as a result does not function.

The WebHCat server provides a REST-like web API for HCatalog. For more information about using WebHCat, see Hive and WebHCat Integration on page 3496.

This section contains the following topics:

Accessing HCatalog Tables from Hive

To access tables created in HCatalog in Hive, use the following command to append paths to your `HADOOP_CLASSPATH` environment variable:

```
export HADOOP_CLASSPATH=${HADOOP_CLASSPATH}:$HCAT_HOME/share/hcatalog/
storage-handlers/hbase/lib/hbase-storage-handler-<version>.jar:$HCAT_HOME/
share/hcatalog/hcatalog-core-<version>-mapr.jar:$HCAT_HOME/share/hcatalog/
hcatalog-pig-adapter-<version>-mapr.jar:$HCAT_HOME/share/hcatalog/
hcatalog-server-extensions-<version>-mapr.jar
```

Loading and Retrieving Data from Pig

To use the HCatalog library `HCatLoader` and `HCatStorer` to load and retrieve data from Pig:

1. Create a table with the `hcat` utility.

   ```
   hcat -e "create table hcatpig(key int, value string)"
   ```

2. Verify that the table and table definition both exist.

   ```
   hcat -e "describe formatted hcatpig"
   ```

3. Load data into the table from Pig: Copy the `$HIVE_HOME/examples/files/kv1.txt` file into the MapRFS file system, then start Pig and load the file with the following commands:

```
pig -useHCatalog -Dfs.default.name=maprfs://CLDB_Host:7222/
grunt> A = LOAD 'kv1.txt' using PigStorage('\u0001') AS(key:INT,
value:chararray);
grunt> STORE A INTO 'hcatpig' USING
org.apache.hive.hcatalog.pig.HCatStorer();
```

4. Retrieve data from the `hcatpig` table with the following Pig commands: Another way to verify that the data is loaded into the `hcatpig` table is by looking at the contents of `maprfs://user/hive/warehouse/hcatpig/`. HCatalog tables are also accessible from the Hive CLI. All Hive queries work on HCatalog tables.

```
B = LOAD 'default.hcatpig' USING
org.apache.hive.hcatalog.pig.HCatLoader();
dump B; // this should display the records in kv1.txt
```

Running MapReduce Applications

This example uses a sample MapReduce program named `HCatalogMRTest.java`.

1. From the command line, issue the following commands to define the environment:

```
export LIB_JARS=
$HCAT_HOME/share/hcatalog/hcatalog-core-<version>-mapr.jar,
$HIVE_HOME/lib/hive-metastore-<version>-mapr.jar,
$HIVE_HOME/lib/libthrift-<version>.jar,
$HIVE_HOME/lib/hive-exec-<version>-mapr.jar,
$HIVE_HOME/lib/libfb303-<version>.jar,
$HIVE_HOME/lib/jdo2-api-<version>-ec.jar,
$HIVE_HOME/lib/slf4j-api-<version>.jar

export HADOOP_CLASSPATH=
$HCAT_HOME/share/hcatalog/hcatalog-core-<version>-mapr.jar:
$HIVE_HOME/lib/hive-metastore-<version>-mapr.jar:
$HIVE_HOME/lib/libthrift-<version>.jar:
$HIVE_HOME/lib/hive-exec-<version>-mapr.jar:
$HIVE_HOME/lib/libfb303-<version>.jar:
$HIVE_HOME/lib/jdo2-api-<version>-ec.jar:
$HIVE_HOME/conf:
$HADOOP_HOME/conf:
$HIVE_HOME/lib/slf4j-api-<version>.jar
```

2. Compile `HCatalogMRTest.java`:

```
javac -cp `hadoop classpath`:${HCAT_HOME}/share/hcatalog/
hcatalog-core-<version>-mapr.jar HCatalogMRTest.java -d .
```

3. Create a JAR file:

```
jar -cf hcatmrtest.jar org
```

4. Create an output table:

```
hcat -e "create table hcatpigoutput(key int, value int)"
```

5. Run the job: At the end of the job, the file `hcatpigoutput` should have entries in the form `key, count`.

```
hadoop --config $HADOOP_HOME/conf jar ./hcatmrtest.jar
org.myorg.HCatalogMRTest -libjars $LIB_JARS hcatpig hcatpigoutput
```

Running Non-MapReduce Applications

This example uses a sample MapReduce program named `TestReaderWriter.java`.

1. Add the following JAR files to your `$HADOOP_CLASSPATH` environment variable with the following command:

```
export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:/opt/
mapr/hive/hive-<version>/lib/antlr-runtime-3.4.jar:/opt/mapr/hive/
hive-<version>/lib/hive-cli-<veresion>-mapr.jar
```

2. Compile the test program with the following command:

```
javac -cp `hadoop classpath`:${HCAT_HOME}/share/hcatalog/
hcatalog-core-<version>-mapr.jar TestReaderWriter.java -d <directory>
```

3. Create a JAR file with the following command:

```
jar -cf hcatrwtest.jar org
```

4. Run the job with the following command:

```
hadoop jar /root/<username>/hcatalog/hcatrwtest.jar
org.apache.hive.catalog.data.TestReaderWriter -libjars $LIB_JARS
```

The last command should result in a table named `mytbl` that is populated with data.

Hive and WebHCat Integration

The WebHCat server provides a REST-like web API for HCatalog. Applications make HTTP requests to run Pig, Hive, and HCatalog DDL from within applications.

This topic contains the following sections:

Configuring the WebHCat Server

The properties to configure WebHCat are in the following file:

```
/opt/mapr/hive/hive-<version>/hcatalog/etc/webhcat/webhcat-site.xml
```

When you set up WebHCat, you can configure MapR File System and Zookeeper as storage.

1. To configure storage for WebHCat, add the MapRFS location property.

```
<property> <name>templeton.storage.class</name>
<value>org.apache.hive.hcatalog.templeton.tool.HDFSStorage</value> </
property> <property> <name>templeton.storage.root</name> <value>/user/
mapr/webhcat</value> <description>The path to the directory to use for
storage</description> </property>
```

2. To configure WebHCat for Pig:

a) Compress the Pig installation, then move the compressed file to the MapRFS layer.

```
# cd /opt/mapr/pig
# tar -czvf /tmp/pig-<version>.tar.gz pig-<version>/
# hadoop fs -mkdir /user/mapr/webhcat
# hadoop fs -put /tmp/pig-<version>.tar.gz /user/mapr/webhcat/
```

b) Set the value of the `templeton.pig.archive` property to the location of the compressed file.

```
<property> <name>templeton.pig.archive</name> <value>maprfs:///user/
mapr/webhcat/pig-<version>.tar.gz</value> </property>
```

c) Set the value of the `templeton.pig.path` property to the path inside the compressed Pig file where the Pig binary is located.

```
<property>
    <name>templeton.pig.path</name>
    <value>pig-<version>.tar.gz/pig-<version>/bin/pig</value>
</property>
```

3. To configure WebHCat for Hive:

a) Compress the Hive installation, then move the compressed file to the MapR File System layer.

```
# cd /opt/mapr/hive
# tar -czvf /tmp/hive-<version>.tar.gz hive-<version>/
# hadoop fs -mkdir /user/mapr/webhcat
# hadoop fs -put /tmp/hive-<version>.tar.gz /user/mapr/webhcat
```

b) Set the value of the `templeton.hive.archive` property to the location of the compressed file.

```
<property> <name>templeton.hive.archive</name> <value>maprfs:///user/
mapr/webhcat/hive-<version>.tar.gz</value> </property>
```

c) Set the value of the `templeton.hive.path` property to the path inside the compressed Hive file where the Hive binary is located.

```
<property>
     <name>templeton.hive.path</name>
     <value>hive-<version>.tar.gz/hive-<version>/bin/hive</value>
</property>
```

4. To Configure WebHCat for streaming:

a) Copy the Streaming JAR to the MapR File System layer.

```
# hadoop fs -put
/opt/mapr/hadoop/hadoop-<version>/contrib/streaming/
hadoop-<version>-dev-streaming.jar /user/mapr/webhcat
```

b) Set the `templeton.streaming.jar` property to the location of the streaming JAR.

```
<property> <name>templeton.streaming.jar</name> <value>maprfs:///user/
mapr/webhcat/hadoop-<version>-dev-streaming.jar</value> </property>
```


last-updated: Apr 11, 2022

Configure WebHCat Server to use SSL Encryption

You can configure WebHCat REST-API to use SSL (Secure Sockets Layer) encryption. The following WebHCat properties are added to enable SSL:

```
templeton.use.ssl
Default value: false
Description: Set this to true for using SSL encryption for  WebHCat server

templeton.keystore.path
Default value: <empty string>
Description: SSL certificate keystore location for WebHCat server

templeton.keystore.password
Default value: <empty string>
Description: SSL certificate keystore password for WebHCat server

templeton.ssl.protocol.blacklist
Default value: SSLv2,SSLv3
Description: SSL Versions to disable for WebHCat server

templeton.host
Default value: 0.0.0.0
Description: The host address the WebHCat server will listen on
```

Modifying the `webhcat-site.xml` file:

Configure the following properties in the `webhcat-site.xml` file to enable SSL encryption on each node where HWebHCat is installed:

```
<!-- WebHCat SSL -->
<property>
  <name>templeton.use.ssl</name>
  <value>true</value>
</property>

<property>
  <name>templeton.keystore.path</name>
  <value>/opt/mapr/conf/ssl_keystore</value>
</property>

<property>
  <name>htempleton.keystore.password</name>
  <value><ssl-keystore-password></value>
</property>
```

Note: After running `/opt/mapr/server/configure.sh -R`, all properties needed to enable SSL encryption for WebHCat are added automatically to `webhcat-site.xml` on the Mapr-SASL secure cluster.

To check status of WebHCat server configured for SSL encryption, use following command:

```
curl -k 'https://<user>:<password>@<host>:50111/templeton/v1/status'
```

Requirements for Using Automatically Generated PEM Files
To use automatically generated PEM files for the WebHCat REST API on a MapR-SASL cluster, you need to have a cluster with a host name that consists at least of three parts: administrator user name and password, and WebHCat REST API host.

Check the status of the WebHCat REST API to make sure you have a cluster with a host name that consists of the administrator user name and password, and WebHCat REST API host service:

```
curl  --cacert /opt/mapr/conf/ssl_truststore.pem  -u
<cluster_admin_user>:<cluster_admin_password>
"https://<myhost.mapr.com>:50111/templeton/v1/status" -v
```

The sample output for this example is as follows:

```
* TCP_NODELAY set
* Connected to c74v610.mapr.com (192.168.122.254) port 50111 (#0)
* Cipher selection: ALL:!EXPORT:!EXPORT40:!EXPORT56:!aNULL:!LOW:!
RC4:@STRENGTH
* successfully set certificate verify locations:
CAfile: /opt/mapr/conf/ssl_truststore.pem
CApath: none
* (303) (OUT), TLS Unknown, Certificate Status (22):
* (303) (OUT), TLS handshake, Client hello (1):
* (303) (IN), TLS handshake, Server hello (2):
* (303) (IN), TLS handshake, Certificate (11):
* (303) (IN), TLS handshake, Server key exchange (12):
* (303) (IN), TLS handshake, Server finished (14):
* (303) (OUT), TLS handshake, Client key exchange (16):
* (303) (OUT), TLS change cipher, Client hello (1):
* (303) (OUT), TLS handshake, Finished (20):
* (303) (IN), TLS change cipher, Client hello (1):
* (303) (IN), TLS handshake, Finished (20):
* SSL connection using unknown / ECDHE-RSA-AES256-GCM-SHA384
* Server certificate:
*  subject: CN=*.mapr.com
*  start date: May 10 15:18:03 2018 GMT
*  expire date: Apr 16 15:18:03 2118 GMT
*  common name: *.mapr.com (matched)
*  issuer: CN=*.mapr.com
*  SSL certificate verify ok.
* Server auth using Basic with user 'mapr'
> GET /templeton/v1/status HTTP/1.1
> Host: c74v610.mapr.com:50111
> Authorization: Basic bWFwcjptYXBy
> User-Agent: curl/7.59.0
> Accept: */*
>
< HTTP/1.1 200 OK
< Set-Cookie:
hadoop.auth="u=mapr&p=mapr&t=multiauth&e=1526001586135&s=dgOtxP2Hs95DBl0Jyxy
V/oJ1BZk="; Path=/; Domain=.mapr.com; Expires=Fri, 11-May-2018 01:19:46
GMT; Secure; HttpOnly
< Content-Type: application/json
< Transfer-Encoding: chunked
< Server: Jetty(7.6.0.v20120127)
<
* Connection #0 to host c74v610.mapr.com left intact
{"version":"v1","status":"ok"}
```

Managing the WebHCat Server

As of Hive 0.13-1504 and Hive 1.0-1504, WebHCat is managed by Warden. Therefore, you can start and stop WebHCat using maprcli and the Control System.

Starting the WebHCat Server

Applies to versions prior to Hive 0.13-1504 and Hive 1.0-1504:

```
# ./webhcat_server.sh start
```

Starting WebHCat Using the maprcli

1. Make a list of nodes on which Hive Metastore is configured.
2. Issue the maprcli node services command:

```
maprcli node services -name hcat -action start -nodes <space delimited
list of nodes>
```

Stopping WebHCat Using the maprcli

1. Make a list of nodes on which Hive Metastore is configured.
2. Issue the maprcli node services command:

```
maprcli node services -name hcat -action stop -nodes <space delimited
list of nodes>
```

Starting or Stopping WebHCat Using the Control System

1. In the Navigation pane, expand the Cluster Views pane and click **Dashboard**.
2. In the Services pane, click **WebHcat** to open the Nodes screen displaying all the nodes on which Hive Metastore is configured.
3. On the Nodes screen, click the hostname of each node to display its Node Properties screen.
4. On each Node Properties screen, use the **Stop/Start** button in the WebHcat row under Manage Node Services to start WebHcat.

Checking the Error Logs

Go to the following folder:

```
/opt/mapr/hive/hive-<version>/logs/<user.name>/webhcat
```

Note: If you are running a Hive 0.13 version prior Hive 0.13-1504, go to the /tmp/<user.name>/webhcat folder to view the error logs.

Verifying the Server's Status

In a web browser, navigate to:

```
http://hostname:50111/templeton/v1/status?user.name=root
```

A healthy server will return the string `{"status":"ok","version":"v1"}`. You can change the port number from the default value of 50111 by editing the `webhcat-site.xml` file.

Running Jobs on the WebHCat Server

REST Calls in WebHCat

The base URI for REST calls in WebHCat is `http://<host>:<port>/templeton/v1/`. The following table lists elements appended to the base URI and DDL commands.

last-updated: Apr 11, 2022

URI	Description
Server Information	
/status	Shows WebHCat server status.
/version	Shows WebHCat server version.
DDL Commands	
/ddl/database	List existing databases.
/ddl/database/<mydatabase>	Shows properties for the database named *mydatabase*.
/ddl/database/<mydatabase>/table	Shows tables in the database named *mydatabase*.
/ddl/database/<mydatabase>/table/<mytable>	Shows the table definition for the table named *mytable* in the database named *mydatabase*.
/ddl/database/<mydatabase>/table/<mytable>/property	Shows the table properties for the table named *mytable* in the database named *mydatabase*.

Launching a MapReduce Job with WebHCat

WebHCat launches two jobs for each MapReduce job. The first job, `TempletonControllerJob`, has one map task. The map task launches the actual job from the REST API call. Check the status of both jobs and the output directory contents.

1. Copy the MapReduce example job to the MapRFS layer:

```
hadoop fs -put /opt/mapr/hadoop/hadoop-<version>/
hadoop-<version>-dev-examples.jar /user/mapr/webhcat/examples.jar
```

2. Use the `curl` utility to launch the job:

```
curl -s -d jar=examples.jar -d class="terasort" -d
arg=teragen.test -d arg=whop3 'http://localhost:50111/templeton/v1/
mapreduce/jar?user.name=<username>'
```

Launching a Streaming MapReduce Job with WebHCat

1. Use the `curl` utility to launch the job:

```
curl -s -d arg=teragen.test -d output=mycounts -d mapper=/bin/cat -d
reducer="/usr/bin/wc -w" 'http://localhost:50111/templeton/v1/mapreduce/
streaming?user.name=<username>'
```

2. Check the job status for both WebHCat jobs at the jobtracker page in the Control System.

Launching a Pig Job with WebHCat

1. Copy a data file into MapRFS:

```
hadoop fs -put $HIVE_HOME/examples/files/kv1.txt /user/<user name>/
```

2. Create a `test.pig` file with the following contents:

```
A = LOAD 'kv1.txt' using PigStorage('\u0001') AS(key:INT,
value:chararray);
STORE A INTO 'pig.output';
```

3. Copy the `test.pig` file into MapR filesystem:

```
hadoop fs -put test.pig /user/<user name>/
```

4. Run the Pig REST API command:

```
curl -s -d file=test1.pig -d arg=-v 'http://localhost:50111/templeton/v1/
pig?user.name=<username>'
```

5. Monitor the contents of the `pig.output` directory.

6. Check the JobTracker page for two jobs: `TempletonControllerJob` and `PigLatin`.

Launching a Hive Job with WebHCat

1. Create a table:

```
curl -s -d execute="create+external+table+ext3(t+TIMESTAMP)+location /
user/<user name>/ext3'" 'http://localhost:50111/templeton/v1/hive?
user.name=<username>'
```

2. Load data into the table:

```
curl -s -d execute="insert+overwrite+table+ext3+select+*+from+datetable"
'http://localhost:50111/templeton/v1/hive?user.name=<username>'
```

3. List the tables:

```
curl -s -d execute="show+tables" -d statusdir='hive.output' 'http://
localhost:50111/templeton/v1/hive?user.name=<username>'
```

   The list of tables is in `hive.output/stdout`.

The Job Queue

To show HCatalog jobs for a particular user, navigate to the following address:

```
http://<hostname>:<port>/templeton/v1/queue/?user.name=<username>
```

The default port for HCatalog is 50111.

Hive and Tez Integration
You can use Tez, instead of MapReduce, for generic data processing tasks. Tez significantly increases the processing speed. Tez, working with Hive, provides lower latency for interactive queries and higher throughput for batch queries.
Configuring Hive and Tez

To configure Hive on Tez, repeat the following steps on each node where you want to configure Hive on Tez. Tez mode for MR jobs is not compatible with all MR jobs, so do not set up the whole cluster to work on Tez.

There is a known issue related to the incomplete removal of previously installed Tez packages. The issue affects platforms on which Tez was installed but later removed using `sudo apt-get remove mapr-tez`. Because of Ubuntu-specific behavior and Tez source-code issues, the `remove` command removes Tez only partially in some installations. If this happens, an error is generated when you try to re-install Tez on Ubuntu, as described following in step 1. If you believe your installation might have this issue, you can prevent the error. Before performing the following steps, use the `purge` command to completely remove all previously installed Tez packages.

1. Install Tez if it is already not installed. To install Tez, run the following command:

On CentOS / RedHat	`yum install mapr-tez`
On SLES	`zypper install mapr-tez`
On Ubuntu	`apt-get install mapr-tez`

   Note: Repeat this step on each node where you want Hive on Tez to be configured.

2. Create the `/apps/tez` directory on MapR filesystem.

   To create, run the following commands:

```
hadoop fs -mkdir /apps
hadoop fs -mkdir /apps/tez
```

3. Upload the Tez libraries to the `/tez` directory on the MapR file system.

   To upload, run the following commands:

```
hadoop fs -put /opt/mapr/tez/tez-<version> /apps/tez
hadoop fs -chmod -R 755 /apps/tez
```

4. Verify the upload.

   To verify, run the following command:

```
hadoop fs -ls /apps/tez/tez-<version>
```

5. Set the Tez environment variables. To set, open the `/opt/mapr/hive/hive-<version>/conf/hive-env.sh` file, add the following lines, and save the file:

```
export TEZ_CONF_DIR=/opt/mapr/tez/tez-<version>/conf
export TEZ_JARS=/opt/mapr/tez/tez-<version>/*:/opt/mapr/tez/
tez-<version>/lib/*
export HADOOP_CLASSPATH=$TEZ_CONF_DIR:$TEZ_JARS:$HADOOP_CLASSPATH
```

   Note: Repeat this step on each node where you want Hive on Tez to be configured.

last-updated: Apr 11, 2022

6. Configure Hive for Tez engine. To configure, open the `/opt/mapr/hive/hive-<version>/conf/hive-site.xml` file, add the following lines, and save the file.

```
<property>
  <name>hive.execution.engine</name>
  <value>tez</value>
</property>
```

Add the `hive.exec.pre.hooks`, `hive.exec.post.hooks`, and `hive.exec.failure.hooks` properties with value `org.apache.hadoop.hive.ql.hooks.ATSHook` to use the Hive queries page in the Tez UI.

Note: Starting from EEP 6.3.4, the following execution-hooks properties are managed by running `configure.sh` command with `-R` option.

```
<property>
  <name>hive.exec.pre.hooks</name>
  <value>org.apache.hadoop.hive.ql.hooks.ATSHook</value>
</property>

<property>
  <name>hive.exec.post.hooks</name>
  <value>org.apache.hadoop.hive.ql.hooks.ATSHook</value>
</property>

<property>
    <name>hive.exec.failure.hooks</name>
    <value>org.apache.hadoop.hive.ql.hooks.ATSHook</value>
</property>
```

Note: Repeat this step on each node where you want Hive on Tez to be configured.

7. Run `configure.sh` with the `-R` option.

```
/opt/mapr/server/configure.sh -R
```

Note: Starting in EEP 6.0.1 and later, Tez should be configured by running the `$MAPR_HOME/server/configure.sh` script with the `-R` option.

8. Configure Tez shuffle on a secured cluster:

   Refer to Tez Shuffle on page 3511 to configure SSL encryption on shuffle.

Known Issues and Restrictions

Sqoop importing data into Hive fails when the entire cluster is configured to use Tez.	This is because of Sqoop's incompatibility with Tez. **Workaround:** Do not configure the entire cluster to use Tez.
Percentage sampling is not supported in `org.apache.hadoop.hive.ql.io.HiveInputFormat`.	Hive uses `org.apache.hadoop.hive.ql.io.HiveInputFormat` by default and so queries like `'SELECT * FROM tablename TABLESAMPLE(20 percent);'` will not work for Hive on Tez. **Workaround:** Instead of `org.apache.hadoop.hive.ql.io.HiveInputFormat`, use

last-updated: Apr 11, 2022

	`org.apache.hadoop.hive.ql.io.CombineHiveInputFormat`.
	To change input format, do one of the following:
	• Set `hive.tez.input.format` in hive shell. For example:
	`hive> set hive.tez.input.format=org.apache.hadoop.hive.ql.io.CombineHiveInputFormat;`
	• Add `org.apache.hadoop.hive.ql.io.CombineHiveInputFormat` to `hive-site.xml` file. For example:
	`<property> <name>hive.tez.input.format</name> <value>org.apache.hadoop.hive.ql.io.CombineHiveInputFormat</value> </property>`

Hive on Tez does not work well with Sequence Files Schema changes	TEZ-2741
Limitations with common joins	HIVE-11693: The CommonMergeJoinOperator only sets big table position when it has inputs for big table. If the input is empty, the method is not called.
HiveServer2 on Tez doesn't support concurrent queries within one session	HIVE-9223: When multiple queries are submitted in the same HS2 session concurrently, some queries fail with an error.

Tez upgrade issues	• No support for preserving configuration from EEP-5.0.0 and EEP-4.1.1 (ECO-1803) to EEP-6.0.0(1808) or EEP-5.0.1(1808) on Ubuntu.
	• No support for preserving Tomcat configuration from previous EEPs to EEP-6.0.0 (1808).
	• You should manually stop the Tomcat service and delete the tomcat folder as a precondition if you are updating or upgrading Tez from the following EEPs:
	`EEP-4.0.0`
	`EEP-4.1.0`

Tez shuffle SSL encryption issue	During a shuffle phase, the `javax.net.ssl.SSLException` error could occur on a multi-node cluster due to insufficient Tez shuffle SSL encryption configuration, see Tez Shuffle on page 3511 for a solution.

SQL Limitations

The following is a list of SQL limitations on Hive on Tez:

Issue	Summary
HIVE-11693	CommonMergeJoinOperator throws exception with Tez.

last-updated: Apr 11, 2022

Issue	Summary
HIVE-9989	Hive on Tez group by with cast(NULL AS BIGINT) throws NPE.
HIVE-11270	Tez gives different responses when run on Physical tables and logical views.
HIVE-9223	HiveServer2 on Tez doesn't support concurrent queries within one session.
HIVE-13623	Hive on Tez produces wrong results when withClause and (outer) joins.
TEZ-2741	Hive on Tez does not work well with Sequence Files Schema changes.
HIVE-13926	Cannot limit reduce (not both Map and Reduce) memory in Tez engine.

Hive-on-Tez User Interface

This section describes how to install, configure, manage, and start the Hive-on-Tez user interface.

Warning: The Hive-on-Tez user interface supports RM HA only starting from the 1803 release (EEP 4.1.1 and EEP-5.0.0).

Installing the Hive-on-Tez User Interface

This topic describes installation of the Hive-on-Tez user interface by using the MapR Installer or manual steps.

Installation Using the MapR Installer

When you use the MapR Installer to install Tez, the timeline server for the Hive-on-Tez user interface is installed automatically. If the **Enable MapR Secure Cluster** option is enabled in the MapR Installer, the timeline server is installed to be secure.

The Tomcat server is installed into this folder:

```
/opt/mapr/tez/tez-<version>/tomcat/apache-tomcat-<version>
```

To start using the Hive-on-Tez user interface if the **Enable MapR Secure Cluster** option is enabled or if the cluster is Kerberized, you must log in to the timeline server user interface:

```
https://<hostname>:8190
```

Manual Installation

To install the Hive-on-Tez user interface manually:

1. Install and configure `mapr-tez` as described in Configuring Hive and Tez on page 3502.
2. Install the timeline server:

On CentOS / Red Hat	`yum install mapr-timelineserver`
On SLES	`zypper install mapr-timelineserver`
On Ubuntu	`apt-get install mapr-timelineserver`

Note: Install the timeline server on a single node. The Hive-on-Tez user interface does not support High Availability (HA).

last-updated: Apr 11, 2022

Configuring the Timeline Server to Use the Hive-on-Tez User Interface

This topic describes how to configure the timeline server to use the Hive-on-Tez user interface. This topic includes security configuration information.

When the timeline server is installed using the MapR Installer, the installer secures the timeline server automatically. When you install the timeline server manually, use these steps.

Note: This procedure assumes that you have previously configured the cluster using the `configure.sh` script.

1. Run `configure.sh -R` (on all Hive nodes), replacing `<hostname>` with the name of your timeline server node:

```
sudo /opt/mapr/server/configure.sh -R -TL <hostname>
```

   Note: Make sure the hostname matches the CN in ssl_keystore for secure clusters. If not, all hive and yarn jobs fail. The hostname can be obtained using the `$hostname -f` command.

   Running `configure.sh -R` configures the timeline server properties in `/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/yarn-site.xml` for enhanced security.

2. To use the timeline server with Kerberos, you need to make additional entries to the `/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/yarn-site.xml` file. Replace the following variables with real values:
   - MAPR_PRINCIPAL
   - PATH_TO_KEYTAB
   - HTTP_PRINCIPAL

```
<property>
  <name>yarn.timeline-service.principal</name>
  <value>MAPR_PRINCIPAL</value>
</property>
<property>
<name>yarn.timeline-service.keytab</name>
  <value>PATH_TO_KEYTAB</value>
</property>
<property>
  <name>yarn.timeline-service.http-authentication.kerberos.principal</
name>
  <value>HTTP_PRINCIPAL</value>
</property>
<property>
 <name>yarn.timeline-service.http-authentication.kerberos.keytab</name>
 <value>PATH_TO_KEYTAB</value>
</property>
```

3. Restart the resource manager:

```
maprcli node services -name resourcemanager -action restart -nodes
<hostname>
```

Configuring the Tomcat Server

This topic describes how to configure and manage the Tomcat server for the Hive-on-Tez user interface.

last-updated: Apr 11, 2022

Extracting the Tomcat Server

You can only extract the Tomcat server after you manually install Tez. Tez has a built-in Tomcat Server archive with the latest version. You can find the archive at:

```
$TEZ_HOME/tomcat/tomcat.tar.gz
```

To extract the Tomcat server, use these commands in the command line:

```
cd $TEZ_HOME/tomcat/
sudo tar -zxvf tomcat.tar.gz -C $TEZ_HOME/tomcat
```

Change the permissions for the tomcat directory to the user who will be running the Tomcat server:

```
sudo chown -R <$USER>:<$USER_GROUP> $TEZ_HOME/tomcat
```

Configuring the Timeline Server Base URL and Resource Manager WEB URL

To set the `timelineBaseUrl` and `RMWebUrl`, update the Tez configuration file.

For EEP 6.2.0 and earlier, the file location is:

```
nano $TEZ_HOME/tomcat/apache-tomcat-<version>/webapps/tez-ui/config/
configs.env
```

For EEP 6.3.0 and later, the file location is:

```
nano $TEZ_HOME/tomcat/apache-tomcat-<version>/webapps/tez-ui/config/
configs.js
```

To configure the Timeline Server Base URL and Resource Manager WEB URL:

1. Replace TIME_LINE_BASE_URL with the real URL, for example:
   - For a non-secure configuration:

     ```
     'http://localhost:8188'
     ```

   - For a secure configuration:

     ```
     'https://localhost:8190'
     ```

2. Replace RM_WEB_URL with the real URL, for example:
   - For a non-secure configuration:

     ```
     'http://localhost:8088'
     ```

   - For a secure configuration:

     ```
     'https://localhost:8090'
     ```

- For a proxy server, specify the user-defined URL in the `yarn-site.xml` file, as shown:

```
<property>
   <name>yarn.web-proxy.address</name>
   <value><hostname>:<port></value>
</property>
```

  Replace RM_WEB_URL with the value specified as the `yarn.web-proxy.address` property.

Configuring SSL for the Tomcat Server on a Secure Cluster

To start the Tomcat server with the exposed SSL port, edit the following properties in the `$TEZ_HOME/tomcat/apache-tomcat-9.0.1/conf/server.xml` file, replacing `<ssl-keystore-password>` with the real SSL keystore password.

1. Find default configuration of the exposed port:

```
<Connector port="9383"
  protocol="HTTP/1.1"
  connectionTimeout="20000"
  redirectPort="8443" />
```

2. Change configuration for SSL:

```
<Connector port="9393"
SSLEnabled="true"
maxThreads="150"
scheme="https"
secure="true"
clientAuth="false"
sslProtocol="TLS"
keystoreFile="/opt/mapr/conf/ssl_keystore"
keystorePass="<ssl-keystore-password>"/>
```

Starting and Stopping the Tomcat Server

To start the Tomcat server, run this script:

```
$TEZ_HOME/tomcat/apache-tomcat-<version>/bin/startup.sh
```

To stop the Tomcat server, run this script:

```
$TEZ_HOME/tomcat/apache-tomcat-<version>/bin/shutdown.sh
```

Note: The `timelineBaseUrl` maps to the YARN Timeline Server, and the `RMWebUrl` maps to the YARN Resource Manager. For default port information, see Ports Used by MapR Software on page 2290.

Integrating the Hive-on-Tez User Interface with Tez

This topic describes how to integrate the Hive-on-Tez user interface with Tez.

last-updated: Apr 11, 2022

1. Add the following entry to the `/opt/mapr/tez/tez-<version>/conf/tez-site.xml` file, replacing `<hostname>:<port>` with the real host name. You can use 9383 or 9393 for the port. 9383 is HTTP and 9393 is HTTPS Tomcat port for the Hive-on-Tez user interface.

```
<property>
  <description>Enable Tez to use the Timeline Server for History
Logging</description>
  <name>tez.history.logging.service.class</name>

<value>org.apache.tez.dag.history.logging.ats.ATSHistoryLoggingService</
value>
</property>

<property>
  <description>URL for where the Tez UI is hosted</description>
  <name>tez.tez-ui.history-url.base</name>
   <value>http(s)://<hostname>:<port>/tez-ui/</value>
</property>
```

Repeat this step on each node where you want the Hive-on-Tez user interface to be configured.

Connecting to the Hive-on-Tez User Interface

This topic describes how to connect to the Hive-on-Tez user interface.

To start using the Hive-on-Tez user interface on a MapR secure or Kerberized cluster or non-secure cluster, you must log in to the timeline server user interface and RM UI:

```
https://<hostname>:8190
```

```
https://<hostname>:8090
```

To connect to the Hive-on-Tez user interface on secure clusters, use a browser to navigate to:

```
https://<hostname>:9393/tez-ui/
```

where `<hostname>` is the host where the Tomcat server is running.

Hive-on-Tez User Interface Known Issues

This topic describes known issues that you should be aware of while troubleshooting.

Timeline Server Known Issues

(**Issue 29538**) After an incremental install or rolling upgrade to MapR 6.1, the timeline server does not start. To resolve this issue, add the following entry to `/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/yarn-env.sh`:

```
export YARN_TIMELINESERVER_OPTS="${YARN_TIMELINESERVER_OPTS} $
{MAPR_LOGIN_OPTS}"
```

To grant administrative privileges to any user(s), modify `yarn-site.xml`, as shown:

```
<property>
  <name>yarn.admin.acl</name>
  <value><user_name></value>
</property>
```

After the `yarn.admin.acl` property takes effect, the user specified by `<user_name>` has administrative privileges and access to all jobs.

For example, User-A can access all the jobs owned by User-A, by default. If User-A needs access to jobs owned by other users, administrative privileges can be granted to User-A through the `yarn.admin.acl` property.

Tez Shuffle

Tez uses `org.apache.hadoop.mapred.ShuffleHandler` provided by MapReduce version 2.0 (MRv2) as an auxiliary service, which you can choose to configure via the `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/mapred-site.xml` file.

On a secured cluster, Tez shuffle, SSL encryption configuration is enabled in `/opt/mapr/tez/tez-<version>/conf/tez-site.xml` by default:

`tez.runtime.shuffle.ssl.enable`	`true`
`tez.runtime.shuffle.keep-alive.enabled`	`true`

Also, you must configure Tez shuffle for YARN by adding the following property to the `mapred-site.xml` file. Edit the `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/mapred-site.xml` file:

`mapreduce.shuffle.ssl.enabled`	`true`

Queue Management with Hive-on-Tez

HiveServer2 provides built-in functionality to set-up and handle a pool of Tez sessions in default queues. Tez initiates a session and keeps it alive to run sequential queries. Quereies can be submitted through HiveServer2 clients, such as Beeline and the Hive CLI. You can manage queues through properties in `hive-site.xml`.

Queue management is strongly connected to the type of YARN Scheduler used. By default, a MapR cluster uses Fair Scheduler and Hive-on-Tez to run queries in queues with a user name. If a query is submitted from the Hive CLI, the real user name is used. If a query is submitted from a HiveServer2 client, such as Beeline, the queue name depends on the HiveServer2 impersonation configuration property, `hive.server2.enable.doAs`, where the queue name could be the real user name or the user name of the Hiveserver2 process.

With Capacity Scheduler, Hive queries submitted from the CLI and Beeline are configured through the `capacity-scheduler.xml` file. Default queue names are chosen from the scheduler settings, but you can also use the `tez.queue.name=<queue_name>` property to run queries in a specific queue.

Application Masters (AM) are strongly bound to YARN. You cannot change the queue for an AM that is already started. If impersonation is enabled for HiveServer2, a new AM starts next to an existing AM for a default queue. Do not use or close a default queue at the end of a lifetime.

Note: HiveServer2 works with or without impersonation. Impersonation is set through the `hive.server2.enable.doAs` property.

Run Queries in a Specific Queue

If you want all queries to run in a specific queue, you can configure a queue name through the `tez.queue.name` property. When you configure a queue name through the `tez.queue.name` property, Tez sets the queue name for all jobs submitted from the client to the configured `tez.queue.name`. You can set this property before each query through the Hive SET command, as shown:

```
set tez.queue.name=<queue_name>;
```


last-updated: Apr 11, 2022

Or, you can set the property in the `hive-site.xml` file, as shown:

```
<property>
   <name>tez.queue.name</name>
   <value>my_queue</value>
</property>
```

Important: If you set `tez.queue.name` in hive-site.xml, and you want the queue name to persist across all queries in the session, you must also set the `hive.server2.tez.unset.tez.queue.name` property in hive-site.xml to `false`, as shown:

```
<property>
   <name>tez.queue.name</name>
   <value>my_queue</value>
</property>
<property>
    <name>hive.server2.tez.unset.tez.queue.name</name>
    <value>false</value>
</property>
```

If `hive.server2.tez.unset.tez.queue.name` is set to `true`, Hive will not persist the `tez.queue.name` across queries and instead uses the default cluster queue names.

Configuration Properties

HiveServer2 has several settings related to queue management. Specify the following properties in the `hive-site.xml` file:

Property	Description	Default Value
tez.queue.name	The queue name for all jobs submitted from a given client. Set through the Hive CLI via the SET command before running a query or through hive-site.xml. If you set the property through hive-site.xml, and you want the setting to persist across all queries that run, set `hive.server2.tez.unset.tez.queue.name` to `false`.	No default. Must be explicitly set.
hive.server2.tez.initialize.default.sessions	When set to true, enables you to use HiveServer2 without turning on Tez for HiveServer2. Useful when you want to run queries over Tez without the pool of sessions.	false
hive.server2.tez.default.queues	A list of comma-separated values that correspond to YARN queues of the same name. When HiveServer2 is launched in Tez mode, this configuration must be set to enable multiple Tez sessions to run in parallel on the cluster.	empty string
hive.server2.tez.sessions.per.default.queue	A positive integer that determines the number of Tez sessions that should launch in each of the queues specified by `hive.server2.tez.default.queues`. Determines the parallelism on each queue. For example, if you specify two default queues and two sessions per default queue, four application masters start.	1
hive.server2.tez.session.lifetime	Defines the lifetime of the Tez sessions launched by HiveServer2 when default sessions are enabled. Set to `0` to disable session expiration.	162h

last-updated: Apr 11, 2022

Property	Description	Default Value
hive.server2.tez.unset.tez.queue.name	Controls whether the `tez.queue.name` persists across all queries in a session. Must be set to `false` for the `tez.queue.name` to persist. When set to `true`, the `tez.queue.name` only applies to the first query that runs; thereafter, the default cluster queue names are used.  **Note:** This functionality was introduced in EEP 7.01 and EEP 6.3.2. A patch for previous EEP versions is available. See Applying a Patch.	true

Managing Hive Services

This section includes the following topics:

Starting Hive

You can start the Hive shell from `HIVE_HOME/bin/` with the `hive` command. Example:

```
/opt/mapr/hive/hive-<version>/bin/hive
```

When the Hive shell starts, it reads an initialization file called `.hiverc` which is located in the `HIVE_HOME/bin/` or `$HOME/` directories. You can edit this file to set custom parameters or commands that initialize the Hive command-line environment, one command per line.

When you run the Hive shell, you can specify a MySQL initialization script file using the `-i` option. Example:

```
/opt/mapr/hive/hive-<version>/bin/hive -i <filename>
```

Setting the Execution Engine

Consider the following definitions:

- Runtime: execution time of job.
- Session time: time from the start of Hive shell or Beeline until you exit.

You can change the execution engine during a session (session time), but not while executing job in the session (runtime). If you specify the execution engine before starting the job, it will override the `hive.execution.engine` property in `hive-site.xml` file. For example, to specify the execution engine:

```
hive> set hive.execution.engine=tez;
hive> *perform some query here*
```

If you open another session of hive shell or beeline, you will not see the setting in the session from before and you can set needed properties for every session.

MapR Technologies highly recommends configuring Tez as an execution engine instead of MR execution engine. MR execution engine is deprecated in Hive.

If you are currently using the MR execution engine for accessing Hive CLI and HS2, Hive will throw the following warning message:

```
Hive-on-MR is deprecated in Hive 2 and may not be available in the future
versions. Consider using a different execution engine (i.e. spark, tez) or
using Hive 1.X releases.
```

To install and configure Tez as an execution engine for Hive, see Configuring Hive and Tez.

last-updated: Apr 11, 2022

Managing Hive Metastore

The Hive Metastore is started automatically by the warden at installation time if the `mapr-hivemetastore` package is installed. It is sometimes necessary to start or stop the service (for example, after changing the configuration). You can start and stop Hive Metastore in two ways:

- Using the `maprcli node services` command - Using this command, you can start Hive Metastore on multiple nodes at one time.
- Using the MapR Control System

To start Hive Metastore using the maprcli:

1. Make a list of nodes on which Hive Metastore is configured.
2. Issue the maprcli node services command:

```
/opt/mapr/bin/maprcli node services -name hivemeta -action start -nodes
<space delimited list of nodes>
```

To stop Hive Metastore using the maprcli:

1. Make a list of nodes on which Hive Metastore is configured.
2. Issue the maprcli node services command:

```
maprcli node services -name hivemeta -action stop -nodes <space
delimited list of nodes>
```

To start or stop Hive Metastore using the Control System:

1. In the Navigation pane, expand the Cluster Views pane and click **Dashboard**.
2. In the Services pane, click **Hive Metastore** to open the Nodes screen displaying all the nodes on which Hive Metastore is configured.
3. On the Nodes screen, click the hostname of each node to display its Node Properties screen.
4. On each Node Properties screen, use the **Stop/Start** button in the Hive Metastore row under Manage Node Services to start Hive Metastore.

Managing Hiveserver2

Hiveserver2 is started automatically at installation time by the warden if the `mapr-hiveserver2` package is installed. It is sometimes necessary to start or stop the service (for example, after changing the configuration). You can start and stop Hiveserver2 in two ways:

- Using the `maprcli node services` command - Using this command, you can start Hiveserver2 on multiple nodes at one time.
- Using the MapR Control System

To start Hiveserver2 using the maprcli:

1. Make a list of nodes on which Hiveserver2 is configured.
2. Issue the maprcli node services command:

```
maprcli node services -name hs2 -action start -nodes <space delimited
list of nodes>
```

To stop Hiveserver2 using the maprcli:

1. Make a list of nodes on which Hiveserver2 is configured.
2. Issue the maprcli node services command:

```
maprcli node services -name hs2 -action stop -nodes <space delimited
list of nodes>
```

To start or stop Hiveserver2 using the Control System:

1. In the Navigation pane, expand the Cluster Views pane and click **Dashboard**.
2. In the Services pane, click **Hiveserver2** to open the Nodes screen displaying all the nodes on which Hiveserver2 is configured.
3. On the Nodes screen, click the hostname of each node to display its Node Properties screen.
4. On each Node Properties screen, use the **Stop/Start** button in the Hiveserver2 row under Manage Node Services to start Hiveserver2.

Connecting to Hive

This section contains the following topics:

Connecting to Hive Metastore

The connection requirements Hive Metastore clients use to connect to Hive Metastore is based on the Hive Metastore authenication method:

Authentication Method	Connection Requirements
MapR-SASL	Client nodes require the following: • They are configured to use MapR-SASL when authenticating with Hive Metastore. • A valid MapR ticket.
Kerberos	Client nodes require the following: • They are configured to use Kerberos when authenticating with Hive Metastore. • A valid Kerberos ticket.
No Authentication	If the cluster is not secure, client nodes do not require any MapR tickets.

Connecting to HMS is provided by the thrift service. You can configure it in hive-site.xml with hive.metastore.uris property:

```
<property>
  <name>hive.metastore.uris</name>
  <value>thrift://<n.n.n.n<:9083</value>
  <description>IP address (or fully-qualified domain name) and port of the
metastore host</description>
</property>
```

Connecting to HiveServer2

The method that HiveServer2 clients use to connect to HiveServer2 is based on the HiveServer2 Authentication method and the type of client.

last-updated: Apr 11, 2022

Using JDBC or Beeline to Connect to HiveServer2
The HiveServer2 authentication method and client type determine how the HiveServer2 clients connect to HiveServer2.

> **Tip:** For details on how to install and use ODBC to connect to Hive, see Using ODBC to Connect to HiveServer2 on page 3522. When connecting to Hive via ODBC, the client must have a valid MapR or Kerberos ticket.

Using JDBC or Beeline to Connect to HiveServer2

The default port for HiveServer2 is 10000.

The following table lists HiveServer2 authentication mechanisms with the connection parameters required in the JDBC connection string. For a complete list of the JDBC connection string parameters, refer to the next section, Hive JDBC Connection String Parameters.

HiveServer2 Authentication	Connection Requirements
No Authentication	**Connection String:** `jdbc:hive2://<hs2_hostname>:10000<database>;` You must enter a valid user name. **For encryption, JDBC requires a truststore and an optional truststore password.** • **Connection String with Encryption**: `jdbc:hive2://<hs2_hostname>:10000/<database>;` You must enter a valid user name.`ssl=true;sslTrustStore=<path-to-truststore>;sslTrustStorePassword=<password>` • **Connection String with Encryption (truststore passed in JVM arguments)**: `jdbc:hive2://<hs2_hostname>:<port>/<database>;ssl=true` **Note:** Prior to connecting to an application that uses JDBC, such as Beeline, you can run the following command to pass the truststore parameters as Java arguments: `export HADOOP_OPTS="-Djavax.net.ssl.trustStore=<path-to-trust-store-file> -Djavax.net.ssl.trustStorePassword=<password>"`
MapR-SASL (included as part of the secure by default configuration)	**Connection String**: `jdbc:hive2://<hs2_hostname>:10000/<database>;auth=maprsasl;ssl=true;` MapR-SASL encryption is enabled by default. For more information, see Configuring JDBC Connection String with SSL Encryption Enabled or Disabled. **Note:** MapR-SASL is not supported for Hive in HTTP mode. **Connection for Java Application**: Use the `-D` flag to append the JVM argument: `-Dhadoop.login=maprsasl`.

last-updated: Apr 11, 2022

HiveServer2 Authentication	Connection Requirements
PAM	**Connection String**: `jdbc:hive2://<hs2_hostname>:10000/<database>;user=<user>;password=<password>`
PAM + SSL (included as part of the secure by default configuration)	**Connection String**: `jdbc:hive2://<hs2_hostname>:10000/<database>;ssl=true;user=<user>;password=<password>`. For more information, see Configuring JDBC Connection String with SSL Encryption Enabled or Disabled.
Kerberos	**Connection String**: `jdbc:hive2://<hostname>:10000/default;principal=mapr/<FQDN@REALM>` **Connection for Java Application**: Use the `-D` flag to append the JVM argument: `-Dhadoop.login=hybrid` **Note:** The client nodes must also have a Kerberos ticket and be configured to connect to HiveServer2 to use Kerberos.
LDAP	**Connection String:** `jdbc:hive2://<hs2_hostname>:10000/<database>;user=<ldap_user>;password=<ldap_password>`
ZooKeeper	**Connection String**: `jdbc:hive2://<hostname>:<port>,<hostname>:<port>/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2` **Example**: `hive --service beeline -u 'jdbc:hive2://zookeeper1.com:5181,zookeeper2.com:5181,zookeeper3.com:5181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2' -n mapr -p`

Hive JDBC Connection String Parameters

The following example shows a common Hive JDBC connection string:

```
jdbc:hive2://zookeeper_quorum|hs2_host:port/[db]
[;principal=<hs2_principal>/<hs2_host>|_HOST@<KDC_REALM>]
[;transportMode=binary|http][;httpPath=<http_path>]
[;serviceDiscoveryMode=zookeeper;zooKeeperNamespace=<zk_namespace>]
[;auth=maprsasl][;ssl=true|false][;sslKeyStore=<key_store_path>]
[;keyStorePassword=<key_store_password][;sslTrustStore=<trust_store_path>]
[;trustStorePassword=<trust_store_password>][;twoWay=true|false]
```

The following table lists all the Hive JDBC connection string parameters with default values where applicable:

JDBC Parameter	Default	Comment
zookeeper_quorum		Zookeeper quorum. Used only if HA mode for HiveServer2 is enabled.

last-updated: Apr 11, 2022

hs2_host		The hostname of the node with an active HS2 server running.
port	10000/10001	HiveServer2 port. Defaults to 10000 in binary mode. Defaults to 10001 in HTTP transport mode.
[db]	default	The database name to which you want to connect.
[;principal=<hs2_principal>/<hs2_host>|_HOST@<KDC_REALM>]		Kerberos principal. Used with Kerberos security only.
[;transportMode=binary|http]	binary	HS2 uses a TThreadPoolServer (from Thrift) for TCP (binary) mode, or a Jetty server for the HTTP mode. HTTP mode is required when a proxy is needed between the client and server, for example, for load balancing or security reasons.
[;httpPath=<http_path>]	cliservice or /	The corresponding HTTP endpoint. The default value is cliservice or /. See conf hive.server2.thrift.http.path
[;serviceDiscoveryMode=zookeeper;zooKeeperNamespace=<zk_namespace>]		<zk_namespace> is the parent node in ZooKeeper used by HiveServer2 when supporting dynamic service discovery.
[;auth=maprsasl]		Used with MapR SASL security.
[;ssl=true|false]	false	Used to enable SSL encryption.
[;sslKeyStore=<key_store_path>]	Default value is read from $MAPR_HOME/conf/ssl-client.xml	This parameter only takes effect when ssl=true. Path is the path to the keystore.
[;keyStorePassword=<key_store_password]	Default value is read from $MAPR_HOME/conf/ssl-client.xml	This param will take effect only when ssl=true. Keystore password.
[;sslTrustStore=<trust_store_path>]	Default value is read from $MAPR_HOME/conf/ssl-client.xml	This param will take effect only when ssl=true. Path is the path to the truststore.
[;trustStorePassword=<trust_store_password>]	Default value is read from $MAPR_HOME/conf/ssl-client.xml	This parameter only takes effect when ssl=true. Password is the truststore password.

[;twoWay=true|false]		HIVE-10447 enabled the JDBC driver to support 2-way SSL in HTTP mode. Currently, HiveServer2 does not support 2-way SSL. This features is useful when there is an intermediate server, such as Knox, which requires the client to support 2-way SSL.

Beeline Examples

This page shows examples for connecting to HiveServer2 using Beeline.

The following table is a guide for interpreting the examples on this page. In the examples, replace the variables (information in brackets) with your site-specific values. Be sure to remove the brackets when you insert your information:

Variable	Description
<hs2_hostname>	The name of the host where HiveServer2 is installed.
<database>	The database name to connect to.
<username>	The JDBC username.
<password>	The password for the JDBC user.
<FQDN@realm>	Fully qualified domain name & Kerberos realm.

Using Beeline with no Encryption and no Authentication

```
hive --service beeline
Beeline version 2.3.3-mapr-1901 by Apache Hive
beeline> !connect jdbc:hive2://<hs2_hostname>:10000/<database>;
Connecting to jdbc:hive2://<hs2_hostname>:10000/<database>;
Enter username for jdbc:hive2://<hs2_hostname>:10000/<database>;: <username>
0: jdbc:hive2://<hs2_hostname>:10000/<database>
```

Using Beeline with Encryption and no Authentication

```
hive --service beeline
Beeline version 2.3.3-mapr-1901 by Apache Hive
beeline> !connect jdbc:hive2://<hs2_hostname>:10000/
<database>;ssl=true;sslTrustStore=truststore.jks;sslTrustStorePassword=tsp
Connecting to jdbc:hive2://<hs2_hostname>:10000/
<database>;ssl=true;sslTrustStore=truststore.jks;sslTrustStorePassword=tsp
Enter username for jdbc:hive2://<hs2_hostname>:10000/
<database>;ssl=true;sslTrustStore=truststore.jks;sslTrustStorePassword=tsp:
<username>
0: jdbc:hive2://<hs2_hostname>:10000/<database>
```

Connecting to HiveServer2 with MapR-SASL Authentication

```
hive --service beeline
Beeline version 2.3.3-mapr-1901 by Apache Hive
beeline> !connect jdbc:hive2://<hs2_hostname>:10000/
<database>;auth=maprsasl;ssl=true
Connecting to jdbc:hive2://<hs2_hostname>:10000/
<database>;auth=maprsasl;ssl=true
19/01/31 12:15:33 [main]: WARN maprsasl.MaprSaslClient: SASL
```


last-updated: Apr 11, 2022

```
Server qopProperty: auth-confis different from Client:
auth-conf,auth-int,auth.Using Server one
Connected to: Apache Hive (version 2.3.3-mapr-1901)
Driver: Hive JDBC (version 2.3.3-mapr-1901)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://<hs2_hostname>:10000/<database>
```

Starting from EEP 6.0.0, with secure by default configuration, it is a default connection string for a secure cluster. For more information, see Configuring JDBC Connection String with SSL Encryption Enabled or Disabled.

Using Beeline with PAM Authentication

```
hive --service beeline
Beeline version 2.3.3-mapr-1901 by Apache Hive
beeline> !connect jdbc:hive2://<hs2_hostname>:10000/<database>;
Connecting to jdbc:hive2://<hs2_hostname>:10000/<database>;
Enter username for jdbc:hive2://<hs2_hostname>:10000/<database>;: <username>
Enter password for jdbc:hive2://<hs2_hostname>:10000/<database>;:
<password>
Connected to: Apache Hive (version 2.3.3-mapr-1901)
Driver: Hive JDBC (version 2.3.3-mapr-1901)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://<hs2_hostname>:10000/<database>
```

Connecting to HiveServer2 with ZooKeeper

```
hive --service beeline -u
'jdbc:hive2://
zookeeper1.com:5181,zookeeper2.com:5181,zookeeper3.com:5181/;serviceDiscover
yMode=zooKeeper;zooKeeperNamespace=hiveserver2' -n mapr -p
```

Connecting to HiveServer2 with PAM Authentication and SSL Encryption

```
hive --service beeline
Beeline version 2.3.3-mapr-1901 by Apache Hive
beeline> !connect jdbc:hive2://<hs2_hostname>:10000/<database>;ssl=true;
Connecting to jdbc:hive2://<hs2_hostname>:10000/<database>;ssl=true;
Enter username for jdbc:hive2://<hs2_hostname>:10000/<database>: <username>
Enter password for jdbc:hive2://<hs2_hostname>:10000/<database>: <password>
Connected to: Apache Hive (version 2.3.3-mapr)
Driver: Hive JDBC (version 2.3.3-mapr)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://<hs2_hostname>:10000/<database>
```

Starting from EEP 6.0.0, with secure-by-default configurations, the default connection string is for a secure cluster. For more information, see Configuring JDBC Connection String with SSL Encryption Enabled or Disabled.

Using Beeline with Kerberos

Beeline must pass the Kerberos principal for HiveServer2 in the JDBC connection string. The connection strings you pass to Beeline must use the principal name that you configured for HiveServer2.

The following example shows a sample Beeline authentication with Kerberos:

```
hive --service beeline
Beeline version 2.3.3-mapr-1901 by Apache Hive
beeline> !connect jdbc:hive2://<hs2_hostname>:10000/
```

```
<database>;principal=mapr<FQDN@REALM>
Connecting to jdbc:hive2://<hs2_hostname>:10000/<database>;principal=mapr/
<FQDN@REALM>
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://<hs2_hostname>:10000/def>
Connected to: Apache Hive (version 2.3.3-mapr)
Driver: Hive JDBC (version 2.3.3-mapr)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://<hs2_hostname>:10000/<database>
```

Using Beeline with Encryption but no Authentication (truststore parameters passed as JVM arguments)

```
Hive --service beeline
Beeline version 2.3.3-mapr-1901 by Apache Hive
beeline> !connect jdbc:hive2://<hs2_hostname>:1000/<database>;ssl=true
Connecting to jdbc:hive2://<hs2_hostname>:10000/<database>;ssl=true
Enter username for jdbc:hive2://<hs2_hostname>:10000/<database>;ssl=true:
<username>
Enter password for jdbc:hive2://<hs2_hostname>:10000/<database>;ssl=true:
<password>
Connected to: Apache Hive (version 2.3.3-mapr)
Driver: Hive JDBC (version 2.3.3-mapr)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://<hs2_hostname>:10000/<database>
```

Generating a Kerberos Ticket

Use the `kinit` utility to generate the ticket and then use `klist` to verify that a ticket exists.

```
# kinit <username>/<FQDN@REALM>
# klist

Credentials cache: API:501:9
        Principal: username/<FQDN@REALM>
    Cache version: 0

Server: krbtgt/<FQDN@REALM>
Client: username/<FQDN@REALM>
Ticket etype: aes128-cts-hmac-sha1-96
Ticket length: 256
Auth time:  Jun 11 10:01:48 2014
End time:   Jun 12 18:01:34 2014
Renew till: Jun 18 10:01:48 2014
Ticket flags: pre-authent, initial, renewable, forwardable
Addresses: addressless
```

Configuring JDBC Connection String with SSL Encryption Enabled or Disabled

You can configure a JDBC connection string with SSL encryption enabled or disabled.

SSL encryption to HiveServer2 is enabled (`hive.server2.use.SSL=true`)

The following table describes the JDBC connection string when SSL encryption is enabled between the Hive client and HiveServer2.

last-updated: Apr 11, 2022

Table

Authentication Type	JDBC Parameter	Example
PAM	`ssl=true`	`jdbc:hive2://<hostname>:10000/default;ssl=true <login> <password>`
MapR-SASL	`ssl=true`	`jdbc:hive2://<hostname>:10000/default;auth=maprsasl;ssl=true`
Kerberos	`ssl=true`	`jdbc:hive2://<hostname>:10000/default;principal=<user/fqdn@EXAMPLE.COM>;ssl=true`

SSL encryption to HiveServer2 is disabled (`hive.server2.use.SSL=false`)

The following table describes the JDBC connection string when SSL encryption is disabled between the Hive client and HiveServer2.

Table

Authentication Type	JDBC Parameter	Example
PAM	--	`jdbc:hive2://<hostname>:10000/default; <login> <password>`
MapR-SASL	--	`jdbc:hive2://<hostname>:10000/default;auth=maprsasl`
Kerberos	--	`jdbc:hive2://<hostname>:10000/default;principal=<user/fqdn@EXAMPLE.COM`

Using ODBC to Connect to HiveServer2

This section contains details about setting up and using the ODBC Connector for Hive.

Before You Begin

The MapR Hive ODBC Connector is an ODBC driver for Apache Hive 0.7.0 and later that complies with the ODBC 3.52 specification. You can download the Hive ODBC connector from https://package.mapr.hpe.com/tools/MapR-ODBC/MapR_Hive/. After downloading the driver, refer to documentation for Hive ODBC Driver to install and configure the driver.

The Hive ODBC Driver supports the following Advanced Options:

- Enable Auto Reconnect
- Driver Config Take Precedence
- Fast SQL Prepare
- Get Tables With Query
- Invalid Session Auto Recover
- Show System Table

- Socket Timeout
- Default String Column Length
- Rows Fetched Per Block
- Use Native Query

To use the ODBC driver, configure a *Data Source Name* (DSN), a definition that specifies how to connect to Hive. DSNs are typically managed by the operating system and may be used by multiple applications. Some applications do not use DSNs. You will need to refer to your particular application's documentation to understand how it connects using ODBC.

The standard query language for ODBC is SQL. HiveQL, the standard query language for Hive, includes a subset of ANSI SQL-92. Applications that connect to Hive using ODBC may need queries altered if the queries use SQL features that are not present in Hive. Applications that use SQL will recognize HiveQL, but might not provide access to HiveQL-specific features such as multi-table insert.

Please refer to the Hive Language Manual for up-to-date information on HiveQL.

The SQL Connector

The SQL Connector feature translates standard SQL-92 queries into equivalent HiveQL queries. The SQL Connector performs syntactical translations and structural transformations. For example:

- Quoted Identifiers: When quoting identifiers, HiveQL uses back quotes (`), while SQL uses double quotes ("). Even when a driver reports the back quote as the quote character, some applications still generate double-quoted identifiers.
- Table Aliases: HiveQL does not support the AS keyword between a table reference and its alias.
- The `JOIN`, `INNER JOIN`, and `CROSS JOIN` SQL syntaxes are translated to the HiveQL `JOIN` syntax.
- SQL `TOP N` queries are transformed to HiveQL `LIMIT` queries.

Hive ODBC Connector on Linux

System Requirements

- The 32-bit and 64-bit version of the following operating systems:
  - Red Hat® Enterprise Linux® (RHEL) 6 or 7
  - CentOS 6 or 7
  - SUSE Linux Enterprise Server (SLES) 11 or 12
  - Debian 8 or 9
  - Ubuntu 14.04, 16.04, or 18.04
- 45 MB of available disk space.
- An installed ODBC driver manager:
  - iODBC 3.52.7 or above (OR)
  - unixODBC 2.2.12 or above

The MapR ODBC Driver with SQL Connector for Apache Hive requires a Hadoop cluster with the Hive service installed and running. The MapR ODBC Driver with SQL Connector for Apache Hive is suitable for use with all versions of Hive. Download the ODBC connector from the following location: MapR_Hive.

The RPM files are applicable for:

- Red Hat® Enterprise Linux® (RHEL) 6 or 7
- CentOS 6 or 7
- SUSE Linux Enterprise Server (SLES) 11 or 12

The DEB files are applicable for:

- Debian 8 or 9
- Ubuntu 14.04, 16.04, or 18.04

The latest version of the Hive ODBC connector is at version 2.6.1.1001.

Install the Hive ODBC Connector on Linux

The MapR ODBC Driver with SQL Connector for Apache Hive driver files are installed in the following directories:

- `/opt/mapr/hiveodbc/ErrorMessages` – Error messages files directory
- `/opt/mapr/hiveodbc/Setup` – Sample configuration files directory
- `/opt/mapr/hiveodbc/lib/32` – 32-bit shared libraries directory
- `/opt/mapr/hiveodbc/lib/64` – 64-bit shared libraries directory

To install the MapR ODBC Driver with SQL Connector for Apache Hive:

1. Log in as the `root` user.
2. Use RPM to install the rpm package corresponding to your Linux distribution:
   - 32-bit
   - 64-bit

   The MapR ODBC Driver with SQL Connector for Apache Hive depends on the following resources:
   - cyrus-sasl-2.1.22-7 or later
   - cyrus-sasl-gssapi-2.1.22-7 or later
   - cyrus-sasl-plain-2.1.22-7 or later

   If the package manager in your Linux distribution cannot resolve the dependencies automatically when installing the driver, download and manually install the packages required by the version of the driver that you want to install.

Configure the Hive ODBC Connector Driver on Linux

The LD_LIBRARY_PATH environment variable must include the paths to the:

- Libraries for the installed ODBC driver manager
- Shared libraries for the MapR ODBC Driver with SQL Connector for Apache Hive

Important: The Linux version of the driver bundles together functionality for both 32-bit and 64-bit environments. Do not include the paths to both 32- and 64-bit shared libraries in LD_LIBRARY PATH at the same time. Include only the path to the shared libraries corresponding to the driver matching the bitness of the client application used. For example, if you are using a 64-bit client application and ODBC driver manager libraries are installed in `/usr/local/lib`, then set LD_LIBRARY_PATH as follows:

```
export LD_LIBRARY_PATH=/usr/local/lib:/opt/mapr/hiveodbc/lib/64
```

For more information about how to set environment variables permanently, refer to your Linux shell documentation.

Configuring ODBC Connections for Linux

Files

ODBC driver managers use configuration files to define and configure ODBC data sources and drivers. By default, the following configuration files residing in the user's home directory are used:

- `.odbc.ini` – The file used to define ODBC data sources (required)
- `.odbcinst.ini` – The file used to define ODBC drivers (optional)
- `.mapr.hiveodbc.ini` – The file used to configure the MapR ODBC Driver with SQL Connector for Apache Hive (required)

Sample Files

The driver installation contains the following sample configuration files in the Setup directory:

- `odbc.ini`
- `odbcinst.ini`
- `mapr.hiveodbc.ini`

The names of the sample configuration files do not begin with a period (.) so that they will appear in directory listings by default. A filename beginning with a period (.) is hidden. For odbc.ini and odbcinst.ini, if the default location is used, then the filenames must begin with a period (.). For mapr.hiveodbc.ini, the filename must begin with a period (.) and must reside in the user's home directory. If the configuration files do not already exist in the user's home directory, then the sample configuration files can be copied to that directory and renamed. If the configuration files already exist in the user's home directory, then the sample configuration files should be used as a guide for modifying the existing configuration files.

Configuring the Environment

By default, the configuration files reside in the user's home directory. However, two environment variables, ODBCINI and ODBCSYSINI, can be used to specify different locations for the `odbc.ini` and `odbcinst.ini` configuration files. Set ODBCINI to point to your `odbc.ini` file. Set ODBCSYSINI to point to the directory containing the odbcinst.ini file. For example, if your `odbc.ini` file is located in `/etc` and your `odbcinst.ini` file is located in `/usr/local/odbc`, then set the environment variables as follows:

```
export ODBCINI=/etc/odbc.ini export ODBCSYSINI=/usr/local/odbc
```

```
export ODBCINI=/etc/odbc.ini export ODBCSYSINI=/usr/local/odbc
```

For version 2.1.8 and above, you must also set MAPRHIVEINI to point to the mapr.hiveodbc.ini file in the user's home directory:

```
export MAPRHIVEINI=/etc/.mapr.hiveodbc.ini
```


last-updated: Apr 11, 2022

For version 2.1.5 and below, you set MAPRINI to point to the mapr.hiveodbc.ini file in the user's home directory:

```
export MAPRINI=/<user_home>/.mapr.hiveodbc.ini
```

Configuring the `odbc.ini` File

ODBC Data Sources are defined in the odbc.ini configuration file. The file is divided into several sections:

- [ODBC] is optional and used to control global ODBC configuration, such as ODBC tracing.
- [ODBC Data Sources] is required, listing DSNs and associating DSNs with a driver.
- A section having the same name as the data source specified in the [ODBC Data Sources] section is required to configure the data source.

Here is an example `odbc.ini` configuration file for Linux:

```
[ODBC Data Sources]
Sample MapR Hive DSN 32=MapR Hive ODBC Driver 32-bit
[Sample Mapr Hive DSN 32]
Driver=/opt/mapr/hiveodbc/lib/32/libmaprhiveodbc32.so
HOST=MyHiveServer
PORT=10000
```

To create a data source:

1. Open the `.odbc.ini` configuration file in a text editor.
2. Add a new entry to the [ODBC Data Sources] section. Type the data source name (DSN) and the driver name.
3. To set configuration options, add a new section having a name matching the data source name (DSN) you specified in step 2. Specify configuration options as keyvalue pairs.
4. Save the `.odbc.ini` configuration file.

Note: You can set configuration options in your `odbc.ini` and `.mapr.hiveodbc.ini` files. Configuration options set in a `.mapr.hiveodbc.ini` file apply to all connections, whereas configuration options set in an `odbc.ini` file are specific to a connection. Configuration options set in `odbc.ini` take precedence over configuration options set in `.mapr.hiveodbc.ini`.

Configuring the `odbcinst.ini` File

ODBC Drivers are defined in the odbcinst.ini configuration file. The configuration file is optional because drivers can be specified directly in the odbc.ini configuration file. The odbcinst.ini file is divided into the following sections:

- [ODBC Drivers] lists the names of all the installed ODBC drivers.
- section having the same name as the driver name specified in the [ODBC Drivers] section lists driver attributes and values.

Here is an example odbcinst.ini file for Linux:

```
[ODBC Drivers]
Mapr Hive ODBC Driver 32-bit=Installed
Mapr Hive ODBC Driver 64-bit=Installed
[Mapr Hive ODBC Driver 32-bit]
Description=Mapr Hive ODBC Driver (32-bit)
```

```
Driver=/opt/mapr/hiveodbc/lib/32/libmaprhiveodbc32.so
[Mapr Hive ODBC Driver 64-bit]
Description=Mapr Hive ODBC Driver (64-bit)
Driver=/opt/mapr/hiveodbc/lib/64/libmaprhiveodbc64.so
```

To define a driver:

1. Open the `.odbcinst.ini` configuration file in a text editor.
2. Add a new entry to the [ODBC Drivers] section. Type the driver name, and then type **=Installed**

   Note: Assign the driver name as the value of the Driver attribute in the data source definition instead of the driver shared library name.
3. In `.odbcinst.ini`, add a new section having a name matching the driver name you typed in step 2, and then add configuration options to the section based on the sample `odbcinst.ini` file provided with the MapR ODBC Driver with SQL Connector for Apache Hive in the Setup directory. Specify configuration options as keyvalue pairs.
4. Save the `.odbcinst.ini` configuration file.

Configuring the `mapr.hiveodbc.ini` File

To configure the MapR ODBC Driver with SQL Connector for Apache Hive to work with your ODBC driver manager:

1. Open the `.mapr.hiveodbc.ini` configuration file in a text editor.
2. Edit the DriverManagerEncoding setting. The value usually must be UTF-16 or UTF-32, depending on the ODBC driver manager you use. iODBC uses UTF-32 and unixODBC uses UTF-16. Consult your ODBC Driver Manager documentation for the correct setting to use.
3. Edit the ODBCInstLib setting. The value is the name of the ODBCInst shared library for the ODBC driver manager you use. The configuration file defaults to the shared library for iODBC. In Linux, the shared library name for iODBC is `libiodbcinst.so`.

   Note: Consult your ODBC driver manager documentation for the correct library to specify. You can specify an absolute or relative filename for the library. If you intend to use the relative filename, then the path to the library must be included in the library path environment variable. In Linux, the library path environment variable is named `LD_LIBRARY_PATH`.
4. Save the `.mapr.hiveodbc.ini` configuration file.

Configuring Authentication

You can configure the following types of authentication:

- No authentication
- User name
- User name and password
- Kerberos

When `hive.server2. authentication` is set to KERBEROS, then you must configure your connection to use Kerberos.

To find out the authentication setting your Hive Server 2 is set to use, review the following properties in the `hive-site.xml` file:

last-updated: Apr 11, 2022

- `hive.server2.authentication`
- `hive.server2.enable.doAs`

Using No Authentication

To use no authentication, set the AuthMech configuration key for the DSN to 0.

Using User Name

To configure User Name authentication:

1. Set the AuthMech configuration key for the DSN to 2.
2. Set the UID key to the appropriate user name recognized by the Hive server.

Using User Name and Password

To configure User Name and Password authentication:

1. Set the AuthMech configuration key for the DSN to 3.
2. Set the UID key to the appropriate user name recognized by the Hive server.
3. Set the PWD key to the password corresponding to the user name you provided in step 2.

Using Kerberos

To configure Kerberos authentication:

1. Set the H2SAuthMech configuration key for the DSN to 1.
2. If your Kerberos setup does not define a default realm or if the realm of your Hive server is not the default, then set the appropriate realm using the HS2KrbRealm key.
3. Set the HS2HostFQDN key to the fully qualified domain name of the Hive Server 2 host.
4. Set the HS2KrbServiceName key to the service name of the Hive Server 2 host.

Hive ODBC Connector on Windows

There are versions of the connector for 32-bit and 64-bit applications. The 64-bit version of the connector works only with 64-bit DSNs; the 32-bit connector works only with 32-bit DSNs. Because 64-bit Windows machines can run both 64-bit and 32-bit applications, install both versions of the connector in order to set up DSNs to work with both types of applications. If both the 32-bit connector and the 64-bit connector are installed, you must configure DSNs for each independently, in their separate Data Source Administrators.

Install the Hive ODBC Connector on Windows

To use MapR Hive ODBC Connector on Windows requires:

- Windows® 7 Professional or Windows® 2008 R2. Both 32 and 64-bit editions are supported.
- The Microsoft Visual C++ 2010 Redistributable Package (runtimes required to run applications developed with Visual C++ on a computer that does not have Visual C++ 2010 installed.)
- A Hadoop cluster with the Hive service installed and running. You should find out from the cluster administrator the hostname or IP address for the Hive service and the port that the service is running on. (The default port for Hive is 10000.)

1. Run the installer to get started:

- To install the 64-bit connector, download and run the following package:

```
https://package.mapr.hpe.com/tools/MapR-ODBC/MapR_Hive/
MapRHive_odbc_2.6.1.1001/MapRHiveODBC64.msi
```

- To install the 32-bit connector, download and run the following package:

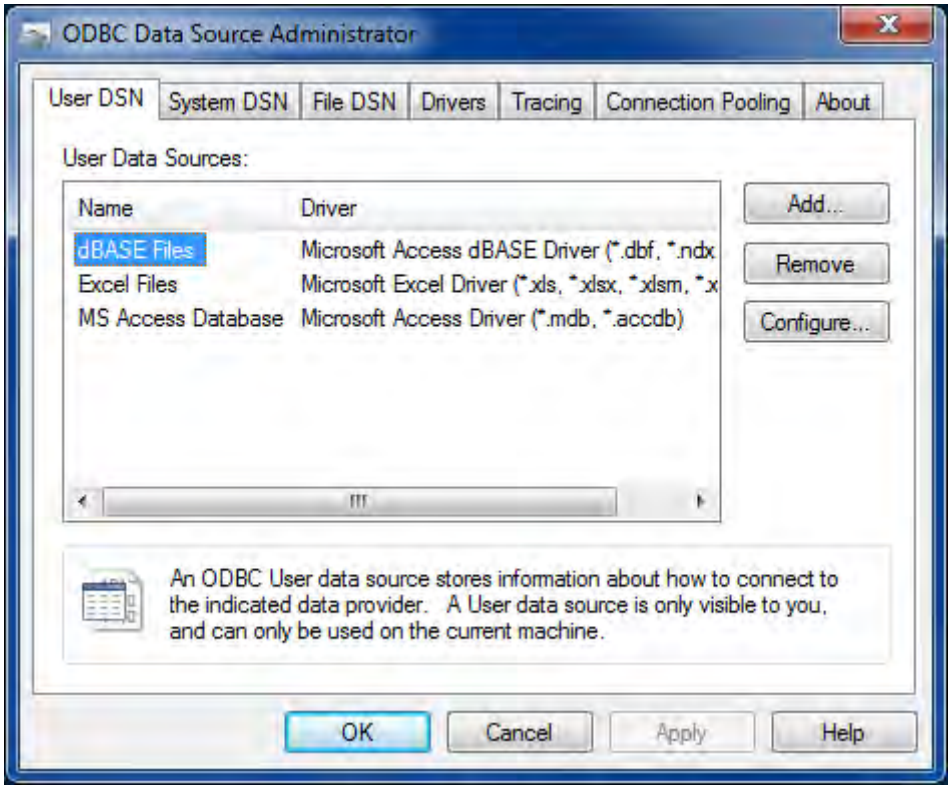
```
https://package.mapr.hpe.com/tools/MapR-ODBC/MapR_Hive/
MapRHive_odbc_2.6.1.1001/MapRHiveODBC32.msi
```

2. Accept the license agreement.
3. Select an installation folder.
4. On the Information window, click **Next**.
5. On the Completing... window, click **Finish.**
6. Install a DSN corresponding to your Hive server.

Configure Hive ODBC Connections on Windows

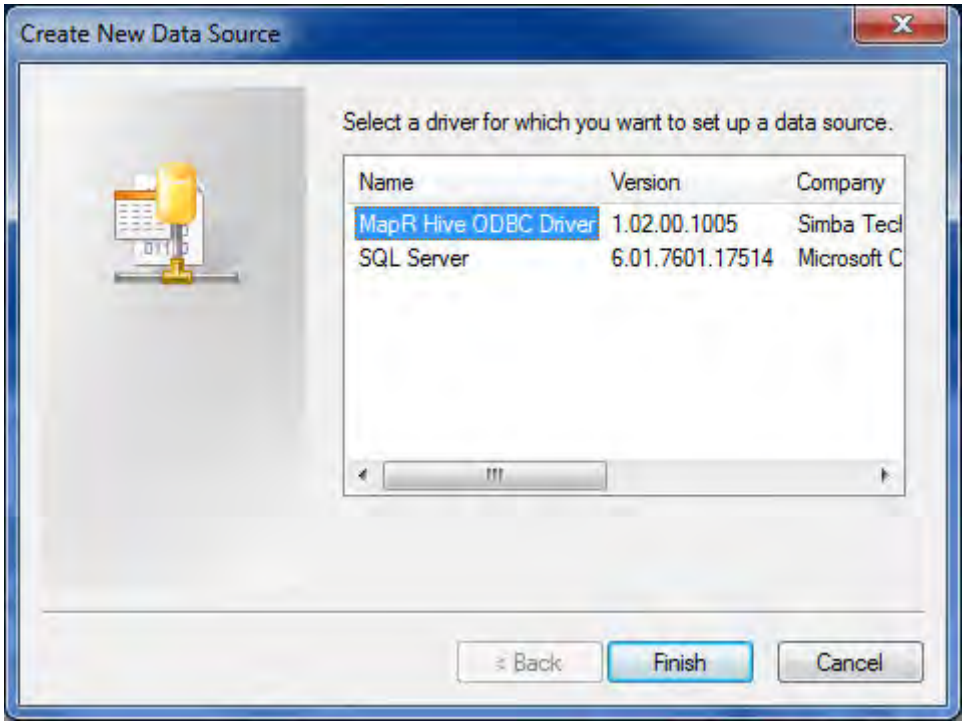
To create a Data Source Name (DSN)

1. Open the Data Source Administrator from the Start menu. Example: **Start > MapR Hive ODBC Driver 2.0 > 64-Bit ODBC Driver Manager**
2. On the **User DSN** tab click **Add** to open the Create New Data Source dialog.



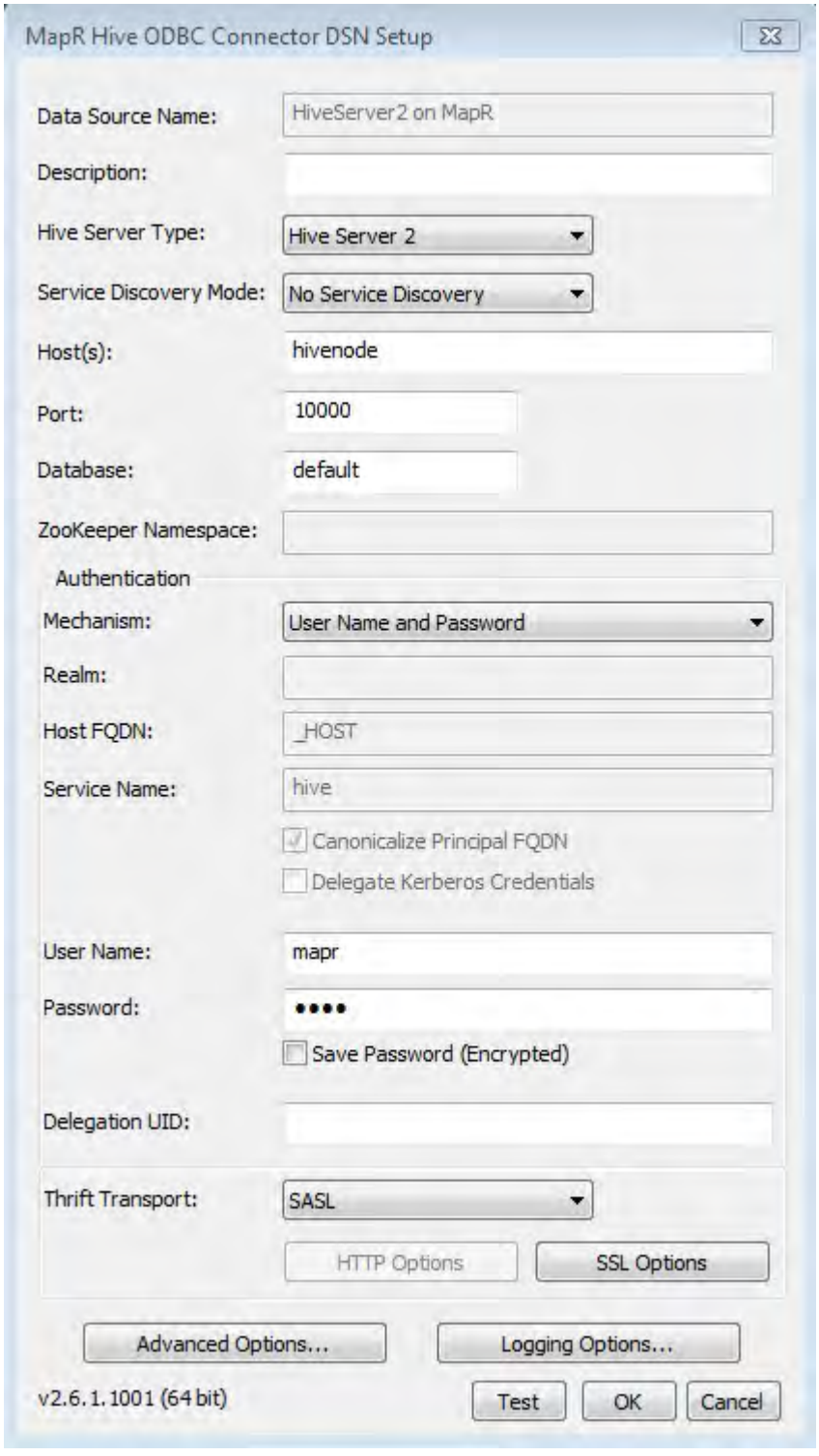
3. Select **MapR Hive ODBC Connector** and click **Finish** to open the Hive ODBC Driver DSN Setup window.

last-updated: Apr 11, 2022



4. Enter the connection information for the Hive instance:

last-updated: Apr 11, 2022

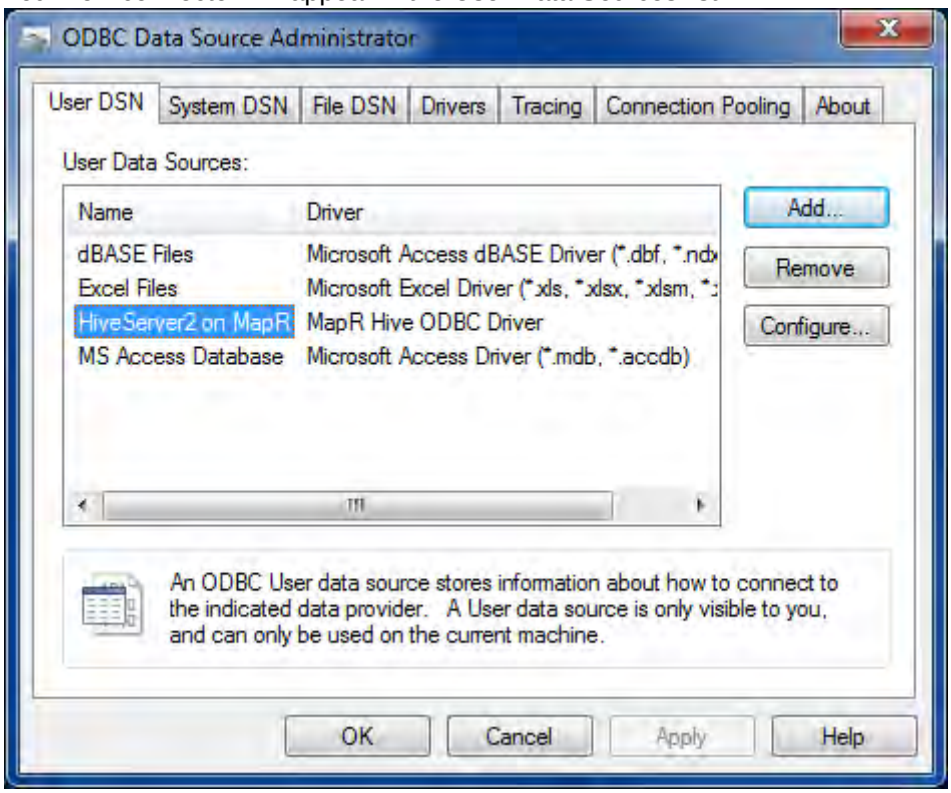


- **Data Source Name** — Specify a name for the DSN.
- **Description** — Enter an optional description for the DSN.
- **Host** — Enter the hostname or IP of the server running HiveServer1 or HiveServer2.
- **Port** — Enter the listening port for the Hive service.

last-updated: Apr 11, 2022

- **Database** — Leave as `default` to connect to the default Hive database, or enter a specific database name.
- **Hive Server Type:** — Set to HiveServer1 or HiveServer2.
- **Authentication** — If you are using HiveServer2, set the following.
  - **Mechanism:** — Set to the authentication mechanism you're using. The MapR ODBC driver supports user name, user name and password, username and password over SSL authentication, and Kerberos.
  - **User Name:** — Set the user to run queries as.
  - **Password:** — The user's password, if your selected authentication mechanism requires one.

5. Optionally, click **Test** to test the connection.
6. Click OK.
   Your new connector will appear in the User Data Sources list.



For steps to apply custom configurations, see Hive ODBC Driver.

Hive ODBC Connector on Mac OS X

System Requirements

- Mac OS X version 10.6.8 or later
- 100 MB of available disk space
- iODBC 3.52.7 or above
- unixODBC 2.2.12 or above

last-updated: Apr 11, 2022

The MapR ODBC Driver with SQL Connector for Apache Hive requires a Hadoop cluster with the Hive service installed and running. The MapR ODBC Driver with SQL Connector for Apache Hive is suitable for use with all versions of Hive. The driver supports both 32- and 64-bit client applications.

Download the MacOS Hive ODBC connector from https://package.mapr.hpe.com/tools/MapR-ODBC/MapR_Hive/MapRHive_odbc_2.6.1.1001/MapRHiveODBC.dmg.

Installation

The MapR ODBC Driver with SQL Connector for Apache Hive driver files are installed in the following directories:

- `/opt/mapr/hiveodbc/ErrorMessages` – Error messages files directory
- `/opt/mapr/hiveodbc/Setup` – Sample configuration files directory
- `/opt/mapr/hiveodbc/lib/universal` – Binaries directory

To install the MapR ODBC Driver with SQL Connector for Apache Hive:

1. Double-click to mount the `MapRHiveODBC.dmg` disk image.
2. Double-click `MapRHiveODBC.pkg` to run the Installer.
3. Follow the instructions in the Installer to complete the installation process.
4. When the installation completes, click **Close**.

Configuration

Setting the DYLD_LIBRARY_PATH Environment Variable

The `DYLD_LIBRARY_PATH` environment variable must include the paths to:

- Installed ODBC driver manager libraries
- Installed MapR ODBC Driver with SQL Connector for Apache Hive shared libraries

For example, if ODBC driver manager libraries are installed in `/usr/local/lib`, then set `DYLD_LIBRARY_PATH` as follows:

```
export DYLD_LIBRARY_PATH=/usr/local/lib/opt/mapr/hiveodbc/lib/universal
```

Refer to your Mac OS X shell documentation for details on how to set environment variables permanently.

Configure Hive ODBC Connections on Mac OS X

See Configuring ODBC Connections for Linux for details on creating ODBC connections.

Hive ODBC Connector License and Copyright Information

Third Party Trademarks

ICU License - ICU 1.8.1 and later

COPYRIGHT AND PERMISSION NOTICE

Copyright (c) 1995-2010 International Business Machines Corporation and others

All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above

last-updated: Apr 11, 2022

copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

All trademarks and registered trademarks mentioned herein are the property of their respective owners.

OpenSSL

Copyright (c) 1998-2008 The OpenSSL Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org/)"

4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org.

5. Products derived from this software may not be called "OpenSSL" nor may "OpenSSL" appear in their names without prior written permission of the OpenSSL Project.

6. Redistributions of any form whatsoever must retain the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)"

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Expat

Copyright (c) 1998, 1999, 2000 Thai Open Source Software Center Ltd

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the ""Software""), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of

the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED ""AS IS"", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NOINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE."

Apache Hive

Copyright 2008-2011 The Apache Software Foundation.

Apache Thrift

Copyright 2006-2010 The Apache Software Foundation.

Configuring Encryption for ODBC Connection

Explains how to configure SSL encryption between ODBC connection to Hiveserver2 on non-secure cluster.

Hive uses cyrus-sasl-plain package for ODBC connection.

1. Generate `ssl_keystore/ssl_truststore` by running the following command:

   ```
   sudo bash /opt/mapr/server/manageSSLKeys.sh create -ug mapr:mapr
   ```

   Important: Make a note of the `CN=HOST_NAME` parameter in the output.

2. Configure SSL for Hive as described in Configure Encryption without Authentication on page 3448.

3. Generate the `.pem` file. To generate:

   a) Verify that `ssl_keystore` and `ssl_truststore` are present on the system.

   ```
   cd /opt/mapr/conf
   ll *ssl*store*
   ```

   If `ssl_keystore` and `ssl_truststore` are not present, then generate them.

   b) Generate `.pem` file using `<ssl-keystore-password>` password.

   ```
   keytool -importkeystore -srckeystore ssl_keystore -destkeystore
   ssl_keystore.p12 -srcstoretype jks -deststoretype pkcs12
   ```

   c) Verify that the `ssl_keystore.p12` and `ssl_keystore.pem` files are created.

   For example:

   ```
   openssl pkcs12 -in ssl_keystore.p12 -out ssl_keystore.pem
   openssl x509 -text -in ssl_keystore.pem
   ```

4. Configure SSL for ODBC driver by making the following changes in the `/etc/odbc.ini`, `/etc/odbcinst.ini`, and `/etc/mapr.hiveodbc.init` files. That is, in the:

   - `/etc/odbc.ini` file:

a. Replace <HOST_NAME> with the host name.

b. Set the value for `TrustedCerts` to path to `ssl_keystore.pem` file.

c. Add the following to the file:

```
[ODBC Data Sources]
Sample MapR Hive DSN=Hive Hive ODBC Driver 64-bit
[Hive]
Driver=/opt/mapr/hiveodbc/lib/64/libmaprhiveodbc64.so
HOST=<HOST_NAME>
PORT=10000
SSL=1
CAIssuedCertNamesMismatch=1
TrustedCerts=/opt/mapr/conf/ssl_keystore.pem
AuthMech=4
```

- `/etc/odbcinst.ini` file, add the following:

```
[ODBC Drivers]
Mapr Hive ODBC Driver=Installed
[Mapr Hive ODBC Driver 64-bit]
Description=Mapr Hive ODBC Driver (64-bit)
Driver=/opt/mapr/hiveodbc/lib/64/libmaprhiveodbc64.so
```

- `etc/mapr.hiveodbc.ini` file, add the following:

```
[Driver]
ErrorMessagesPath=/opt/mapr/hiveodbc/ErrorMessages/
LogLevel=0
LogPath=
SwapFilePath=/tmp
```

Sample /etc/odbc.ini file

```
[ODBC Data Sources]
Sample MapR Hive DSN=Hive Hive ODBC Driver 64-bit
[Hive]
Driver=/opt/mapr/hiveodbc/lib/64/libmaprhiveodbc64.so
HOST=<HOST_NAME>
PORT=10000
SSL=1
CAIssuedCertNamesMismatch=1
TrustedCerts=/opt/mapr/conf/ssl_keystore.pem
AuthMech=4
```

Sample /etc/odbcinst.ini file

```
[ODBC Drivers]
Mapr Hive ODBC Driver=Installed
[Mapr Hive ODBC Driver 64-bit]
Description=Mapr Hive ODBC Driver (64-bit)
Driver=/opt/mapr/hiveodbc/lib/64/libmaprhiveodbc64.so
```

Sample /etc/mapr.hiveodbc.ini file

```
[Driver]
ErrorMessagesPath=/opt/mapr/hiveodbc/ErrorMessages/
LogLevel=0
```


last-updated: Apr 11, 2022

```
LogPath=
SwapFilePath=/tmp
```

Connecting to WebHCat

The method that WebHCat clients use to connect to WebHCat is based on the WebHCat Authentication method:

WebHCat Authentication	Connection Requirements
Simple	Clients pass the username as the `user.name` parameter in the REST call. No password is required. Example: `http://<hostname>:50111/templeton/v1/ddl/database/default/table/table01?user.name=juser`
PAM	Clients enter username and password authentication through a pop-up dialog box in the web browser session.
Kerberos with SPNEGO	Clients can use one of the following methods: • curl example: `curl --negotiate -i -u : 'http://<FQDN>:50111/templeton/v1/ddl/database/'` • Web browser with user name and password. For more information, see Configuring SPNEGO on MapR on page 1436 **Note:** With either method you must also have a Kerberos ticket in the cache. See Example: Generating a Kerberos Ticket

Enabling High Availability for Hive

This section describes how to enable High Availability for HiveServer2 and HiveMetastore.

Note: You can achieve High Availability(HA) through HA tools like HAProxy or F5. Based on the tools used, you need to configure reverse DNS lookups and implement other security features. However, the MapR Data Platform does not support any HA tool.

Related concepts

Enabling High Availability for Spark Thrift Server on page 4046

Enabling High Availability for HiveServer2

Perform the following steps to enable High Availability for HiveServer2.

Configuring Hive

1. Modify the `warden.hs2.conf` file as shown below on all the nodes where Hive is installed.

   ```
   services=hs2:all
   ```

2. Add the following properties to the `hive-site.xml` file on all the nodes where HiveServer2 is installed.

last-updated: Apr 11, 2022

Property	Value	Description
`hive.server2.support.dynamic.service.discovery`	`true` (default is `false`)	Set to `true` to enable HiveServer2 dynamic service discovery for its clients.
`hive.server2.zookeeper.namespace`	hiveserver2 (default value)	The parent node in ZooKeeper, which is used by HiveServer2 when supporting dynamic service discovery.
`hive.zookeeper.quorum`	**<hostname>**:5181,**<hostname>**:5181,**<hostname>**:5181	List of ZooKeeper servers to talk to. Used in connection string by JDBC/ODBC clients instead of URI of specific HiveServer2 instance.
`hive.zookeeper.client.port`	5181 (default value)	The port of the ZooKeeper servers to talk to. If the list of Zookeeper servers specified in `hive.zookeeper.quorum` does not contain port numbers and so, this value is used.
`hive.zookeeper.session.timeout`	600000 (default value)	Zookeeper client's session timeout value. The client is disconnected, and as a result, all locks are released if a heartbeat is not sent within the timeout period.

3. Restart all the nodes where Hive service is installed after updating the configuration.

Connecting with JDBC/ODBC Clients

- Connect to HiveServer2 with JDBC/ODBC clients using the following connection string:

```
jdbc:hive2://<zookeeper_ensemble>/;serviceDiscoveryMode=zooKeeper;
zooKeeperNamespace=<hiveserver2_zookeeper_namespace>
```

Here:

<zookeeper_ensemble>	Specifies a comma-separated list of ZooKeeper servers that form the ensemble. For example: `<zk_host1>:<zk_port1>,<zk_host2>:<zk_port2>,<zk_host3>:<zk_p`
<hiveserver2_zookeeper_namespace>	Specifies the namespace on Zookeeper under which HiveServer2 znodes are add The namespace value is configured in `hive.server2.zookeeper.name` space

Deregistering HiveServer2 Instances from Zookeeper

Remove a HiveServer2 instance from Zookeeper by running the following commands (in the ZooKeeper command line interface) to deregister the server.

1. Launch the ZooKeeper command line interface and get the HiveServer2 znode by running the following commands:

```
/opt/mapr/zookeeper/zookeeper-<version>/bin/zkCli.sh -server <ip:port of
zookeeper instance>
ls /<hive.server2.zookeeper.namespace>
```

2. Run the command to deregister HiveServer2. To deregister:

last-updated: Apr 11, 2022

- A particular HiveServer2, run the following command:

```
delete /hiveserver2 serverUri=<hostname:port>;version=<hive
version>;sequence=<sequence number>
```

  After you deregister the HiveServer2 from Zookeeper, it will not return the deregistered HiveServer2 for new client connections. However, active client sessions are not affected by deregistering the HiveServer2 from Zookeeper.

- All HiveServer2 instances of a particular version, run the following command:

```
hive --service hiveserver2 --deregister <version_number>
```

Example HiveServer2 High Availability Setup

This section describes a High Availability set up for HiveServer2 on a sample MapR Data Platform cluster. Suppose a three node cluster with the following (optional) IP addresses and host names:

IP Address	Host Name
192.168.33.11	node1
192.168.33.12	node2
192.168.33.13	node3

Connection to HiveServer2 can be accomplished using the following string:

```
jdbc:hive2://
node1:5181,node2:5181,node3:5181/;serviceDiscoveryMode=zooKeeper;zooKeeperNa
mespace=hiveserver2
```

To deregister HiveServer2:

1. Launch the ZooKeeper command line interface using the following command:

```
/opt/mapr/zookeeper/zookeeper-3.4.5/bin/zkCli.sh -server
192.168.33.13:5181
```

2. Look at the ZooKeeper namespace using the following command:

```
ls /hiveserver2
```

   Output:

```
[serverUri=node3:10000;version=2.1.1-mapr-1703;sequence=0000000004,
serverUri=node1:10000;version=2.1.1-mapr-1703;sequence=0000000006]
```

3. Deregister:

   - HiveServer2 on node3:

```
delete
serverUri=node3:10000;version=2.1.1-mapr-1703;sequence=0000000004
```

- All HiveServer2 instances:

```
hive --service hiveserver2 --deregister 2.1.1-mapr-1703
```

Enabling High Availability for Hive Metastore

To enable High Availability for Hive Metastore, perform the following steps:

1. Enable remote access to the underlying database from different nodes.
2. Add all Metastore instances to `hive.metastore.uris` on all the nodes:

```
<property>
  <name>hive.metastore.uris</name>
  <value>thrift://<hostname>:9083,thrift://<hostname>:9083</value>
</property>
```

3. Restart Hive.

Note that enabling high availability for the Hive Metastore does not require changes to the `warden.hivemetastore.conf` file. Active-active mode is not supported for Hive Metastore.

Example

Suppose HiveMetastore is on the following two nodes:

IP Address	Host Name
192.168.33.11	node1
192.168.33.12	node2

1. Change the MySQL configuration:

```
nano /etc/my.cnf
```

2. Comment out the following properties:

```
#bind-address
#skip-networking
```

If these properties are not in `my.cnf`, you can skip editing `my.cnf`. Restart the MySQL server.

3. Enable remote access for the underlying database by granting permissions in the underlying database:

```
mysql> GRANT ALL PRIVILEGES ON metastore.* TO 'root'@'192.168.33.11'
IDENTIFIED BY 'secret' WITH GRANT OPTION;
```

```
mysql> GRANT ALL PRIVILEGES ON metastore.* TO 'root'@'192.168.33.12'
IDENTIFIED BY 'secret' WITH GRANT OPTION;
```

4. Verify the connection to the MySQL server from `node1` and `node2`. For example, run the following commands:
   - On `node1`:

```
mysql -h node1 -uroot -psecret
```

- On node2:

```
mysql -h node2 -uroot -psecret
```

5. Add all Metastore instances to `hive.metastore.uris` on all nodes with the Hive instance:

```
<property>
  <name>hive.metastore.uris</name>
  <value>thrift://192.168.33.11:9083,thrift://192.168.33.12:9083</value>
</property>
```

6. Restart Hive.

Hive Features in MapR Data Platform
Describes MapR Data Platform-specific features in Hive.

Removing Temporary Hive Files

Starting from EEP 8.1.0 and EEP 6.3.6, to remove the temporary Hive files created during the Hive session, set the value of `hive.scratchdir.lock` property to `true` on `hive-site.xml` file.

```
<property>
   <name>hive.scratchdir.lock</name>
   <value>true</value>
</property>
```

For the previous EEP versions, manually remove the temporary Hive files that are not used by the active Hive sessions.

You have two different situations:

- If you have configured the HiveServer2 in a node, set `hive.scratchdir.lock` property on the `hive-site.xml` file to automatically remove the temporary Hive files.
- If you have not configured the HiveServer2 in a node, set the `hive.scratchdir.lock` property and run the following command to remove the temporary Hive files.

```
hive --service cleardanglingscratchdir
```

Hive 2.3 API Changes
This topic describes the public API changes that occurred between Hive 2.1 EEP 5.0.0 and Hive 2.3 EEP 6.0.0.

For more information, see the Hive 2.1.1-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes on page 5978 and Hive 2.3.3-1808 (EEP 6.0.0) Release Notes on page 5953.

JDBC classes API changes

This section contains changes made to classes related to the JDBC API in Hive.

Table

Method	Description
`List<String> parseInitFile(String initFile)`	Parses initial SQL file skipping comments that starts with # or `--`.

last-updated: Apr 11, 2022

Table

Method	Description
`void setInPlaceUpdateStream(InPlaceUpdate stream)`	Only used by the beeline client to set the stream on which in place progress updates are to be shown.

Table

Method	Description

`JdbcConnectionParams parseURL(String uri)`	Parse JDBC connection URL The new format of the URL is:
	`jdbc:hive2://:,:/dbName;sess_var_list?hive_conf_list#hive_var_list`
	where the optional `sess`, `conf`, and `var` lists are semicolon separated = pairs. For utilizing dynamic service discovery with HiveServer2, multiple comma-separated host:port pairs can be specified as shown above. The JDBC driver resolves the list of URIs and picks a specific server instance to connect to. Currently, dynamic service discovery using ZooKeeper is supported, in which case the host:port pairs represent a ZooKeeper ensemble. As before, if the host/port is not specified, it the driver runs an embedded Hive:
	• `jdbc:hive2://ubuntu:11000/db2?hive.cli.conf.printheader=true;hive.exec.mode.local.auto.inputbytes.max=9999#stab=salesTable;icol=customerID`
	• `jdbc:hive2://?hive.cli.conf.printheader=true;hive.exec.mode.local.auto.inputbytes.max=9999#stab=salesTable;icol=customerID`
	• `jdbc:hive2://ubuntu:11000/db2;user=foo;password=bar`
	Connect to `http://server:10001/hs2`, with specified `basicAuth` credentials and initial database:
	`jdbc:hive2://server:10001/db;user=foo;password=bar?hive.server2.transport.mode=http;hive.server2.thrift.http.path=hs2`

Security-related API changes

The following properties are removed from the default `hive-site.xml` configuration on a secured cluster:

Table

Property	Value
`hive.server2.webui.keystore.path`	`/opt/mapr/conf/ssl_keystore`.
`hive.server2.webui.keystore.password`	`Default keystore password`.

The following property is added to the default `hive-site.xml` configuration on a secured cluster:

Table

Method	Description
`hive.server2.use.SSL true`	`true`

Since the HiveServer2 server is configured to use SSL encryption by default starting from Hive-2.3 EEP-6.0.0, add `ssl=true;` to a JDBC connection string when PAM or MAPR-SASL authentication is used.

For example:

Old JDBC connection string with PAM authentication:

```
beeline> !connect jdbc:hive2://<host>:10000/default;
```

New JDBC connection string with PAM authentication:

```
beeline> !connect jdbc:hive2://<host>:10000/default;ssl=true;
```

Note: All API functionality changes are compatible with previous versions.

Hive 2.1 API

This section contains the following:

New Classes in Hive 2.1

Hive 2.1 includes the following new classes:

Class	Description
org.apache.hadoop.hive.common.DiskRangeInfo	Contains disk range information including disk ranges and total length.
org.apache.hadoop.hive.common.JvmPauseMonitor	This is based on the JvmPauseMonitor from Hadoop.
org.apache.hadoop.hive.common.StringableMap	A utility class that can convert a HashMap of Properties into a colon separated string, and can take the same format of string and convert it to a HashMap of Properties.
org.apache.hadoop.hive.common.ValidCompactorTxnList	An implementation of org.apache.hadoop.hive.common.ValidTxnList for use by the compactor.
org.apache.hadoop.hive.common.io .DiskRange	The sections of a file.
org.apache.hadoop.hive.common.io.DiskRangeList	Alternative for Java linked list iterator interface to support concurrent modifications of the same list by multiple iterators.
org.apache.hadoop.hive.common.jsonexplain.tez.Printer	Creation of output string to show JSON plan.
org.apache.hadoop.hive.common.jsonexplain.tez.TezJsonParserUtils	JsonParser for Tez that prints a JSONObject nto outputStream.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.common.metrics.LegacyMetrics	The Metrics Subsystem allows exposure of a number of named parameters/counters via JMX, is intended to be used as a static subsystem, and has a couple of primary ways in which it can be used: • Using the set and get methods to set and get named parameters. • Using the incrementCounter method to increment and set named parameters in one go, rather than having to make a get and then a set. • Using the startScope and endScope methods to start and end named "scopes" that record the number of times they have been instantiated and amount of time (in milliseconds) spent inside the scopes.
org.apache.hadoop.hive.common.type.RandomTypeUtil	Creates random data of different object types.
org.apache.hadoop.hive.conf.VariableSubstitution	Substitution of environment variables.
org.apache.hadoop.hive.contrib.genericudf.example.GenericUDFAdd10	Intializes the GenericUDF (once per instance), evaluates the GenericUDF with the arguments, and gets the string to display.
org.apache.hadoop.hive.io.HdfsUtils	Utils to resolve file properties in MAPR filesystem.
org.apache.hadoop.hive.metastore.AcidEventListener	It handles cleanup of dropped partition/table/database in ACID related metastore tables.
org.apache.hadoop.hive.metastore.FileMetadataHandler	The base implementation of a file metadata handler for a specific file type.
org.apache.hadoop.hive.metastore.FileMetadataManager	Handle storage functions of metadata.
org.apache.hadoop.hive.metastore.HMSMetricsListener	Report metrics of metadata added and deleted by this Hive Metastore.
org.apache.hadoop.hive.metastore.Metastore	Class to arrange work with Metastore.
org.apache.hadoop.hive.metastore.PartFilterExprUtil	Utility functions for working with partition filter expressions.
org.apache.hadoop.hive.metastore.api.AbortTxnsRequest	Class for handling transactions request.
org.apache.hadoop.hive.metastore.api.AddForeignKeyRequest	Class for handling foreign key request.
org.apache.hadoop.hive.metastore.api.AddPrimaryKeyRequest	Class for handling primary key request.
org.apache.hadoop.hive.metastore.api.CacheFileMetadataRequest	Class for caching metadata requests.
org.apache.hadoop.hive.metastore.api.CacheFileMetadataResult	Class for caching metadata results.
org.apache.hadoop.hive.metastore.api.ClearFileMetadataResult	Class for clearing metadata results.
org.apache.hadoop.hive.metastore.api.DropConstraintRequest	Class for dropping constraint requests.
org.apache.hadoop.hive.metastore.api.ForeignKeysRequest	Class for handling foreign key requests.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.metastore.api.ForeignKeysResponse	Class for getting response from all functions.
org.apache.hadoop.hive.metastore.api.GetAllFunctionsResponse	Class for getting response from all functions.
org.apache.hadoop.hive.metastore.api.GetFileMetadataByExprRequest	Class for getting metadata from expression response.
org.apache.hadoop.hive.metastore.api.GetFileMetadataByExprResult	Class for getting metadata from expression result.
org.apache.hadoop.hive.metastore.api.MetadataPpdResult	Class for describing metadata rpd result.
org.apache.hadoop.hive.metastore.api.PrimaryKeysRequest	Class for describing primary key result.
org.apache.hadoop.hive.metastore.api.PrimaryKeysResponse	Class for describing primary key response.
org.apache.hadoop.hive.metastore.api.PutFileMetadataRequest	Class for output metadata request to file.
org.apache.hadoop.hive.metastore.api.PutFileMetadataResult	Class for output metadata result to file.
org.apache.hadoop.hive.metastore.api.SQLForeignKey	Class for describing SQL foreign key.
org.apache.hadoop.hive.metastore.api.SQLPrimaryKey	Class for describing SQL primary key.
org.apache.hadoop.hive.metastore.api.TableMeta	Class for describing table metadata.
org.apache.hadoop.hive.metastore.model.MConstraint	Model of constraints stored in metastore.
org.apache.hadoop.hive.metastore.txn.TxnUtils	Class for handling transactions.
org.apache.hadoop.hive.ql.CompilationOpContext	Contains the operator sequence ID and a subset of compilation context that is passed to operators to get rid of some globals.
org.apache.hadoop.hive.ql.QueryDisplay	Contains limited query information to save for WebUI. The class is synchronized, as WebUI may access information about a running query.
org.apache.hadoop.hive.ql.QueryState	The class to store query level info such as queryId.
org.apache.hadoop.hive.ql.exec.AbstractMapOperator	Abstract Map operator.
org.apache.hadoop.hive.ql.exec.GlobalWorkMapFactory	Get job that has been executed on cluster as a map value.
org.apache.hadoop.hive.ql.exec.ObjectCacheWrapper	Wapping class for ObjectCache class.
org.apache.hadoop.hive.ql.exec.SerializationUtilities	Utilities related to serialization and deserialization.
org.apache.hadoop.hive.ql.exec.UDFClassLoader	UDFClassLoader is used to dynamically register udf (and related) jars. This was introducted to fix HIVE-11878. Each session will have its own instance of UDFClassLoader to support HiveServer2, which can contain multiple active sessions.
org.apache.hadoop.hive.ql.exec.spark.CacheTran	Class for making cache persistent.
org.apache.hadoop.hive.ql.exec.spark.SmallTableCache	Class for cache cleaning if new query is present.
org.apache.hadoop.hive.ql.exec.spark.SparkDynamicPartitionPruner	The spark version of DynamicPartitionPruner.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.spark.status.impl.SparkJobUtils	Utilities for spark job.
org.apache.hadoop.hive.ql.exec.tez.ColumnarSplitSizeEstimator	Split size estimator for columnar file formats.
org.apache.hadoop.hive.ql.exec.tez.HostAffinitySplitLocationProvider	This maps a split (path + offset) to an index based on the number of locations provided.
org.apache.hadoop.hive.ql.exec.tez.InPlaceUpdates	Class responsible for inplace updates.
org.apache.hadoop.hive.ql.exec.tez.KeyValuesFromKeyValue	Provides a key/values (note the plural values) interface out of a KeyValueReader, needed by ReduceRecordSource when reading input from a key/value source.
org.apache.hadoop.hive.ql.exec.tez.KeyValuesFromKeyValues	Provides a key/values interface out of a KeyValuesReader for use by ReduceRecordSource.
org.apache.hadoop.hive.ql.exec.tez.LlapObjectCache	Llap implementation for the shared object cache.
org.apache.hadoop.hive.ql.exec.tez.Utils	Utilities for running tez jobs.
org.apache.hadoop.hive.ql.exec.vector.IntervalDayTimeColumnVector	This class represents a nullable interval day time column vector capable of handing a wide range of interval day time values.
org.apache.hadoop.hive.ql.exec.vector.ListColumnVector	The representation of a vectorized column of list objects.
org.apache.hadoop.hive.ql.exec.vector.MapColumnVector	The representation of a vectorized column of map objects.
org.apache.hadoop.hive.ql.exec.vecto.MultiValuedColumnVector	The representation of a vectorized column of multi-valued objects, such as lists and maps.
org.apache.hadoop.hive.ql.exec.vector.StructColumnVector	The representation of a vectorized column of struct objects.
org.apache.hadoop.hive.ql.exec.vector.TimestampColumnVector	This class represents a nullable timestamp column vector capable of handing a wide range of timestamp values.
org.apache.hadoop.hive.ql.exec.vector.UnionColumnVector	The representation of a vectorized column of struct objects.
org.apache.hadoop.hive.ql.exec.vector.VectorSparkHashTableSinkOperator	Vectorized version of SparkHashTableSinkOperator. It delegates all the work to super class Copied from VectorFileSinkOperator.
org.apache.hadoop.hive.ql.exec.vector.VectorSparkPartitionPruningSinkOperator	Vectorized version for SparkPartitionPruningSinkOperator.
org.apache.hadoop.hive.ql.exec.vector.expressions.BRoundWithNumDigitsDoubleToDouble	Banking rounding implementation.
org.apache.hadoop.hive.ql.exec.vector.expressions.CastDoubleToTimestamp	Cast double type to timestamp type.
org.apache.hadoop.hive.ql.exec.vector.expressions.CastLongToTimestamp	Cast long type to timestamp type.
org.apache.hadoop.hive.ql.exec.vector.expressions.CastMillisecondsLongToTimestamp	Cast milliseconds long type to timestamp.
org.apache.hadoop.hive.ql.exec.vector.expressions.CastStringGroupToString	Cast string group type to string type.
org.apache.hadoop.hive.ql.exec.vector.expressions.CastTimestampToBoolean	Cast timestamp type to boolean.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.vector.expressions.CastTimestampToDate	Cast timestamp type to decimal type.
org.apache.hadoop.hive.ql.exec.vector.expressions.CastTimestampToDouble	Cast timestamp type to double type.
org.apache.hadoop.hive.ql.exec.vector.expressions.CastTimestampToLong	Cast timestamp type to long type.
org.apache.hadoop.hive.ql.exec.vector.expressions.DateColSubtractDateColumn	Substract two variables of date type.
org.apache.hadoop.hive.ql.exec.vector.expressions.DateColSubtractDateScalar	Substract two variables of date type.
org.apache.hadoop.hive.ql.exec.vector.expressions.DateScalarSubtractDateColumn	Substract two variables of date type.
org.apache.hadoop.hive.ql.exec.vector.expressions.FilterStructColumnInList	Evaluates an IN filter on a batch for a vector of structs.
org.apache.hadoop.hive.ql.exec.vector.expressions.FilterTimestampColumnInList	Evaluates IN filter on a batch for a vector of timestamps.
org.apache.hadoop.hive.ql.exec.vector.expressions.FuncBRoundWithNumDigitsDecimalToDecimal	Banking rounding for decimal digits.
org.apache.hadoop.hive.ql.exec.vector.expressions.FuncDecimalToTimestamp	This is a superclass for unary decimal functions and expressions returning timestamps that operate directly on the input and set the output.
org.apache.hadoop.hive.ql.exec.vector.expressions.FuncTimestampToDecimal	This is a superclass for unary timestamp functions and expressions returning decimals that operate directly on the input and set the output.
org.apache.hadoop.hive.ql.exec.vector.expressions.FuncTimestampToLong	This is a superclass for unary timestamp functions and expressions returning long that operate directly on the input and set the output.
org.apache.hadoop.hive.ql.exec.vector.expressions.IfExprDoubleColumnDoubleColumn	Computes IF(expr1, expr2, expr3) for 3 input column expressions.
org.apache.hadoop.hive.ql.exec.vector.expressions.IfExprIntervalDayTimeColumnColumn	Computes IF(expr1, expr2, expr3) for 3 input column expressions.
org.apache.hadoop.hive.ql.exec.vector.expressions.IfExprIntervalDayTimeColumnScalar	Computes IF(expr1, expr2, expr3) for 3 input column expressions.
org.apache.hadoop.hive.ql.exec.vector.expressions.IfExprIntervalDayTimeScalarColumn	Computes IF(expr1, expr2, expr3) for 3 input column expressions.
org.apache.hadoop.hive.ql.exec.vector.expressions.IfExprIntervalDayTimeScalarScalar	Computes IF(expr1, expr2, expr3) for 3 input column expressions.
org.apache.hadoop.hive.ql.exec.vector.expressions.IfExprLongColumnLongColumn	Computes IF(expr1, expr2, expr3) for 3 input column expressions.
org.apache.hadoop.hive.ql.exec.vector.expressions.IfExprTimestampColumnColumn	Computes IF(expr1, expr2, expr3) for 3 input column expressions.
org.apache.hadoop.hive.ql.exec.vector.expressions.IfExprTimestampColumnColumnBase	Computes IF(expr1, expr2, expr3) for 3 input column expressions.
org.apache.hadoop.hive.ql.exec.vector.expressions.IfExprTimestampColumnScalar	Computes IF(expr1, expr2, expr3) for 3 input column expressions.
org.apache.hadoop.hive.ql.exec.vector.expressions.IfExprTimestampColumnScalarBase	Computes IF(expr1, expr2, expr3) for 3 input column expressions.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.vector.expressions.IfExprTimestampScalarColumn	Computes IF(expr1, expr2, expr3) for 3 input column expressions.
org.apache.hadoop.hive.ql.exec.vector.expressions.IfExprTimestampScalarColumnBase	Computes IF(expr1, expr2, expr3) for 3 input column expressions.
org.apache.hadoop.hive.ql.exec.vector.expressions.IfExprTimestampScalarScalar	Computes IF(expr1, expr2, expr3) for 3 input column expressions.
org.apache.hadoop.hive.ql.exec.vector.expressions.IfExprTimestampScalarScalarBase	Computes IF(expr1, expr2, expr3) for 3 input column expressions.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongColEqualLongColumn	If equal two columns as vectors.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongColEqualLongScalar	If equal long column and long scalar as vectors.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongColGreaterEqualLongColumn	If greater or equal two columns as vectors.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongColGreaterEqualLongScalar	If greater or equal long column and long scalar.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongColGreaterLongColumn	If greater two columns as vectors.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongColGreaterLongScalar	If greater long column and long scalar as vectors.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongColLessEqualLongColumn	If less equal two columns as vectors.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongColLessEqualLongScalar	If less equal long column and long scalar as vectors.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongColLessLongColumn	If less two columns as vectors.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongColLessLongScalar	If less long column and long scalar as vectors.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongColNotEqualLongColumn	If not equal two columns as vectors.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongColNotEqualLongScalar	If not equal long column and long scalar as vectors.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongScalarEqualLongColumn	If equal long scalar and long column.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongScalarGreaterEqualLongColumn	If greater equal long scalar and long column as vector.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongScalarGreaterLongColumn	If greater long scalar and long column as vectors.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongScalarLessEqualLongColumn	If less equal long scalar and long column as vector.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongScalarLessLongColumn	If less long scalar and long column as vectors.
org.apache.hadoop.hive.ql.exec.vector.expressions.LongScalarNotEqualLongColumn	If not equal long scalar and long column as vectors.
org.apache.hadoop.hive.ql.exec.vector.expressions.SelectStringColLikeStringScalar	Select like statement for string column and string scalar.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.vector.expressions.StructColumnInList	Evaluates an IN boolean expression (not a filter) on a batch for a vector of structs.
org.apache.hadoop.hive.ql.exec.vector.expressions.TimestampColumnInList	Returns a boolean value indicating if a column is IN a list of constants.
org.apache.hadoop.hive.ql.exec.vector.expressions.TimestampToStringUnaryUDF	This is a superclass for unary long functions returning strings that operate directly on the input and set the output.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFDateTimestamp	Vectorized version of TO_DATE(timestamp).
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFDayOfMonthDate	Expression to get day of month.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFDayOfMonthTimestamp	Expression to get day of month.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFHourDate	Returns hour of day.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFHourTimestamp	Returns hour of day.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFMinuteDate	Returns minute value.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFMinuteTimestamp	Returns minute value.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFMonthDate	Returns month value.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFMonthTimestamp	Returns month value.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFSecondDate	Expression to get seconds.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFSecondTimestamp	Expression to get seconds.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFTimestampFieldDate	Abstract class to return various fields from a Timestamp or Date.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFTimestampFieldTimestamp	Abstract class to return various fields from a Timestamp.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFUnixTimeStampDate	Returns Unix Timestamp.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFUnixTimeStampTimestamp	Returns Unix Timestamp.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFWeekOfYearDate	Expression to get week of year.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFWeekOfYearTimestamp	Expression to get week of year.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFYearDate	Expression to get year as a long.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFYearTimestamp	Expression to get year as a long.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.vector.expressions.aggregates.VectorUDAFAvgTimestamp	Generated from template VectorUDAFAvg.txt.
org.apache.hadoop.hive.ql.exec.vector.expressions.aggregates.VectorUDAFStdPopTimestamp	Vectorized implementation for VARIANCE aggregates.
org.apache.hadoop.hive.ql.exec.vector.expressions.aggregates.VectorUDAFStdSampTimestamp	Vectorized implementation for VARIANCE aggregates.
org.apache.hadoop.hive.ql.exec.vector.expressions.aggregates.VectorUDAFVarPopTimestamp	Vectorized implementation for VARIANCE aggregates.
org.apache.hadoop.hive.ql.exec.vector.expressions.aggregates.VectorUDAFVarSampTimestamp	Vectorized implementation for VARIANCE aggregates.
org.apache.hadoop.hive.ql.hooks.LineageLogger	Implementation of a post execute hook that logs lineage info to a log file.
org.apache.hadoop.hive.ql.hooks.PostExecOrcFileDump	Post execution hook to print orc file dump for files that will be read by fetch task.
org.apache.hadoop.hive.ql.hooks.PostExecTezSummaryPrinter	Post execution hook to print hive tez counters to console error stream.
org.apache.hadoop.hive.ql.io.HdfsUtils	Utilities for hadoop fs.
org.apache.hadoop.hive.ql.io.IOContextMap	Uses the global static map of IOContext-s inside IOContext, uses threadlocal for Spark, and creates inheritable threadlocal with attemptId (only set in LLAP), which will propagate to all the Tez threads.
org.apache.hadoop.hive.ql.io.NullScanFileSystem	Filesystem that does not allow Hive to read files for nullscans.
org.apache.hadoop.hive.ql.io.ProxyLocalFileSystem	This class is to workaround existing issues on LocalFileSystem.
org.apache.hadoop.hive.ql.io.SyntheticFileId	Create synthetic ID for file.
org.apache.hadoop.hive.ql.io.orc.ExternalCache	Metastore-based footer cache storing serialized footers.
org.apache.hadoop.hive.ql.io.orc.MetastoreExternalCachesByConf	An implementation of external cache and factory based on metastore.
org.apache.hadoop.hive.ql.io.orc.OrcFileFormatProxy	File format proxy for ORC.
org.apache.hadoop.hive.ql.io.orc.RecordReaderImpl	Implementation of record reader.
org.apache.hadoop.hive.ql.io.parquet.read.ParquetFilterPredicateConverter	Translate the search argument to the filter predicate parquet uses.
org.apache.hadoop.hive.ql.io.sarg.ConvertAstToSearchArg	Converting asterisk and use it as a search argument.
org.apache.hadoop.hive.ql.io.sarg.SearchArgumentImpl	The implementation of SearchArguments.
org.apache.hadoop.hive.ql.lib.PreOrderOnceWalker	This class takes list of starting nodes and walks them in pre-order.
org.apache.hadoop.hive.ql.log.HiveEventCounter	A log4J2 appender that simply counts logging events in four levels: fatal, error, warn, and info.
org.apache.hadoop.hive.ql.log.NoDeleteRollingFileAppender	Instantiate a RollingFileAppender and open the file designated by `filename`. The opened filename will become the output destination for this appender.
org.apache.hadoop.hive.ql.log.NullAppender	A NullAppender that never outputs a message to any device.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.log.PidFilePatternConverter	FilePattern converter that converts %pid pattern to @ information obtained at runtime.
org.apache.hadoop.hive.ql.metadata.ForeignKeyInfo	ForeignKeyInfo is a metadata structure containing the foreign keys associated with a table.
org.apache.hadoop.hive.ql.metadata.PrimaryKeyInfo	PrimaryKeyInfo is a metadata structure containing the primary key associated with a table.
org.apache.hadoop.hive.ql.metadata.TableIterable	Gets Table objects for a table list.
org.apache.hadoop.hive.ql.optimizer.OperatorComparatorFactory	Comparator for table operators.
org.apache.hadoop.hive.ql.optimizer.PartitionColumnsSeparator	Takes a Filter expression, and if its predicate contains an IN operator whose children are constant structs or structs containing constant fields, it will try to generate predicate with IN clauses containing only partition columns.
org.apache.hadoop.hive.ql.optimizer.PointLookupOptimizer	Takes a Filter expression, and if its predicate contains an OR operator whose children are constant equality expressions, it will try to generate an IN clause (which is more efficient).
org.apache.hadoop.hive.ql.optimizer.RedundantDynamicPruningConditionsRemoval	Takes a Filter operator on top of a TableScan and removes dynamic pruning conditions if static partition pruning has been triggered already.
org.apache.hadoop.hive.ql.optimizer.SparkRemoveDynamicPruningBySize	Disables pruning if the number of keys for dynamic pruning is too large.
org.apache.hadoop.hive.ql.optimizer.calcite.HivePlannerContext	Creating context for Hive Planner.
org.apache.hadoop.hive.ql.optimizer.calcite.HiveRelBuilder	Builder for relational expressions in Hive.
org.apache.hadoop.hive.ql.optimizer.calcite.HiveRelFactories	Factory class for creating relational operators for queries.
org.apache.hadoop.hive.ql.optimizer.calcite.HiveRexExecutorImpl	Hive registry executor implementation.
org.apache.hadoop.hive.ql.optimizer.calcite.HiveRexUtil	Utilities for hive registry executor.
org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveBetween	Operand type-inference strategy where an unknown operand type is derived from the first operand with a known type, but the first operand is a boolean.
org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveIn	Create in clause instance for hive queries.
org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveMultiJoin	A HiveMultiJoin represents a succession of binary joins.
org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveSortLimit	Sorting limit in hive queries.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveAggregateJoinTransposeRule	Planner rule that pushes an org.apache.calcite.rel.core.Aggregate past a org.apache.calcite.rel.core.Join.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveAggregateProjectMergeRule	Planner rule that recognizes a HiveAggregate on top of a HiveProject and if possible, aggregates through the project or removes the project.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveAggregatePullUpConstantsRule	Rule for pull up constants aggregation.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveFilterAggregateTransposeRule	Transpose rule for filter aggregation.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveFilterProjectTSTransposeRule	Transpose rule for filtering project TST.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveFilterSortTransposeRule	Transpose rule for filtering sort.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveJoinProjectTransposeRule	Transpose rule for join project.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HivePointLookupOptimizerRule	Takes a Filter expression, and if its predicate contains an OR operator whose children are constant equality expressions, tries to generate an IN clause (which is more efficient).
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveProjectFilterPullUpConstantsRule	Planner rule that infers constant expressions from Filter into a Project operator.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveProjectSortTransposeRule	Transpose rule for project sort.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveReduceExpressionsRule	Collection of planner rules that apply various simplifying transformations on RexNode trees.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveReduceExpressionsWithStatsRule	This rule simplifies the condition in Filter operators using the column statistics (if available).
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveRelColumnsAlignment	Infers the order in Aggregate columns and the order of conjuncts in a Join condition that might be more beneficial to avoid additional sort stages.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveSortJoinReduceRule	Planner rule that pushes a org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveSortLimit past a org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveJoin.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveSortLimitPullUpConstantsRule	Planner rule that pulls up constant keys through a SortLimit operator.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveSortMergeRule	This rule will merge two HiveSortLimit operators.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveSortProjectTransposeRule	Transpose rule for sort project.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveSortRemoveRule	Planner rule that removes a org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveSortLimit.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveSortUnionReduceRule	Planner rule that pushes a org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveSortLimit past aorg.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveUnion.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveUnionPullUpConstantsRule	Planner rule that pulls up constants through a Union operator.
org.apache.hadoop.hive.ql.optimizer.calcite.stats.HiveRelMdPredicates	Infers predicates for a project.
org.apache.hadoop.hive.ql.optimizer.physical.LlapDecider	LlapDecider takes care of tagging certain vertices in the execution graph as "llap", which in turn causes them to be submitted to an llap daemon instead of a regular yarn container.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.optimizer.physical.MemoryDecider	MemoryDecider is a simple physical optimizer that adjusts the memory layout of tez tasks.
org.apache.hadoop.hive.ql.optimizer.physical.SerializeFilter	SerializeFilter is a simple physical optimizer that serializes all filter expressions in Tablescan Operators.
org.apache.hadoop.hive.ql.optimizer.spark.CombineEquivalentWorkResolver	CombineEquivalentWorkResolver searches inside SparkWork, finds and combines equivalent works.
org.apache.hadoop.hive.ql.optimizer.spark.SparkPartitionPruningSinkDesc	Description of spark partition pruning sink.
org.apache.hadoop.hive.ql.parse.AnalyzeCommandUtils	Utilities for command analysis.
org.apache.hadoop.hive.ql.parse.ColumnStatsAutoGatherContext	ColumnStatsAutoGatherContext is passed to the compiler when set hive.stats.autogather is true during the INSERT OVERWRITE command.
org.apache.hadoop.hive.ql.parse.MaskAndFilterInfo	Information for masking and filtering.
org.apache.hadoop.hive.ql.parse.TableMask	The main purpose for this class is for authorization.
org.apache.hadoop.hive.ql.parse.spark.SparkPartitionPruningSinkOperator	This operator gets partition info from the upstream operators and writes them to HDFS.
org.apache.hadoop.hive.ql.parse.spark.SplitOpTreeForDPP	This processor triggers on SparkPartitionPruningSinkOperator.
org.apache.hadoop.hive.ql.plan.AbortTxnsDesc	Descriptor for aborting transactions.
org.apache.hadoop.hive.ql.plan.CacheMetadataDesc	Description for metadata cache.
org.apache.hadoop.hive.ql.plan.ShowCreateDatabaseDesc	Shows the name of the database.
org.apache.hadoop.hive.ql.plan.VectorPartitionConversion	PartitionConversion.
org.apache.hadoop.hive.ql.plan.VectorPartitionDesc	VectorMapDesc.
org.apache.hadoop.hive.ql.plan.VectorReduceSinkDesc	VectorReduceSinkDesc.
org.apache.hadoop.hive.ql.plan.VectorReduceSinkInfo	VectorGroupByAggregrationInfo.
org.apache.hadoop.hive.ql.ppd.SimplePredicatePushDown	Implementation of predicate push down.
org.apache.hadoop.hive.ql.security.authorization.DefaultHiveAuthorizationTranslator	Default implementation of HiveAuthorizationTranslator.
org.apache.hadoop.hive.ql.security.authorization.plugin.AbstractHiveAuthorizer	Abstract class that extends HiveAuthorizer.
org.apache.hadoop.hive.ql.session.ClearDanglingScratchDir	A tool to remove dangling scratch directory.
org.apache.hadoop.hive.ql.stats.StatsCollectionContext	Creating context for stats collection.
org.apache.hadoop.hive.ql.txn.compactor.HouseKeeperServiceBase	Housekeeper for running services.
org.apache.hadoop.hive.ql.udf.UDFChr	UDFChr converts an integer into its ASCII equivalent.
org.apache.hadoop.hive.ql.udf.UDFCrc32	UDFCrc32.
org.apache.hadoop.hive.ql.udf.UDFMd5	UDFMd5.
org.apache.hadoop.hive.ql.udf.UDFReplace	UDFReplace replaces all substrings that are matched with a replacement substring.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.udf.UDFSha1	UDFSha.
org.apache.hadoop.hive.ql.udf.UDFVersion	UDFVersion
org.apache.hadoop.hive.ql.udf.generic.BaseMaskUDF	User defined function for masking.
org.apache.hadoop.hive.ql.udf.generic.GenericUDAFSumEmptyIsZero	User defined aggregation function for summing empy as zeros.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFAesBase	Base for user defined functions.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFAesDecrypt	User defined function for decryption.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFAesEncrypt	User defined function for encryption.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFBRound	User defined function for banking rounding.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFBaseNwayCompare	Base class for comparison UDF's (Greatest and Least).
org.apache.hadoop.hive.ql.udf.generic.GenericUDFMask	User defined function for masking.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFMaskFirstN	User defined function for masking first n symbols.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFMaskHash	User defined function that returns a hashed value based on str.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFMaskLastN	User defined function for masking last n symbols.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFMaskShowFirstN	User defined function for showing masked first n symbols.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFMaskShowLastN	User defined function for showing masked last n symbols.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFParamUtils	Generic UDF params utility class.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFQuarter	GenericUDFQuarter.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFRegExp	UDF to extract a specific group identified by a java regex.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFSha2	GenericUDFSha2.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFSubstringIndex	GenericUDFSubstringIndex.
org.apache.hadoop.hive.ql.udf.generic.GenericUDTFGetSplits	GenericUDTFGetSplits.
org.apache.hadoop.hive.ql.util.DependencyResolver	Query dependency resolver.
org.apache.hadoop.hive.ql.util.ResourceDownloader	Resource downloader.
org.apache.hadoop.hive.ql.util.TimestampUtils	Utitilities for Timestamps and the relevant conversions.
org.apache.hadoop.hive.serde2.DefaultFetchFormatter	Serializes row by user specified serde and calls toString() to make string type result.
org.apache.hadoop.hive.serde2.NoOpFetchFormatter	A No-op fetch formatter.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.serde2.binarysortable.BinarySortableSerDeWithEndPrefix	Serializer desrializer for binary sortable.
org.apache.hadoop.hive.serde2.thrift.ColumnBuffer	Column buffer.
org.apache.hadoop.hive.serde2.thrift.ThriftFormatter	Thrift formatter.
org.apache.hadoop.hive.serde2.thrift.ThriftJDBCBinarySerDe	Serializes the final output to thrift-able objects directly in the SerDe.
org.apache.hadoop.hive.thrift.HiveDelegationTokenManager	Delegation token manager.
org.apache.hive.beeline.ClientCommandHookFactory	Updates some client side information after executing some Hive commands.
org.apache.hive.beeline.ClientHook	This is the client's hook and used for new Hive CLI.
org.apache.hive.common.util.DateParser	Date parser class for Hive.
org.apache.hive.common.util.FixedSizedObjectPool	Simple object pool of limited size.
org.apache.hive.common.util.HashCodeUtil	Utilities for hash code.
org.apache.hive.common.util.IntervalDayTimeUtils	DateUtils.
org.apache.hive.hcatalog.streaming.AbstractRecordWriter	Class for defining record writer.
org.apache.hive.jdbc.HttpTokenAuthInterceptor	The class is instantiated with the username and password, it is then used to add header with these credentials to HTTP requests
org.apache.hive.jdbc.XsrfHttpRequestInterceptor	Http request interceptor for xsrf token.
org.apache.hive.service.cli.operation.GetCrossReferenceOperation	GetCrossReferenceOperation.
org.apache.hive.service.cli.operation.GetPrimaryKeysOperation	GetPrimaryKeysOperation.
org.apache.hive.service.cli.operation.SQLOperationDisplay	Used to display some info in the HS2 WebUI.
org.apache.hive.service.cli.operation.SQLOperationDisplayCache	Cache some SQLOperation information for WebUI
org.apache.hive.service.cli.thrift.RetryingThriftCLIServiceClient	RetryingThriftCLIServiceClient.

New Interfaces in Hive 2.1

Hive 2.1 includes the following new interfaces:

Interface	Description
org.apache.hadoop.hive.common.*Pool*	Simple object pool to prevent GC on small objects passed between threads.
org.apache.hadoop.hive.common.io.*Allocator*	An allocator provided externally to storage classes to allocate MemoryBuffer-s.
org.apache.hadoop.hive.common.io.*DataCache*	An abstract data cache that IO formats can use to retrieve and cache data.
org.apache.hadoop.hive.conf.*HiveVariableSource*	Getting hive variables.
org.apache.hadoop.hive.metastore.*FileFormatProxy*	Same as PartitionExpressionProxy, but for file format specific methods for metadata cache.

last-updated: Apr 11, 2022

org.apache.hadoop.hive.metastore.*HouseKeeperService*	Runs arbitrary background logic inside the metastore service.
org.apache.hadoop.hive.metastore.txn.*TxnStore*	A handler to answer transaction related calls that come into the metastore server.
org.apache.hadoop.hive.ql.exec.tez.*KeyValuesAdapter*	Key-values interface for the Reader used by ReduceRecordSource
org.apache.hadoop.hive.ql.exec.vector.expressions.*IStructInExpr*	Interface used for both filter and non-filter versions of IN to simplify VectorizationContext code.
org.apache.hadoop.hive.ql.exec.vector.expressions.*ITimestampInExpr*	Interface used to proccess timestamp in expression.
org.apache.hadoop.hive.ql.io.*ColumnarSplit*	Interface when implemented should return the estimated size of columnar projections that will be read from the split.
org.apache.hadoop.hive.ql.io.*LlapAwareSplit*	Split that is aware that it could be executed in LLAP.
org.apache.hadoop.hive.ql.io.*LlapWrappableInputFormatInterface*	Marker interface for LLAP; serves no other purpose.
org.apache.hadoop.hive.ql.io.*SelfDescribingInputFormatInterface*	Marker interface to indicate a given input format is self-describing and can perform schema evolution itself.
org.apache.hadoop.hive.ql.io.*StreamingOutputFormat*	Marker interface for streaming output formats.
org.apache.hadoop.hive.ql.security.authorization.plugin.*HiveAuthorizationTranslator*	This interface has functions that provide the ability to customize the translation from Hive internal representations of Authorization objects to the public API objects This is an interface that is not meant for general use, it is targeted to some specific use cases of Apache Sentry (incubating).
org.apache.hadoop.hive.serde2.*FetchFormatter*	(**For internal-use only**) Used in ListSinkOperator for formatting final output.

Changed Classes in Hive 2.1

The following classes have changes in Hive 2.1:

Class	Description
org.apache.hadoop.hive.accumulo.mr.HiveAccumuloTableOutputFormat	Output format for accumulo tables.
org.apache.hadoop.hive.cli.CliDriver	CliDriver.
org.apache.hadoop.hive.cli.OptionsProcessor	OptionsProcessor.
org.apache.hadoop.hive.common.CompressionUtils	Contains methods used for the purposes of compression. This class should not be accessed from code run in Hadoop.
org.apache.hadoop.hive.common.FileUtils	Collection of file manipulation utilities common across Hive.
org.apache.hadoop.hive.common.HiveStatsUtils	HiveStatsUtils.
org.apache.hadoop.hive.common.JavaUtils	Collection of Java class loading/reflection related utilities common across Hive.
org.apache.hadoop.hive.common.LogUtils	Utilities common to logging operations.
org.apache.hadoop.hive.common.ObjectPair	Creating pair out of templates.
org.apache.hadoop.hive.common.ServerUtils	ServerUtils (specific to HiveServer version 1)

Class	Description
org.apache.hadoop.hive.common.StatsSetupConst	Defines the constant strings used by the statistics implementation.
org.apache.hadoop.hive.common.ValidReadTxnList	An implementation of org.apache.hadoop.hive.common.ValidTxnList for use by readers.
org.apache.hadoop.hive.common.cli.CommonCliOptions	Reusable code for Hive Cli's.
org.apache.hadoop.hive.common.io.NonSyncByteArrayInputStream	A thread-not-safe version of ByteArrayOutputStream, which removes all synchronized modifiers.
org.apache.hadoop.hive.common.type.HiveDecimal	HiveDecimal.
org.apache.hadoop.hive.common.type.HiveIntervalDayTime	Day-time interval type representing an offset in days/hours/minutes/seconds, with nanosecond precision.
org.apache.hadoop.hive.common.type.HiveVarchar	HiveVarChar.
org.apache.hadoop.hive.conf.HiveConf	Hive Configuration.
org.apache.hadoop.hive.conf.HiveConfUtil	Hive Configuration utils
org.apache.hadoop.hive.contrib.serde2.MultiDelimitSerDe	This SerDe allows user to use multiple characters as the field delimiter for a table.
org.apache.hadoop.hive.contrib.serde2.RegexSerDe	RegexSerDe uses regular expression (regex) to serialize/deserialize.
org.apache.hadoop.hive.contrib.serde2.TypedBytesSerDe	TypedBytesSerDe uses typed bytes to serialize/deserialize.
org.apache.hadoop.hive.contrib.serde2.s3.S3LogDeserializer	S3LogDeserializer.
org.apache.hadoop.hive.hbase.AbstractHBaseKeyPredicateDecomposer	Simple abstract class to help with creation of a DecomposedPredicate.
org.apache.hadoop.hive.hbase.CompositeHBaseKeyFactory	Factory that creates composite keys.
org.apache.hadoop.hive.hbase.HBaseLazyObjectFactory	Replaces original keyOI with OI which is create by HBaseKeyFactory provided by serde property for hbase.
org.apache.hadoop.hive.hbase.HBaseSerDe	HBaseSerDe can be used to serialize object into an HBase table and deserialize objects from an HBase table.
org.apache.hadoop.hive.hbase.HBaseSerDeHelper	Helper class for HBaseSerDe
org.apache.hadoop.hive.hbase.HiveHBaseTableInputFormat	HiveHBaseTableInputFormat implements InputFormat for HBase storage handler tables, decorating an underlying HBase TableInputFormat with extra Hive logic such as column pruning and filter pushdown.
org.apache.hadoop.hive.hbase.HiveHBaseTableOutputFormat	HiveHBaseTableOutputFormat implements HiveOutputFormat for HBase tables.
org.apache.hadoop.hive.hbase.LazyHBaseCellMap	LazyHBaseCellMap refines LazyMap with HBase column mapping.
org.apache.hadoop.hive.hwi.HWIContextListener	After getting a contextInitialized event, this component starts an instance of the HiveSessionManager.
org.apache.hadoop.hive.hwi.HWIServer	This is the entry point for HWI.
org.apache.hadoop.hive.hwi.HWISessionItem	HWISessionItem can be viewed as a wrapper for a Hive shell.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.hwi.HWISessionManager	HiveSessionManager is a Runnable started inside a web application context.
org.apache.hadoop.hive.metastore.Deadline	Monitors long running methods in a thread.
org.apache.hadoop.hive.metastore.HiveAlterHandler	Hive specific implementation of alter.
org.apache.hadoop.hive.metastore.HiveMetaStore	Removes application logic to a separate interface.
org.apache.hadoop.hive.metastore.HiveMetaStoreClient	Hive Metastore Client.
org.apache.hadoop.hive.metastore.HiveMetaStoreFsImpl	Class to handle methods for filesystem and data related to metastore.
org.apache.hadoop.hive.metastore.LockComponentBuilder	A builder for LockComponents.
org.apache.hadoop.hive.metastore.LockRequestBuilder	Builder class to make constructing LockRequest easier.
org.apache.hadoop.hive.metastore.MetaStoreSchemaInfo	Information about metastore schemas stored in database.
org.apache.hadoop.hive.metastore.MetaStoreUtils	Utilities to handle metastore data.
org.apache.hadoop.hive.metastore.ObjectStore	Interface between the application logic and the database store that contains the objects.
org.apache.hadoop.hive.metastore.PartitionDropOptions	Generalizes the switches for dropPartitions().
org.apache.hadoop.hive.metastore.RetryingHMSHandler	Handler for hive metastore.
org.apache.hadoop.hive.metastore.RetryingMetaStoreClient	RetryingMetaStoreClient.
org.apache.hadoop.hive.metastore.StatObjectConverter	Contains conversion logic that creates Thrift stat objects from JDO stat objects and plain arrays from DirectSQL.
org.apache.hadoop.hive.metastore.Warehouse	Represents a warehouse where data of Hive tables is stored.
org.apache.hadoop.hive.metastore.api.AbortTxnRequest	Aborting transaction request.
org.apache.hadoop.hive.metastore.api.AddDynamicPartitions	Adding dynamic partitions.
org.apache.hadoop.hive.metastore.api.AddPartitionsRequest	Adding partition request.
org.apache.hadoop.hive.metastore.api.AddPartitionsResult	Adding partition result.
org.apache.hadoop.hive.metastore.api.AggrStats	Aggregation statistics.
org.apache.hadoop.hive.metastore.api.BinaryColumnStatsData	Binary column statistics data.
org.apache.hadoop.hive.metastore.api.BooleanColumnStatsData	Boolean column statistics data.
org.apache.hadoop.hive.metastore.api.CheckLockRequest	Checking request on acquiring lock.
org.apache.hadoop.hive.metastore.api.ColumnStatistics	Column statistics class.
org.apache.hadoop.hive.metastore.api.ColumnStatisticsDesc	Column statistics description.
org.apache.hadoop.hive.metastore.api.ColumnStatisticsObj	Column statistics object.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.metastore.api.CommitTxnRequest	Commit transaction request.
org.apache.hadoop.hive.metastore.api.CompactionRequest	Compaction request.
org.apache.hadoop.hive.metastore.api.CurrentNotificationEventId	Current notification event ID.
org.apache.hadoop.hive.metastore.api.Database	Class that describes database.
org.apache.hadoop.hive.metastore.api.Date	Date class.
org.apache.hadoop.hive.metastore.api.DateColumnStatsData	Date column statistics data.
org.apache.hadoop.hive.metastore.api.Decimal	Handling decimal type.
org.apache.hadoop.hive.metastore.api.DecimalColumnStatsData	Decimal column statistics data.
org.apache.hadoop.hive.metastore.api.DoubleColumnStatsData	Double column statistics data.
org.apache.hadoop.hive.metastore.api.DropPartitionsExpr	Drop partitions expression.
org.apache.hadoop.hive.metastore.api.DropPartitionsRequest	Drop partitions request.
org.apache.hadoop.hive.metastore.api.DropPartitionsResult	Drop partitions result.
org.apache.hadoop.hive.metastore.api.EnvironmentContext	Environment context structure.
org.apache.hadoop.hive.metastore.api.FieldSchema	Field schema structure.
org.apache.hadoop.hive.metastore.api.FireEventRequest	Fire event request.
org.apache.hadoop.hive.metastore.api.FireEventResponse	Fire event response.
org.apache.hadoop.hive.metastore.api.Function	Function structure.
org.apache.hadoop.hive.metastore.api.GetOpenTxnsInfoResponse	Getter for open transactions information about response.
org.apache.hadoop.hive.metastore.api.GetOpenTxnsResponse	Getter for open transactions response.
org.apache.hadoop.hive.metastore.api.GetPrincipalsInRoleRequest	Getting request for principals in role.
org.apache.hadoop.hive.metastore.api.GetPrincipalsInRoleResponse	Getting response for principals in role.
org.apache.hadoop.hive.metastore.api.GetRoleGrantsForPrincipalRequest	Getting request for granting role for principal.
org.apache.hadoop.hive.metastore.api.GetRoleGrantsForPrincipalResponse	Getting response for granting role for principal.
org.apache.hadoop.hive.metastore.api.GrantRevokePrivilegeRequest	Request for revoking granted privilege.
org.apache.hadoop.hive.metastore.api.GrantRevokePrivilegeResponse	Response for revoking granted privilege.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.metastore.api.GrantRevokeRoleRequest	Request for revoking granted role.
org.apache.hadoop.hive.metastore.api.GrantRevokeRoleResponse	Response for revoking granted role.
org.apache.hadoop.hive.metastore.api.HeartbeatRequest	Request for heartbeat.
org.apache.hadoop.hive.metastore.api.HeartbeatTxnRangeRequest	Request for transaction range request.
org.apache.hadoop.hive.metastore.api.HeartbeatTxnRangeResponse	Response for transaction range response.
org.apache.hadoop.hive.metastore.api.HiveObjectPrivilege	Description of privileges for hive object.
org.apache.hadoop.hive.metastore.api.HiveObjectRef	Hive object reference.
org.apache.hadoop.hive.metastore.api.Index	Description of index.
org.apache.hadoop.hive.metastore.api.InsertEventRequestData	Class to handle data about insert event on request.
org.apache.hadoop.hive.metastore.api.LockComponent	Description of lock component.
org.apache.hadoop.hive.metastore.api.LockRequest	Description of lock request.
org.apache.hadoop.hive.metastore.api.LockResponse	Description of lock response.
org.apache.hadoop.hive.metastore.api.LongColumnStatsData	Description of long column statistics data.
org.apache.hadoop.hive.metastore.api.NotificationEvent	Description of notification event.
org.apache.hadoop.hive.metastore.api.NotificationEventRequest	Description of notification event on request.
org.apache.hadoop.hive.metastore.api.NotificationEventResponse	Description of notification event on response.
org.apache.hadoop.hive.metastore.api.OpenTxnRequest	Description of open transactions on request.
org.apache.hadoop.hive.metastore.api.OpenTxnsResponse	Description of open transactions on response.
org.apache.hadoop.hive.metastore.api.Order	Description of order.
org.apache.hadoop.hive.metastore.api.Partition	Description of partition.
org.apache.hadoop.hive.metastore.api.PartitionListComposingSpec	Description of partition list composing specification.
org.apache.hadoop.hive.metastore.api.PartitionSpec	Description of partition specification.
org.apache.hadoop.hive.metastore.api.PartitionSpecWithSharedSD	Description of partition specification with shared sd.
org.apache.hadoop.hive.metastore.api.PartitionWithoutSD	Description of partition without sd.
org.apache.hadoop.hive.metastore.api.PartitionsByExprRequest	Description of partitions by expression request.
org.apache.hadoop.hive.metastore.api.PartitionsByExprResult	Description of partitions by expression result.
org.apache.hadoop.hive.metastore.api.PartitionsStatsRequest	Description of partition statistics request.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.metastore.api.PartitionsStatsResult	Description of partitions statistics result.
org.apache.hadoop.hive.metastore.api.PrincipalPrivilegeSet	Description of setting principal privilege.
org.apache.hadoop.hive.metastore.api.PrivilegeBag	Description of privilege bag.
org.apache.hadoop.hive.metastore.api.PrivilegeGrantInfo	Description of granted privilege info.
org.apache.hadoop.hive.metastore.api.ResourceUri	Description of resource URI.
org.apache.hadoop.hive.metastore.api.Role	Description of role.
org.apache.hadoop.hive.metastore.api.RolePrincipalGrant	Description of granted principal role.
org.apache.hadoop.hive.metastore.api.Schema	Description of schema.
org.apache.hadoop.hive.metastore.api.SerDeInfo	Description of serializer deserializer information.
org.apache.hadoop.hive.metastore.api.SetPartitionsStatsRequest	Description of setting partition statistics on request.
org.apache.hadoop.hive.metastore.api.ShowCompactRequest	Show compaction on request.
org.apache.hadoop.hive.metastore.api.ShowCompactResponse	Show compaction on response.
org.apache.hadoop.hive.metastore.api.ShowCompactResponseElement	Show compaction response element.
org.apache.hadoop.hive.metastore.api.ShowLocksRequest	Show locks on request.
org.apache.hadoop.hive.metastore.api.ShowLocksResponse	Show locks on response.
org.apache.hadoop.hive.metastore.api.ShowLocksResponseElement	Show locks response element.
org.apache.hadoop.hive.metastore.api.SkewedInfo	Description for skewed information.
org.apache.hadoop.hive.metastore.api.StorageDescriptor	Description for storage descriptor.
org.apache.hadoop.hive.metastore.api.StringColumnStatsData	Description for string column statistics data.
org.apache.hadoop.hive.metastore.api.Table	Description for data.
org.apache.hadoop.hive.metastore.api.TableStatsRequest	Description for table statistics request.
org.apache.hadoop.hive.metastore.api.TableStatsResult	Description for table statistics result.
org.apache.hadoop.hive.metastore.api.AlreadyExistsException	Custom exception to handle already exists error.
org.apache.hadoop.hive.metastore.api.ConfigValSecurityException	Custom exception to handle configuration value security error.
org.apache.hadoop.hive.metastore.api.IndexAlreadyExistsException	Custom exception to handle index already exists error.
org.apache.hadoop.hive.metastore.api.InvalidInputException	Custom exception for invalid input error.
org.apache.hadoop.hive.metastore.api.InvalidObjectException	Custom exception for invalid object error.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.metastore.api.InvalidOperationException	Custom exception for invalid operation error.
org.apache.hadoop.hive.metastore.api.InvalidPartitionException	Custom exception for invalid partition error.
org.apache.hadoop.hive.metastore.api.MetaException	Custom exception for metastore related error.
org.apache.hadoop.hive.metastore.api.NoSuchLockException	Custom exception in case of invalid lock.
org.apache.hadoop.hive.metastore.api.NoSuchObjectException	Custom exception in case of invalid object.
org.apache.hadoop.hive.metastore.api.NoSuchTxnException	Custom exception in case of invalid transaction.
org.apache.hadoop.hive.metastore.api.TxnAbortedException	Custom exception in case of aborted transaction.
org.apache.hadoop.hive.metastore.api.TxnOpenException	Custom exception in case of not close transaction.
org.apache.hadoop.hive.metastore.api.UnknownDBException	Custom exception in case of unknown database.
org.apache.hadoop.hive.metastore.api.UnknownPartitionException	Custom exception in case of unknown partition.
org.apache.hadoop.hive.metastore.api.UnknownTableException	Custom partition in case of unknown table.
org.apache.hadoop.hive.metastore.events.EventCleanerTask	Cleaning tasks from event table.
org.apache.hadoop.hive.metastore.parser.ExpressionTree	Represents the filter as a binary tree.
org.apache.hadoop.hive.metastore.txn.CompactionInfo	Information on a possible or running compaction.
org.apache.hadoop.hive.metastore.txn.CompactionInfo	Utility methods for creating and destroying txn database/ schema, plus methods for querying against metastore tables.
org.apache.hadoop.hive.ql.Context	Context for Semantic Analyzers.
org.apache.hadoop.hive.ql.Driver	Driver to process commands on cluster.
org.apache.hadoop.hive.ql.QueryPlan	QueryPlan can be serialized to disk to restart/resume the progress of it in the future, either within or outside of the current JVM.
org.apache.hadoop.hive.ql.QueryProperties	QueryProperties.
org.apache.hadoop.hive.ql.exec.AbstractFileMergeOperator	Fast file merge operator for ORC and RCfile.
org.apache.hadoop.hive.ql.exec.AbstractMapJoinOperator	Class to handle join input's join keys.
org.apache.hadoop.hive.ql.exec.AppMasterEventOperator	AppMasterEventOperator sends any rows it receives to the Tez AM.
org.apache.hadoop.hive.ql.exec.AutoProgressor	AutoProgressor periodically sends updates to the job tracker so that it doesn't consider this task attempt dead if there is a long period of inactivity.
org.apache.hadoop.hive.ql.exec.CollectOperator	Buffers rows emitted by other operators.
org.apache.hadoop.hive.ql.exec.ColumnStatsTask	ColumnStatsTask implementation.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.ColumnStatsUpdateTask	ColumnStatsUpdateTask implementation.
org.apache.hadoop.hive.ql.exec.CommonJoinOperator	Join operator implementation.
org.apache.hadoop.hive.ql.exec.CommonMergeJoinOperator	Consolidate the join algorithms to either hash based joins (MapJoinOperator) or sort-merge based joins, this operator is being introduced.
org.apache.hadoop.hive.ql.exec.ConditionalTask	Conditional Task implementation.
org.apache.hadoop.hive.ql.exec.DDLTask	DDLTask implementation.
org.apache.hadoop.hive.ql.exec.DefaultBucketMatcher	Finding right bucket.
org.apache.hadoop.hive.ql.exec.DemuxOperator	DemuxOperator is an operator used by MapReduce Jobs optimized by CorrelationOptimizer.
org.apache.hadoop.hive.ql.exec.DummyStoreOperator	For SortMerge joins, this is a dummy operator, which stores the row for the small table before it reaches the sort merge join operator.
org.apache.hadoop.hive.ql.exec.ExplainTask	ExplainTask implementation.
org.apache.hadoop.hive.ql.exec.FetchOperator	FetchTask implementation.
org.apache.hadoop.hive.ql.exec.FetchTask	FetchTask implementation.
org.apache.hadoop.hive.ql.exec.FileSinkOperator	File Sink operator implementation.
org.apache.hadoop.hive.ql.exec.FilterOperator	Filter operator implementation.
org.apache.hadoop.hive.ql.exec.ForwardOperator	Forward Operator Just forwards.
org.apache.hadoop.hive.ql.exec.FunctionRegistry	FunctionRegistry.
org.apache.hadoop.hive.ql.exec.FunctionTask	FunctionTask.
org.apache.hadoop.hive.ql.exec.GroupByOperator	GroupBy operator implementation.
org.apache.hadoop.hive.ql.exec.HashTableDummyOperator	Hash table operator implementation.
org.apache.hadoop.hive.ql.exec.HashTableSinkOperator	Hash table sink operator implementation.
org.apache.hadoop.hive.ql.exec.JoinOperator	Join operator implementation.
org.apache.hadoop.hive.ql.exec.LateralViewForwardOperator	LateralViewForwardOperator.
org.apache.hadoop.hive.ql.exec.LateralViewJoinOperator	The lateral view join operator is used for FROM src LATERAL VIEW udtf()...
org.apache.hadoop.hive.ql.exec.LimitOperator	Limit operator implementation Limits the number of rows to be passed on.
org.apache.hadoop.hive.ql.exec.ListSinkOperator	For fetch task with operator tree, row read from FetchOperator is processed via operator tree and finally arrives to this operator.
org.apache.hadoop.hive.ql.exec.MapJoinOperator	Map side Join operator implementation.
org.apache.hadoop.hive.ql.exec.MapOperator	Map operator.
org.apache.hadoop.hive.ql.exec.MapredContext	Runtime context of MapredTask providing additional information to GenericUDF
org.apache.hadoop.hive.ql.exec.MoveTask	MoveTask implementation.
org.apache.hadoop.hive.ql.exec.MuxOperator	MuxOperator is used in the Reduce side of MapReduce jobs optimized by Correlation Optimizer.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.ObjectCacheFactory	ObjectCacheFactory returns the appropriate cache depending on settings in the hive conf.
org.apache.hadoop.hive.ql.exec.Operator	Base operator implementation.
org.apache.hadoop.hive.ql.exec.OperatorFactory	OperatorFactory.
org.apache.hadoop.hive.ql.exec.OperatorUtils	Utilities to handle operators.
org.apache.hadoop.hive.ql.exec.OrcFileMergeOperator	Fast file merge operator for ORC files.
org.apache.hadoop.hive.ql.exec.PTFOperator	Class to handle partitioned table functions operators.
org.apache.hadoop.hive.ql.exec.PTFPartition	Represents a collection of rows that is acted upon by a TableFunction or a WindowFunction.
org.apache.hadoop.hive.ql.exec.PTFRollingPartition	Represents a collection of rows that is acted upon by a TableFunction or a WindowFunction.
org.apache.hadoop.hive.ql.exec.PTFUtils	Utilities to handle partitioned table functions.
org.apache.hadoop.hive.ql.exec.PartitionKeySampler	Class to handle partition key sampler.
org.apache.hadoop.hive.ql.exec.RCFileMergeOperator	Fast file merge operator for RC files.
org.apache.hadoop.hive.ql.exec.ReduceSinkOperator	Reduce Sink Operator sends output to the reduce stage.
org.apache.hadoop.hive.ql.exec.Registry	Function registry.
org.apache.hadoop.hive.ql.exec.SMBMapJoinOperator	Sorted Merge Map Join Operator.
org.apache.hadoop.hive.ql.exec.ScriptOperator	ScriptOperator.
org.apache.hadoop.hive.ql.exec.SelectOperator	Select operator implementation.
org.apache.hadoop.hive.ql.exec.SkewJoinHandler	At runtime in Join, output big keys in one table into one corresponding directories, and all same keys in other tables into different dirs (one for each table).
org.apache.hadoop.hive.ql.exec.SparkHashTableSinkOperator	Operator for spark hashtable sink.
org.apache.hadoop.hive.ql.exec.StatsNoJobTask	StatsNoJobTask is used in cases where stats collection is the only task for the given query (no parent MR or Tez job).
org.apache.hadoop.hive.ql.exec.TableScanOperator	Table Scan Operator If the data is coming from the map-reduce framework, just forward it.
org.apache.hadoop.hive.ql.exec.Task	Task implementation.
org.apache.hadoop.hive.ql.exec.TaskResult	TaskResult implementation.
org.apache.hadoop.hive.ql.exec.TemporaryHashSinkOperator	Operator temporary hash sink.
org.apache.hadoop.hive.ql.exec.TerminalOperator	Terminal Operator Base Class.
org.apache.hadoop.hive.ql.exec.TezDummyStoreOperator	A dummy store operator same as the dummy store operator but for tez.
org.apache.hadoop.hive.ql.exec.TopNHash	Stores binary key/value in sorted manner to get top-n key/value TODO: rename to TopNHeap?
org.apache.hadoop.hive.ql.exec.UDTFOperator	UDTFOperator.
org.apache.hadoop.hive.ql.exec.UnionOperator	Union Operator Just forwards.
org.apache.hadoop.hive.ql.exec.Utilities	Utilities.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.mr.ExecDriver	ExecDriver is the central class in co-ordinating execution of any map-reduce task.
org.apache.hadoop.hive.ql.exec.mr.ExecMapper	ExecMapper is the generic Map class for Hive.
org.apache.hadoop.hive.ql.exec.mr.ExecMapperContext	ExecMapperContext is the generic Map context class for Hive.
org.apache.hadoop.hive.ql.exec.mr.HadoopJobExecHelper	Handle information about hadoop job.
org.apache.hadoop.hive.ql.exec.mr.JobDebugger	JobDebugger takes a RunningJob that has failed and grabs the top 4 failing tasks and outputs this information to the Hive CLI.
org.apache.hadoop.hive.ql.exec.mr.MapredLocalTask	MapredLocalTask represents any local work (i.e.: client side work) that hive needs to execute.
org.apache.hadoop.hive.ql.exec.mr.Throttle	Intelligence to make clients wait if the cluster is in a bad state.
org.apache.hadoop.hive.ql.exec.persistence.BytesBytesMultiHashMap	HashMap that maps byte arrays to byte arrays with limited functionality necessary for MapJoin hash tables, with small memory overhead.
org.apache.hadoop.hive.ql.exec.persistence.HashMapWrapper	Simple wrapper for persistent Hashmap implementing only the put/get/remove/clear interface.
org.apache.hadoop.hive.ql.exec.persistence.HybridHashTableContainer	Hash table container that can have many partitions -- each partition has its own hashmap, as well as row container for small table and big table.
org.apache.hadoop.hive.ql.exec.persistence.KeyValueContainer	An eager key/value container that puts every row directly to output stream.
org.apache.hadoop.hive.ql.exec.persistence.MapJoinBytesTableContainer	Table container that serializes keys and values using LazyBinarySerDe into BytesBytesMultiHashMap, with very low memory overhead.
org.apache.hadoop.hive.ql.exec.persistence.MapJoinKey	The base class for MapJoinKey.
org.apache.hadoop.hive.ql.exec.persistence.MapJoinTableContainerSerDe	Serialization/deserialization of table container for join.
org.apache.hadoop.hive.ql.exec.persistence.ObjectContainer	An eager object container that puts every row directly to output stream.
org.apache.hadoop.hive.ql.exec.persistence.RowContainer	Simple persistent container for rows.
org.apache.hadoop.hive.ql.exec.persistence.UnwrapRowContainer	Unwraps values from current key with valueIndex in mapjoin desc.
org.apache.hadoop.hive.ql.exec.spark.GroupByShuffler	Shuffle group by operator.
org.apache.hadoop.hive.ql.exec.spark.HiveSparkClientFactory	Factory class for spark client.
org.apache.hadoop.hive.ql.exec.spark.LocalHiveSparkClient	LocalSparkClient submit Spark job in local driver, it's responsible for build spark client environment and execute spark work.
org.apache.hadoop.hive.ql.exec.spark.MapInput	Input for mapper.
org.apache.hadoop.hive.ql.exec.spark.MapTran	Mapper tran.
org.apache.hadoop.hive.ql.exec.spark.ReduceTran	Reduce tran.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.spark.RemoteHiveSparkClient	RemoteSparkClient is a wrapper of org.apache.hive.spark.client.SparkClient, which wrap a spark job request and send to an remote SparkContext.
org.apache.hadoop.hive.ql.exec.spark.ShuffleTran	Shuffle tran.
org.apache.hadoop.hive.ql.exec.spark.SortByShuffler	Sorting class for shuffler.
org.apache.hadoop.hive.ql.exec.spark.SparkTask	Description of spark task.
org.apache.hadoop.hive.ql.exec.spark.SparkUtilities	Contains utilities methods used as part of Spark tasks.
org.apache.hadoop.hive.ql.exec.spark.session.SparkSessionImpl	Implementation of spark session.
org.apache.hadoop.hive.ql.exec.spark.status.impl.JobMetricsListener	Listener for job metrics.
org.apache.hadoop.hive.ql.exec.spark.status.impl.LocalSparkJobStatus	Spark job local status.
org.apache.hadoop.hive.ql.exec.spark.status.impl.RemoteSparkJobStatus	Used with remove spark client.
org.apache.hadoop.hive.ql.exec.tez.DagUtils	DagUtils.
org.apache.hadoop.hive.ql.exec.tez.DynamicPartitionPruner	DynamicPartitionPruner takes a list of assigned partitions at runtime (split generation) and prunes them using events generated during execution of the dag.
org.apache.hadoop.hive.ql.exec.tez.HiveSplitGenerator	Generates splits inside the AM on the cluster.
org.apache.hadoop.hive.ql.exec.tez.MapRecordProcessor	Process input from tez LogicalInput and write output - for a map plan Just pump the records through the query plan.
org.apache.hadoop.hive.ql.exec.tez.MapRecordSource	Process input from tez LogicalInput and write output - for a map plan Just pump the records through the query plan.
org.apache.hadoop.hive.ql.exec.tez.MergeFileRecordProcessor	Record processor for fast merging of files.
org.apache.hadoop.hive.ql.exec.tez.RecordProcessor	Process input from tez LogicalInput and write output It has different subclasses for map and reduce processing
org.apache.hadoop.hive.ql.exec.tez.ReduceRecordProcessor	Process input from tez LogicalInput and write output - for a map plan Just pump the records through the query plan.
org.apache.hadoop.hive.ql.exec.tez.ReduceRecordSource	Process input from tez LogicalInput and write output - for a map plan Just pump the records through the query plan.
org.apache.hadoop.hive.ql.exec.tez.SplitGrouper	SplitGrouper is used to combine splits based on head room and locality.
org.apache.hadoop.hive.ql.exec.tez.TezJobMonitor	TezJobMonitor keeps track of a tez job while it's being executed.
org.apache.hadoop.hive.ql.exec.tez.TezProcessor	Hive processor for Tez that forms the vertices in Tez and processes the data.
org.apache.hadoop.hive.ql.exec.tez.TezSessionPoolManager	This class is for managing multiple tez sessions particularly when HiveServer2 is being used to submit queries.
org.apache.hadoop.hive.ql.exec.tez.TezSessionState	Holds session state related to Tez

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.tez.TezTask	TezTask handles the execution of TezWork.
org.apache.hadoop.hive.ql.exec.tez.tools.KeyValueInputMerger	A KeyValuesReader implementation that returns a sorted stream of key-values by doing a sorted merge of the key-value in LogicalInputs.
org.apache.hadoop.hive.ql.exec.tez.tools.KeyValuesInputMerger	A KeyValuesReader implementation that returns a sorted stream of key-values by doing a sorted merge of the key-value in LogicalInputs.
org.apache.hadoop.hive.ql.exec.vector.BytesColumnVector	This class supports string and binary data by value reference.
org.apache.hadoop.hive.ql.exec.vecto.ColumnVector	ColumnVector contains the shared structure for the sub-types, including NULL information, and whether this vector repeats, i.e.
org.apache.hadoop.hive.ql.exec.vector.DecimalColumnVector	A vector of HiveDecimalWritable objects.
org.apache.hadoop.hive.ql.exec.vector.DoubleColumnVector	This class represents a nullable double precision floating point column vector.
org.apache.hadoop.hive.ql.exec.vector.LongColumnVector	This class represents a nullable int column vector.
org.apache.hadoop.hive.ql.exec.vector.TimestampUtils	Utilities for Timestamps and the relevant conversions.
org.apache.hadoop.hive.ql.exec.vector.VectorAppMasterEventOperator	App Master Event operator implementation.
org.apache.hadoop.hive.ql.exec.vector.VectorAssignRow	This class assigns specified columns of a row from a Writable row objects.
org.apache.hadoop.hive.ql.exec.vector.VectorColumnOrderedMap	This class collects column information for mapping vector columns, including the hive type name.
org.apache.hadoop.hive.ql.exec.vector.VectorColumnSetInfo	Class to keep information on a set of typed vector columns.
org.apache.hadoop.hive.ql.exec.vector.VectorCopyRow	This class copies specified columns of a row from one VectorizedRowBatch to another.
org.apache.hadoop.hive.ql.exec.vector.VectorDeserializeRow	This class deserializes a serialization format into a row of a VectorizedRowBatch.
org.apache.hadoop.hive.ql.exec.vector.VectorExtractRow	This class extracts specified VectorizedRowBatch row columns into writables.
org.apache.hadoop.hive.ql.exec.vector.VectorFileSinkOperator	File Sink operator implementation.
org.apache.hadoop.hive.ql.exec.vector.VectorFilterOperator	Filter operator implementation.
org.apache.hadoop.hive.ql.exec.vector.VectorGroupByOperator	Vectorized GROUP BY operator implementation.
org.apache.hadoop.hive.ql.exec.vector.VectorHashKeyWrapper	A hash map key wrapper for vectorized processing.
org.apache.hadoop.hive.ql.exec.vector.VectorLimitOperator	Limit operator implementation Limits the number of rows to be passed on.
org.apache.hadoop.hive.ql.exec.vector.VectorMapJoinBaseOperator	The *NON-NATIVE* base vector map join operator class used by VectorMapJoinOperator and VectorMapJoinOuterFilteredOperator.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.vector.VectorMapJoinOperator	The vectorized version of the MapJoinOperator.
org.apache.hadoop.hive.ql.exec.vector.VectorMapJoinOuterFilteredOperator	This is the *NON-NATIVE* vector map join operator for just LEFT OUTER JOIN and filtered.
org.apache.hadoop.hive.ql.exec.vector.VectorMapOperator	The vectorized MapOperator.
org.apache.hadoop.hive.ql.exec.vector.VectorReduceSinkOperator	The vectorized reduce sink operator.
org.apache.hadoop.hive.ql.exec.vector.VectorSMBMapJoinOperator	VectorSMBJoinOperator.
org.apache.hadoop.hive.ql.exec.vector.VectorSelectOperator	Select operator implementation.
org.apache.hadoop.hive.ql.exec.vector.VectorSerializeRow	This class serializes columns from a row in a VectorizedRowBatch into a serialization format.
org.apache.hadoop.hive.ql.exec.vector.VectorizationContext	Context class for vectorization execution.
org.apache.hadoop.hive.ql.exec.vector.VectorizedBatchUtil	The vectorized MapOperator.
org.apache.hadoop.hive.ql.exec.vector.VectorizedRowBatch	A VectorizedRowBatch is a set of rows, organized with each column as a vector.
org.apache.hadoop.hive.ql.exec.vector.VectorizedRowBatchCtx	Context for Vectorized row batch.
org.apache.hadoop.hive.ql.exec.vector.expressions.CastDecimalToTimestamp	Type cast decimal to timestamp.
org.apache.hadoop.hive.ql.exec.vector.expressions.CastTimestampToDecimal	To be used to cast timestamp to decimal.
org.apache.hadoop.hive.ql.exec.vector.expressions.ColAndCol	Evaluate AND of 2 or more boolean columns and store the boolean result in the output boolean column.
org.apache.hadoop.hive.ql.exec.vector.expressions.ColOrCol	Evaluate OR of 2 or more boolean columns and store the boolean result in the output boolean column.
org.apache.hadoop.hive.ql.exec.vector.expressions.ConstantVectorExpression	Constant is represented as a vector with repeating values.
org.apache.hadoop.hive.ql.exec.vector.expressions.DecimalUtil	Utility functions for vector operations on decimal values.
org.apache.hadoop.hive.ql.exec.vector.expressions.FilterExprOrExpr	Represents an Or expression.
org.apache.hadoop.hive.ql.exec.vector.expressions.FilterStringColumnInList	Evaluate an IN filter on a batch for a vector of strings.
org.apache.hadoop.hive.ql.exec.vector.expressions.FuncRoundWithNumDigitsDecimalToDecimal	Function for rounding decimals.
org.apache.hadoop.hive.ql.exec.vector.expressions.MathExpr	Math expression evaluation helper functions.
org.apache.hadoop.hive.ql.exec.vector.expressions.NullUtil	Utility functions to handle null propagation.
org.apache.hadoop.hive.ql.exec.vector.expressions.StringColumnInList	Evaluate an IN boolean expression (not a filter) on a batch for a vector of strings.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.vector.expressions.StringExpr	String expression evaluation helper functions.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFDateAddColCol	Vectorized user defined function for adding columns.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFDateAddColScalar	Vectorized user defined function for adding column and scalar.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFDateDiffColCol	Vectorized user defined function forfinding difference between two columns.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFDateLong	Vectorized version of TO_DATE(TIMESTAMP)/ TO_DATE(DATE).
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFDateString	Vectorized version of TO_DATE(STRING) As TO_DATE() now returns DATE type, this should be the same behavior as the DATE cast operator.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinCommonOperator	Common operator class for native vectorized map join.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinGenerateResultOperator	Contains methods for generating vectorized join results and forwarding batchs.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinInnerBigOnlyGenerateResultOperator	This class has methods for generating vectorized join results for the big table only variation of inner joins.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinInnerBigOnlyLongOperator	Specialized class for doing a vectorized map join that is an inner join on a Single-Column Long and only big table columns appear in the join result so a hash multi-set is used.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinInnerBigOnlyMultiKeyOperator	Specialized class for doing a vectorized map join that is an inner join on Multi-Key and only big table columns appear in the join result so a hash multi-set is used.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinInnerBigOnlyStringOperator	Specialized class for doing a vectorized map join that is an inner join on a Single-Column String and only big table columns appear in the join result so a hash multi-set is used.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinInnerGenerateResultOperator	Contains methods for generating vectorized join results for inner joins.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinInnerLongOperator	Specialized class for doing a vectorized map join that is an inner join on a Single-Column Long and only big table columns appear in the join result so a hash multi-set is used.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinInnerMultiKeyOperator	Specialized class for doing a vectorized map join that is an inner join on a Multi-Key using a hash map.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinInnerStringOperator	Specialized class for doing a vectorized map join that is an inner join on a Single-Column String using a hash map.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinLeftSemiGenerateResultOperator	Contains methods for generating vectorized join results for left semi joins.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinLeftSemiLongOperator	Specialized class for doing a vectorized map join that is an left semi join on a Single-Column Long using a hash set.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinLeftSemiMultiKeyOperator	Specialized class for doing a vectorized map join that is an left semi join on Multi-Key using hash set.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinLeftSemiStringOperator	Specialized class for doing a vectorized map join that is an left semi join on a Single-Column String using a hash set.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinOuterGenerateResultOperator	Contains methods for generating vectorized join results for outer joins.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinOuterLongOperator	Specialized class for doing a vectorized map join that is an outer join on a Single-Column Long using a hash map.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinOuterMultiKeyOperator	Specialized class for doing a vectorized map join that is an outer join on Multi-Key using a hash map.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinOuterStringOperator	Specialized class for doing a vectorized map join that is an outer join on a Single-Column String using a hash map.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.VectorMapJoinRowBytesContainer	An eager bytes container that puts row bytes to an output stream.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastBytesHashMap	Bytes key hash map optimized for vector map join. This is the abstract base for the multi-key and string bytes key hash map implementations.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastBytesHashMultiSet	Bytes key hash multi-set optimized for vector map join. This is the abstract base for the multi-key and string bytes key hash multi-set implementations.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastBytesHashUtil	Utilities for bytes key hash multi-set optimized for vector map join.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastHashTable	Vector map join fast hash table.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastKeyStore	Vector map join fast key store.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastLongHashMap	Vector map join fact long hash map.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastLongHashMultiSet	Vector mapr join fast long hash multi set.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastLongHashSet	A single LONG key hash set optimized for vector map join.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastLongHashTable	A single long value map optimized for vector map join.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastLongHashUtil	Utilities for vector map join of single long value.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastMultiKeyHashMap	A multi-key value hash map optimized for vector map join. The key is stored as the provided bytes (uninterpreted).
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastMultiKeyHashMultiSet	A multi-key hash multi-set optimized for vector map join. The key is stored as the provided bytes (uninterpreted).
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastMultiKeyHashSet	A multi-key hash set optimized for vector map join. The key is stored as the provided bytes (uninterpreted).
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastStringCommon	A single byte array value hash map optimized for vector map join.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastTableContainer	HashTableLoader for Tez constructs the hashtable from records read from a broadcast edge.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.vector.mapjoin.hashtable.VectorMapJoinHashMapResult	Abstract class for a hash map result. For reading the values, one-by-one.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.hashtable.VectorMapJoinHashTableResult	Root abstract class for a hash table result.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.optimized.VectorMapJoinOptimizedCreateHashTable	Create hash table for vector map join.
org.apache.hadoop.hive.ql.hooks.HookContext	Hook Context keeps all the necessary information for all the hooks.
org.apache.hadoop.hive.ql.hooks.PostExecutePrinter	Implementation of a post execute hook that simply prints out its parameters to standard output.
org.apache.hadoop.hive.ql.hooks.PreExecutePrinter	Implementation of a pre execute hook that simply prints out its parameters to standard output.
org.apache.hadoop.hive.ql.index.HiveIndex	Holds index related constants.
org.apache.hadoop.hive.ql.index.HiveIndexQueryContext	Used to pass information between the IndexProcessor and the plugin IndexHandler during query processing
org.apache.hadoop.hive.ql.index.HiveIndexResult	HiveIndexResult parses the input stream from an index query to generate a list of file splits to query.
org.apache.hadoop.hive.ql.index.HiveIndexedInputFormat	Input format for doing queries that use indexes.
org.apache.hadoop.hive.ql.index.IndexPredicateAnalyzer	IndexPredicateAnalyzer decomposes predicates, separating the parts which can be satisfied by an index from the parts which cannot.
org.apache.hadoop.hive.ql.index.IndexSearchCondition	IndexSearchCondition represents an individual search condition found by IndexPredicateAnalyzer.
org.apache.hadoop.hive.ql.index.TableBasedIndexHandler	Index handler for indexes that use tables to store indexes.
org.apache.hadoop.hive.ql.index.compact.HiveCompactIndexInputFormat	Hive compact index input format.
org.apache.hadoop.hive.ql.io.AcidUtils	Utilities that are shared by all of the ACID input and output formats.
org.apache.hadoop.hive.ql.io.BucketizedHiveInputFormat	BucketizedHiveInputFormat serves the similar function as hiveInputFormat but its getSplits() always group splits from one input file into one wrapper split.
org.apache.hadoop.hive.ql.io.CombineHiveInputFormat	CombineHiveInputFormat is a parameterized InputFormat which looks at the path name and determine the correct InputFormat for that path name from mapredPlan.pathToPartitionInfo().
org.apache.hadoop.hive.ql.io.HiveFileFormatUtils	An util class for various Hive file format tasks.
org.apache.hadoop.hive.ql.io.HiveInputFormat	HiveInputFormat is a parameterized InputFormat which looks at the path name and determine the correct InputFormat for that path name from mapredPlan.pathToPartitionInfo().
org.apache.hadoop.hive.ql.io.IOConstants	Input output constants.
org.apache.hadoop.hive.ql.io.IOContext	IOContext basically contains the position information of the current key/value.
org.apache.hadoop.hive.ql.io.NullRowsInputFormat	NullRowsInputFormat outputs null rows, maximum 100.
org.apache.hadoop.hive.ql.io.OneNullRowInputFormat	OneNullRowInputFormat outputs one null row.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.io.RCFileInputFormat	RCFileInputFormat.
org.apache.hadoop.hive.ql.io.SequenceFileInputFormatChecker	SequenceFileInputFormatChecker.
org.apache.hadoop.hive.ql.io.avro.AvroContainerOutputFormat	Write to an Avro file from a Hive process.
org.apache.hadoop.hive.ql.io.merge.MergeFileMapper	Mapper for fast file merging of ORC and RC files.
org.apache.hadoop.hive.ql.io.merge.MergeFileTask	Task for fast merging of ORC and RC files.
org.apache.hadoop.hive.ql.io.orc.CompressionKind	An enumeration that lists the generic compression algorithms that can be applied to ORC files.
org.apache.hadoop.hive.ql.io.orc.OrcFile	Contains factory methods to read or write ORC files.
org.apache.hadoop.hive.ql.io.orc.OrcFileKeyWrapper	Key for OrcFileMergeMapper task.
org.apache.hadoop.hive.ql.io.orc.OrcFileStripeMergeRecordReader	Record reader for orc file stripe merge.
org.apache.hadoop.hive.ql.io.orc.OrcFileValueWrapper	Value for OrcFileMergeMapper.
org.apache.hadoop.hive.ql.io.orc.OrcInputFormat	A MapReduce/Hive input format for ORC files.
org.apache.hadoop.hive.ql.io.orc.OrcNewSplit	OrcFileSplit.
org.apache.hadoop.hive.ql.io.orc.OrcRecordUpdater	A RecordUpdater where the files are stored as ORC.
org.apache.hadoop.hive.ql.io.orc.OrcSerde	A serde class for ORC.
org.apache.hadoop.hive.ql.io.orc.OrcSplit	OrcFileSplit.
org.apache.hadoop.hive.ql.io.orc.ReaderImpl	Implementation of record reader.
org.apache.hadoop.hive.ql.io.orc.VectorizedOrcInputFormat	A MapReduce/Hive input format for ORC files.
org.apache.hadoop.hive.ql.io.orc.WriterImpl	An ORC file writer.
org.apache.hadoop.hive.ql.io.parquet.LeafFilterFactory	Factory class for leaf filter.
org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat	A Parquet OutputFormat for Hive (with the deprecated package mapred)
org.apache.hadoop.hive.ql.io.parquet.convert.DataWritableRecordConverter	A MapWritableReadSupport, encapsulates the tuples
org.apache.hadoop.hive.ql.io.parquet.convert.ETypeConverter.BinaryConverter	ETypeConverter is an easy way to set the converter for the right type.
org.apache.hadoop.hive.ql.io.parquet.convert.HiveCollectionConverter	Converter for collections.
org.apache.hadoop.hive.ql.io.parquet.convert.HiveGroupConverter	Converter for groups.
org.apache.hadoop.hive.ql.io.parquet.convert.HiveStructConverter	A MapWritableGroupConverter, real converter between hive and parquet types recursively for complex types.
org.apache.hadoop.hive.ql.io.parquet.read.DataWritableReadSupport	A MapWritableReadSupport Manages the translation between Hive and Parquet
org.apache.hadoop.hive.ql.io.parquet.read.ParquetRecordReaderWrapper	Wrapper for parquet record reader.
org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe	A ParquetHiveSerDe for Hive (with the deprecated package mapred)

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.io.parquet.write.DataWritableWriter	DataWritableWriter sends a record to the Parquet API with the expected schema in order to be written to a file.
org.apache.hadoop.hive.ql.io.parquet.write.ParquetRecordWriterWrapper	Wrapper for parquet record writer.
org.apache.hadoop.hive.ql.io.rcfile.stats.PartialScanMapper	PartialScanMapper.
org.apache.hadoop.hive.ql.io.rcfile.stats.PartialScanTask	PartialScanTask.
org.apache.hadoop.hive.ql.io.rcfile.stats.PartialScanWork	Partial Scan Work.
org.apache.hadoop.hive.ql.io.rcfile.truncate.ColumnTruncateMapper	A factory for creating SearchArguments, as well as modifying those created by this factory.
org.apache.hadoop.hive.ql.io.rcfile.truncate.ColumnTruncateTask	Base class for operator graph walker this class takes list of starting ops and walks them one by one.
org.apache.hadoop.hive.ql.io.sarg.SearchArgumentFactory	A factory for creating SearchArguments, as well as modifying those created by this factory.
org.apache.hadoop.hive.ql.lib.DefaultGraphWalker	Base class for operator graph walker this class takes list of starting ops and walks them one by one.
org.apache.hadoop.hive.ql.lib.RuleExactMatch	Implentation of the Rule interface for Nodes Used in Node dispatching to dispatch process/visitor functions for Nodes.
org.apache.hadoop.hive.ql.lockmgr.DbLockManager	An implementation of HiveLockManager for use with org.apache.hadoop.hive.ql.lockmgr.DbTxnManager.
org.apache.hadoop.hive.ql.lockmgr.DbTxnManager	An implementation of HiveTxnManager that stores the transactions in the metastore database.
org.apache.hadoop.hive.ql.lockmgr.HiveLockObject	The class is used to uniquely identify a HiveLockObject.
org.apache.hadoop.hive.ql.lockmgr.LockException	Exception from lock manager.
org.apache.hadoop.hive.ql.lockmgr.zookeeper.ZooKeeperHiveLock	The class is used to uniquely identify ZookeeperHiveLock.
org.apache.hadoop.hive.ql.lockmgr.zookeeper.ZooKeeperHiveLockManager	Zookeeper lock manager.
org.apache.hadoop.hive.ql.log.PerfLogger	PerfLogger.
org.apache.hadoop.hive.ql.metadata.Hive	Contains functions that implement meta data/DDL operations using calls to the metastore.
org.apache.hadoop.hive.ql.metadata.HiveException	Generic exception class for Hive.
org.apache.hadoop.hive.ql.metadata.HiveMetaStoreChecker	Verify that the information in the metastore matches what is on the filesystem.
org.apache.hadoop.hive.ql.metadata.HiveUtils	General collection of helper functions.
org.apache.hadoop.hive.ql.metadata.Partition	A Hive Table Partition: is a fundamental storage unit within a Table.
org.apache.hadoop.hive.ql.metadata.SessionHiveMetaStoreClient	Client for hivemetastore during session.
org.apache.hadoop.hive.ql.metadata.Table	A Hive Table: is a fundamental unit of data in Hive that shares a common schema/DDL.
org.apache.hadoop.hive.ql.metadata.VirtualColumn	Provides metadata that is not stored in table itself.
org.apache.hadoop.hive.ql.metadata.formatting.JsonMetaDataFormatter	Format table and index information for machine readability using json.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.metadata.formatting.MetaDataFormatUtils	This class provides methods to format table and index information.
org.apache.hadoop.hive.ql.optimizer.BucketMapJoinOptimizer	this transformation does bucket map join optimization.
org.apache.hadoop.hive.ql.optimizer.BucketingSortingReduceSinkOptimizer	This transformation does optimization for enforcing bucketing and sorting.
org.apache.hadoop.hive.ql.optimizer.ColumnPruner	Implementation of one of the rule-based optimization steps.
org.apache.hadoop.hive.ql.optimizer.ColumnPrunerProcCtx	This class implements the processor context for Column Pruner.
org.apache.hadoop.hive.ql.optimizer.ColumnPrunerProcFactory	Factory for generating the different node processors used by ColumnPruner.
org.apache.hadoop.hive.ql.optimizer.ConstantPropagate	Implementation of one of the rule-based optimization steps.
org.apache.hadoop.hive.ql.optimizer.ConstantPropagateProcCtx	Implements the processor context for Constant Propagate.
org.apache.hadoop.hive.ql.optimizer.ConstantPropagateProcFactory	Factory for generating the different node processors used by ConstantPropagate.
org.apache.hadoop.hive.ql.optimizer.ConvertJoinMapJoin	ConvertJoinMapJoin is an optimization that replaces a common join (aka shuffle join) with a map join (aka broadcast or fragment replicate join when possible.
org.apache.hadoop.hive.ql.optimizer.GenMRProcContext	Processor Context for creating map reduce task.
org.apache.hadoop.hive.ql.optimizer.GenMapRedUtils	General utility common functions for the Processor to convert operator into map-reduce tasks.
org.apache.hadoop.hive.ql.optimizer.GlobalLimitOptimizer	This optimizer is used to reduce the input size for the query for queries which are specifying a limit.
org.apache.hadoop.hive.ql.optimizer.GroupByOptimizer	This transformation does group by optimization.
org.apache.hadoop.hive.ql.optimizer.IdentityProjectRemover	This optimization tries to remove SelectOperator from tree which don't do any processing except forwarding columns from its parent to its children.
org.apache.hadoop.hive.ql.optimizer.JoinReorder	Implementation of rule-based join table reordering optimization.
org.apache.hadoop.hive.ql.optimizer.LimitPushdownOptimizer	Make RS calculate top-K selection for limit clause.
org.apache.hadoop.hive.ql.optimizer.MapJoinProcessor	Implementation of one of the rule-based map join optimization.
org.apache.hadoop.hive.ql.optimizer.NonBlockingOpDeDupProc	Merges SEL-SEL or FIL-FIL into single operator
org.apache.hadoop.hive.ql.optimizer.ReduceSinkMapJoinProc	This processor addresses the RS-MJ case that occurs in Tez on the small/hash table. The work that RS will be a part of must be connected to the MJ work via a broadcast edge.
org.apache.hadoop.hive.ql.optimizer.SamplePruner	The transformation step that does sample pruning.
org.apache.hadoop.hive.ql.optimizer.SimpleFetchAggregation	Execute final aggregation stage for simple fetch query on fetch task.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.optimizer.SimpleFetchOptimizer	Tries to convert simple fetch query to single fetch task, which fetches rows directly from location of table/ partition.
org.apache.hadoop.hive.ql.optimizer.SkewJoinOptimizer	SkewJoinOptimizer.
org.apache.hadoop.hive.ql.optimizer.SortedDynPartitionOptimizer	When dynamic partitioning (with or without bucketing and sorting) is enabled, this optimization sorts the records on partition, bucket and sort columns respectively before inserting records into the destination table.
org.apache.hadoop.hive.ql.optimizer.SortedMergeBucketMapJoinOptimizer	Replace a bucket map join with a sorted merge map join.
org.apache.hadoop.hive.ql.optimizer.StatsOptimizer	There is a set of queries which can be answered entirely from statistics stored in metastore.
org.apache.hadoop.hive.ql.optimizer.Transform	Optimizer interface.
org.apache.hadoop.hive.ql.optimizer.calcite.HiveCalciteUtil	Generic utility functions needed for Calcite based Hive CBO.
org.apache.hadoop.hive.ql.optimizer.calcite.HiveRelOptUtil	Splits different join conditions.
org.apache.hadoop.hive.ql.optimizer.calcite.RelOptHiveTable	Class for handling all table metadata.
org.apache.hadoop.hive.ql.optimizer.calcite.TraitsUtil	Traits utilities.
org.apache.hadoop.hive.ql.optimizer.calcite.cost.HiveRelMdCost	HiveRelMdCost supplies the implementation of cost model.
org.apache.hadoop.hive.ql.optimizer.calcite.cost.HiveVolcanoPlanner	Refinement of org.apache.calcite.plan.volcano.VolcanoPlanner for Hive.
org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveAggregate	Describing aggregate function as relational operator.
org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveFilter	Describing filter function as relational operator.
org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveJoin	Describing join function as relational operator.
org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveProject	Creates a HiveProject.
org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveSemiJoin	Describing semi join operator as relational operator.
org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveTableScan	Relational expression representing a scan of a HiveDB collection.
org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveUnion	Describing union operator as relational operator.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveFilterJoinRule	Creates a PushFilterPastJoinRule with an explicit root operand.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveJoinProjectTransposeRule	Transpose rule for hive join project.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveFilterSetOpTransposeRule	Creates a HiveFilterSetOpTransposeRule.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveInsertExchange4JoinRule	Not an optimization rule.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveJoinAddNotNullRule	Creates an HiveJoinAddNotNullRule.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveJoinPushTransitivePredicatesRule	Planner rule that infers predicates from on a org.apache.calcite.rel.core.Join and creates org.apache.calcite.rel.core.Filters if those predicates can be pushed to its inputs.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveJoinToMultiJoinRule	Rule that merges a join with multijoin/join children if the equi compared the same set of input columns.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HivePreFilteringRule	Pull out deterministic expressions from non-deterministic and push down deterministic expressions as a separate filter.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveProjectMergeRule	ProjectMergeRule merges a org.apache.calcite.rel.core.Project into another org.apache.calcite.rel.core.Project, provided the projects aren't projecting identical sets of input references.
org.apache.hadoop.hive.ql.optimizer.calcite.rules.HiveRelFieldTrimmer	Hive relational expression field trimmer.
org.apache.hadoop.hive.ql.optimizer.calcite.stats.HiveRelMdCollation	Hive relational expression metadata collation.
org.apache.hadoop.hive.ql.optimizer.calcite.stats.HiveRelMdDistinctRowCount	Hive relational expression metadata.
org.apache.hadoop.hive.ql.optimizer.calcite.stats.HiveRelMdDistribution	Hive relational expression metadata distribuiton.
org.apache.hadoop.hive.ql.optimizer.calcite.stats.HiveRelMdMemory	Hive relational expression metadata memory
org.apache.hadoop.hive.ql.optimizer.calcite.stats.HiveRelMdParallelism	Hive relational expression metadata parallelism.
org.apache.hadoop.hive.ql.optimizer.calcite.stats.HiveRelMdRowCount	Hive relational expression metadata row count.
org.apache.hadoop.hive.ql.optimizer.calcite.stats.HiveRelMdSelectivity	Hive relational expression metadata selectivity.
org.apache.hadoop.hive.ql.optimizer.calcite.stats.HiveRelMdSize	Hive relational expression metadata size.
org.apache.hadoop.hive.ql.optimizer.calcite.stats.HiveRelMdUniqueKeys	Hive relational expression metadata unique keys.
org.apache.hadoop.hive.ql.optimizer.calcite.translator.ASTConverter	Abstract syntax tree converter.
org.apache.hadoop.hive.ql.optimizer.calcite.translator.ExprNodeConverter	Expression node converter.
org.apache.hadoop.hive.ql.optimizer.calcite.translator.HiveOpConverter	Hive operation converter.
org.apache.hadoop.hive.ql.optimizer.calcite.translator.HiveOpConverterPostProc	Post processing hive operation converter.
org.apache.hadoop.hive.ql.optimizer.calcite.translator.PlanModifierForASTConv	Modifying plan for converting abstract syntax tree.
org.apache.hadoop.hive.ql.optimizer.calcite.translator.PlanModifierForReturnPath	Modifying plan for operation tree.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.optimizer.calcite.translator.RexNodeConverter	Row expression node converter.
org.apache.hadoop.hive.ql.optimizer.calcite.translator.SqlFunctionConverter	Converting SQL function.
org.apache.hadoop.hive.ql.optimizer.correlation.CorrelationOptimizer	Implementation of Correlation Optimizer.
org.apache.hadoop.hive.ql.optimizer.correlation.CorrelationUtilities	Utilities for both CorrelationOptimizer and ReduceSinkDeDuplication.
org.apache.hadoop.hive.ql.optimizer.correlation.ReduceSinkDeDuplication	If two reducer sink operators share the same partition/ sort columns and order, they can be merged.
org.apache.hadoop.hive.ql.optimizer.index.RewriteGBUsingIndex	RewriteGBUsingIndex is implemented as one of the Rule-based Optimizations.
org.apache.hadoop.hive.ql.optimizer.index.RewriteParseContextGenerator	RewriteParseContextGenerator is a class that offers methods to generate operator tree for input queries.
org.apache.hadoop.hive.ql.optimizer.lineage.ExprProcFactory	Expression processor factory for lineage.
org.apache.hadoop.hive.ql.optimizer.lineage.Generator	Generates the lineage information for the columns and tables from the plan before it goes through other optimization phases.
org.apache.hadoop.hive.ql.optimizer.lineage.LineageCtx	Contains the lineage context that is passed while walking the operator tree in Lineage.
org.apache.hadoop.hive.ql.optimizer.lineage.OpProcFactory	Operator factory for the rule processors for lineage.
org.apache.hadoop.hive.ql.optimizer.listbucketingpruner.ListBucketingPruner	The transformation step that does list bucketing pruning.
org.apache.hadoop.hive.ql.optimizer.metainfo.annotation.AnnotateWithOpTraits	This class annotates each operator with its traits. The OpTraits class specifies the traits that are populated for each operator.
org.apache.hadoop.hive.ql.optimizer.pcr.PartitionConditionRemover	The transformation step that does partition condition remover.
org.apache.hadoop.hive.ql.optimizer.pcr.PcrExprProcFactory	Expression processor factory for partition condition removing.
org.apache.hadoop.hive.ql.optimizer.physical.CrossProductCheck	Check each MapJoin and ShuffleJoin Operator to see they are performing a cross product.
org.apache.hadoop.hive.ql.optimizer.physical.Vectorizer	Class to define vectorization.
org.apache.hadoop.hive.ql.optimizer.ppr.PartitionExpressionForMetastore	The basic implementation of PartitionExpressionProxy that uses ql package classes.
org.apache.hadoop.hive.ql.optimizer.ppr.PartitionPruner	The transformation step that does partition pruning.
org.apache.hadoop.hive.ql.optimizer.spark.SparkReduceSinkMapJoinProc	This processor addresses the RS-MJ case that occurs in spark on the small/hash table side of things. The work that RS will be a part of must be connectedto the MJ work via be a broadcast edge.
org.apache.hadoop.hive.ql.optimizer.stats.annotation.AnnotateWithStatistics	Create a list of top op nodes
org.apache.hadoop.hive.ql.optimizer.unionproc.UnionProcessor	FileSinkProcessor is a simple rule to remember seen unions for later processing.
org.apache.hadoop.hive.ql.parse.ASTNode	Definition of abstract syntax tree node.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.parse.BaseSemanticAnalyzer	BaseSemanticAnalyzer.
org.apache.hadoop.hive.ql.parse.CalcitePlanner	Cost based optimizer planner.
org.apache.hadoop.hive.ql.parse.ColumnAccessAnalyzer	Analysis of column access.
org.apache.hadoop.hive.ql.parse.ColumnStatsSemanticAnalyzer	ColumnStatsSemanticAnalyzer.
org.apache.hadoop.hive.ql.parse.DDLSemanticAnalyzer	DDLSemanticAnalyzer.
org.apache.hadoop.hive.ql.parse.ExplainSQRewriteSemanticAnalyzer	ExplainSQRewriteSemanticAnalyzer.
org.apache.hadoop.hive.ql.parse.ExplainSemanticAnalyzer	ExplainSemanticAnalyzer.
org.apache.hadoop.hive.ql.parse.ExportSemanticAnalyzer	ExportSemanticAnalyzer.
org.apache.hadoop.hive.ql.parse.FunctionSemanticAnalyzer	FunctionSemanticAnalyzer.
org.apache.hadoop.hive.ql.parse.GenTezProcContext	GenTezProcContext.
org.apache.hadoop.hive.ql.parse.GenTezUtils	GenTezUtils is a collection of shared helper methods to produce TezWork.
org.apache.hadoop.hive.ql.parse.GlobalLimitCtx	context for pruning inputs.
org.apache.hadoop.hive.ql.parse.ImportSemanticAnalyzer	ImportSemanticAnalyzer.
org.apache.hadoop.hive.ql.parse.LoadSemanticAnalyzer	LoadSemanticAnalyzer.
org.apache.hadoop.hive.ql.parse.MacroSemanticAnalyzer	MacroSemanticAnalyzer.
org.apache.hadoop.hive.ql.parse.MapReduceCompiler	Compiling list of tasks.
org.apache.hadoop.hive.ql.parse.MetaDataExportListener	Listens for drop events and, if set, exports the table's metadata as JSON to the trash of the user performing the drop
org.apache.hadoop.hive.ql.parse.ParseContext	Parse Context: The current parse context.
org.apache.hadoop.hive.ql.parse.ParseDriver	ParseDriver.
org.apache.hadoop.hive.ql.parse.ParseUtils	Library of utility functions used in the parse code.
org.apache.hadoop.hive.ql.parse.QB	Implementation of the query block.
org.apache.hadoop.hive.ql.parse.QBParseInfo	Implementation of the parse information related to a query block.
org.apache.hadoop.hive.ql.parse.RowResolver	Implementation of the Row Resolver.
org.apache.hadoop.hive.ql.parse.SemanticAnalyzer	Implementation of the semantic analyzer.
org.apache.hadoop.hive.ql.parse.SemanticAnalyzerFactory	SemanticAnalyzerFactory.
org.apache.hadoop.hive.ql.parse.SplitSample	Stores all the information specified in the TABLESAMPLE(...) clause.
org.apache.hadoop.hive.ql.parse.TaskCompiler	TaskCompiler is a the base class for classes that compile operator pipelines into tasks.
org.apache.hadoop.hive.ql.parse.TezCompiler	TezCompiler translates the operator plan into TezTasks.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.parse.TypeCheckCtx	This class implements the context information that is used for typechecking phase in query compilation.
org.apache.hadoop.hive.ql.parse.TypeCheckProcFactory	The Factory for creating typecheck processors.
org.apache.hadoop.hive.ql.parse.UpdateDeleteSemanticAnalyzer	A subclass of the org.apache.hadoop.hive.ql.parse.SemanticAnalyzer that just handles update and delete statements.
org.apache.hadoop.hive.ql.parse.WindowingSpec	Windowing Specification.
org.apache.hadoop.hive.ql.parse.spark.GenSparkProcContext	GenSparkProcContext maintains information about the tasks and operators as we walk the operator tree to break them into SparkTasks.
org.apache.hadoop.hive.ql.parse.spark.GenSparkUtils	GenSparkUtils is a collection of shared helper methods to produce SparkWork Cloned from GenTezUtils.
org.apache.hadoop.hive.ql.parse.spark.GenSparkWorkWalker	Walks the operator tree in DFS fashion.
org.apache.hadoop.hive.ql.parse.spark.OptimizeSparkProcContext	OptimizeSparkProcContext.
org.apache.hadoop.hive.ql.plan.AbstractOperatorDesc	Operator description.
org.apache.hadoop.hive.ql.plan.AlterTableDesc	AlterTableDesc.
org.apache.hadoop.hive.ql.plan.AlterTableSimpleDesc	Contains information needed to modify a partition or a table
org.apache.hadoop.hive.ql.plan.BaseWork	BaseWork.
org.apache.hadoop.hive.ql.plan.ColumnStatsDesc	Contains the information needed to persist column level statistics
org.apache.hadoop.hive.ql.plan.ColumnStatsWork	ColumnStats Work.
org.apache.hadoop.hive.ql.plan.CommonMergeJoinDesc	Description of merge join operator.
org.apache.hadoop.hive.ql.plan.CreateTableDesc	CreateTableDesc.
org.apache.hadoop.hive.ql.plan.CreateViewDesc	CreateViewDesc.
org.apache.hadoop.hive.ql.plan.DDLWork	DDLWork.
org.apache.hadoop.hive.ql.plan.DropTableDesc	DropTableDesc.
org.apache.hadoop.hive.ql.plan.DynamicPartitionCtx	Dynamic partition context.
org.apache.hadoop.hive.ql.plan.DynamicPruningEventDesc	Dynamic pruning event description.
org.apache.hadoop.hive.ql.plan.ExplainWork	ExplainWork.
org.apache.hadoop.hive.ql.plan.ExprNodeDesc	ExprNodeDesc.
org.apache.hadoop.hive.ql.plan.ExprNodeDescUtils	Utilities for expression node description.
org.apache.hadoop.hive.ql.plan.FetchWork	FetchWork.
org.apache.hadoop.hive.ql.plan.FileSinkDesc	FileSinkDesc.
org.apache.hadoop.hive.ql.plan.FilterDesc	FilterDesc.
org.apache.hadoop.hive.ql.plan.GroupByDesc	GroupByDesc.
org.apache.hadoop.hive.ql.plan.HashTableSinkDesc	Map Join operator Descriptor implementation.
org.apache.hadoop.hive.ql.plan.JoinCondDesc	Join conditions Descriptor implementation.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.plan.JoinDesc	Join operator Descriptor implementation.
org.apache.hadoop.hive.ql.plan.LateralViewJoinDesc	LateralViewJoinDesc.
org.apache.hadoop.hive.ql.plan.LimitDesc	LimitDesc.
org.apache.hadoop.hive.ql.plan.LoadTableDesc	LoadTableDesc.
org.apache.hadoop.hive.ql.plan.MapJoinDesc	Map Join operator Descriptor implementation.
org.apache.hadoop.hive.ql.plan.MapWork	MapWork represents all the information used to run a map task on the cluster.
org.apache.hadoop.hive.ql.plan.MapredWork	MapredWork.
org.apache.hadoop.hive.ql.plan.MergeJoinWork	Creating merge join work.
org.apache.hadoop.hive.ql.plan.PartitionDesc	PartitionDesc.
org.apache.hadoop.hive.ql.plan.PlanUtils	PlanUtils.
org.apache.hadoop.hive.ql.plan.ReduceSinkDesc	ReduceSinkDesc.
org.apache.hadoop.hive.ql.plan.ReduceWork	ReduceWork represents all the information used to run a reduce task on the cluster.
org.apache.hadoop.hive.ql.plan.SelectDesc	SelectDesc.
org.apache.hadoop.hive.ql.plan.SparkHashTableSinkDesc	Map Join operator Descriptor implementation.
org.apache.hadoop.hive.ql.plan.Statistics	Statistics.
org.apache.hadoop.hive.ql.plan.StatsWork	ConditionalStats.
org.apache.hadoop.hive.ql.plan.TableScanDesc	Table Scan Descriptor Currently, data is only read from a base source as part of map-reduce framework.
org.apache.hadoop.hive.ql.plan.TezWork	TezWork.
org.apache.hadoop.hive.ql.plan.UnionWork	Simple wrapper for union all cases.
org.apache.hadoop.hive.ql.plan.VectorGroupByDesc	VectorGroupByDesc.
org.apache.hadoop.hive.ql.plan.ptf.BoundaryDef	Map-reduce boundaries definition.
org.apache.hadoop.hive.ql.plan.ptf.CurrentRowDef	Current row definition.
org.apache.hadoop.hive.ql.plan.ptf.OrderExpressionDef	Order expression definition.
org.apache.hadoop.hive.ql.plan.ptf.RangeBoundaryDef	Range boundary definition.
org.apache.hadoop.hive.ql.plan.ptf.ValueBoundaryDef	Value boundary definition.
org.apache.hadoop.hive.ql.plan.ptf.WindowFrameDef	Window frame definition.
org.apache.hadoop.hive.ql.ppd.ExprWalkerInfo	Context for Expression Walker for determining predicate pushdown candidates It contains a ExprInfo object for each expression that is processed.
org.apache.hadoop.hive.ql.ppd.OpProcFactory	Operator factory for predicate pushdown processing of operator graph Each operator determines the pushdown predicates by walking the expression tree.
org.apache.hadoop.hive.ql.ppd.PredicatePushDown	Implements predicate pushdown.
org.apache.hadoop.hive.ql.ppd.PredicateTransitivePropagate	Propagates filters to other aliases based on join condition
org.apache.hadoop.hive.ql.ppd.SyntheticJoinPredicate	Creates synthetic predicates that represent "IN (keylist other table)"

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.processors.AddResourceProcessor	AddResourceProcessor.
org.apache.hadoop.hive.ql.processors.CommandProcessorResponse	Encapsulates the basic response info returned by classes the implement the CommandProcessor interface.
org.apache.hadoop.hive.ql.processors.CompileProcessor	Processor allows users to build code inside a hive session, then use this code as a UDF, Serde, or even a more complex entity like an input format or hook.
org.apache.hadoop.hive.ql.processors.CryptoProcessor	Processes HADOOP commands used for HDFS encryption.
org.apache.hadoop.hive.ql.processors.DeleteResourceProcessor	DeleteResourceProcessor.
org.apache.hadoop.hive.ql.processors.DfsProcessor	DfsProcessor.
org.apache.hadoop.hive.ql.processors.SetProcessor	SetProcessor.
org.apache.hadoop.hive.ql.security.authorization.AuthorizationPreEventListener	AuthorizationPreEventListener : A MetaStorePreEventListener that performs authorization/ authentication checks on the metastore-side.
org.apache.hadoop.hive.ql.security.authorization.AuthorizationUtils	Utility code shared by hive internal code and sql standard authorization plugin implementation
org.apache.hadoop.hive.ql.security.authorization.HiveAuthorizationProviderBase	Class for authorization that returns userNames and groupNames.
org.apache.hadoop.hive.ql.security.authorization.plugin.AuthorizationMetaStoreFilterHook	Metastore filter hook for filtering out the list of objects that the current authorization implementation does not allow user to see
org.apache.hadoop.hive.ql.security.authorization.plugin.HiveAuthorizerImpl	Convenience implementation of HiveAuthorizer.
org.apache.hadoop.hive.ql.security.authorization.plugin.HiveAuthzContext	Provides context information in authorization check call that can be used for auditing and/or authorization.
org.apache.hadoop.hive.ql.security.authorization.plugin.HivePrivilegeObject	Represents the object on which privilege is being granted/revoked, and objects being used in queries.
org.apache.hadoop.hive.ql.security.authorization.plugin.HiveV1Authorizer	Hive v1 authorization class.
org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.DummyHiveAuthorizationValidator	A no-op HiveAuthorizationValidator for use from hive cli.
org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.SQLAuthorizationUtils	Utilities for SQL based authorization.
org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.SQLStdHiveAccessController	Implements functionality of access control statements for sql standard based authorization
org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.SQLStdHiveAuthorizationValidator	Class to check if user has privileges to perform certain action according to SQL standart hive authorization.
org.apache.hadoop.hive.ql.session.LineageState	LineageState.
org.apache.hadoop.hive.ql.session.OperationLog	OperationLog wraps the actual operation log file, and provides interface for accessing, reading, writing, and removing the file.
org.apache.hadoop.hive.ql.session.SessionState	SessionState encapsulates common data associated with a session.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.stats.StatsFactory	A factory of stats publisher and aggregator implementations of the StatsPublisher and StatsAggregator interfaces.
org.apache.hadoop.hive.ql.stats.StatsUtils	Utilities of stats publisher and aggregator.
org.apache.hadoop.hive.ql.stats.fs.FSStatsAggregator	File system stats aggregator.
org.apache.hadoop.hive.ql.stats.fs.FSStatsPublisher	File system stats publisher.
org.apache.hadoop.hive.ql.txn.compactor.CompactorMR	Performs compactions via an MR job.
org.apache.hadoop.hive.ql.txn.compactor.Worker	Performs compactions.
org.apache.hadoop.hive.ql.udf.generic.GenericUDAFAverage	GenericUDAFAverage.
org.apache.hadoop.hive.ql.udf.generic.GenericUDAFStreamingEvaluator	User defined aggregate function streaming evaluator.
org.apache.hadoop.hive.ql.udf.generic.GenericUDAFSum	GenericUDAFSum.
org.apache.hadoop.hive.ql.udf.generic.GenericUDF	A Generic User-defined function (GenericUDF) for the use with Hive.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFBasePad	A Generic User-defined function (GenericUDF) for the use with Hive.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFBridge	GenericUDFBridge encapsulates UDF to provide the same interface as GenericUDF.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFDateAdd	UDFDateAdd.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFDateSub	UDFDateSub.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFFromUtcTimestamp	Generic user defined fuction to compute UTC timestamp.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFGreatest	GenericUDF Class for SQL construct "greatest(v1, v2, ..
org.apache.hadoop.hive.ql.udf.generic.GenericUDFLeast	GenericUDF Class for SQL construct "least(v1, v2, ..
org.apache.hadoop.hive.ql.udf.generic.GenericUDFLpad	UDFLpad.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFOPAnd	GenericUDF Class for computing and.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFOPEqual	GenericUDF Class for operation EQUAL.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFOPEqualOrGreaterThan	GenericUDF Class for operation EqualOrGreaterThan.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFOPEqualOrLessThan	GenericUDF Class for operation EqualOrLessThan.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFOPGreaterThan	GenericUDF Class for operation GreaterThan.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFOPLessThan	GenericUDF Class for operation LessThan.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFOPNotEqual	GenericUDF Class for operation Not EQUAL.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFOPNotNull	GenericUDFOPNotNull.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.udf.generic.GenericUDFOPNull	GenericUDFOPNull.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFOPOr	GenericUDF Class for computing or.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFRound	Rounding function permits rounding off integer digits in decimal numbers, which essentially downgrades the scale to negative territory.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFRpad	UDFRpad.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFToUnixTimeStamp	Deterministic version of UDFUnixTimeStamp.
org.apache.hadoop.hive.ql.udf.generic.GenericUDFUtils	Util functions for GenericUDF classes.
org.apache.hadoop.hive.ql.udf.generic.NumDistinctValueEstimator	Take the average of the index for all the bit vectors and get the estimated NDV (estimateNumDistinctValues).
org.apache.hadoop.hive.ql.udf.generic.NumericHistogram	A generic, re-usable histogram class that supports partial aggregations.
org.apache.hadoop.hive.ql.udf.generic.RoundUtils	Utility class for generic round UDF.
org.apache.hadoop.hive.ql.udf.ptf.WindowingTableFunction	A window function performs a calculation across a set of table rows that are somehow related to the current row.
org.apache.hadoop.hive.ql.util.DateTimeMath	Operations involving/returning year-month intervals.
org.apache.hadoop.hive.ql.util.DosToUnix	Converting windows script to UNIX.
org.apache.hadoop.hive.ql.util.ZooKeeperHiveHelper	Get the ensemble server addresses from the configuration. The format is: host1:port,host2:port
org.apache.hadoop.hive.serde2.AbstractSerDe	Abstract class for implementing SerDe.
org.apache.hadoop.hive.serde2.ColumnProjectionUtils	ColumnProjectionUtils.
org.apache.hadoop.hive.serde2.DelimitedJSONSerDe	DelimitedJSONSerDe.
org.apache.hadoop.hive.serde2.MetadataTypedColumnsetSerDe	MetadataTypedColumnsetSerDe.
org.apache.hadoop.hive.serde2.OpenCSVSerde	OpenCSVSerde use opencsv to deserialize CSV format.
org.apache.hadoop.hive.serde2.RegexSerDe	RegexSerDe uses regular expression (regex) to deserialize data.
org.apache.hadoop.hive.serde2.SerDeUtils	SerDeUtils.
org.apache.hadoop.hive.serde2.WriteBuffers	The structure storing arbitrary amount of data as a set of fixed-size byte buffers.
org.apache.hadoop.hive.serde2.avro.AvroLazyObjectInspector	Lazy objectinspector for avro serialization
org.apache.hadoop.hive.serde2.avro.AvroSerDe	Read or write Avro data from Hive.
org.apache.hadoop.hive.serde2.avro.AvroSerdeUtils	Utilities useful only to the AvroSerde itself.
org.apache.hadoop.hive.serde2.binarysortable.BinarySortableSerDe	BinarySortableSerDe can be used to write data in a way that the data can be compared byte-by-byte with the same order.
org.apache.hadoop.hive.serde2.binarysortable.fast.BinarySortableDeserializeRead	Directly deserialize with the caller reading field-by-field the LazyBinary serialization format.
org.apache.hadoop.hive.serde2.binarysortable.fast.BinarySortableSerializeWrite	Directly serialize, field-by-field, the BinarySortable format.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.serde2.io.DateWritable	DateWritable Writable equivalent of java.sql.Date.
org.apache.hadoop.hive.serde2.io.HiveDecimalWritable	Get a HiveDecimal instance from the writable and constraint it with maximum precision/scale.
org.apache.hadoop.hive.serde2.io.TimestampWritable	TimestampWritable Writable equivalent of java.sq.Timestamp Timestamps are of the format YYYY-MM-DD HH:MM:SS.[fff...] We encode Unix timestamp in seconds in 4 bytes, using the MSB to signify whether the timestamp has a fractional portion.
org.apache.hadoop.hive.serde2.lazy.LazyBinary	LazyBinary stores a binary object in a LazyObject.
org.apache.hadoop.hive.serde2.lazy.LazyHiveDecimal	LazyHiveDecimal stores hive decimal object in LazyObject.
org.apache.hadoop.hive.serde2.lazy.LazyMap	LazyMap stores a map of Primitive LazyObjects to LazyObjects.
org.apache.hadoop.hive.serde2.lazy.LazySerDeParameters	SerDeParameters.
org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe	LazySimpleSerDe can be used to read the same data format as MetadataTypedColumnsetSerDe and TCTLSeparatedProtocol.
org.apache.hadoop.hive.serde2.lazy.LazyUtils	LazyUtils.
org.apache.hadoop.hive.serde2.lazy.fast.LazySimpleDeserializeRead	Directly deserialize with the caller reading field-by-field the LazySimple (text) serialization format.
org.apache.hadoop.hive.serde2.lazy.fast.LazySimpleSerializeWrite	Directly serialize, field-by-field, the LazyBinary format.
org.apache.hadoop.hive.serde2.lazy.objectinspector.LazyListObjectInspector	LazyListObjectInspector works on array data that is stored in LazyArray.
org.apache.hadoop.hive.serde2.lazy.objectinspector.LazyMapObjectInspector	LazyMapObjectInspector works on struct data that is stored in LazyStruct.
org.apache.hadoop.hive.serde2.lazy.objectinspector.LazyUnionObjectInspector	LazyUnionObjectInspector works on union data that is stored in LazyUnion.
org.apache.hadoop.hive.serde2.lazybinary.LazyBinarySerDe	The LazyBinarySerDe class combines the lazy property of LazySimpleSerDe class and the binary property of BinarySortable class.
org.apache.hadoop.hive.serde2.lazybinary.fast.LazyBinaryDeserializeRead	Directly deserialize with the caller reading field-by-field the LazyBinary serialization format.
org.apache.hadoop.hive.serde2.lazybinary.fast.LazyBinarySerializeWrite	Directly serialize, field-by-field, the LazyBinary format.
org.apache.hadoop.hive.serde2.objectinspector.ObjectInspectorFactory	ObjectInspectorFactory is the primary way to create new ObjectInspector instances.
org.apache.hadoop.hive.serde2.objectinspector.ObjectInspectorUtils	ObjectInspectorFactory is the primary way to create new ObjectInspector instances.
org.apache.hadoop.hive.serde2.objectinspector.ReflectionStructObjectInspector	ReflectionStructObjectInspector works on struct data that is stored as a native Java object.
org.apache.hadoop.hive.serde2.objectinspector.SettableUnionObjectInspector	SettableUnionObjectInspector.
org.apache.hadoop.hive.serde2.objectinspector.StandardStructObjectInspector	ListStructObjectInspector works on struct data that is stored as a Java List or Java Array object.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.serde2.objectinspector.StandardUnionObjectInspector	StandardUnionObjectInspector works on union data that is stored as UnionObject.
org.apache.hadoop.hive.serde2.objectinspector.ThriftUnionObjectInspector	Always use the ObjectInspectorFactory to create new ObjectInspector objects, instead of directly creating an instance of this class.
org.apache.hadoop.hive.serde2.typeinfo.HiveDecimalUtils	Utilities for decimal precision and scale.
org.apache.hadoop.hive.serde2.typeinfo.TypeInfoUtils	TypeInfoUtils.
org.apache.hadoop.hive.shims.Hadoop23Shims	Implemention of shims against Hadoop 0.23.0.
org.apache.hadoop.hive.shims.HadoopShimsSecure	Base implemention for shims against secure Hadoop 0.20.3/0.23.
org.apache.hadoop.hive.shims.ShimLoader	ShimLoader.
org.apache.hadoop.hive.shims.Utils	Utilities for split location provider.
org.apache.hadoop.hive.thrift.DelegationTokenSecretManager	A Hive specific delegation token secret manager.
org.apache.hadoop.hive.thrift.HadoopThriftAuthBridge	Functions that bridge Thrift's SASL transports to Hadoop's SASL callback handlers and authentication classes.
org.apache.hadoop.hive.thrift.TokenStoreDelegationTokenSecretManager	Extension of DelegationTokenSecretManager to support alternative to default in-memory token management for fail-over and clustering through plug-able token store (ZooKeeper etc.).
org.apache.hive.beeline.BeeLine	A console SQL shell with command completion.
org.apache.hive.beeline.Commands	Implementation of beeline commands.
org.apache.hive.common.util.BloomFilter	BloomFilter is a probabilistic data structure for set membership check.
org.apache.hive.common.util.DateUtils	DateUtils.
org.apache.hive.common.util.HiveStringUtils	HiveStringUtils General string utils Originally copied from o.a.hadoop.util.StringUtils
org.apache.hive.common.util.HiveTestUtils	Utilities for testing hive.
org.apache.hive.common.util.Murmur3	Murmur3 is successor to Murmur2 fast non-crytographic hash algorithms.
org.apache.hive.common.util.ShutdownHookManager	The ShutdownHookManager enables running shutdownHook in a deterministic order, higher priority first.
org.apache.hive.common.util.StreamPrinter	StreamPrinter.
org.apache.hive.hcatalog.cli.HCatCli	HCatalog command line interface.
org.apache.hive.hcatalog.common.HCatConstants	List of constants used by HCatalog.
org.apache.hive.hcatalog.streaming.ConnectionError	Exception to catch connection errors.
org.apache.hive.hcatalog.streaming.DelimitedInputWriter	Streaming Writer handles delimited input (eg.
org.apache.hive.hcatalog.streaming.HiveEndPoint	Information about the hive end point (i.e.
org.apache.hive.hcatalog.streaming.InvalidTable	Exception to catch invalid table.
org.apache.hive.hcatalog.streaming.StrictJsonWriter	Streaming Writer handles utf8 encoded Json (Strict syntax).

last-updated: Apr 11, 2022

Class	Description
org.apache.hive.hcatalog.templeton.AppConfig	The configuration for Templeton.
org.apache.hive.hcatalog.templeton.Main	The main executable that starts up and runs the Server.
org.apache.hive.jdbc .HiveConnection	HiveConnection.
org.apache.hive.jdbc .HiveDatabaseMetaData	HiveDatabaseMetaData.
org.apache.hive.jdbc .HivePreparedStatement	HivePreparedStatement.
org.apache.hive.jdbc .HiveQueryResultSet	HiveQueryResultSet.
org.apache.hive.jdbc .HiveStatement	HiveStatement.
org.apache.hive.jdbc .JdbcColumn	Column metadata.
org.apache.hive.jdbc .Utils	Utilities for jdbc.
org.apache.hive.service.ServiceUtils	Utilities to correctly process domain names etc.
org.apache.hive.service.auth.AuthenticationProviderFactory	Helps select a PasswdAuthenticationProvider for a given {@code AuthMethod}.
org.apache.hive.service.auth.HiveAuthFactory	Helps in some aspects of authentication.
org.apache.hive.service.auth.LdapAuthenticationProviderImpl	Utilities to correctly process domain names etc.
org.apache.hive.service.auth.TSetIpAddressProcessor	Sets the ipAddress for operations executed via HiveServer2.
org.apache.hive.service.cli.CLIService	CLIService.
org.apache.hive.service.cli.CLIServiceUtils	CLIServiceUtils.
org.apache.hive.service.cli.ColumnBasedSet	ColumnBasedSet.
org.apache.hive.service.cli.ColumnDescriptor	ColumnDescriptor.
org.apache.hive.service.cli.ColumnValue	Protocols before HIVE_CLI_SERVICE_PROTOCOL_V6 (used by RowBasedSet)
org.apache.hive.service.cli.EmbeddedCLIServiceClient	Embedded CLI Service Client.
org.apache.hive.service.cli.FetchOrientation	Fetch Orientation.
org.apache.hive.service.cli.GetInfoType	Get Info type.
org.apache.hive.service.cli.GetInfoValue	Get Info value.
org.apache.hive.service.cli.Handle	Handle.
org.apache.hive.service.cli.HandleIdentifier	Handle identifier.
org.apache.hive.service.cli.HiveSQLException	Hive SQL exception.

last-updated: Apr 11, 2022

Class	Description
org.apache.hive.service.cli.OperationHandle	Handler for operation.
org.apache.hive.service.cli.OperationState	Operation State.
org.apache.hive.service.cli.OperationStatus	Operation Status.
org.apache.hive.service.cli.OperationType	OperationType.
org.apache.hive.service.cli.RowBasedSet	Row Based Set.
org.apache.hive.service.cli.RowSetFactory	Row set factory class.
org.apache.hive.service.cli.SessionHandle	Session Handle.
org.apache.hive.service.cli.TableSchema	Table Schema.
org.apache.hive.service.cli.TypeDescriptor	Type Descriptor.
org.apache.hive.service.cli.TypeQualifiers	Holds type qualifier information for a primitive type, such as char/varchar length or decimal precision/scale.
org.apache.hive.service.cli.operation.ClassicTableTypeMapping	Classic Table Type Mapping.
org.apache.hive.service.cli.operation.ExecuteStatementOperation	Implementation of statement execution.
org.apache.hive.service.cli.operation.HiveCommandOperation	Executes a HiveCommand.
org.apache.hive.service.cli.operation.HiveTableTypeMapping	Hive Table Type Mapping.
org.apache.hive.service.cli.operation.LogDivertAppender	Divert appender to redirect operation logs to separate files.
org.apache.hive.service.cli.operation.MetadataOperation	Metadata Operation.
org.apache.hive.service.cli.operation.Operation	Class to define operation.
org.apache.hive.service.cli.operation.OperationManager	Operation Manager.
org.apache.hive.service.cli.operation.SQLOperation	SQL Operation.
org.apache.hive.service.cli.session.HiveSessionImpl	Hive Session.
org.apache.hive.service.cli.session.HiveSessionImplwithUGI	Hive session implementation with UGI.
org.apache.hive.service.cli.session.SessionManager	Session Manager.
org.apache.hive.service.cli.thrift.ThriftBinaryCLIService	Initialize worker threads in hive CLI startup.
org.apache.hive.service.cli.thrift.ThriftCLIService	Thrift CLI Service.
org.apache.hive.service.cli.thrift.ThriftCLIServiceClient	Thrift CLI Service Client.
org.apache.hive.service.cli.thrift.ThriftHttpCLIService	Service to handle requests over HTTP.
org.apache.hive.service.cli.thrift.ThriftHttpServlet	Thrift Http servlet.
org.apache.hive.spark.client.MetricsCollection	Provides metrics collected for a submitted job.
org.apache.hive.spark.client.SparkClientUtilities	Utilities for spark client.
org.apache.hive.spark.client.rpc.Rpc	Encapsulates the RPC functionality.
org.apache.hive.spark.client.rpc.RpcConfiguration	Definitions of configuration keys and default values for the RPC layer.
org.apache.hive.spark.client.rpc.RpcServer	An RPC server.

last-updated: Apr 11, 2022

Changed Interfaces in Hive 2.1

The following interfaces have changed in Hive 2.1:

Interface	Description
org.apache.hadoop.hive.common.*ValidTxnList*	Models the list of transactions that should be included in a snapshot.
org.apache.hadoop.hive.metastore.*AlterHandler*	Interface for Alter Table and Alter Partition code
org.apache.hadoop.hive.metastore.*IMetaStoreClient*	Wrapper around hive metastore thrift api
org.apache.hadoop.hive.metastore.*RawStore*	
org.apache.hadoop.hive.metastore.*PartitionExpressionProxy*	The proxy interface that metastore uses for variety of QL operations (metastore can't depend on QL because QL depends on metastore; creating metastore-client module would be a proper way to solve this problem).
org.apache.hadoop.hive.ql.exec.persistence.*MapJoinTableContainer*	
org.apache.hadoop.hive.ql.exec.spark.*SparkShuffler*	
org.apache.hadoop.hive.ql.exec.spark.*SparkTran*	
org.apache.hadoop.hive.ql.exec.spark.session.*SparkSession*	
org.apache.hadoop.hive.ql.exec.spark.status.*SparkJobStatus*	SparkJobStatus identify what Hive want to know about the status of a Spark job.
org.apache.hadoop.hive.ql.exec.vector.expressions.*VectorExpressionWriter*	Interface used to create Writable objects from vector expression primitives.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.hashtable.*VectorMapJoinHashTable*	
org.apache.hadoop.hive.ql.io.*InputFormatChecker*	Check for validity of the input files.
org.apache.hadoop.hive.ql.io.orc.*Reader*	The interface for reading ORC files.
org.apache.hadoop.hive.ql.io.orc.*RecordReader*	A row-by-row iterator for ORC files.
org.apache.hadoop.hive.ql.io.orc.*Writer*	The HIVE interface for writing ORC files.
org.apache.hadoop.hive.ql.io.sarg.*SearchArgument*	Primary interface for SearchArgument, which are the subset of predicates that can be pushed down to the RecordReader.
org.apache.hadoop.hive.ql.lockmgr.*HiveTxnManager*	An interface that allows Hive to manage transactions.
org.apache.hadoop.hive.ql.metadata.formatting.*MetaDataFormatter*	Interface to format table and index information.
org.apache.hadoop.hive.ql.plan.*OperatorDesc*	
org.apache.hadoop.hive.ql.security.authorization.plugin.*HiveAuthorizationValidator*	Interface used to check if user has privileges to perform certain action.
org.apache.hadoop.hive.ql.security.authorization.plugin.*HiveAuthorizer*	Interface for hive authorization plugins.
org.apache.hadoop.hive.ql.stats.*StatsAggregator*	An interface for any possible implementation for gathering statistics.
org.apache.hadoop.hive.ql.stats.*StatsPublisher*	An interface for any possible implementation for publishing statics.

last-updated: Apr 11, 2022

Interface	Description
org.apache.hadoop.hive.shims.*HadoopShims*	In order to be compatible with multiple versions of Hadoop, all parts of the Hadoop interface that are not cross-version compatible are encapsulated in an implementation of this class.
org.apache.hive.hcatalog.streaming.*StreamingConnection*	Represents a connection to a HiveEndPoint.
org.apache.hive.hcatalog.streaming.*TransactionBatch*	Represents a set of Transactions returned by Hive.
org.apache.hive.service.cli.*ICLIService*	
org.apache.hive.service.cli.*RowSet*	
org.apache.hive.service.cli.operation.*TableTypeMapping*	
org.apache.hive.service.cli.session.*HiveSession*	
org.apache.hive.service.cli.session.*HiveSessionBase*	Methods that don't need to be executed under a doAs context are here.
org.apache.hive.spark.client.*JobContext*	Holds runtime information about the job execution context.
org.apache.hive.spark.client.*SparkClient*	Defines the API for the Spark remote client.

Removed API in Hive 2.1

The following classes and interfaces are not available with Hive 2.1:

Removed Classes

Class	Description
org.apache.hadoop.hive.common.metrics.Metrics	Metrics Subsystem - allows exposure of a number of named parameters/counters via jmx, intended to be used as a static subsystem Has a couple of primary ways it can be used: (i) Using the set and get methods to set and get named parameters (ii) Using the incrementCounter method to increment and set named parameters in one go, rather than having to make a get and then a set.
org.apache.hadoop.hive.hbase.HBaseStatsAggregator	A class that implements the StatsAggregator interface through HBase.
org.apache.hadoop.hive.hbase.HBaseStatsPublisher	A class that implements the StatsPublisher interface through HBase.
org.apache.hadoop.hive.hbase.HBaseStatsSetupConstants	HBase constants statistics setup.
org.apache.hadoop.hive.hbase.HBaseStatsUtils	Utilities for hbase statistics.
org.apache.hadoop.hive.metastore.ProtectMode	Protection Mode.
org.apache.hadoop.hive.metastore.txn.CompactionTxnHandler	Extends the transaction handler with methods needed only by the compactor threads.
org.apache.hadoop.hive.metastore.txn.TxnHandler	A handler to answer transaction related calls that come into the metastore server.
org.apache.hadoop.hive.metastore.txn.ValidCompactorTxnList	And implmentation of org.apache.hadoop.hive.common.ValidTxnList for use by the compactor.
org.apache.hadoop.hive.ql.exec.DefaultFetchFormatter	Serializes row by user specified serde and call toString() to make string type result

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.exec.Heartbeater	Class to handle heartbeats for MR and Tez tasks.
org.apache.hadoop.hive.ql.exec.vector.RandomRowObjectSource	Generates object inspector and random row object[].
org.apache.hadoop.hive.ql.exec.vector.VectorAssignRowDynBatch	Assigns specified columns of a VectorizedRowBatch row from a Writable row Object[].
org.apache.hadoop.hive.ql.exec.vector.VectorAssignRowSameBatch	Assigns specified columns of a VectorizedRowBatch row from a Writable row Object[].
org.apache.hadoop.hive.ql.exec.vector.VectorSerializeRowNoNulls	Serializes columns from a row in a VectorizedRowBatch into a serialization format.
org.apache.hadoop.hive.ql.exec.vector.VectorizedColumnarSerDe	VectorizedColumnarSerDe is used by Vectorized query execution engine for columnar based storage supported by RCFile.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFDayOfMonthLong	Expression to get day of month.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFHourLong	Returns hour of day.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFMinuteLong	Returns minute value.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFMonthLong	Returns month value.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFSecondLong	Expression to get seconds.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFTimestampFieldLong	Abstract class to return various fields from a Timestamp or Date.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFUnixTimeStampLong	Return Unix Timestamp.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFWeekOfYearLong	Expression to get week of year.
org.apache.hadoop.hive.ql.exec.vector.expressions.VectorUDFYearLong	Expression to get year as a long.
org.apache.hadoop.hive.ql.exec.vector.mapjoin.fast.VectorMapJoinFastIntHashUtil	Utilities for mapr join in vectorization mode.
org.apache.hadoop.hive.ql.io.VectorizedRCFileInputFormat	A MapReduce/Hive Vectorized input format for RC files.
org.apache.hadoop.hive.ql.io.VectorizedRCFileRecordReader	RCFileRecordReader.
org.apache.hadoop.hive.ql.io.orc.FileDump	A tool for printing out the file structure of ORC files.
org.apache.hadoop.hive.ql.io.orc.InStream	Class to define input stream.
org.apache.hadoop.hive.ql.io.orc.Metadata	Metadata stored in underlying db.
org.apache.hadoop.hive.ql.io.orc.MetadataReader	Class to read and process metadata.
org.apache.hadoop.hive.ql.io.orc.OrcProto	Class to serialize data stored in orc.
org.apache.hadoop.hive.ql.io.orc.OrcUtils	Utilities to process orc files.
org.apache.hadoop.hive.ql.io.orc.RecordReaderFactory	Factory to create ORC tree readers.
org.apache.hadoop.hive.ql.io.orc.RecordReaderUtils	Stateless methods shared between RecordReaderImpl and EncodedReaderImpl.

last-updated: Apr 11, 2022

Class	Description
org.apache.hadoop.hive.ql.io.orc.StripeStatistics	Information about index data stored in stripe.
org.apache.hadoop.hive.ql.io.orc.TreeReaderFactory	Factory for creating ORC tree readers.
org.apache.hadoop.hive.ql.log.PidDailyRollingFileAppender	Logging pids in file.
org.apache.hadoop.hive.ql.optimizer.calcite.HiveConfigContext	Hive configuration context.
org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveLimit	Define limit operator.
org.apache.hadoop.hive.ql.optimizer.calcite.reloperators.HiveSort	Define sort operator.
org.apache.hadoop.hive.ql.parse.VariableSubstitution	The Hive variable substitution mechanism was designed to avoid some of the code that was getting baked into the scripting language on top of Hive.
org.apache.hadoop.hive.ql.session.DependencyResolver	Creating list of dependency jars.
org.apache.hadoop.hive.ql.stats.CounterStatsAggregator	Counter statistics aggregator.
org.apache.hadoop.hive.ql.stats.CounterStatsAggregatorSpark	Counter statistics aggregator for Spark.
org.apache.hadoop.hive.ql.stats.CounterStatsAggregatorTez	This class aggregates stats via counters and does so for Tez Tasks.
org.apache.hadoop.hive.ql.stats.CounterStatsPublisher	Counter statistics publisher.
org.apache.hadoop.hive.ql.udf.UDFRegExp	UDFRegExp.
org.apache.hadoop.hive.shims.Hadoop20SShims	Implemention of shims against Hadoop 0.20 with Security.
org.apache.hadoop.hive.shims.HiveEventCounter	Hive event counter.
org.apache.hadoop.hive.shims.Jetty20SShims	In order to be compatible with multiple versions of Jetty, all parts of the Jetty interface that are not cross-version compatible are encapsulated in an implementation of this class.
org.apache.hadoop.mapred.WebHCatJTShim20S	This is in org.apache.hadoop.mapred package because it relies on JobSubmissionProtocol which is package private
org.apache.hive.benchmark.vectorization.VectorizationBench	Measures the performance for vectorization.
org.apache.hive.jdbc.ZooKeeperHiveClientHelper	Resolve to a host:port by connecting to ZooKeeper and picking a host randomly.
org.apache.hive.service.cli.Column	Column.
org.apache.hive.service.cli.Type	Type.
org.apache.hive.service.cli.thrift.TArrayTypeEntry	Array type entry.
org.apache.hive.service.cli.thrift.TBinaryColumn	Binary column.
org.apache.hive.service.cli.thrift.TBoolColumn	Boolean column.
org.apache.hive.service.cli.thrift.TBoolValue	Boolean value.
org.apache.hive.service.cli.thrift.TByteColumn	Byte column.
org.apache.hive.service.cli.thrift.TByteValue	Byte value.

last-updated: Apr 11, 2022

Class	Description
org.apache.hive.service.cli.thrift.TCLIService	Command line interface service.
org.apache.hive.service.cli.thrift.TCLIServiceConstants	Command line interface constatnts.
org.apache.hive.service.cli.thrift.TCancelDelegationTokenReq	Cancel delegation token request.
org.apache.hive.service.cli.thrift.TCancelDelegationTokenResp	Cancel delegation token response.
org.apache.hive.service.cli.thrift.TCancelOperationReq	Cancel operation request.
org.apache.hive.service.cli.thrift.TCancelOperationResp	Cancel operation response.
org.apache.hive.service.cli.thrift.TCloseOperationReq	Close operation request.
org.apache.hive.service.cli.thrift.TCloseOperationResp	Close operation response.
org.apache.hive.service.cli.thrift.TCloseSessionReq	Close session request.
org.apache.hive.service.cli.thrift.TCloseSessionResp	Close session response.
org.apache.hive.service.cli.thrift.TColumn	Column.
org.apache.hive.service.cli.thrift.TColumnDesc	Column description.
org.apache.hive.service.cli.thrift.TColumnValue	Column value.
org.apache.hive.service.cli.thrift.TDoubleColumn	Double column.
org.apache.hive.service.cli.thrift.TDoubleValue	Double value.
org.apache.hive.service.cli.thrift.TExecuteStatementReq	Execute statement request.
org.apache.hive.service.cli.thrift.TExecuteStatementResp	Execute statement response.
org.apache.hive.service.cli.thrift.TFetchOrientation	Fetch orientation.
org.apache.hive.service.cli.thrift.TFetchResultsReq	Fetch results request.
org.apache.hive.service.cli.thrift.TFetchResultsResp	Fetch results response.
org.apache.hive.service.cli.thrift.TGetCatalogsReq	Get catalogs request.

Removed Interfaces

Interface	Description
org.apache.hadoop.hive.ql.exec.*FetchFormatter*	(For internal-use only) Used in ListSinkOperator for formatting final output
org.apache.hadoop.hive.ql.io.orc.*BinaryColumnStatistics*	Statistics for binary columns.
org.apache.hadoop.hive.ql.io.orc.*BooleanColumnStatistics*	Statistics for boolean columns.
org.apache.hadoop.hive.ql.io.orc.*ColumnStatistics*	Statistics that are available for all types of columns.
org.apache.hadoop.hive.ql.io.orc.*CompressionCodec*	Compress the in buffer to the out buffer.
org.apache.hadoop.hive.ql.io.orc.*CompressionCodec.Modifier*	Compress the in buffer to the out buffer.
org.apache.hadoop.hive.ql.io.orc.*ConversionTreeReaderFactory*	Factory for creating ORC tree readers.
org.apache.hadoop.hive.ql.io.orc.*DateColumnStatistics*	Statistics for DATE columns.

Interface	Description
org.apache.hadoop.hive.ql.io.orc.*DecimalColumnStatistics*	Statistics for decimal columns.
org.apache.hadoop.hive.ql.io.orc.*DirectDecompressionCodec*	Decompression codec.
org.apache.hadoop.hive.ql.io.orc.*DoubleColumnStatistics*	Statistics for float and double columns.
org.apache.hadoop.hive.ql.io.orc.*IntegerColumnStatistics*	Statistics for all of the integer columns, such as byte, short, int, and long.
org.apache.hadoop.hive.ql.io.orc.*PositionProvider*	An interface used for seeking to a row index.
org.apache.hadoop.hive.ql.io.orc.*StringColumnStatistics*	Statistics for string columns.
org.apache.hadoop.hive.ql.io.orc.*StripeInformation*	Information about the stripes in an ORC file that is provided by the Reader.
org.apache.hadoop.hive.ql.io.orc.*TimestampColumnStatistics*	Statistics for Timestamp columns.
org.apache.hadoop.hive.ql.stats.*StatsCollectionTaskIndependent*	Marker interface to differentiate between stats publisher / aggregator which don't track stats per task, as oppose to others which do.

Deprecated API in Hive 2.1

The following classes, interfaces, and fields have been deprecated in Hive 2.1.

Deprecated Classes

org.apache.hadoop.hive.ql.exec.ByteWritable
org.apache.hadoop.hive.serde.Constants
org.apache.hadoop.hive.ql.io.FlatFileInputFormat
org.apache.hadoop.hive.ql.io.FlatFileInputFormat.FlatFileRecordReader
org.apache.hadoop.hive.ql.io.IgnoreKeyTextOutputFormat *use HiveIgnoreKeyTextOutputFormat instead}*
org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe.SerDeParameters
org.apache.hadoop.hive.ql.exec.UDAF *Either implement* GenericUDAFResolver2 *or extend AbstractGenericUDAFResolver instead.*

Deprecated Interfaces

org.apache.hadoop.hive.serde2.Deserializer
org.apache.hadoop.hive.ql.udf.generic.GenericUDAFEvaluator.AggregationBuffer *use GenericUDAFEvaluator.AbstractAggregationBuffer instead*
org.apache.hadoop.hive.ql.udf.generic.GenericUDAFResolver *Use GenericUDAFResolver2 instead.*
org.apache.hadoop.hive.serde2.SerDe
org.apache.hadoop.hive.serde2.Serializer

Deprecated Fields

org.apache.hadoop.hive.serde2.avro.AvroSerdeUtils.AVRO_SERDE_SCHEMA
org.apache.hadoop.hive.serde2.avro.AvroSerdeUtils.SCHEMA_DOC
org.apache.hadoop.hive.serde2.avro.AvroSerdeUtils.SCHEMA_LITERAL
org.apache.hadoop.hive.serde2.avro.AvroSerdeUtils.SCHEMA_NAME
org.apache.hadoop.hive.serde2.avro.AvroSerdeUtils.SCHEMA_NAMESPACE
org.apache.hadoop.hive.serde2.avro.AvroSerdeUtils.SCHEMA_RETRIEVER
org.apache.hadoop.hive.serde2.avro.AvroSerdeUtils.SCHEMA_URL
org.apache.hive.hplsql.HplsqlParser.tokenNames *Use HplsqlParser.VOCABULARY instead.*
org.apache.hive.hplsql.HplsqlLexer.tokenNames *Use HplsqlLexer.VOCABULARY instead.*

Deprecated Methods

org.apache.hadoop.hive.metastore.IMetaStoreClient.addDynamicPartitions(long, String, String, List<String>) *in Hive 1.3.0/2.1.0 - will be removed in 2 releases*
org.apache.hadoop.hive.metastore.HiveMetaStoreClient.addDynamicPartitions(long, String, String, List<String>)
org.apache.hadoop.hive.serde2.ColumnProjectionUtils.appendReadColumnIDs(Configuration, List<Integer>) *for backwards compatibility with <= 0.12, use appendReadColumns*
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.bucketCols(List<String>, int)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.collectionItemsTerminatedBy(char)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.comments(String)
org.apache.hadoop.hive.metastore.IMetaStoreClient.compact(String, String, String, CompactionType)
org.apache.hadoop.hive.metastore.HiveMetaStoreClient.compact(String, String, String, CompactionType)
org.apache.hadoop.hive.ql.metadata.HiveStorageHandler.configureTableJobProperties(TableDesc, Map<String, String>)
org.apache.orc.impl.InStream.create(String, ByteBuffer[], long[], long, CompressionCodec, int)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.create(String, String, List<HCatFieldSchema>)
org.apache.hive.hcatalog.api.HCatAddPartitionDesc.create(String, String, String, Map<String, String>)
org.apache.hadoop.hive.serde2.lazy.LazyFactory.createColumnarStructInspector(List<String>, List<TypeInfo>, byte[], Text, boolean, byte)
org.apache.hadoop.hive.serde2.lazy.LazyFactory.createLazyObjectInspector(TypeInfo, byte[], int, Text, boolean, byte)
org.apache.hadoop.hive.serde2.lazy.LazyFactory.createLazyObjectInspector(TypeInfo, byte[], int, Text, boolean, byte, boolean)
org.apache.hadoop.hive.serde2.lazy.LazyFactory.createLazyObjectInspector(TypeInfo, byte[], int, Text, boolean, byte, boolean, ObjectInspectorFactory.ObjectInspectorOptions)
org.apache.hadoop.hive.serde2.lazy.LazyFactory.createLazyObjectInspector(TypeInfo, byte[], int, Text, boolean, byte, ObjectInspectorFactory.ObjectInspectorOptions)

last-updated: Apr 11, 2022

org.apache.hadoop.hive.serde2.lazy.LazyFactory.createLazyStructInspector(List<String>, List<TypeInfo>, byte[], Text, boolean, boolean, byte)
org.apache.hadoop.hive.serde2.lazy.LazyFactory.createLazyStructInspector(List<String>, List<TypeInfo>, byte[], Text, boolean, boolean, byte, boolean)
org.apache.hadoop.hive.metastore.IMetaStoreClient.dropTable(String, boolean) *As of release 0.6.0 replaced by IMetaStoreClient.dropTable(String, String, boolean, boolean). This method will be removed in release 0.7.0.*
org.apache.hadoop.hive.metastore.HiveMetaStoreClient.dropTable(String, boolean)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.escapeChar(char)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.fieldsTerminatedBy(char)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.fileFormat(String)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.getBucketCols()
org.apache.hive.hcatalog.api.HCatCreateTableDesc.getCols()
org.apache.hadoop.hive.serde2.dynamic_type.SimpleCharStream.getColumn()
org.apache.hive.hcatalog.api.HCatCreateTableDesc.getComments()
org.apache.hadoop.hive.ql.io.RCFile.Writer.getCompressionCodec()
org.apache.hive.hcatalog.api.HCatCreateTableDesc.getDatabaseName()
org.apache.hive.hcatalog.api.HCatAddPartitionDesc.getDatabaseName()
org.apache.hive.hcatalog.api.HCatCreateTableDesc.getExternal()
org.apache.hadoop.hive.ql.exec.Utilities.getFileExtension(JobConf, boolean) *Use Utilities.getFileExtension(JobConf, boolean, HiveOutputFormat)*
org.apache.hive.hcatalog.api.HCatCreateTableDesc.getFileFormat()
org.apache.hive.hcatalog.common.HCatUtil.getHiveClient(HiveConf)
org.apache.hadoop.hive.serde2.lazy.objectinspector.LazyObjectInspectorFactory.getLazySimpleListObjectInspector(ObjectInspector, byte, Text, boolean, byte)
org.apache.hadoop.hive.serde2.lazy.objectinspector.LazyObjectInspectorFactory.getLazySimpleMapObjectInspector(ObjectInspector, ObjectInspector, byte, byte, Text, boolean, byte)
org.apache.hadoop.hive.serde2.lazy.objectinspector.LazyObjectInspectorFactory.getLazySimpleStructObjectInspector(List<String>, List<ObjectInspector>, byte, Text, boolean, boolean, byte)
org.apache.hadoop.hive.serde2.lazy.objectinspector.LazyObjectInspectorFactory.getLazySimpleStructObjectInspector(List<String>, List<ObjectInspector>, byte, Text, boolean, boolean, byte, ObjectInspectorFactory.ObjectInspectorOptions)
org.apache.hadoop.hive.serde2.lazy.objectinspector.LazyObjectInspectorFactory.getLazySimpleStructObjectInspector(List<String>, List<ObjectInspector>, List<String>, byte, Text, boolean, boolean, byte)
org.apache.hadoop.hive.serde2.lazy.objectinspector.LazyObjectInspectorFactory.getLazySimpleStructObjectInspector(List<String>, List<ObjectInspector>, List<String>, byte, Text, boolean, boolean, byte, ObjectInspectorFactory.ObjectInspectorOptions)
org.apache.hadoop.hive.serde2.lazy.objectinspector.LazyObjectInspectorFactory.getLazyUnionObjectInspector(List<ObjectInspector>, byte, Text, boolean, byte)
org.apache.hadoop.hive.serde2.dynamic_type.SimpleCharStream.getLine()
org.apache.hive.hcatalog.api.HCatCreateTableDesc.getLocation()
org.apache.hive.hcatalog.api.HCatAddPartitionDesc.getLocation()

org.apache.hive.hcatalog.data.schema.HCatFieldSchema.getMapKeyType() *as of 0.13, slated for removal with 0.15 use HCatFieldSchema.getMapKeyTypeInfo() instead*
org.apache.hive.hcatalog.api.HCatCreateTableDesc.getNumBuckets()
org.apache.hadoop.hive.ql.udf.generic.SimpleGenericUDAFParameterInfo.getParameters()
org.apache.hadoop.hive.ql.udf.generic.GenericUDAFParameterInfo.getParameters()
org.apache.hive.hcatalog.api.HCatCreateTableDesc.getPartitionCols()
org.apache.hive.hcatalog.api.HCatAddPartitionDesc.getPartitionSpec()
org.apache.hive.hcatalog.api.HCatCreateTableDesc.getSerdeParams()
org.apache.hive.hcatalog.api.HCatCreateTableDesc.getSortCols()
org.apache.hive.hcatalog.api.HCatCreateTableDesc.getStorageHandler()
org.apache.hadoop.hive.metastore.IMetaStoreClient.getTable(String) *As of release 0.6.0 replaced by IMetaStoreClient.getTable(String, String). This method will be removed in release 0.7.0.*
org.apache.hadoop.hive.metastore.HiveMetaStoreClient.getTable(String)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.getTableName()
org.apache.hive.hcatalog.api.HCatAddPartitionDesc.getTableName()
org.apache.hive.hcatalog.api.HCatCreateTableDesc.getTblProps()
org.apache.hive.hplsql.HplsqlParser.getTokenNames()
org.apache.hive.hplsql.HplsqlLexer.getTokenNames()
org.apache.hive.hcatalog.data.schema.HCatFieldSchema.getType() *as of 0.13, slated for removal with 0.15 use HCatFieldSchema.getTypeInfo() instead*
org.apache.orc.Reader.getTypes() *use getSchema instead*
org.apache.hadoop.hive.serde2.AbstractSerDe.initialize(Configuration, Properties)
org.apache.hadoop.hive.serde2.AbstractEncodingAwareSerDe.initialize(Configuration, Properties)
org.apache.hadoop.hive.ql.udf.generic.GenericUDTF.initialize(ObjectInspector[])
org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe.initSerdeParams(Configuration, Properties, String)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.isTableExternal(boolean)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.linesTerminatedBy(char)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.location(String)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.mapKeysTerminatedBy(char)
org.apache.hive.hcatalog.streaming.HiveEndPoint.newConnection(boolean) *As of release 1.3/2.1. Replaced by HiveEndPoint.newConnection(boolean, String)*
org.apache.hive.hcatalog.streaming.HiveEndPoint.newConnection(boolean, HiveConf) *As of release 1.3/2.1. Replaced by HiveEndPoint.newConnection(boolean, HiveConf, String)*

org.apache.hive.hcatalog.streaming.HiveEndPoint.newConnection(boolean, HiveConf, UserGroupInformation) *As of release 1.3/2.1. Replaced by HiveEndPoint.newConnection(boolean, HiveConf, UserGroupInformation, String)*
org.apache.hadoop.hive.ql.io.RCFile.Reader.nextColumnsBatch()
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.nullDefinedAs(char)
org.apache.hive.service.cli.CLIService.openSession(TProtocolVersion, String, String, Map<String, String>) *Use CLIService.openSession(TProtocolVersion, String, String, String, Map)*
org.apache.hive.service.cli.CLIService.openSessionWithImpersonation(TProtocolVersion, String, String, Map<String, String>, String) *Use CLIService.openSessionWithImpersonation(TProtocolVersion, String, String, String, Map, String)*
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.partCols(List<HCatFieldSchema>)
org.apache.hadoop.hive.ql.io.parquet.ProjectionPusher.pushProjectionsAndFilters(JobConf, Path)
org.apache.hadoop.hive.ql.io.NonSyncDataInputBuffer.readLine() *Use BufferedReader*
org.apache.hadoop.hive.ql.hooks.PostExecute.run(SessionState, Set<ReadEntity>, Set<WriteEntity>, LineageInfo, UserGroupInformation)
org.apache.hadoop.hive.ql.hooks.PreExecute.run(SessionState, Set<ReadEntity>, Set<WriteEntity>, UserGroupInformation)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.serdeParam(String, String)
org.apache.hadoop.hive.ql.io.RCFile.ValueBuffer.setColumnValueBuffer(NonSyncDataOutputBuffer, int)
org.apache.hive.hcatalog.mapreduce.HCatInputFormat.setFilter(String) *as of 0.13, slated for removal with 0.15 Use HCatInputFormat.setInput(org.apache.hadoop.conf.Configuration, String, String, String) instead, to specify a partition filter to directly initialize the input with.*
org.apache.hadoop.hive.serde2.ColumnProjectionUtils.setFullyReadColumns(Configuration) *for backwards compatibility with <= 0.12, use setReadAllColumns*
org.apache.hadoop.hive.serde2.ColumnProjectionUtils.setReadColumnIDs(Configuration, List<Integer>) *for backwards compatibility with <= 0.12, use setReadAllColumns and appendReadColumns*
org.apache.hadoop.hive.metastore.IMetaStoreClient.showLocks()
org.apache.hadoop.hive.metastore.HiveMetaStoreClient.showLocks()
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.sortCols(ArrayList<Order>)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.storageHandler(String)
org.apache.hadoop.hive.metastore.IMetaStoreClient.tableExists(String) *As of release 0.6.0 replaced by IMetaStoreClient.tableExists(String, String). This method will be removed in release 0.7.0.*
org.apache.hadoop.hive.metastore.HiveMetaStoreClient.tableExists(String)
org.apache.hive.hcatalog.api.HCatCreateTableDesc.Builder.tblProps(Map<String, String>)
org.apache.hadoop.hive.metastore.ObjectStore.updateMStorageDescriptorTblPropURI(URI, URI, String, boolean)
org.apache.hadoop.hive.metastore.HiveMetaStoreClient.updatePartitionColumnStatistics(ColumnStatistics)
org.apache.hadoop.hive.metastore.HiveMetaStoreClient.updateTableColumnStatistics(ColumnStatistics)

Troubleshooting Hive and Tez

This section includes Hive and Tez troubleshooting tips.

HDFS Literal Deprecated

Starting in Hive-2.3, the `hdfs` literal is deprecated. Specifying a table location using the `hdfs` URI scheme will cause queries to fail because the Hive parser recognizes the `hdfs` literal in the LOCATION key word and triggers HDFS encryption, which is not supported.

If you use the `hdfs` literal with the LOCATION keyword in Hive queries:

```
CREATE TABLE IF NOT EXISTS i (id INT) LOCATION 'hdfs:///i';
```

The system logs the following warning:

```
LOG.warn("hdfs:// is deprecated filesystem and will be removed in future
releases. Use maprfs://
        instead");
```

To avoid `hdfs` literal issues, update all instances of `hdfs` with `maprfs` in tables, partitions, and databases. Also update the `hive-site.xml` file to remove `hdfs` from the URI scheme list.

Update hive-site.xml

Remove `hdfs` from the `hive.exim.uri.scheme.whitelist` Hive configuration property in `hive-site.xml`, as shown:

```
<property>

<name>hive.exim.uri.scheme.whitelist</
name>
    <value>maprfs,…,…,s3</value>
</property>
```

Update Tables and Partitions

To replace the table and partition location with `maprfs`, run:

```
MariaDB [hive]>  update SDS set
LOCATION = REPLACE(LOCATION, 'hdfs',
'maprfs') where LOCATION like
'%hdfs%';
```

Update Databases

To replace the database location with `maprfs`, run:

```
MariaDB [hive]> update
DBS set DB_LOCATION_URI =
REPLACE(DB_LOCATION_URI, 'hdfs',
'maprfs') where DB_LOCATION_URI like
'%hdfs%';
```

Prohibited usage of `datanucleus.schema.autoCreateAll` property

The usage of the `datanucleus.schema.autoCreateAll` property is prohibited in all cases. Instead of using this property, you must run the `schematool` command. Refer to HIVE-21302 for more information.

WebHCat

Secure WebHCat operations depend on the Hive metastore having Kerberos enabled. If Kerberos is not enabled for the Hive metastore, null pointer exceptions similar to the following will appear:

```
2013-10-06 20:38:55,198 ERROR metastore.RetryingHMSHandler
(RetryingHMSHandler.java:invoke(134)) -
MetaException(message:java.lang.NullPointerException)
       at
org.apache.hadoop.hive.metastore.HiveMetaStore$HMSHandler.get_delegation_tok
en(HiveMetaStore.java:3972)
       at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
       at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39
)
       at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl
.java:25)
       at java.lang.reflect.Method.invoke(Method.java:597)
       at
org.apache.hadoop.hive.metastore.RetryingHMSHandler.invoke(RetryingHMSHandle
r.java:102)
       at com.sun.proxy.$Proxy5.get_delegation_token(Unknown Source)
       at
org.apache.hadoop.hive.metastore.api.ThriftHiveMetastore$Processor$get_deleg
ation_token.getResult(ThriftHiveMetastore.java:8063)
       at
org.apache.hadoop.hive.metastore.api.ThriftHiveMetastore$Processor$get_deleg
ation_token.getResult(ThriftHiveMetastore.java:8047)
       at org.apache.thrift.ProcessFunction.process(ProcessFunction.java:39)
       at org.apache.thrift.TBaseProcessor.process(TBaseProcessor.java:39)
       at
org.apache.hadoop.hive.metastore.TSetIpAddressProcessor.process(TSetIpAddres
sProcessor.java:48)
       at
org.apache.thrift.server.TThreadPoolServer$WorkerProcess.run(TThreadPoolServ
er.java:206)
       at
java.util.concurrent.ThreadPoolExecutor$Worker.runTask(ThreadPoolExecutor.ja
va:895)
       at
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:9
18)
```

If you are updating to the `mapr-hive-2.3.6-mapr-1912` (EEP-6.3.0) package, you should manually replace the old `webhcat-default.xml` configuration file with the new one and restart the WebHCat service:

```
cp /opt/mapr/hive/hive-2.3/hcatalog/etc/webhcat.new/
webhcat-default.xml /opt/mapr/hive/hive-2.3/hcatalog/etc/webhcat/
```

Hive in an Azure Cluster

When Hive services are installed on an Azure cluster, it is possible that the services will not start because Azure assigns too long (over 64 symbols) host names. Perform following steps to fix this issue:

Note: This issue is fixed on MapR core 6.0.1 starting from build `20180320175756.GA-1.x86_64`.

last-updated: Apr 11, 2022

1. Edit the `/etc/hosts` file:

```
nano /etc/hosts
```

   This is an example of a `/etc/hosts` file for an Azure cluster:

```
127.0.0.1   localhost localhost.localdomain localhost4
localhost4.localdomain4
::1         localhost localhost.localdomain localhost6
localhost6.localdomain6
172.24.8.4
anaikregtestc73522602-cluster-com-mapr-vm0.izqafobxqxbuzkv4led1p3snic.dx.
internal.cloudapp.net anaikregtestc73522602-cluster-com-mapr-vm0
172.24.8.5
anaikregtestc73522602-cluster-com-mapr-vm1.izqafobxqxbuzkv4led1p3snic.dx.
internal.cloudapp.net anaikregtestc73522602-cluster-com-mapr-vm1
172.24.8.6
anaikregtestc73522602-cluster-com-mapr-vm2.izqafobxqxbuzkv4led1p3snic.dx.
internal.cloudapp.net anaikregtestc73522602-cluster-com-mapr-vm2
172.24.8.7
anaikregtestc73522602-cluster-com-mapr-vm3.izqafobxqxbuzkv4led1p3snic.dx.
internal.cloudapp.net anaikregtestc73522602-cluster-com-mapr-vm3
172.24.8.8
anaikregtestc73522602-cluster-com-mapr-vm4.izqafobxqxbuzkv4led1p3snic.dx.
internal.cloudapp.net anaikregtestc73522602-cluster-com-mapr-vm4
```

2. Add a short alias for each node:

   Note: You can use any short alias. In this example, `vm0`, `vm1`, `vm2`, `vm3`, and `vm4` are used:

```
127.0.0.1   localhost localhost.localdomain localhost4
localhost4.localdomain4
::1         localhost localhost.localdomain localhost6
localhost6.localdomain6
172.24.8.4
anaikregtestc73522602-cluster-com-mapr-vm0.izqafobxqxbuzkv4led1p3snic.dx.
internal.cloudapp.net anaikregtestc73522602-cluster-com-mapr-vm0 vm0
172.24.8.5
anaikregtestc73522602-cluster-com-mapr-vm1.izqafobxqxbuzkv4led1p3snic.dx.
internal.cloudapp.net anaikregtestc73522602-cluster-com-mapr-vm1 vm1
172.24.8.6
anaikregtestc73522602-cluster-com-mapr-vm2.izqafobxqxbuzkv4led1p3snic.dx.
internal.cloudapp.net anaikregtestc73522602-cluster-com-mapr-vm2 vm2
172.24.8.7
anaikregtestc73522602-cluster-com-mapr-vm3.izqafobxqxbuzkv4led1p3snic.dx.
internal.cloudapp.net anaikregtestc73522602-cluster-com-mapr-vm3 vm3
172.24.8.8
anaikregtestc73522602-cluster-com-mapr-vm4.izqafobxqxbuzkv4led1p3snic.dx.
internal.cloudapp.net anaikregtestc73522602-cluster-com-mapr-vm4 vm4
```

3. Perform step 1 and 2 for each node in the cluster.

Tez Upgrade Issues

- Preserving configuration on Ubuntu is not supported from EEP 4.1.1 and EEP 5.0.0 (1803) to EEP 6.0.0 (1808) or EEP 5.0.1 (1808).
- Preserving Tomcat configuration is not supported from any previous EEP to EEP 6.0.0 (1808).

last-updated: Apr 11, 2022

- You must manually stop the Tomcat service and delete the Tomcat folder as a precondition if you are updating or upgrading Tez from the following EEPs:
  - EEP 4.0.0
  - EEP 4.1.0

User Names, Group Names, and LDAP

LDAP configuration allows you to use group names and usernames with spaces, so it is possible to name groups with spaces in them, for example, `domain users`. The following structure is possible in the MapR FileSystem:

```
drwxr-xr-x   - afischer      domain users          0 2018-10-03 16:10 /
user/abc
drwxr-xr-x   - mapr          mapr                  0 2018-10-05 16:51 /
user/def
drwxr-xr-x   - dschexnayder  domain users          8 2018-10-10 13:30 /
user/xyz
drwxr-xr-x   - mapr          mapr                  1 2018-10-09 14:23 /user/
hive
drwxr-xr-x   - mapr          mapr                 11 2018-10-10 01:56 /user/
mapr
drwxr-xr-x   - mlitovsky     domain users          0 2018-10-06 11:08 /user/
hjbs
drwxr-xr-x   - pcurtis       domain users          5 2018-10-04 19:33 /user/
jknd
drwxr-xr-x   - mapr          mapr                  3 2018-10-08 16:29 /user/
ewkd
drwxr-xr-x   - talvarez      domain users          0 2018-10-04 17:02 /
user/lkd
```

According to HADOOP-12505, the Hadoop community does not allow spaces in group names, and because of that so does Hive. Each time you perform a query in Hive on a group name that has a space, you will see the following exception:

```
-chgrp: 'domain users' does not match expected pattern for group
```

The workaround is to not use spaces in group names or user names.

HiveServer 2 takes time to start because of `get_all_databases`

Materialized view registry and cache is introduced in HIVE-14496 for Hive 2.3.0.

The goal of the cache is to avoid parsing and creating logical plans for the materialized views at query runtime. When a query arrives, you need to consult this cache and extract the logical plans for the views (which are already parsed) from it. Materialized view registry class scans all databases and tables in each database during initialization and that may cause long time to start HiveServer2.

Property `hive.materializedview.enable.views.registry` is added to control the usage of materialized view registry:

Property: `hive.materializedview.enable.views.registry`

Default value: true

Description: In case of a large amount of databases and tables in Hive, usage of materialized view registry and cache force HiveServer2 to scan all of them in order to cache the query plan for a view. This leads to an extremely long time for HiveServer2 to start.

This property is used to disable view registry and cache for this case. To disable materialized view registry and cache, add the following to `hive-site.xml` and restart Hive services.

```
<property>
  <name>hive.materializedview.enable.views.registry</name>
  <value>false</value>
</property>
```

Database and Table Names Containing a Dot (.)

HIVE-16907 rejects queries with database and table names that contain a dot (`.`), and this behavior is backported to Hive 2.3.

Databases and tables that contain a dot (`.`) in the name are not supported now. For example:

```
{code}
insert into `tdb.t1` select * from t2;
{code}
Throws error:
{code}
FAILED: SemanticException
org.apache.hadoop.hive.ql.parse.SemanticException: Line 1:12 Table or
database name may not contain dot(.) character 'tdb.t1'
{code}
```

Avoid using unsupported characters in database and table names.

Hive Logging

This section describes Hive logging for Hive 2.1 and later releases and includes information about log splitting.

Hive Logging (Hive 2.3 and Later)

For Hive 2.3 and later starting with EEP 6.3.0, this topic describes the folder structure of the Hive logs and includes details about the log-file contents and how log files are installed in multinode installations.

Hive Log Folder Structure

Table 1 shows the Hive log folder structure:

Table

Folder or File			Description
`${HIVE_HOME}/log`			Root folder for all Hive logs

	`hive-${ADMIN_USER}-hiveserver2-${HOSTNAME}.out`
	`hive-${ADMIN_USER}-metastore-${HOSTNAME}.out`
	`init_derby_db_${TIMESTAMP}.log`
			Each service has a separate file for logging

`${HIVE_HOME}/log/${ADMIN_USER}`			Root folder for admin cluster logs

		`${ADMIN_USER}-hiveserver2-${HOSTNAME}.log`
		`${ADMIN_USER}-metastore-${HOSTNAME}.log`
		`${ADMIN_USER}-cli-${HOSTNAME}.log`
			Cluster admin log files

last-updated: Apr 11, 2022

Table (Continued)

Folder or File	Description
`${HIVE_HOME}/log/${ADMIN_USER}/webhcat/`	Root folder for webHcat logs

`webhcat.log`
`webhcat-console.log`
`webhcat-console-error.log`
	WebHcat log files

`${HIVE_HOME}/log/${OTHER_USER}`	Root folder for a user other than the admin user
`${OTHER_USER}-cli-${HOSTNAME}.log`	CLI log for a user other than the admin user

In Table 1:

This element	Represents
`${HIVE_HOME}`	The Hive home folder, which is usually `/opt/mapr/hive/hive`.
`${ADMIN_USER}`	The admin user of a cluster that runs HiveServer2 and HiveMetastore daemons. Usually, this is the `mapr` user.
`${HOSTNAME}`	The name of the host where the daemon runs.
`${TIMESTAMP}`	The date and time of log creation.
`${OTHER_USER}`	A user other than the admin user.
`${OTHER_USER}-cli-${HOSTNAME}.log`	The log file that is created when `${OTHER_USER}` launches the Hive CLI.

Content of Log Files

Table 2 shows the content of the log files:

Table

File Name	Description
`hive-${ADMIN_USER}-hiveserver2-${HOSTNAME}.out`	Contains information about when the HiveServer2 daemon was started and the PID of the file.
`hive-${ADMIN_USER}-metastore-${HOSTNAME}.out`	Contains information about when the HiveMetastore daemon was started and the PID of the file.
`${ADMIN_USER}-hiveserver2-${HOSTNAME}.log`	Contains information from the HiveServer2 daemons. This file also contains the job progress.
`${ADMIN_USER}-metastore-${HOSTNAME}.log`	Contains information from the HiveMetastore daemons.
`${ADMIN_USER}-cli-${HOSTNAME}.log`	Created when a user runs the Hive CLI over the ${ADMIN_USER}. This file contains the job progress.
`${OTHER_USER}-cli-${HOSTNAME}.log`	Created when a user runs the Hive CLI over the ${OTHER_USER}. This file also contains the job progress.
`init_derby_db_${TIMESTAMP}.log`	Created if and only if Hive was configured for Apache Derby through cluster installation.

last-updated: Apr 11, 2022

Log Files in a Multinode Hive Installation

Table 3 shows a Hive multinode installation (that is, Hive packages installed on different nodes):

Table

	Hive Metastore	HiveServer2	HiveWebHCat
node1	⅓		
node2		⅓	
node3			⅓

See Table 4 for the log configurations:

Table

File or Folder Name	node1	node2	node3
`hive-${ADMIN_USER}-metastore- ${HOSTNAME}.out`	⅓		
`${ADMIN_USER}-metastore- ${HOSTNAME}.log`	⅓		
`hive-${ADMIN_USER}-hiveserver2- ${HOSTNAME}.out`		⅓	
`${ADMIN_USER}-hiveserver2- ${HOSTNAME}.log`		⅓	
`/webhcat/`			⅓
`webhcat.log`			⅓
`webhcat-console.log`			⅓
`webhcat-console-error.log`			⅓
`/${ADMIN_USER}/`	⅓	⅓	⅓

Related concepts

Disabling Log Splitting of Hive Log Files on page 3606
By default, Hive log files are split into HiveServer2 and Metastore log files, but you can disable log splitting by editing the `hive-env.sh` file.

Splitting Hive Logs into HiveServer2 and Metastore logs by Process ID on page 3606
Starting from the 1904 release, you can split Hive log files into HiveServer2 and Metestore log files by process ID.

Hive Logging (Hive 2.1 and Later)

For certain Hive 2.1 and later releases, this topic describes the folder structure of the Hive logs and includes details about the log-file contents and how log files are installed in multinode installations.

The Hive log information in this topic applies to Hive 2.1 and later releases beginning with the 1803 release-date identifier. Included are the Hive releases in MapR Ecosystem Packs (EEPs) 3.0.3, 3.0.4, 3.0.5, 4.1.1, 4.1.2, 4.1.3, 4.1.4, 5.0.x, 6.0.x, 6.1.1, and 6.2.0. For more information about release-date identifiers, see Release History for EEPs on page 5623.

Default log folder structure

Hive logs have the following folder structure:

```
${HIVE_HOME}/log
${HIVE_HOME}/log/hive-${ADMIN_USER}-hiveserver2-${HOSTNAME}.out
${HIVE_HOME}/log/hive-${ADMIN_USER}-metastore-${HOSTNAME}.out
${HIVE_HOME}/log/init_derby_db_${TIMESTAMP}.log
${HIVE_HOME}/log/${ADMIN_USER}
```


last-updated: Apr 11, 2022

```
${HIVE_HOME}/log/${ADMIN_USER}/${ADMIN_USER}-hiveserver2-${HOSTNAME}.log
${HIVE_HOME}/log/${ADMIN_USER}/${ADMIN_USER}-metastore-${HOSTNAME}.log
${HIVE_HOME}/log/${ADMIN_USER}/webhcat/
${HIVE_HOME}/log/${ADMIN_USER}/webhcat/webhcat.log
${HIVE_HOME}/log/${ADMIN_USER}/webhcat/webhcat-console.log
${HIVE_HOME}/log/${ADMIN_USER}/webhcat/webhcat-console-error.log
${HIVE_HOME}/log/${OTHER_USER}
${HIVE_HOME}/log/${OTHER_USER}/${OTHER_USER}-hiveserver2-${HOSTNAME}.log
```

Here:

```
${HIVE_HOME} - Hive home folder. Usually this is /opt/mapr/hive/hive.
${ADMIN_USER} - Admin user of cluster that runs HiveServer2 and
HiveMetastore daemons. Usually this is mapr.
${HOSTNAME} - Name of the host where a daemon runs.
${TIMESTAMP} - Date and time of log creation.
${OTHER_USER} - Not an admin user.
```

Content of log files

Files `$HIVE_HOME/log/hive-${ADMIN_USER}-hiveserver2-${HOSTNAME}.out` and `$HIVE_HOME/log/hive-${ADMIN_USER}-metastore-${HOSTNAME}.out` contain information about when HiveServer2 and HiveMetastore daemons are stated, and what are their PIDs.

Files `$HIVE_HOME/log/${ADMIN_USER}/${ADMIN_USER}-hiveserver2-${HOSTNAME}.log` and `$HIVE_HOME/log/${ADMIN_USER}/${ADMIN_USER}-hiveserver2-${HOSTNAME}.log` contain information from HiveServer2 and HiveMetastore daemons. File `${ADMIN_USER}-hiveserver2-${HOSTNAME}.log` also contains job progress.

The `${HIVE_HOME}/log/${OTHER_USER}/${OTHER_USER}-hiveserver2-${HOSTNAME}.log` file is created when somebody runs Hive CLI over the `${OTHER_USER}`. The `${OTHER_USER}-hiveserver2-${HOSTNAME}.log` file contains job progress.

The `${HIVE_HOME}/log/init_derby_db_${TIMESTAMP}.log` file is created if and only if Hive was configured for Derby Db through cluster installation.

Log files on multi node Hive installation

Consider Hive multi node installation (that is Hive packages are installed on different nodes). See Table 1:

Table

	Hive Metastore	HiveServer2	HiveWebHCat
node1	⅓		
node2		⅓	
node3			⅓

See Table 2 for log configurations.

Table

	node1	node2	node3
`${HIVE_HOME}/logs/hive-${ADMIN_USER}-metastore- ${HOSTNAME}.out`	⅓		
`${HIVE_HOME}/logs/${ADMIN_USER}/${ADMIN_USER}-metastore- ${HOSTNAME}.log`	⅓		
`${HIVE_HOME}/logs/hive-${ADMIN_USER}-hiveserver2- ${HOSTNAME}.out`		⅓	

Table (Continued)

	node1	node2	node3
`${HIVE_HOME}/logs/${ADMIN_USER}/${ADMIN_USER}-hiveserver2- ${HOSTNAME}.log`		⅓	
`${HIVE_HOME}/logs/${ADMIN_USER}/webhcat/`			⅓
`${HIVE_HOME}/logs/${ADMIN_USER}/webhcat/webhcat.log`			⅓
`${HIVE_HOME}/logs/${ADMIN_USER}/webhcat/webhcat-console.log`			⅓
`${HIVE_HOME}/logs/${ADMIN_USER}/webhcat/webhcat-console-error.log`			⅓
`${HIVE_HOME}/logs/${ADMIN_USER}`	⅓	⅓	⅓

Related concepts

Disabling Log Splitting of Hive Log Files on page 3606
By default, Hive log files are split into HiveServer2 and Metastore log files, but you can disable log splitting by editing the `hive-env.sh` file.

Splitting Hive Logs into HiveServer2 and Metastore logs by Process ID on page 3606
Starting from the 1904 release, you can split Hive log files into HiveServer2 and Metestore log files by process ID.

Disabling Log Splitting of Hive Log Files

By default, Hive log files are split into HiveServer2 and Metastore log files, but you can disable log splitting by editing the `hive-env.sh` file.

This information is valid for Hive-2.1+ starting from the EEP-1803 release.

Disabling Log Splitting of Hive Log Files

You can disable splitting the Hive log files into HiveServer2 and Metastore log files. To write all logs to the `hive.log` file, use these steps:

1. Edit the `hive-env.sh` file to set `SPLIT_HIVE_LOGS_INTO_FILES` property to `false`.

   ```
   export SPLIT_HIVE_LOGS_INTO_FILES="false"
   ```

   Note: To restore the default behavior from your previous Hive log configuration, set the `SPLIT_HIVE_LOGS_INTO_FILES` property to `true`, or comment out this property and restart Hive services.

2. Restart Hive services.

Splitting Hive Logs into HiveServer2 and Metastore logs by Process ID

Starting from the 1904 release, you can split Hive log files into HiveServer2 and Metestore log files by process ID.

To enable this feature, you must create a `hive-log4j2.properties` file, if one does not already exist, and then edit it:

1. If the `hive-log4j2.properties` file does not exist, create it from the template:

```
cp /opt/mapr/hive/hive-<version>/conf/
hive-log4j2.properties.template  /opt/mapr/hive/hive-<version>/conf/
hive-log4j2.properties
```

2. Edit the `hive-log4j2.properties` file to replace Daily Rolling File Appender (DRFA) with the PID appender:

```
#property.hive.root.logger = DRFA
property.hive.root.logger = PID
#appenders = console, DRFA
appenders = console, PID
```

3. Restart Hive services.

The resultant Hive log structure is as follows:

The HiveServer2 log is located at:

```
${HIVE_HOME}/log/<ADMIN_USER>/
<ADMIN_USER>-hiveserver2-<HOSTNAME>.log.<PID>@<HOSTNAME>
```

Where:

- `${HIVE_HOME}` is the home folder for Hive.
- `<ADMIN_USER>` is the administrator user of the cluster. Typically, `mapr`.
- `<HOSTNAME>` is the host where HiveServer2 log file is placed.
- `<PID>` is the process ID of HiveServer2.

The Metastore log is located at:

```
${HIVE_HOME}/log/<ADMIN_USER>/
<ADMIN_USER>-metastore-<HOSTNAME>.log.<PID>@<HOSTNAME>
```

Where:

- `${HIVE_HOME}` is the home folder for Hive.
- `<ADMIN_USER>` is the administrator user of the cluster. Typically, `mapr`.
- `<HOSTNAME>` is the host where Hive Metastore log file is placed.
- `<PID>` is the process ID of Hive Metastore.

Logging CLI session

After splitting logs for HiveServer2 and Hive Metastore, CLI log appears separately for each CLI session at `${HIVE_HOME}/logs/<USERNAME>`.

A log file is created for every launched CLI session:

```
${HIVE_HOME}/log/<USERNAME>/
<USERNAME>-hiveserver2-<HOSTNAME>.log.<PID>@<HOSTNAME>
```

Where `<PID>` is process identifier of the CLI session.

last-updated: Apr 11, 2022

Viewing Hive Audit Logs

Starting in EEP 6.3.4, you can view Hive audit logs for connected, disconnected, and total connected users.

To view audit logs, add the following property in the `hive-site.xml` file:

```
<property>
    <name>hive.enable.full.list.of.connected.users</name>
    <value>true</value>
</property>
```

By default, logs are updated every five seconds.

The following table describes the Hive Parameters used to manage the user audit logs:

Parameter	Default value	Description
hive.enable.full.list.of.connected.users	false	Enables the logging of the users currently connected to Hive when set to true. Use for debugging purposes only.
hive.full.list.of.connected.users.update.interval	5	Enables the log updates for currently connected Hive users in seconds. Must be used with the `hive.enable.full.list.of.connected.users` parameter. Use for debugging purposes only.

How to View Audit Logs

Enable the `hive.enable.full.list.of.connected.users` property in `hive-site.xml` file. You can view audit logs for connected, disconnected, and total connected users in HiveServer2 logs located in `${HIVE_HOME}/logs/mapr/mapr-hiveserver2-<hostname>.log` directory.

The following examples show you how the audit logs look in different scenarios:

Logs display for new user connection

Log entries for connected users provide the current session ID, username, IP address of the user, and the authentication type.

```
INFO [HiveServer2-Handler-Pool:
Thread-51] HiveSessionImpl.audit:
Connected:
sessionId=4c25b6d6-6e8e-4d56-83ba-52ea
271d0545 user=mapr ip=192.168.33.11
auth=MAPRSASL
```

Logs display for disconnected user

Log entries for disconnected users provide the current session ID, username, IP address of the user, and the authentication type.

```
INFO [HiveServer2-Handler-Pool:
Thread-51] HiveSessionImpl.audit:
Disconnected:
sessionId=4c25b6d6-6e8e-4d56-83ba-52ea
271d0545 user=mapr ip=192.168.33.11
auth=MAPRSASL
```

Logs display for total connected users

Log entries for total connected users start with a message -Start of connected users list, and provides the current session ID, username, IP address of the user, operation count, active time, idle time,

last-updated: Apr 11, 2022

authentication type, and end with a message- End of the connected user's list.

```
INFO [pool-4-thread-1]
SessionManager.audit: Start of the
connected users list

INFO [pool-4-thread-1]
SessionManager.audit:
sessionId=c6261d49-1a71-4404-8cad-9cac
11a28151 user=mapr ip=192.168.33.11
operationCount=0 activeTime(s)=268
IdleTime(s)=268, auth=MAPRSASL

INFO [pool-4-thread-1]
SessionManager.audit:
sessionId=36b4d8d4-f201-43da-90eb-cb68
3d343b80 user=mapr ip=192.168.33.11
operationCount=0 activeTime(s)=198
IdleTime(s)=197, auth=MAPRSASL

INFO [pool-4-thread-1]
SessionManager.audit:
sessionId=32b50c8a-28ca-46a5-bbcd-963c
9b22af7f user=mapruser1
ip=192.168.33.11 operationCount=0
activeTime(s)=4 IdleTime(s)=4,
auth=PAM

INFO [pool-4-thread-1]
SessionManager.audit: End of the
connected user's list
```

How to Audit a Hive Query

The audit log in HiveServer2 allows you to trace the activities of a Hive query. The log entries for a Hive query includes username, user's IP address, query ID, query type, and query string.

To audit a Hive query, run any Hive query and then see the HiveServer2 logs located in `${HIVE_HOME}/logs/mapr/mapr-hiveserver2-<hostname>.log` directory.

```
INFO [HiveServer2-Background-Pool: Thread-54]
Driver.audit: user=mapr ip=192.168.33.11
queryId=mapr_20210426155754_ace67f82-9a0c-4d0e-9ac5-c529b9798ec7 query
type=SHOWTABLES queryStr=show tables
```

HttpFS

HttpFS provides a service that enables you to submit HTTP REST calls to distributed file systems. You can use HttpFS to perform read and write operations on the filesystem.

This section includes the following topics:

Authentication on Secure Clusters for HttpFS

In secure clusters, HttpFS can use any of the following authentication methods:

- HttpFS authentication, such as native security (data-fabric SASL)
- Kerberos (for which additional configuration is required)

- Plain security using PAM, which is determined automatically

In a secure cluster, HttpFS runs a script to set the following properties by default. In a non-secure cluster, you must add the following properties manually to the `httpfs-site.xml` file:

```
httpfs.hadoop.authentication.type=multiauth
httpfs.authentication.type=multiauth
```

Configuring HttpFS

You can configure the following features to perform distributed filesystem operations securely through HttpFS.

The following topics describe how to configure various security mechanisms for HttpFS.

Kerberos Authentication for HttpFS

Complete the following steps to enable Kerberos security on nodes that run the HttpFS service:

Step 1: Set up a Kerberos Principal and keytab File

Each node running the HttpFS service must have a keytab file (`/opt/mapr/conf/mapr.keytab`) and these two principals:

- HTTP/<fully.qualified.domain.name>
- mapr/<fully.qualified.domain.name>

Note: For complete instructions on generating a Kerberos principal and keytab file, see Configuring Kerberos.

To check whether the keytab already exists, and if it contains the two necessary principals, run the `klist` command with the `-k` (keytab keys), `-e` (encryption type) and `-t` (timestamp) options:

```
$ klist -ket /opt/mapr/conf/mapr.keytab
```

The output from this command displays the following information:

- KVNO (key version number)
- Timestamp (the time the key was generated)
- Principal names
- Encryption types

If the keytab file does not exist, or does not contain both principals, generate them by following these steps:

1. Generate a Kerberos principal for the `mapr` user. The principal is of the form `mapr/<fully.qualified.domain.name>@<your-realm>.com`, where `<fully.qualified.domain.name>` is unique for each HttpFS node. In the following example, `perfnode153.perf.lab@dev-maprtech.com` is used for the `<fully.qualified.domain.name>@<your-realm>.com`.

   ```
   $ kadmin
   kadmin: addprinc -randkey mapr/perfnode153.perf.lab@dev-maprtech.com
   ```

2. Generate a Kerberos principal for `HTTP/<fully.qualified.domain.name>`. This is required for Kerberos authentication of the HttpFS server using HTTP SPNEGO.

```
$ kadmin
kadmin: addprinc -randkey HTTP/perfnode153.perf.lab@dev-maprtech.com
```

3. If the current node does not already have a keytab file created for another service, create one and name it `mapr.keytab`. Note that each node references the same keytab file (usually located at `/opt/mapr/conf/mapr.keytab`), and each keytab file can have multiple principals.

```
kadmin: ktadd -k /opt/mapr/conf/mapr.keytab mapr/perfnode153.perf.lab
```

4. Change the owner of the keytab file from the `root` user (the default) to the `mapr` user.

```
$ chown mapr:mapr /opt/mapr/conf/mapr.keytab
```

5. Set read-only permissions on the `mapr.keytab` file.

```
$ chmod 600 mapr:mapr /opt/mapr/conf/mapr.keytab
```

Step 2: Verify Credentials in the keytab File

To test that the credentials in the `mapr.keytab` file work, run the `klist` command with the three options:

- **`-k`** for keytab keys
- **`-e`** for encryption type
- **`-t`** for timestamp

Example

```
$ klist -ket /opt/mapr/conf/mapr.keytab
```

Verify that the output lists only one key version number (KVNO) for each principal name. If you see the same principal listed more than once with a different KVNO, this could indicate a problem. The latest version number is used, which means you might not be able to log in to the node and authenticate with your user credentials.

Here is sample output for a node that has the HttpFS and CLDB services installed.

```
Keytab name: FILE:/opt/mapr/conf/mapr.keytab
KVNO Timestamp Principal
---- -----------------
  -------------------------------------------------------
 2 07/18/14 18:50:07 mapr/perfnode153.perf.lab@dev-maprtech
  (aes256-cts-hmac-sha1-96)
 2 07/18/14 18:50:07 mapr/perfnode153.perf.lab@dev-maprtech (arcfour-hmac)
 2 07/18/14 18:50:08 mapr/perfnode153.perf.lab@dev-maprtech (des3-cbc-sha1)
 2 07/18/14 18:50:08 mapr/perfnode153.perf.lab@dev-maprtech (des-cbc-crc)
 2 07/18/14 18:50:26 HTTP/perfnode153.perf.lab@dev-maprtech
  (aes256-cts-hmac-sha1-96)
 2 07/18/14 18:50:26 HTTP/perfnode153.perf.lab@dev-maprtech (arcfour-hmac)
 2 07/18/14 18:50:26 HTTP/perfnode153.perf.lab@dev-maprtech (des3-cbc-sha1)
 2 07/18/14 18:50:26 HTTP/perfnode153.perf.lab@dev-maprtech (des-cbc-crc)
 6 07/18/14 18:50:56 mapr/my.cluster.com@dev-maprtech
  (aes256-cts-hmac-sha1-96)
```

```
 6 07/18/14 18:50:56 mapr/my.cluster.com@dev-maprtech (arcfour-hmac)
 6 07/18/14 18:50:56 mapr/my.cluster.com@dev-maprtech (des3-cbc-sha1)
 6 07/18/14 18:50:57 mapr/my.cluster.com@dev-maprtech (des-cbc-crc)
```

In the example, the following principals are listed for the `perfnode153.perf.lab` node:

- **`mapr/perfnode153.perf.lab@dev-maprtech`** for authenticating to the HttpFS service
- **`HTTP/perfnode153.perf.lab@dev-maprtech`** for communicating securely over HTTP
- **`mapr/my.cluster.com`** for authenticating to the CLDB service

Step 3: Modify the httpfs-site.xml File

A Kerberos-ready version of the `httpfs-site.xml` file called `httpfs-site.xml.kerberos` is provided in `/opt/mapr/httpfs/httpfs-<version>/etc/hadoop`. Edit this file and specify the Kerberos principal name for the nodes running HttpFS, restart the HttpFS server, and then test the set-up. Each step is explained here.

Note that the directory differs based on the httpfs version:

- In EEP 7.0.1 and earlier, the directory is `/opt/mapr/httpfs/httpfs-1.0/etc/hadoop/httpfs-site.xml`.
- In EEP 7.1.0 and later, the directory is `/opt/mapr/httpfs/httpfs-1.1.0/etc/hadoop/httpfs-site.xml`.

To set up the `httpfs-site.xml` file for each node running the HttpFS service, follow these steps:

1. Assign a new name to the existing `httpfs-site.xml` file (to preserve the original version when the file gets overwritten in step 2).
   - In 7.0.1 and earlier, run:

     ```
     cp /opt/mapr/httpfs/httpfs-1.0/etc/hadoop httpfs-site.xml
     httpfs-site.xml.original
     ```

   - In 7.1.0 and later, run:

     ```
     cp /opt/mapr/httpfs/httpfs-1.1.0/etc/hadoop httpfs-site.xml
     httpfs-site.xml.original
     ```

2. Copy the kerberos version (`httpfs-site.xml.kerberos`) to the existing `httpfs-site.xml` file.
   - In 7.0.1 and earlier, run:

     ```
     cp /opt/mapr/httpfs/httpfs-1.0/etc/hadoop httpfs-site.xml.kerberos
     httpfs-site.xml
     ```

   - In 7.1.0 and later, run:

     ```
     cp /opt/mapr/httpfs/httpfs-1.1.0/etc/hadoop httpfs-site.xml.kerberos
     httpfs-site.xml
     ```

3. Edit the `httpfs-site.xml` file and insert the principal name as shown, substituting your fully qualified domain name and realm for `perfnode153.perf.lab@mapr.com`.

```
<property>
  <name>
    httpfs.authentication.kerberos.principal
  </name>
  <value>
    HTTP/perfnode153.perf.lab@mapr.com
  </value>
</property>
```

4. Restart the HttpFS server so the changes will take effect.

```
maprcli node services -name httpfs -action restart -nodes <node_name>
```

5. Test that security is in place by entering the following command to create a file in the MapR filesystem. The command will fail if security is not set up correctly.

```
curl --negotiate -u : -b ~/cookiejar.txt -c ~/cookiejar.txt -i -X PUT
"http://perfnode153.perf.lab:14000/webhdfs/v1/user/mapr/some_file?
op=MKDIRS"
```

(Optional) Configure the HTTP Header Size

For EEP 7.0.1 and earlier, the `maxHttpHeaderSize` parameter defines the maximum size of the request and response HTTP header, specified in bytes. If it is not specified, this parameter defaults to 8192 (8KB).

When Kerberos security is enabled, you may need to increase this value in the `server.xml` file:

```
/opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/conf/server.xml
```

Example

```
<Connector port="${httpfs.http.port}" maxHttpHeaderSize="32000"
protocol="HTTP/1.1" connectionTimeout="20000" redirectPort="8443"/>
```

If you do not increase this value, you may encounter errors of the following form:

```
HTTP/1.1 400 Bad Request
```

Note: After making this configuration change, restart the HttpFS server.

PAM Authentication for HttpFS

Complete the following steps to enable PAM authentication for HttpFS.

1. Add the `httpfs.hadoop.authentication.type` and `httpfs.authentication.type` properties to the `/opt/mapr/httpfs/httpfs-version/etc/hadoop/`**`httpfs-site.xml`** file, as shown. Note that the directory differs based on the httpfs version:
   - In EEP 7.0.1 and earlier, the directory is `/opt/mapr/httpfs/httpfs-1.0/etc/hadoop/httpfs-site.xml`.

- In EEP 7.1.0 and later, the directory is `/opt/mapr/httpfs/httpfs-1.1.0/etc/hadoop/httpfs-site.xml`.

```
<property>
        <name>httpfs.hadoop.authentication.type</name>
        <value>multiauth</value>
</property>

<property>
        <name>httpfs.authentication.type</name>
        <value>multiauth</value>
</property>
```

Note: On secure clusters, the **`multiauth`** authentication is enabled by default.

2. Restart the HttpFS service.

```
sudo -u mapr /opt/mapr/httpfs/httpfs-1.0/sbin/httpfs.sh stop
sudo -u mapr /opt/mapr/httpfs/httpfs-1.0/sbin/httpfs.sh start
```

3. After restarting the service, run cURL with the PUT operation, as shown in this example:

Note: If HttpFS is configured with plain authentication through PAM, the cURL request must contain a username and password.

```
curl -X PUT "https://mapr:mapr@node1:14000/webhdfs/v1/tmp/example?
op=mkdirs"
```

SSL Security for HttpFS

Use this procedure with MapR 6.0 and earlier to explicitly enable security on HttpFS. Although SSL for HttpFS is enabled by default on secure clusters in MapR 6.0.1, you must manually configure it in MapR 6.0 and earlier, including whether you want it to use certificate-based authentication or not.

You also need to enable SSL if custom security is enabled.

Enabling SSL Security for HttpFS
Enable SSL security for HttpFS using a `ssl_keystore` and `ssl_truststore`. These are generated automatically for a secure cluster in `/opt/mapr/conf/`. When using SSL on insecure clusters, you must manually generate a `keystore` and `truststore`.
Configuring Certificate-Based Authentication for HttpFS
To configure certificate-based authentication for HttpFS you need to make changes to the `server.xml` and `web.xml` file and restart the HttpFS server. To use this method, each client requires a client certificate issued by trusted CA.

1. To enable certificate-based authentication, perform the following steps:

a) Verify that the `clientAuth` attribute is set to "**true**" and set properties related to `keystore` and `truststore` in **`server.xml`** (`/opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/conf/server.xml`).

   For example:

```
<Connector port="${httpfs.http.port}" SSLEnabled="true"
maxThreads="150" scheme="https" secure="true" clientAuth="true"
sslProtocol="TLS"
keystoreFile="/opt/mapr/conf/ssl_keystore"
keystorePass="<ssl-keystore-password>"
truststoreFile="/opt/mapr/conf/ssl_truststore"
truststorePass="<ssl-keystore-password>"/>
```

b) In **`web.xml`** (`/opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/webapps/webhdfs/WEB-INF/`**`web.xml`**), un-comment the following section:

```
<security-constraint>
  <web-resource-collection>
    <web-resource-name>Protected Context</web-resource-name>
    <url-pattern>/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>sample</role-name>
  </auth-constraint>
  <user-data-constraint>
    <transport-guarantee>CONFIDENTIAL</transport-guarantee>
  </user-data-constraint>
</security-constraint>

<security-role>
  <role-name>sample</role-name>
</security-role>

<login-config>
  <auth-method>CLIENT-CERT</auth-method>
</login-config>
```

c) Verify that **tomcat-users.xml** (/opt/mapr/httpfs/httpfs-1.0/share/hadoop/httpfs/tomcat/conf/**tomcat-users.xml**) contains the roles and users in the certificates.

```
<tomcat-users>
  <role rolename="sample"/>
  <user name="CN=<hostname>" password="null" roles="sample" />
</tomcat-users>
```

Note: The name value should include information from your certificate. For example, <tomcat-users> <role rolename="**sample**"/> <user name="**CN=www.mapr.com, OU=mapr, O=mapr, L=San Jose, ST=San Jose, C=CA**" password="**null**" roles="**sample**" /> </tomcat-users> You can run the following command to view the contents of the certificate file: openssl x509 -text -in /opt/mapr/hue/hue-<version>/**cert.pem**

```
<tomcat-users>
  <role rolename="sample"/>
  <user name="CN=www.mapr.com, OU=mapr, O=mapr, L=San Jose,
       ST=San Jose, C=CA" password="null" roles="sample" />
</tomcat-users>
```

2. Restart the HttpFS server using the following command:

```
maprcli node services -name httpfs -action restart -nodes <space
delimited list of nodes>
```

Verifying SSL Security for HttpFS
You need to run curl commands to verify that HTTPS is enabled for HttpFS.

Run one of the following curl commands to check that HTTPS is enabled. These commands fetch the file **some_file.txt** from the file system under /user/**mapr** and attempt to open it securely over HTTPS.

- To check if HTTPS is enabled, run the following command (which differs for non-secure and secure clusters):
  - For non-secure clusters:

```
curl "http://localhost:14000/webhdfs/v1/user/mapr/some_file.txt?
op=open&user.name=mapr"
```

  - For secure clusters:

```
curl -u <user_name> -k
 "https://localhost:14000/webhdfs/v1/user/mapr/some_file.txt?op=open"
```

- If you configured Hue to use SSL encryption with certificate-based authentication for communication with HttpFS, run the following command (which differs for non-secure and secure clusters):
  - For non-secure clusters:

```
curl
  --cert /opt/mapr/hue/hue-<version>/cert.pem
  --key /opt/mapr/hue/hue-<version>/hue_private_keystore.pem
  "http://localhost:14000/webhdfs/v1/user/mapr/some_file.txt?
op=open&user.name=mapr"
```

- For secure clusters:

```
curl -u <user_name> -k
 --cert /opt/mapr/hue/hue-<version>/cert.pem
 --key /opt/mapr/hue/hue-<version>/hue_private_keystore.pem
 "https://localhost:14000/webhdfs/v1/user/mapr/some_file.txt?op=open"
```

User Impersonation for HttpFS

You can set up proxy user functionality if you want HttpFS to impersonate a user from a set of hosts, or to impersonate a user that belongs to a set of groups. When you configure proxy user functionality, the proxy user can perform “doAs” operations. Add configuration properties to the `httpfs-site.xml` and `core-site.xml` files to configure proxy user functionality.

Complete the following steps to configure user impersonation for HttpFS:

1. Add the following configuration properties to the `httpfs-site-site.xml` file:
   - `httpfs.proxyuser.#USER#.hosts`
   - `httpfs.proxyuser.#USER#.groups`
2. Replace #USER# with the username of the proxy that can perform “doAs” operations. For the host property, you can add a list of host names as the value. For the group property, you can add a list of groups as the value. Alternatively, you can add a wildcard character (*) as the value for host and group properties. To add multiple users, copy the property and replace #USER# with the proxy user name.

   Host Example

```
<property>
        <name>httpfs.proxyuser.mapr.hosts</name>
        <value>*</value>
</property>
```

   Group Example

```
<property>
        <name>httpfs.proxyuser.mapr.groups</name>
        <value>*</value>
</property>
```

To use impersonation, issue a cURL command with the `doas=<impersonated_user's name>` parameter.

Example 1

Where `user.name` is `mapr` and `doas` (or the impersonated user's name) is **sampleusername**.

```
curl -i -X PUT -T one
"http://<node_name>:14000/webhdfs/v1/user/mapr/TEST/one
?op=CREATE&user.name=mapr&doas=sampleusername&data=true"
-H "Content-Type:application/octet-stream"
```

Example 2

For any user (and password) other than the `mapr` user (for example, **test_user1**), set the `hadoop.proxyuser.<user_name>.hosts</name>` property in the `/opt/mapr/httpfs/httpfs-<version>/etc/hadoop/httpfs-site.xml` file, as shown. Note that the directory differs based on the version of httpfs, for example:

last-updated: Apr 11, 2022

- In EEP 7.0.1 and earlier, the directory is `/opt/mapr/httpfs/httpfs-1.0/etc/hadoop/httpfs-site.xml`.
- In EEP 7.1.0 and later, the directory is `/opt/mapr/httpfs/httpfs-1.1.0/etc/hadoop/httpfs-site.xml`.

```
<property>
          <name>hadoop.proxyuser.<test_user1>.hosts</name>
          <value>*</value>
</property>
```

Run cURL.

Where `trueuser.name` is **test_user1** and `doas` (or the impersonated user's name) is **test_user2**.

```
curl -u fred -i -X PUT -T /etc/hosts --header "Content-Type:application/
octet-stream"
"http://<node_name>:14000/webhdfs/v1/<path_to_test_file>
?op=CREATE&doas=<test_user2>&data=true&user.name=<test_user1>"
```

Finishing HTTPFS Configuration Changes

After making configuration changes for the `mapr-httpfs` package provided by EEP 4.0.0 or later, run the `configure.sh` script on all nodes where the `mapr-httpfs` package was installed:

```
sudo bash /opt/mapr/server/configure.sh -R
```

Troubleshooting HttpFS

To debug authentication issues, follow these steps:

1. Edit the `log4j` properties file located at `/opt/mapr/httpfs/httpfs-1.0/etc/hadoop/httpfs-log4j.properties` and insert the following lines to activate debug capabilities:

   ```
   log4j.logger.org.apache.hadoop.fs.http.server=DEBUG, httpfs
   log4j.logger.org.apache.hadoop.lib=DEBUG, httpfs
   log4j.logger.org.apache.hadoop.security.authentication.server=DEBUG,
   httpfs
   ```

2. Search the logs located at `/opt/mapr/httpfs/httpfs-1.0/logs` for the words *ERROR* or *Exception*.

Hue

Hue is the open source UI that interacts with Apache Hadoop and its ecosystem components, such as Hive, Pig, and Oozie. It is also a framework for creating interactive Web applications.

For information about Hue versions, see the Ecosystem Support Matrix.

Hue is supported on the following browsers:

Windows	Linux	Mac
Chrome	Chrome	Chrome

Windows	Linux	Mac
Firefox 3.6+	Firefox 3.6+	Firefox 3.6+
Safari 5+		Safari 5+
Internet Explorer 8+		

Hue Feature Support

The following table lists supported and unsupported Hue functionality:

Supported	Not Supported
Query editors • Hive (for performing queries on Apache Hive) • Impala (for submitting interactive SQL and HiveQL queries) • DB Query (for viewing data in MySQL, PostgreSQL, Oracle and Sqlite) • Pig (for submitting Pig scripts) • Job Designer (for creating and submitting MapReduce/Streaming/Java jobs) • Spark (beta feature for submitting Spark jobs for hue-3.9.0/3.10.0) • Drill (for performing queries on Apache Drill through JDBC) in hue-3.12 Data browsers • Metastore Tables (for managing databases, tables, and partitions of the Hive metastore) • HBase browser (for creating, editing, and searching tables) • Sqoop Transfer (for transferring bulk data between Hadoop and various types of structured datastores) Workflows • Oozie (for creating and running workflow and coordinator jobs) Hue 4.X supports ADLS browser for accessing files and directories in Azure Data Lake Store. S3 Browser (for accessing files and directories in Amazon S3) File Browser (for accessing files and directories in MapR File System) • Job Browser (for accessing MapReduce applications) • User Admin (for adding, deleting, and managing Hue users and groups)	Hue integration with the following components is not supported: • Sentry 1.6, 1.7 on a secure cluster that uses MapR-SASL authentication. **Note:** Sentry 1.6, 1.7 on a secure cluster that uses Kerberos authentication is supported. • Solr Search • Zookeeper

Configure Hue

After you install Hue, perform the following configuration steps:

last-updated: Apr 11, 2022

1. Complete the general configuration steps. This includes integrating Hue with ResourceManager and HttpFS.
2. Perform the steps to integrate each additional component that you want to use with Hue.
   - Hive
   - MapR Database
   - Impala
   - Oozie
   - Spark
   - Sqoop2

You may also want to:

- Configure Security
- Configure DB Query
- Configure Hue Interface Authentication

Note: The `hue.ini` file is the main configuration file for running Hue on a cluster. This file is located at `/opt/mapr/hue/hue-<version>/desktop/conf/hue.ini`. When you update the value of a property in the `hue.ini`, remove any hashes (##) that appear directly before the property name. You must also restart Hue for these changes to take effect.

Configure General Hue Settings

The following topics provide instructions for configuring general Hue settings:

Enable User Impersonation for Hue

To enable Hue to submit requests on behalf of any other user, complete the following steps:

1. Verify or configure the following lines to the `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/core-site.xml` file for all nodes running ResourceManager:

```
<property>
  <name>hadoop.proxyuser.<default_user>.hosts</name>
  <value>*</value>
</property>

<property>
  <name>hadoop.proxyuser.<default_user>.groups</name>
  <value>*</value>
</property>
```

2. To enable the Hue file browser to view files in the MapR filesystem, add the following proxy user settings in the configuration block of the `httpfs-site.xml`:

```
<!-- Hue HttpFS proxy user setting -->
<configuration>
  <property>
    <name>httpfs.proxyuser.<default_user>.hosts</name>
    <value>*</value>
  </property>

  <property>
    <name>httpfs.proxyuser.<default_user>.groups</name>
    <value>*</value>
  </property>
</configuration>
```

3. Perform any additional Hue configurations and then restart Hue so that the changes will take effect. See Starting the Hue Webserver.

   In most cases, mapr is the <default_user>. The <default_user> you specify must also be the default_user that is configured in the [desktop] section of the `hue.ini`.

   Note: Based on the ecosystem components that you want to use, addition configuration may be required.

Disable an Application in the Hue Interface (optional)

If you want to disable an application (such as Impala), follow these steps:

1. In the `[desktop]` section of the `hue.ini` file, uncomment the `# app_blacklist=` statement and insert the name of the app you want to disable (impala in this example).

   Note: Do not remove `search` from the app_blacklist. The Hue UI will not work if the search application is enabled.

```
# Comma-separated list of apps not to load at server startup.
# Note that rdbms is the name used for dbquery.
app_blacklist=spark,zookeeper,search,impala,sqoop,rdbms
```

   Note: After removing an application from `app_blacklist`, you must update the Hue internal database to create the tables required for the application that was enabled:

```
sudo /opt/mapr/server/configure.sh -R
```

2. Once all changes are made, restart Hue so the changes will take effect.

   Note: You can re-enable a blacklisted application at any time, and then restart Hue.

```
maprcli node services -name hue -action restart -nodes <ip_address>
```

Change the File Size Restriction for the File Browser (optional)

The Hue File Browser will not open files that are 1.0 GB or greater. Starting with Hue 4.2, file size limitation equals 1.0 GB with no way to modify it.
Prevent Hue from Creating User Home Directories
Describes how to disable the automatic creation of user home directories.

By default, Hue creates a directory in the filesystem for a user when the user logs in to the Hue service.

last-updated: Apr 11, 2022

For example, if a /user volume is configured in the filesystem, Hue creates a /user/<username> directory in the volume each time a user logs in to the Hue service. If a quota is not placed on that /user volume, a user could potentially place an unlimited amount of data in the volume.

If you do not want Hue to create a home directory for each user that logs in to the Hue service, disable the `ensure_home_directory` option in the `[desktop] [[auth]]` section of the `hue.ini` file, as shown:

```
[desktop]
  [[auth]]
    ensure_home_directory=false
```

Restart the Hue service for the setting to take effect.

Note: This functionality is available by default starting in EEP 7.1.0. Previous versions of EEP can obtain this functionality through a patch. See Applying a Patch on page 437.

Configure Hue Interface Authentication

You can configure the following user authentication methods for the Hue interface:

Authentication Method	Description
Hue User Administration	Use the Hue interface to create and manage user accounts for each Hue user.
LDAP	Import LDAP users into Hue and then use LDAP to authenticate users with their LDAP credentials. For more information, see Configure Hue with LDAP.
PAM	Use multiple PAM modules to authenticate users. PAM authentication is configured by default. When you use this method, you cannot edit users in the Hue interface.

Using a Non-Default Authentication Method

The default authentication method is PAM.

To edit the authentication method used for the Hue interface, complete the following steps:

1. Set the `backend` property equal to your selected authentication method. For example, to use Hue's user authentication, select `desktop.auth.backend.AllowFirstUserDjangoBackend.`
2. If you choose not to use PAM, comment the `pam_service property.`

Example hue.ini configued to use PAM Authentication

```
[[auth]]
# Authentication backend. Common settings are:
# - django.contrib.auth.backends.ModelBackend (entirely Django backend)
# - desktop.auth.backend.AllowAllBackend (allows everyone)
# - desktop.auth.backend.AllowFirstUserDjangoBackend
# (Default. Relies on Django and user manager, after the first login)
# - desktop.auth.backend.LdapBackend
# - desktop.auth.backend.PamBackend - WARNING: existing users in Hue may be
unaccessible if they not exist in OS
# - desktop.auth.backend.SpnegoDjangoBackend
# - desktop.auth.backend.RemoteUserDjangoBackend
# - libsaml.backend.SAML2Backend
# - libopenid.backend.OpenIDBackend
# - liboauth.backend.OAuthBackend
# (Support Twitter, Facebook, Google+ and Linkedin
backend=desktop.auth.backend.PamBackend
```


last-updated: Apr 11, 2022

```
# The service to use when querying PAM.
pam_service=sudo sshd login
```

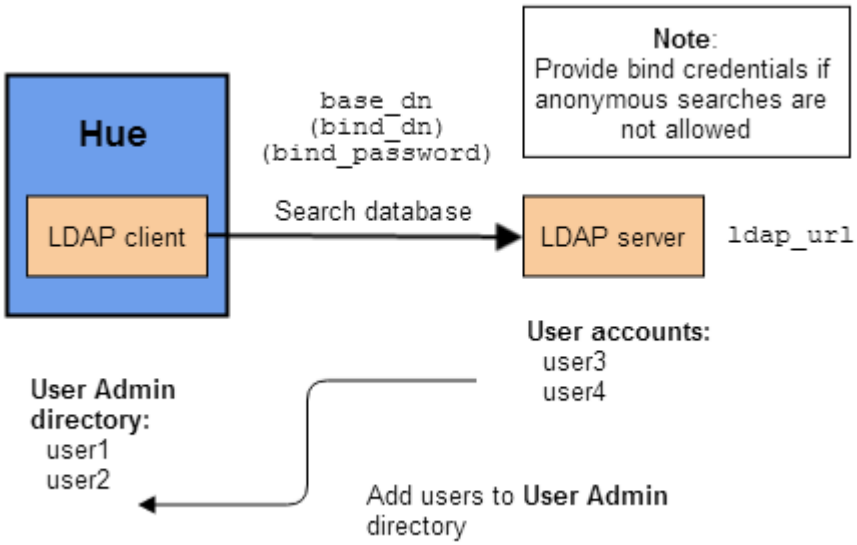
Configure Hue with LDAP

If you use LDAP to authenticate users, you can retrieve user account information from your LDAP database and import it directly into Hue's `User Admin` directory. This way, you do not have to use the Hue interface to create user accounts for each Hue user individually.

Once you import users, you can also use LDAP with Hue to authenticate users with their LDAP credentials. Each of these tasks is explained in the following sections:

Setting up Users from an LDAP Database

This diagram shows how the LDAP client embedded in Hue searches the LDAP server's database for user names, and then adds them to the `User Admin` directory for Hue.



The following table shows the parameters you need to set in the `ldap` section of the `hue.ini` file so you can import users.

Warning: The `hue.ini` file is located at `/opt/mapr/hue/hue-<version>/desktop/conf/.`

Parameter	Description	Comments
`ldap_url`	The URL of your LDAP server.	
`base_dn`	Top of the search tree, which defines the search scope.	
`bind_dn`	Distinguished name (DN) of the user to bind as.	Can be omitted for anonymous searches.
`bind_password`	Password of the bind user.	Can be omitted for anonymous searches.
`user_filter`	Limits the scope of the search by applying a filter.	This parameter is optional.
`user_name_attr`	The attribute used for username in the LDAP schema.	Examples: `cn` (for common name) or `uid` (for user ID).

last-updated: Apr 11, 2022

To set up Hue users by importing information from an LDAP database:

1. Establish communication with the LDAP server by setting the `ldap_url` parameter in the `ldap` section of the `hue.ini` file. Uncomment the line and change the value from the default (`ldap://localhost`) to the URL for your LDAP server.

```
# URL of the LDAP server
    ##ldap_url=ldap://localhost
```

2. Provide the `base_dn` information to define the search scope. Uncomment the line where `base_dn` is defined and replace with your `base_dn`.

```
# The search base for finding users and groups
    ## base_dn="DC=mycompany,DC=com"
```

3. If your LDAP server does not support anonymous searches, you need to provide the `bind_dn` and `bind_password`. Uncomment the lines with these parameters and change the values to your `bind_dn` and your `bind_password`.

```
# Distinguished name of the user to bind as -- not necessary if the LDAP
server
    # supports anonymous searches
    ## bind_dn="CN=ServiceAccount,DC=mycompany,DC=com"

    # Password of the bind user -- not necessary if the LDAP server
supports
    # anonymous searches
    ## bind_password=
```

4. If you want to narrow the scope of the directory search, specify a `user_filter` in the `users` section under the `ldap` section of the `hue.ini` file. This is optional.

```
[[[users]]]

      # Base filter for searching for users
      ## user_filter="objectclass=*"
```

5. Set the `user_name_attr` parameter in the `users` section under the `ldap` section of the `hue.ini` file. If your LDAP directory schema does *not* use the attribute `sAMAccountName` for the username, uncomment the line and change the value of the `user_name_attr` to the attribute you use. For example, if the directory schema uses the `uid` attribute, change the value of the parameter as shown:

```
[[[users]]]

      # The username attribute in the LDAP schema
      ## user_name_attr=sAMAccountName
```

```
user_name_attr=uid
```

6. Restart httpfs so ldap settings will take effect.
7. Restart Hue once all configuration changes have been made so the changes will take effect.

Authenticating Hue Users with LDAP Credentials

last-updated: Apr 11, 2022

This section explains how to edit the `ldap` section of the `hue.ini` file to enable Hue user authentication with LDAP credentials. These instructions assume you have completed the steps in Setting up Users from an LDAP Database.

Warning:

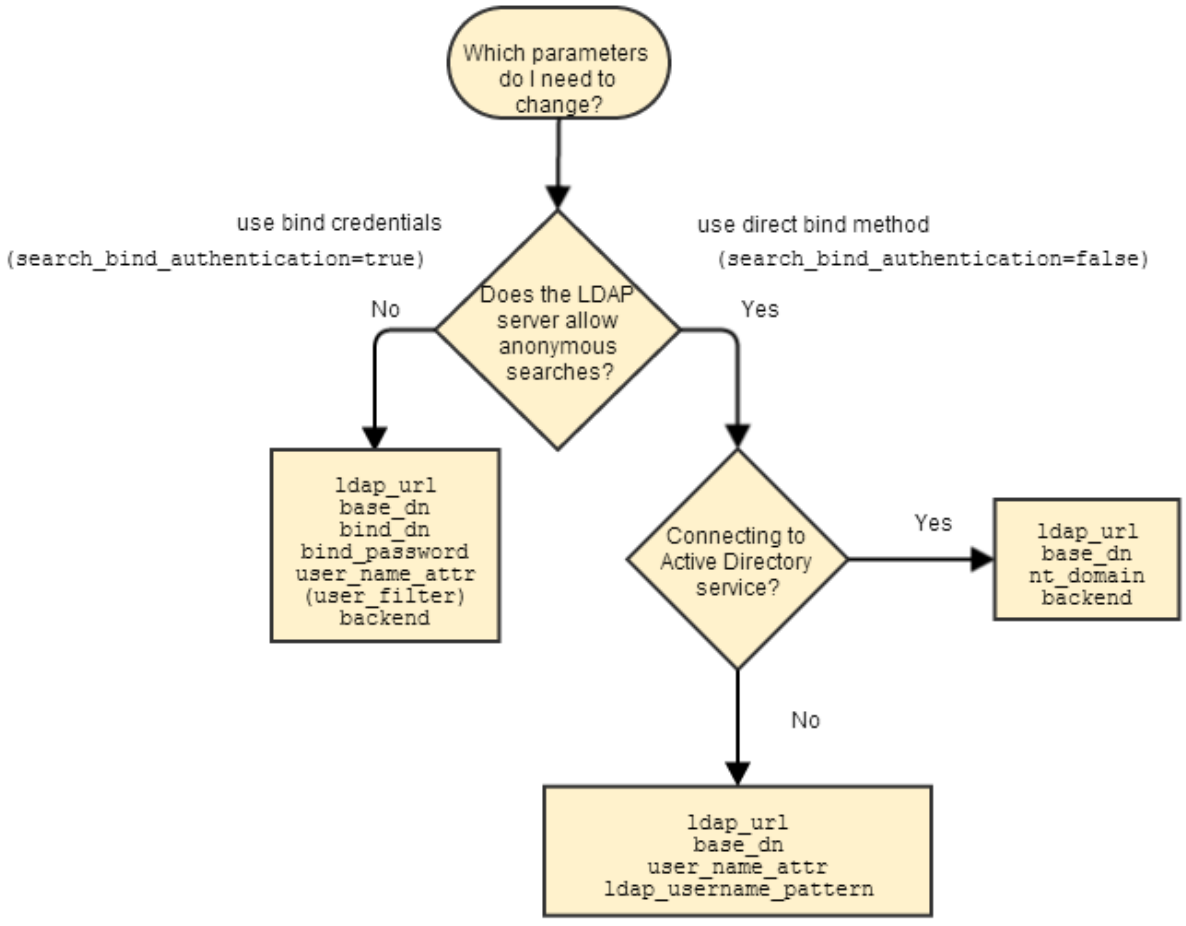
If you switch to authentication through LDAP credentials, the Hue User Admin users will lose superuser privileges unless you take one of the following actions:

- Import one or more superuser accounts from LDAP and assign them superuser permission.
- If you have already enabled the LDAP authentication back end, log into Hue using the LDAP back end, which will create an LDAP user. Next, disable the LDAP authentication back end and use User Admin to give the superuser permission to the new LDAP user.

Before you edit the parameters in the `hue.ini` file, determine whether your LDAP server allows anonymous searches.

- If anonymous searches *are* allowed, use the **direct bind** method.
- If anonymous searches *are not* allowed, use bind credentials (also known as **search and bind**).

The following flow chart shows which parameters you must specify for each of these authentication methods:



These are the parameters you need to set in the `ldap` section of the `hue.ini` file so you can authenticate Hue users with LDAP credentials:

last-updated: Apr 11, 2022

Parameter	Description	Comments
`search_bind_authentication`	Determines which authentication method to use: search and bind, or direct bind.	When set to *true*, Hue performs an LDAP search using `bind_dn` and `bind_password` as provided in `hue.ini`. The search can be further limited by the search filter `user_filter`. When set to *false*, Hue performs a direct bind to LDAP using the credentials provided from one of these sources: • the UPN, formed by concatenating `<shortname>` (the user name provided on the Hue login page) and `nt_domain` (if `nt_domain` is specified) • the `ldap_username_pattern` (if `nt_domain` is not specified)
`nt_domain`	The NT domain to connect to. This parameter is *only* used with Active Directory.	Used with the *direct bind* method of authentication. If `nt_domain` is specified, then `ldap_username_pattern` is ignored.
`ldap_username_pattern`	Used to connect to directory services other than Active Directory.	Used with the *direct bind* method of authentication. Usually takes the form `"cn=<username>,dc=example,dc=com"`
`backend`	The backend to use for authenticating users.	Needs to be set to `desktop.auth.backend.LdapBackend` for Hue authentication.

Using Bind Credentials (Search and Bind)

To use the search and bind method for LDAP authentication, edit these parameters in the `ldap` section of the `hue.ini` file:

1. Set `search_bind_authentication=true`.
2. In the `Authentication backend` section, add the following line after the `##backend=` statement: Hue searches `base_dn` for an entry with `user_name_attr` that contains the user name provided on the Hue login page.

   ```
   backend=desktop.auth.backend.LdapBackend
   ```

3. Restart Hue once all configuration changes have been made so the changes will take effect.

Using Direct Bind

To use the direct bind method for LDAP authentication, edit these parameters in the `ldap` section of the `hue.ini` file:

1. Set `search_bind_authentication=false`.

2. If you are using the Active Directory directory service, uncomment the line with the `nt_domain` parameter. Change the value from `nt_domain=mycompany.com` to the NT domain you want to connect to.
3. If you are using any other directory service, uncomment the line with `ldap_username_pattern` and specify the format, such as the one shown here: Note that `<username>` will be replaced by the information provided on the Hue login page.

   ```
   ldap_username_pattern="cn=<username>,dc=example,dc=com"
   ```

4. Restart Hue once all configuration changes have been made so the changes will take effect.

Configure Hue with DB Query

Hue's DB Query application provides a way to view data in these four database formats:

- SQLite
- MySQL
- PostgreSQL
- Oracle

To configure Hue to work with DB Query, edit the `[librdbms]` section of the `hue.ini` file and change the parameters related to the particular database you are using.

Warning: To disable the DB Query application, add `rdbms` to the blacklist as shown in Disabling an Application.

Default librdbms Settings in hue.ini

Database settings are contained in the `[librdbms]` section of the `hue.ini` file. Two databases (SQLite and MySQL) are already configured in their own sections under the `[[databases]]` heading. Note that PostgreSQL and Oracle work similarly to MySQL, so the `[[[mysql]]]` section can be treated as a template that can be edited for these two databases.

The DB Query application supports four types of databases, which are configured in the `[[databases]]` section.

The default version of the `librdbms` section of the `hue.ini` file looks like this:

```
###########################################################################
# Settings for the RDBMS application
###########################################################################
[librdbms]
 # The RDBMS app can have any number of databases configured in the
databases
 # section. A database is known by its section name
 # (i.e. sqlite, mysql, psql, and oracle in the list below).
 [[databases]]
   # sqlite configuration.
   ## [[[sqlite]]]
     # Name to show in the UI.
     ## nice_name=SQLite
     # For SQLite, name defines the path to the database.
     ## name=/opt/mapr/hue/hue-<version>/desktop/desktop.db
     # Database backend to use.
     ## engine=sqlite
   # mysql, oracle, or postgresql configuration.
```


last-updated: Apr 11, 2022

```
   ## [[[mysql]]]
     # Name to show in the UI.
     ## nice_name="My SQL DB"
     # For MySQL and PostgreSQL, name is the name of the database.
     # For Oracle, Name is instance of the Oracle server. For express
edition
     # this is 'xe' by default.
     ## name=mysqldb
     # Database backend to use. This can be:
     # 1. mysql
     # 2. postgresql
     # 3. oracle
     ## engine=mysql

     # IP or hostname of the database to connect to.
     ## host=localhost

     # Port the database server is listening to. Defaults are:
     # 1. MySQL: 3306
     # 2. PostgreSQL: 5432
     # 3. Oracle Express Edition: 1521
     ## port=3306

     # Username to authenticate with when connecting to the database.
     ## user=example

     # Password matching the username to authenticate with when
     # connecting to the database.
     ## password=example
```

Using DB Query with SQLite

To configure Hue to work with SQLite, edit the `sqlite` section of the `hue.ini` file and replace default values as necessary.

Default sqlite Section

The default version of the `sqlite` section is shown here.

```
# sqlite configuration
## [[[sqlite]]]
  # Name to show in the UI.
  ## nice_name=SQLite

  # For SQLite, name defines the path to the database.
  ## name=/tmp/sqlite.db

  # Database backend to use.
  ## engine=sqlite
```

Modifying the sqlite Section

To configure Hue's DB Query application to work with SQLite, edit the `sqlite` section of the `hue.ini` file as follows:

1. Remove the comment characters (##) in front of the [[[sqlite]]] statement.
2. Replace the default values for the following parameters as necessary.

last-updated: Apr 11, 2022

Parameter	Default Value	New Value
`nice_name`	`SQLite`	The name that you want to appear in the UI
`name`	`/tmp/sqlite.db`	The path to the SQLite database

3. Remove the comment characters (##) for the parameter values you changed. Otherwise, the default values are used.

Using DB Query with MySQL, PostgreSQL, or Oracle

Instructions for MySQL, PostgreSQL, and Oracle are similar. These instructions show how to configure Hue to work with MySQL. You can adapt the instructions to configure Hue to work with PostgreSQL or Oracle.

The next section shows the default `mysql` section of the `hue.ini` file for illustration. For instructions on configuring Hue so the DB Query application can display data from a MySQL database, see Modifying the mysql Section.

Default mysql Section

The default version of the `mysql` section is shown here.

```
# mysql, oracle, or postgresql configuration.
## [[[mysql]]]
  # Name to show in the UI.
  ## nice_name="My SQL DB"
  # For MySQL and PostgreSQL, name is the name of the database.
  # For Oracle, Name is the instance of the Oracle server. For    express
edition
  # this is 'xe' by default.
  ## name=mysqldb
  # Database backend to use. This can be:
  # 1. mysql
  # 2. postgresql
  # 3. oracle
  ## engine=mysql

  # IP or hostname of the database to connect to.
  ## host=localhost

  # Port the database server is listening to. Defaults are:
  # 1. MySQL: 3306
  # 2. PostgreSQL: 5432
  # 3. Oracle Express Edition: 1521
  ## port=3306

  # Username to authenticate with when connecting to the database.
  ## user=example

  # Password matching the username to authenticate with when
  # connecting to the database.
  ## password=example
```

Modifying the mysql Section

To configure Hue so the DB Query application can work with MySQL, edit the `mysql` section of the `hue.ini` file as follows:

1. Replace the default values for the following parameters.

last-updated: Apr 11, 2022

Parameter	Default Value	New Value
`nice_name`	`"My SQL DB"`	The name that you want to appear in the UI
`name`	`mysqldb`	The name of the database
`host`	`localhost`	IP address or hostname of the database to connect to
`user`	`example`	The username to use for authentication when connecting to the database
`password`	`example`	The password to use with the username (above) for authentication when connecting to the database

2. Remove the comment characters (##) for the parameter values you changed. Otherwise, the default values are used.

Activating DB Query with mySQL

To activate DBQuery with MySQL, follow these steps:

1. Move the mysql python library to a back-up location and regenerate it to match the version on the deployment machine. mv /opt/mapr/hue/hue-<VERSION>/python2.6/site-packages/MySQL_python-1.2.3c1-py2.6-linux-x86_64.egg MySQL_python-1.2.3c1-py2.6-linux-x86_64.egg.bk
2. Regenerate the mysql python library by running the following command: /opt/mapr/hue/hue-<VERSION>/build/env/bin/pip install mysql-python

Modifying the mysql Section for PostgreSQL

If you are working with a PostgreSQL database, edit the `mysql` section of the `hue.ini` file and rename it `postgresql`. Next, replace the MySQL default values as shown:

1. Replace the following MySQL default values with PostgreSQL values.

Parameter	Default Value	New Value
`nice_name`	`"My SQL DB"`	The name that you want to appear in the UI
`name`	`mysqldb`	The name of the database
`engine`	`mysql`	postgresql (the database backend)
`port`	`3306`	5432

2. Replace the following default values with your host machine, username, and password.

Parameter	Default Value	New Value
`host`	`localhost`	IP address or hostname of the database to connect to
`user`	`example`	The username to use for authentication when connecting to the database

last-updated: Apr 11, 2022

Parameter	Default Value	New Value
password	example	The password to use with the username (above) for authentication when connecting to the database

3. Remove the comment characters (##) at the beginning of each line where you changed parameter values. Otherwise, the MySQL default values are used.

Modifying the mysql Section for Oracle

If you are working with an Oracle database, edit the `mysql` section of the `hue.ini` file and rename it `oracle`. Next, replace the MySQL default values as shown:

1. Replace the following MySQL default values with Oracle values.

Parameter	Default Value	New Value
nice_name	"My SQL DB"	The name that you want to appear in the UI
name	mysqldb	The instance of the Oracle server (for express edition, this is xe by default)
engine	mysql	oracle (the database backend)
port	3306	1521 (for Oracle Express Edition)

2. Replace the following default values with your host machine, username, and password.

Parameter	Default Value	New Value
host	localhost	IP address or hostname of the database to connect to
user	example	The username to use for authentication when connecting to the database
password	example	The password to use with the username (above) for authentication when connecting to the database

3. Remove the comment characters (##) at the beginning of each line where you changed parameter values. Otherwise, the mysql default values are used.

Configure the Hue Database

The Hue server stores user-account information, the job submission history, and Hive queries in one of the following supported databases:

- SQLite, the embedded Hue database (the default database)
- MySQL
- PostgreSQL
- Oracle
- MariaDB

last-updated: Apr 11, 2022

Note: Using SQLite is not recommend.

Configure Hue to Store Data in MySQL or MariaDB

Hue supports MySQL 5.5.x - 5.7.x. Hue does not support MySQL 8 and later.

1. Configure the database connection parameters in the `[desktop][[database]]` section of `hue.ini`. Set the following properties with your database connection parameters:

```
[desktop]
  ...
  [[database]]
    engine=mysql
    host=<host>
    port=3306
    user=<user>
    password='<password>'
    name=<database>
```

   Example of configuration:

```
[desktop]
  ...
  [[database]]
    engine=mysql
    host=node1
    port=3306
    user=hue
    password=hue_password
    name=hue
```

   If the connection is secured through SSL/TLS, specify the CA, Cert, and Key paths, as shown in the following example:

```
[desktop]
[[database]]
...
options='{"ssl": {"key": "/path/to/client-key.pem", "cert": "/path/to/
client-cert.pem", "ca": "/path/to/ca.pem"}}'
```

   You can find detailed documentation for other options in the Django Connecting to the Database documentation.

2. Perform the initial data migration:

```
sudo /opt/mapr/server/configure.sh -R
```

3. Restart Hue:

```
maprcli node services -name hue -action restart -nodes <node_with_hue>
```

Related information

https://docs.djangoproject.com/en/1.11/ref/databases/#version-support

Configure Hue to Store Data in PostgreSQL

1. Install the following packages:

CentOS	`yum install gcc python-devel postgresql-devel`
SLES	`zypper install gcc python-devel postgresql-devel`
Ubuntu	`apt-get install gcc python-dev postgresql-server-dev-all`

2. Ensure that the `pg_config` command is in your `PATH`. For example, on CentOS with PostgreSQL 9.4 development package from the postgresql.org official repository, you need to add the directory with `pg_config` manually:

```
export PATH="/usr/pgsql-9.4/bin:$PATH"
```

3. Install the Python `psycopg2` package in Hue:

```
cd /opt/mapr/hue/hue-<version>
source ./bin/activate
pip install psycopg2
deactivate
```

4. Configure database connection parameters in the `[desktop][[database]]` section of `hue.ini`. Set the following properties for your database connection parameters:

```
[desktop]
  ...
  [[database]]
    engine=postgresql_psycopg2
    host=<host>
    port=5432
    user=<user>
    password=<password>
    name=<database>
    schema=<schema>
```

Example of Configuration

```
[desktop]
  ...
  [[database]]
    engine=postgresql_psycopg2
    host=node1
    port=5432
    user=hue
    password=hue_password
    name=hue
    schema=public
```

5. Perform the initial data migration:

```
sudo /opt/mapr/server/configure.sh -R
```


last-updated: Apr 11, 2022

6. Restart Hue:

```
maprcli node services -name hue -action restart -nodes <node_with_hue>
```

Configure Hue to Store Data in Oracle Database

Note: To configure Hue with Oracle 12, you need the Oracle 11 Instant Client Base and SDK.

1. Install the following packages:

CentOS	`yum install gcc python-devel`
SLES	`zypper install gcc python-devel`
Ubuntu	`apt-get install gcc python-dev`

2. Ensure that the library required to install the Oracle module in Hue is available through the `LD_LIBRARY_PATH` environment variable. The module that provides support for Oracle in Hue requires the `libclntsh.so` library to be available through the `LD_LIBRARY_PATH` environment variable. Typically, the library is located under the `$ORACLE_HOME` or `$ORACLE_HOME/lib` directories. Also, the library might include a version in the filename (for example, `libclntsh.so.11.1`), but the Oracle module for Hue requires it to be named `libclntsh.so`.
   a) Ensure that the `ORACLE_HOME` environment variable is set:

```
export ORACLE_HOME=<path_to_oracle_installation>
```

   b) Use the `find` command to locate the library:

```
find "$ORACLE_HOME" -name "libclntsh.so*"
```

   c) Go to the directory and ensure that the library is available and has the proper filename. If not, you can create a symbolic link:

```
ln -s libclntsh.so.11.* libclntsh.so
```

   d) Add the following variables to the Hue environment configuration by creating a file in `/opt/mapr/hue/hue-<version>/bin/env.d/`. For example, create `/opt/mapr/hue/hue-<version>/bin/env.d/99custom` with the following content:

```
export ORACLE_HOME="<path_to_oracle_installation>"
export LD_LIBRARY_PATH="$ORACLE_HOME:$LD_LIBRARY_PATH"
```

   or the following, depending on your Oracle configuration:

```
export ORACLE_HOME="<path_to_oracle_installation>"
export LD_LIBRARY_PATH="$ORACLE_HOME/lib:$LD_LIBRARY_PATH"
```

3. Depending on the Oracle instant client version installed:
   a) Run the shell script:

```
$ sudo sh -c "echo /usr/lib/oracle/<version-number>/client64/lib
> /etc/ld.so.conf.d/oracle-instantclient.conf"
```


last-updated: Apr 11, 2022

b) Use the `ldconfig` command to create the linking:

```
$ sudo ldconfig
```

c) Verify the dynamic linking:

```
 $ sudo ldconfig -p | grep -i oracle
```

For example:

```
ldconfig -p | grep -i oracle
libocijdbc11.so (libc6,x86-64) => /opt/oracle/instantclient_11_2/
libocijdbc11.so
libociei.so (libc6,x86-64) => /opt/oracle/instantclient_11_2/
libociei.so
libocci.so.11.1 (libc6,x86-64) => /opt/oracle/instantclient_11_2/
libocci.so.11.1
libnnz11.so (libc6,x86-64) => /opt/oracle/instantclient_11_2/
libnnz11.so
libclntsh.so.11.1 (libc6,x86-64) => /opt/oracle/instantclient_11_2/
libclntsh.so.11.1
```

4. Install the Python `cx_Oracle` package in Hue:

```
cd /opt/mapr/hue/hue-<version>
source ./bin/activate
pip install cx_Oracle==5.3
deactivate
```

Note: Python cx_Oracle version 5.3 is supported.

5. Configure the database connection parameters in the `[desktop][[database]]` section of `hue.ini`. Note these considerations:

   - Make sure that you have the appropriate permissions to use LOBs; for example, `SQL GRANT on SYS.DBMS_LOB`.
   - SID refers to the Oracle system ID, which is used to uniquely identify the database.

```
[desktop]
  ...
  [[database]]
    engine=oracle
    host=<host>
    port=1521
    user=<user>
    password=<password>
    name=<SID of the database>
```

- To achieve a multithreading environment, you can specify the `options={'threaded':true}` parameter in this section. For example:

```
[desktop]
  ...
  [[database]]
    engine=oracle
    host=node1
    port=1521
    user=hue
    password=hue_password
    name=XE
    options={'threaded':true}
```

By setting the `port=0` parameter, you can use the Oracle Service Name instead of specifying the SSID. For example:

```
[desktop]
  ...
  [[database]]
    engine=oracle
    port=0
    user=hue
    password=hue_password
    name=node1:1521/hue
```

6. Perform the initial data migration:

```
sudo /opt/mapr/server/configure.sh -R
```

7. Restart Hue:

```
maprcli node services -name hue -action restart -nodes <node_with_hue>
```

Configuring an Oracle Schema

You must create schemas for Oracle databases manually.

Edit the `hue.ini` file to use an Oracle database:

For detailed steps, see `inspectdb` and `dumpdb` Hue commands, http://gethue.com/hue-api-execute-some-builtin-commands/.

Configure Hue with Security

Based on the Hue version and the version of the other components that Hue communicates with, you can configure the following security features:

Note: You cannot enable both Kerberos and MapR-SASL on the same cluster.

Configure Hue to use Kerberos

After you set up a Kerberos principal and keytab file, you can configure Hue to use the Kerberos authentication protocol.

After you set up a Kerberos principal and keytab file, enable the Kerberos Ticket Renewal service, update `hue.ini` and `core-site.xml` with the required parameters, and restart the Warden and Hue services.

last-updated: Apr 11, 2022

Enabling the Kerberos Ticket Renewer Service

Important: This functionality requires a patch. The patch works with EEP-6.3.0 (Core-6.1.0 and Hue 4.3.0). To install patches, see Applying a Patch.

The Kerberos Ticket Renewer service (kt_renewer) renews tickets for the Hue service. Hue automatically starts the kt_renewer process on clusters that use Kerberos for authentication. Kerberos tickets have a default expiration time of 7 days. The kt_renewer service extracts the Kerberos ticket from the keytab file and renews the ticket before it expires.

You can access the kt_renewer process log files in the following locations:

- `${HUE_HOME}/logs/kt_renewer.out`
- `${HUE_HOME}/logs/kt_renewer.log`

To use the Kerberos Ticket Renewer service:

1. Enable the Kerberos Ticket Renewer Service:
   - In the `kdc.conf` file, add the `max_renewable_life` parameter.
   - In the `krb5.conf` file, add the `renew_lifetime` parameter.
2. Update the `hue.ini` file to include the Kerberos credentials cache path (`ccache_path`) and ticket renewal frequency (`keytab_reinit_frequency`), as shown in the following example:

```
[desktop]
               [[kerberos]]
                 …
                 # Path to keep Kerberos credentials cached
                 # ccache_path=/tmp/custom_hue_krb5_ccache
                 # Frequency in seconds with which Hue will renew its
keytab
                 # keytab_reinit_frequency=86400
                 …
```

Modifying the hue.ini File

In the `kerberos` section of the `hue.ini` file, make the following changes:

1. Supply the path to Hue's Kerberos keytab file.
2. Supply the Kerberos principal name for Hue.
3. Supply the path to kinit.
4. In the `[[yarn_clusters]] [[[default]]]` section:
   - If you are using a certificate signed by the CA (Certificate Authority), set the `ssl_cert_ca_verify` value to `True`.
   - If you are using a self-signed certificate or no certificate, leave the value set to `False`.
5. **For Hue with secure Hive**: In the `beeswax` section, make sure that the `hive_conf_dir` property points to a directory containing a valid `hive-site.xml` file (either the original or a synced copy).
6. **Optional**: To enable SSL encryption, see Enable SSL Encryption Between Hue and Hive.

last-updated: Apr 11, 2022

7. Make sure that you specified a fully-qualified domain name (FQDN) for all services integrated with Hue that uses Kerberos:

   HttpFS: Set the `webhdfs_url` property in the `[hadoop] [[hdfs_clusters]] [[[default]]]` section.

   HiveServer2: Set the `hive_server_host` property in the `[beeswax]` section.

   Impala: Set the `server_host` property in the `[impala]` section.

   Spark: Set the `livy_server_url` property in the `[impala]` section.

   Note: Support for Kerberos integration with Livy was introduced in Hue 4.X.

   Sqoop2: Set the `server_url` property in the `[sqoop]` section.

   Oozie: Set the `oozie_url` property in the `[liboozie]` section.

   HBase: Set the `hbase_clusters` property in the `[hbase]` section.

   Drill: Refer to section.

The changes are summarized in the following `hue.ini` files, which you can use as a template:

```
[desktop]
   [[kerberos]]
     # Path to Hue's Kerberos keytab file
     hue_keytab=/opt/mapr/conf/mapr.keytab

     # Kerberos principal name for Hue
     # hue_principal=mapr/<hostname>@<realm>
     # Substitute your hostname and realm in the example below
     hue_principal=mapr/perfnode181.perf.lab@dev-maprtech

     # Path to keep Kerberos credentials cached
     # ccache_path=/tmp/custom_hue_krb5_ccache
     # Frequency in seconds with which Hue will renew its keytab
     # keytab_reinit_frequency=86400

     # Path to kinit
     # Note that the actual path depends on which Linux OS you are using
     kinit_path=/usr/bin/kinit

[beeswax]
  # If Kerberos security is enabled, use fully-qualified domain name
  # (FQDN)
  hive_server_host=<FQDN of Hive Server>
  # Hive configuration directory, where hive-site.xml is located.
  hive_conf_dir=/opt/mapr/hive/hive-<version>/conf
  # Change this if your Hive is secured
  security_enabled=true
  # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY
  mechanism=GSSAPI

[impala]
  # Host of the Impala Server (one of the Impalad)
  server_host=<FQDN of Impalad>
  # Kerberos principal
  impala_principal=mapr/perfnode181.perf.lab@dev-maprtech

[hadoop]
  ...
  [[hdfs_clusters]]
     [[[default]]]
```

```
  # Enter the filesystem uri
  fs_defaultfs=maprfs:///

  # Use WebHdfs/HttpFs as the communication mechanism.
  # Domain should be the NameNode or HttpFs host.
  # Default port is 14000 for HttpFs.
  webhdfs_url=https://<FQDN of HttpFS>:14000/webhdfs/v1

  # Change this if your HDFS cluster is secured
  security_enabled=True

  # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY
  mechanism=GSSAPI
  ...
[[yarn_clusters]]
  [[[default]]]
  # Enter the host on which you are running the ResourceManager
  ## resourcemanager_host=localhost

  # The port where the ResourceManager IPC listens on
  ## resourcemanager_port=8032

  # Whether to submit jobs to this cluster
  submit_to=true

  # Change this if your YARN cluster is secured
  security_enabled=true

  # URL of the ResourceManager API
  ## resourcemanager_api_url=https://localhost:8090

  # URL of the ProxyServer API
  ## proxy_api_url=https://localhost:8090

  # URL of the HistoryServer API
  history_server_api_url=https://localhost:19890

  # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY
  mechanism=GSSAPI

  # In secure mode (HTTPS), if SSL certificates from Resource Manager's
  # Rest Server have to be verified against certificate authority
  ssl_cert_ca_verify=False

[spark]
  # The Livy Server URL.
  livy_server_url=https://<FQDN of Livy Server>:8998

  # Whether Livy requires client to perform Kerberos authentication.
  security_enabled=True

  # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY.
  mechanism=GSSAPI

[liboozie]
  # The URL where the Oozie service runs on. This is required in order for
  # users to submit jobs.
  oozie_url=https://<FQDN of Oozie>:<oozie_port_number>/oozie

  # Requires FQDN in oozie_url if enabled
  security_enabled=true
  # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY
 mechanism=GSSAPI
```


last-updated: Apr 11, 2022

```
[hbase]
  # Comma-separated list of HBase Thrift servers for clusters in the format
of '(name|host:port)'.
  # Use full hostname with security.
  # If using Kerberos we assume GSSAPI SASL, not PLAIN.
  hbase_clusters=(Cluster|<FQDN of Hbase Thrift Server>:9090)
  # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY
  mechanism=GSSAPI
```

Note: You need to manually set `security_enabled` property to `true` and `mechanism` property to `GSSAPI` for a Kerberised environment. These options are automatically configured only on a MapR-SASL cluster.

Modifying the core-site.xml File

In the `core-site.xml` file, provide the shortname for the Kerberos principal as shown. In addition, verify that you configured the proxyuser during configuration. See Configure Hue for details.

```
<!-- Hue security configuration -->
<property>
  <name>hue.kerberos.principal.shortname</name>
  <value>mapr</value>
</property>
<property>
  <name>hadoop.proxyuser.mapr.groups</name>
  <value>*</value> <!-- A group that all users of Hue belong to, or the
wildcard value "*" -->
</property>
<property>
  <name>hadoop.proxyuser.mapr.hosts</name>
  <value><hue_server_FQDN></value>
</property>
```

Restarting Warden and Hue

After you make all the changes to the files listed above, restart Warden and Hue so the changes will take effect.

Configure Hue to use MapR-SASL

You can configure Hue to use MapR-SASL for its communications with various components on a secure MapR cluster. Hue automatically detects and sets the security configuration of the cluster and its components. Therefore, in some cases, minimal configuration is required.

The following components are supported by Hue with MapR-SASL:

- HttpFS
- YARN and Spark History Server
- Hive
- Livy
- Sqoop2
- HBase Thrift
- Oozie
- Drill

last-updated: Apr 11, 2022

Note: For secure by default clusters or for clusters where the `customSecurity` flag is not added, Hue automatically sets `security_enabled` to `true` and `mechanism` to `MAPR-SECURITY` for these components. In all other cases, `${security_enabled}` and `${mechanisms}` variables are set to `false` and `none` respectively, but these options can be configured manually for custom setups.

An example of a default configuration with automatically defined `security_enabled` and `mechanism` properties for HttpFS is as follows:

```
[hadoop]
 [[hdfs_clusters]]
   # HA support by using HttpFs
   [[[default]]]
        ...
   # Change this if your HDFS cluster is secured
   security_enabled=${security_enabled}

   # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY
   mechanism=${mechanism}
        ...
```

An example of a manual configuration of `security_enabled` and `mechanism` properties for HttpFS is as follows:

```
[hadoop]
 [[hdfs_clusters]]
   # HA support by using HttpFs
   [[[default]]]
        ...
   # Change this if your HDFS cluster is secured
   security_enabled=true

   # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY
   mechanism=MAPR-SECURITY
        ...
```

Note: After you configure the `hue.ini`, you must restart Hue. However, if you configure multiple sections of the same file, you can restart Hue one time after your updates are complete.

Configure LDAP Authentication Between Hue and Hive

You can configure Hue to use LDAP Authentication when it communicates with HiveServer2. Before you configure Hue to use LDAP authentication with HiveServer2, verify that HiveServer2 is configured to use LDAP authentication. For more information, see Configure HiveServer2 to use LDAP Authentication on page 3440.

Complete the following steps to configure LDAP authentication between Hue and Hive:

1. Configure Hue to connect to Hive with LDAP authentication:
   a) Configure the `[beeswax]` section of the `hue.ini:` set `mechanism` option.

```
[beeswax]
  ...
  # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY
  mechanism=LDAP
```

b) Configure the `[beeswax]` section of the `hue.ini` (for Hive integration only):

```
[beeswax]
  ...
 # Override the default desktop username and password of the hue user
used for authentications with other services.
  # e.g. Used for LDAP/PAM pass-through authentication.
  auth_username=sampleuser
  auth_password=123456
  ...
```

Or configure the `[desktop]` section of the `hue.ini` to set the username and password for all services that require username/password authentication:

```
[desktop]
  ...
  # Default LDAP/PAM/.. username and password of the hue user used
for authentications with other services.
  # Inactive if password is empty.
  # e.g. LDAP pass-through authentication for HiveServer2 or Impala.
Apps can override them individually.
  auth_username=sampleuser
  auth_password=123456
  ...
```

2. **Optional:** Configure Hue to authenticate users through LDAP. See Configure Hue with LDAP.

3. Restart Hue:

```
maprcli node services -name hue -action restart -nodes <space delimited
list of nodes>
```

Configure PAM Authentication Between Hue and Hive

In Hue 3.10, you can configure Hue to use PAM authentication when it communicates with HiveServer2. Before you configure Hue to use PAM authentication with HiveServer2, verify that HiveServer2 is configured to use PAM authentication. See Configure HiveServer2 to Use PAM Authentication for more information.

Complete the following steps to configure PAM authentication between Hue and Hive:

1. Configure the `[beeswax]` section of the `hue.ini` file. Set the `mechanism` option to `none`. Set the `auth_username` and `auth_password` options in the `[desktop]` or `[beeswax]` sections of `hue.ini` (where `auth_username` and `auth_password` are the user credentials for the user who authenticates the Hue service with HiveServer2).
   The following example summarizes these changes:

```
[desktop]
     ...
     # Default LDAP/PAM/.. username and password of the Hue user used
for authentication with other services.
     # Inactive if password is empty.
     # e.g. LDAP pass-through authentication for HiveServer2 or Impala.
 Apps can override them individually.
       auth_username=mapr
       auth_password=<user_password>
     ...
 [beeswax]
     ...
     # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY
       mechanism=none
```

2. Restart Hue:

```
maprcli node services -name hue -action restart -nodes <Hue node>
```

Configure Kerberos Between Hue and Sqoop2

You can configure Kerberos authentication between Hue and Sqoop2.

- The MapR cluster must be configured to use Kerberos. For more information, see Configuring Kerberos User Authentication.
- Sqoop2 must be configured to use Kerberos. For more information, see Configure Kerberos Authentication for Sqoop2.

To configure Kerberos authentication between Sqoop2 and Hue, complete the following steps:

1. Configure Hue to use Kerberos for YARN. For more information, see Configure Hue to use Kerberos.
2. In the `[sqoop]` section of the `hue ini`, set `mechanism` to `GSSAPI`. For example:

```
[sqoop]
  # For autocompletion, fill out the librdbms section.
  # Sqoop server URL
  server_url=http://localhost:12000/sqoop
  # Change this if your cluster is secured
  security_enabled=${security_enabled}
  # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY
  mechanism=GSSAPI
```

3. Restart Hue.

```
maprcli node services -name hue -action start -nodes <space delimited
list of nodes>
```

Enable SSL Encryption Between Hue and Hive

Hue automatically determines when SSL encryption is enabled in Hive by reading the `hive-site.xml` file.

Note: Starting from EEP 6.0.0, SSL encryption is enabled by default for Hive in a secured cluster. To configure SSL for Hive on a non-secure cluster, refer to Hive Encryption on page 3447.

1. The following example shows how to correctly configure the directory path where the `hive-site.xml` file is located in the `hive_conf_dir` property in `[beeswax]` section of the `hue.ini` file:

```
[beeswax]
   ...
   # Hive configuration directory, where hive-site.xml is located
   hive_conf_dir=/opt/mapr/hive/hive-2.3/conf
```

2. The following examples show how to enable or disable Hue verification of service certificates by configuring `ssl_cacerts` and `ssl_validate` properties in `[desktop]` section of the `hue.ini` file:

   Example for enabling certificate verification:

```
[desktop]
   ...

  # Path to default Certificate Authority certificates. As example: /
path/to/cacert.pem
  ssl_cacerts=/opt/mapr/conf/ssl_truststore.pem

  # Choose whether Hue should validate certificates received from the
server.
  ssl_validate=true
```

   Example for disabling certificate verification:

```
[desktop]
   ...

   # Path to default Certificate Authority certificates. As an example: /
path/to/cacert.pem
   # ssl_cacerts=

   # Choose whether Hue should validate certificates received from the
server.
   ssl_validate=false
```

3. After you change these properties, restart Hue to apply your changes:

```
maprcli node services -name hue -action start -nodes <hostname>
```

Enable SSL Encryption Between Hue and HttpFS

As of HttpFS 1.0-1504 and Hue 3.7-1505, you can enable SSL encryption and mutual-based authentication between Hue and HttpFS on a secure MapR cluster that is version 4.0.2 or greater.

Complete the following steps to enable SSL encryption and mutual-based authentication between Hue and HttpFS on a secure cluster:

1. Configure HttpFS to use SSL or verify that HttpFS is configured to use SSL. For details, see SSL Security for HttpFS.

2. Set the `webhdfs_url` property in the `[hadoop] [[hdfs_clusters]] [[[default]]]` section of the `hue.ini` file to contain the correct URL for HttpFS with the HTTPS schema and domain of the HttpFS server:

```
[hadoop]
  [[hdfs_clusters]]
    [[[default]]]
      # Use WebHdfs/HttpFs as the communication mechanism.
      # Domain should be the NameNode or HttpFs host.
      # Default port is 14000 for HttpFs.
      webhdfs_url=https://node1.cluster.com:14000/webhdfs/v1
```

3. You can enable or disable Hue verification of service certificates by configuring `ssl_cacerts` and `ssl_validate` properties in the `[desktop]` section of the `hue.ini` file.

   Example for enabling certificate verification:

```
[desktop]
   ...

  # Path to default Certificate Authority certificates. As example: /
path/to/cacert.pem
  ssl_cacerts=/opt/mapr/conf/ssl_truststore.pem

  # Choose whether Hue should validate certificates received from the
server.
  ssl_validate=true
```

   Example for disabling certificate verification:

```
[desktop]
   ...

  # Path to default Certificate Authority certificates. As example: /
path/to/cacert.pem
  # ssl_cacerts=

  # Choose whether Hue should validate certificates received from the
server.
  ssl_validate=false
```

4. [OPTIONAL] Configure mutual authentication between Hue and HttpFS. Add the following configuration in the `hue.ini` file under the `[hadoop] [[hdfs_clusters]] [[[default]]]` section.

   - `mutual_ssl_auth=True`
   - `ssl_cert=/path/to/certificate.pem`
   - `ssl_key=/path/to/private_key.pem`

   Use absolute paths for `ssl_cert` and `ssl_key`. Hue does not support private keys with a passphrase in this step.

last-updated: Apr 11, 2022

The changes are summarized in the following example in the `hue.ini` file, which you can use as a template:

```
[hadoop]
 [[hdfs_clusters]]
  # HA support by using HttpFs
 [[[default]]]
  # Use WebHdfs/HttpFs as the communication mechanism.
  # Domain should be the NameNode or HttpFs host.
  # Default port is 14000 for HttpFs.
  webhdfs_url=https://node1.cluster.com:14000/webhdfs/v1
      ....
  # SSL certificate based authentication
  ssl_cert=/path/to/certificate.pem
  ssl_key=/path/to/private_key.pem
```

5. Restart Hue.

```
maprcli node services -name hue -action start -nodes <ip_address>
```

6. To test that SSL encryption is enabled for HttpFS, run the following command:

```
curl -k --cert /path/to/certificate.pem --key /path/to/private_key.pem
"https://node1.cluster.com:14000/webhdfs/v1?
op=GETFILESTATUS&user.name=mapr"
```

Enable SSL Encryption Between Hue and Sqoop2

The following procedure explains how to enable SSL encryption between Hue and Sqoop2.

1. Configure Sqoop2 to use SSL, or verify that Sqoop2 is configured to use SSL. For details, see "Enabling SSL Encryption in Sqoop2."
2. In the `hue.ini` file:
   a) Add the following entry under the [desktop] section:

   ```
   ssl_cacerts=/opt/mapr/conf/ssl_truststore.pem
   ```

   b) Add the following entry under the [sqoop] section:

   ```
   server_url=https://<host>:12000/sqoop
   ```

   The following example `hue.ini` file summarizes the changes. You can use this example as a template:

   ```
   ….
   [desktop]
   ….
   # Path to default Certificate Authority certificates.
   ssl_cacerts=/opt/mapr/hue/hue-4.2.0/cert.pem
   ….
   [sqoop]
   # For autocompletion, fill out the librdbms section.

   # Sqoop server URL
   server_url=https://ubuntu500:12000/sqoop
   ```

3. Restart Hue:

```
maprcli node services -name hue -action restart -nodes <ip_address>
```

Troubleshoot Hue Security Issues

To troubleshoot Kerberos security issues, enable the debugger by changing the following setting in the `/opt/mapr/conf/env.sh` file:

```
# uncomment the following line to debug client kerberos issues
#MAPR_KERBEROS_DEBUG="-Dsun.security.krb5.debug=true -Dsun.security.spnego.d
ebug=true -Djavax.net.debug=all"
```

Under the Hue installation directory, check `logs/runcpserver.log` for errors. Some sample error messages are shown below.

Could not start SASL
If you see this message, try using renewable tickets:

```
TypeError: TTransportException('Could not start SASL: Error in
sasl_client_start (-1) SASL(-1): generic failure: GSSAPI Error: Unspecified
GSS failure.  Minor code may provide more information (Ticket expired)',)
is not JSON serializable
```

Run the `kinit` command to generate a new ticket with a long running lifetime, then restart the Hue webserver.

Configuration Error

If you see this message, it means that the ticket generated by the `kinit` command from `maprlogin kerberos` was not copied to `/tmp/hue_krb5_ccache`:

```
Caused by: javax.security.auth.login.LoginException: Configuration Error -
useTicketCache should be set to true to use the ticket cache /tmp/
hue_krb5_ccache
```

This can happen when you generate a new ticket after the original ticket expires and forget to copy it into the ticket cache. Run the following command to copy the ticket into the ticket cache:

```
kinit -k -t /opt/mapr/conf/mapr.keytab -c /tmp/hue_krb5_ccache mapr/
perfnode181.perf.lab@dev-maprtech
```

Password incorrect while getting initial credentials

This message (`Password incorrect while getting initial credentials`) appears when you create a keytab file, but try to authenticate with a password. The act of creating a keytab causes a new random key to be placed in the Kerberos database and into the keytab file (`/opt/mapr/conf/mapr.keytab`). That key does not have a password associated with it, so you can only authenticate using the keytab.

If you want to authenticate with a password, run the `cpw` command in `kadmin` instead of the `ktadd` command.

Integrate Hue

This section contains the following topics with information for integrating Hue with other ecosystem components:

last-updated: Apr 11, 2022

Integrate Hue with MapR Database Binary Tables

You can use the Hue HBase application to access MapR Database binary tables.

Note: In order to use the Hue HBase application, you need to install the HBase Client and the HBase Thrift Gateway. For more information, see the Installing MapR and MapR Ecosystem Components on page 128.

Step 1: Setting up MapR Database Binary Table Mapping

To use the Hue HBase application to access MapR Database binary tables, you need to set the `hbase.table.namespace.mappings` property.

Table Mapping Naming Conventions

A table mapping takes the form *`name`*`:`*`map`*, where *`name`* is the table name to redirect and *`map`* is the modification made to the name. The value in *`name`* can be a literal string or contain the `*` wildcard. When mapping a name with a wild card, the mapping is treated as a directory. Requests to tables with names that match the wild card are sent to the directory in the mapping.

When mapping a name that is a literal string, you can choose from two different behaviors:

- End the mapping with a slash to indicate that this mapping is to a directory. For example, the mapping `mytable1:/user/aaa/` sends requests for table `mytable1` to the full path `/user/aaa/mytable1`.
- End the mapping without a slash, which creates an alias and treats the mapping as a full path. For example, the mapping `mytable1:/user/aaa` sends requests for table `mytable1` to the full path `/user/aaa`.

Example: Map Table Names to MapR Database

In the following example, the `hbase.table.namespace.mappings` property is set so that any flat table name, such as `mytable`, is treated as a MapR Database table in the directory `/tables_dir/mytable`.

```
<property>
  <name>hbase.table.namespace.mappings</name>
  <value>*:/tables_dir</value>
</property>
```

Once you finish enabling table mapping in the `core-site.xml` file, start (or restart) the HBase thrift server so the changes will take effect.

```
maprcli node services -name hbasethrift -action start -nodes node001
```

Step 2: Configure Hue for MapR Database

To configure Hue for MapR Database, edit the `hbase` section of the `hue.ini` file, which looks like this:

```
[hbase]
  # Comma-separated list of HBase Thrift servers for
  # clusters in the format of '(name|host:port)'.
  ## hbase_clusters=(Cluster|localhost:9090)

  # Hard limit of rows or columns per row fetched before truncating.
  ## truncate_limit = 500
```


last-updated: Apr 11, 2022

In this file, make the following changes:

1. Uncomment the `## hbase_clusters=(Cluster|localhost:9090)` statement and provide the list of HBase Thrift servers.

```
hbase_clusters=(<clustername1>|<hostname1>:9090),(<clustername2>|
<hostname2>:9090)[,...]
```

2. Uncomment the `truncate_limit` statement and change the value if necessary.

Integrate Hue with Hive
Describes how to integrate Hue with Hive through settings in the `hue.ini` file.

By default, Hue connects to a single instance of HiveServer2 through the `hive_server_host` parameter in `hue.ini`; however, if high availability (HA) is enabled for HiveServer2, Hue can leverage that. If one instance of HiveServer2 goes down, the client automatically connects to another instance of HiveServer2.

Verify the Hive Version

Before you integrate Hue with HiveServer2 (HA or single instance), verify that the path specified in the `hive_conf_dir` property applies to the Hive version that you have installed. If needed, update the path to reflect the Hive version that you have installed.

```
# Hive configuration directory, where hive-site.xml is located
hive_conf_dir=/opt/mapr/hive/hive-<version>/conf
```

If Hue and Hive are installed on separate nodes, you must also copy the Hive `conf` directory to the Hue node.

Integrating Hue with HiveServer2 High Availability

If you want Hue to leverage HiveServer2 HA, enable high availability for HiveServer2, and update the `hue.ini` file to include the following properties and settings:

```
[beeswax]
  #Whether to use service discovery for llap.
  hive_discovery_llap = true
  #Is llap (hive server interactive) running in HA.
  hive_discovery_llap_ha = true
  #Whether to use service discovery for HiveServer2.
  hive_discovery_hs2 = true
[libzookeeper]
  #ZooKeeper ensemble; comma-separated list of host/port.
  ensemble=<host:port>:5181
```

Note that the `hive_server_host` and `hive_server_port` properties in `hue.ini` are not required if using HiveServer2 HA. Service discovery overrides the server and thrift port.

Perform any additional Hue configurations and then restart Hue for changes to take effect. See Starting the Hue Webserver.

Integrating Hue with a Single Instance of HiveServer2

Update the beeswax section of the `hue.ini` file to include the following properties and settings:

Note: This is not required on a single node cluster.

```
[beeswax]

# Host where HiveServer2 is running.
```

```
# If Kerberos security is enabled, use fully-qualified domain name (FQDN).
  hive_server_host=<FQDN of Hive Server>

# Port that HiveServer2 Thrift server runs on.
  hive_server_port=10000
```

Perform any additional Hue configurations and then restart Hue for changes to take effect. See Starting the Hue Webserver.

Configuring Data and Metadata Directories

When Hue and Hive are used together, they are usually configured to share metadata and data directories. However, you can create separate directories for Hue and Hive.

The locations of the shared directories are specified by the following properties in the `hive-site.xml` file:

- `hive.metastore.uris` (the hostname and port of the Hive Metastore node)
- `hive.metastore.warehouse.dir` (the directory where the default database for the warehouse is located)

See Configure Shared Hive Data and Metadata Directories for Hue and Configure Separate Hive Data and Metadata Directories for Hue for more information.

Configure Shared Hive Data and Metadata Directories for Hue

To configure shared Hive data and metadata directories for Hue:

1. Change the `hive.metastore.uris` property as shown:

```
<property>
  <name>hive.metastore.uris</name>
  <value>thrift://localhost:9083</value>
  <description> URI where clients contact Hive metastore server </
description>
</property>
```

   Note: The `hive.metastore.warehouse.dir` property can keep its default value and does not need to be changed.

2. Enable Hue impersonation by setting the following property to `true`.

```
<property>
  <name>hive.metastore.execute.setugi</name>
  <value>true</value>
  <description> Set this property to enable Hive Metastore service
impersonation in unsecure mode.
   In unsecure mode, setting this property to true causes the metastore
to execute DFS operations
   using the client's reported user and group permissions. Note that
this property must be set on
   BOTH the client and server sides. </description>
</property>
```

3. Set the location of the `sharelib`.

```
<property>
  <name>oozie.service.WorkflowAppService.system.libpath</name>
  <value>/oozie/share/lib</value>
</property>
```

4. To enable the Hive Metastore service to share the embedded Derby database, add the following property blocks to the `hive-site.xml` file on the node running hiveserver2 to point to the location of the Derby metastore:

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:derby:;databaseName=/<local dir>/metastore_db;create=true</
value>
  <description>JDBC connect string for a JDBC metastore</description>
</property>

<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>org.apache.derby.jdbc.EmbeddedDriver</value>
  <description>Driver class name for a JDBC metastore</description>
</property>
```

5. To enable the Hive Metastore service to share a MySQL database, add the following property blocks to the `hive-site.xml` file:

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:mysql://<ip_address>:3306/hive_11?
createDatabaseIfNotExist=true</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>com.mysql.jdbc.Driver</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionUserName</name>
  <value><UserName></value>
  <description>Substitute the actual username</description>
</property>

<property>
  <name>javax.jdo.option.ConnectionPassword</name>
  <value><Password></value>
  <description>Substitute the actual password</description>
</property>
```

Configure Separate Hive Data and Metadata Directories for Hue

If you want to store Hue data and metadata in separate directories from Hive data and metadata, follow these steps:

1. Copy `hive-site.xml` to a new location. (The original `hive-site.xml` file remains in the previous location for use by Hive.)

last-updated: Apr 11, 2022

2. Edit `hue.ini` and change the `hive_conf_dir` property so it points to the new location for `hive-site.xml`.
3. Change the `hive.metastore.warehouse.dir` property in the new `hive-site.xml` file so it points to the directory where Hue data will be located.
4. Change the `hive.metastore.uris` property so it points to the directory for Hue's `metastore_db`.
5. Set the `hive.metastore.execute.setugi` property to `true`.

```
<property>
  <name>hive.metastore.execute.setugi</name>
  <value>true</value>
  <description> Set this property to enable Hive Metastore service
impersonation in non-secure mode.
   In non-secure mode, setting this property to true causes the
metastore to execute DFS operations
   using the client's reported user and group permissions. Note that
this property must be set on
   BOTH the client and server sides. </description>
</property>
```

Integrate Hue with Impala

Hue's Impala application supports interactive SQL and HiveQL queries from within your browser. Hue provides additional functionality with Impala that is not available from the command line such as parameterized queries, syntax highlighting, and saving queries.

For information about installing Impala, see Installing Impala.

Warning: To disable the Impala application, see Disabling an Application.

To configure Hue to work with Impala, edit the `impala` section of the `hue.ini` file and change parameter values as necessary. The next section shows the default version of the Impala settings in `hue.ini`. For instructions on how to change Impala settings, see Modifying the hue.ini File.

Default Impala Settings in hue.ini

The default version of the `impala` section of the `hue.ini` file is shown below.

```
###########################################################################
# Settings to configure Impala
###########################################################################

[impala]
 # Host of the Impala Server (one of the Impalad)
 ## server_host=localhost

 # Port of the Impala Server
 ## server_port=21050

 # Turn on/off impersonation mechanism when talking to Impala
 ## impersonation_enabled=False
```

Modifying the hue.ini File

To configure Hue to work with Impala, change the values of the following parameters in the `impala` section of the `hue.ini` file:

Parameter	Default Value	Description
`server_host`	localhost	The hostname or IP address of the Impala server
`server_port`	21050	The port of the Impala server
`impersonation_enabled`	False	Turns the impersonation mechanism on or off when talking to Impala

1. If `server_host` is not localhost, change the hostname (and remove the ## characters to uncomment the line).

   ```
   ## server_host=<hostname>
   ```

2. If you are *not* using the default server port (21050), change the `server_port` setting to the port you are using (and remove the ## characters to uncomment the line).

   ```
   ## server_port=<port_number>
   ```

3. Enable impersonation in Impala so the user `mapr` can impersonate any Linux PAM user on the cluster:
   a) Edit the `/opt/mapr/impala/impala-<version>/conf/env.sh` file, and add the following property to the section `IMPALA_SERVER_ARGS=*/`:

      ```
      authorized_proxy_user_config=mapr=*"
      ```

   b) Restart Impala.

      ```
      sudo maprcli node services -name impalaserver -action restart -nodes
      <ip_address>
      ```

4. Enable impersonation for Impala in the `hue.ini` file by setting `impersonation_enabled` to `true` (and remove the ## characters to uncomment the line).

   ```
   ## impersonation_enabled=true
   ```

5. Restart Hue so the changes will take effect.

Integrate Hue with Oozie

Complete the following steps to integrate Hue with Oozie:

1. Edit the `hue.ini` file to configure the location where the Oozie service is running. The path to the `hue.ini` file is: `/opt/mapr/hue/hue-<version>/desktop/conf/hue.ini`.

   ```
   [liboozie]
   # The URL (host IP address) where the Oozie service is running. This is
   required in order for # users to submit jobs.
   oozie_url=http://<ip_address>:<oozie_port_number>/oozie
   ```

   The `<oozie_port_number>` depends on whether your cluster is secure.

2. Enable user impersonation for Oozie through Hue:

last-updated: Apr 11, 2022

a) Add the following lines to the `oozie-site.xml` file (the path to the file is: `/opt/mapr/oozie/oozie-4.2.0/conf/oozie-site.xml`):

```
<property>

<name>oozie.service.ProxyUserService.proxyuser.<default_user>.hosts</
name>
  <value>*</value>
</property>
<property>

<name>oozie.service.ProxyUserService.proxyuser.<default_user>.groups</
name>
  <value>*</value>
</property>
```

In most cases, mapr is the <default_user>. This should be the same default user that is specified in the `hue.ini` and the `core-site.xml`.

b) Restart Oozie. To restart Oozie, first stop Oozie then start it:

```
maprcli node services -name oozie -action restart -nodes <ip_address>
```

To verify that the Oozie server started, enter:

```
lsof -i:<oozie_port_number>
```

The output from this command should look similar to this:

```
COMMAND PID USER FD TYPE DEVICE SIZE/OFF NODE NAME java 16644 mapr
35u IPv6 69926776 0t0 TCP *:irisa (LISTEN)
```

You can also check Oozie logs to verify that Oozie started. The log is found here: `/opt/mapr/oozie/oozie-<version>/logs/oozie.log`

3. Perform any additional Hue configurations and then restart Hue so that the changes will take effect. See Starting the Hue Webserver.

Integrate Hue with Spark (Experimental Only)

You can configure Hue to use the Spark Notebook UI. This allows users to submit Spark jobs from Hue.

Note: Spark Notebook is a feature that utilizes the Spark REST Job Server (Livy). The `mapr-livy` package must be installed on a node were the mapr-spark package is installed or the Livy service will not start.

1. In the `[spark]` section of the `hue.ini`, set the `livy_server_host` parameter to the host where the Livy server is running.

```
[spark]
# IP or hostname of livy server.
livy_server_url=https://<host>:8998
```

Note: If the Livy server runs on the same node as the Hue UI, you are not required to set this property as the value defaults to the local host.

2. Restart Hue.

```
maprcli node services -name hue -action restart -nodes <hue node>
```

Additional Information

- If needed, you can use the Control System or `maprcli` to start, stop, or restart the Livy Server. For more information, see Managing Services on page 827.

Note: Troubleshooting Tip

If you have more that one version of Python installed, you may see the following error when executing Python samples:

```
Py4JJavaError: An error occurred while calling
z:org.apache.spark.api.python.PythonRDD.collectAndServe...
```

Workaround:

Set the following environment variables in /opt/mapr/spark/spark-<version>/conf/spark-env.sh:

```
export PYSPARK_PYTHON=/usr/bin/python2.7
export PYSPARK_DRIVER_PYTHON=/usr/bin/python2.7
```

Integrate Hue with Drill

Starting in EEP 6.0, Drill is officially supported with Hue. When you integrate Drill with Hue, users can run Drill queries from the Hue interface and visualize data.

Drill integrates with Hue through configuration options in the `/opt/mapr/hue/hue-<version>/conf/hue.ini` file. A user can authenticate to Hue through Plain, Kerberos, or MapR-SASL authentication. The user that authenticates to Hue is the user that runs the Drill queries from Hue.

When connecting to Drill, Hue performs outbound impersonation to Drill as the user that authenticated to Hue. Drill accepts the outbound impersonation from Hue as an inbound impersonation. Drill then performs outbound impersonation to the MapR filesystem or MapR Database.

Note: In a secure cluster, you can only access the Hue interface through HTTPS.

Note: SSL encryption is currently not supported.

Prerequisites

Note the following prerequisites before you integrate Hue with Drill:

- The cluster must have the latest versions of Hue, Drill, and HTTPFS installed. Hue uses HTTPFS to communicate with the MapR file system. You can see the latest component versions in the Component Versions for Released EEPs. If you install Hue and Drill on a secure MapR cluster, Hue and Drill are installed with the default MapR security configurations and outbound impersonation is enabled.
  - **MapR Installer**

    When you install Hue and Drill using the MapR Installer, HTTPFS is installed automatically, and Hue is automatically configured to integrate Drill without having to perform any manual configuration. The installer configures a Zookeeper connection to Drill in `hue.ini`, by default.

  - **Manual Installation**
    - Install HTTPFS and then configure Hue, as described in the following section, *Configuring Hue*
- In a secure cluster, Drill must have user impersonation enabled.

- Drill has an inbound impersonation policy option, `exec.impersonation.inbound_policies`, that allows the Hue process user (proxy user) to impersonate the Hue authenticated user as an outbound impersonation from Hue to Drill. This option is automatically configured when Drill and Hue are installed using the MapR Installer with MapR default security enabled, or when you run configure.sh on a secure cluster. If you do not run configure.sh, you must manually add this option to the impersonation configuration in the `/opt/mapr/drill/drill-<version>/conf/drill-override.conf` file, as shown:

```
impersonation.enabled: true,
  impersonation.max_chained_user_hops: 3,
  exec.impersonation.inbound_policies: "[{proxy_principals:{users:
[\"mapr\"]},target_principals:{users:[\"*\"]}}]",
```

- If you plan to use Kerberos for authentication, you will need to include the Hue keytab file and Kerberos principal name for Hue in the hue.ini file. If needed, complete the steps listed in the Creating a Kerberos Principal and Extracting the Kerberos Ticket from the keytab File sections on the Configure Hue to use Kerberos on page 3636 page.

Configuring Hue

If you manually installed Hue, Drill, and HTTPFS, you must modify the `hue.ini` file to include the configuration information needed for Hue to connect with Drill and HTTPFS. The `hue.ini` file contains sections where you configure Hue to integrate with various components, like Drill and HTTPFS. You can access the `hue.ini` file in the `/opt/mapr/hue/hue-<version>/desktop/conf` directory. Start/Restart the services after you update the `hue.ini` file.

Note: If you installed Hue and Drill using the MapR Installer, these options are populated automatically in the `hue.ini` file; no configuration is required. In a secure MapR cluster, the authentication mechanism defaults to MAPR-SECURITY.

Complete the following steps to integrate Hue with Drill:

1. **Edit the Drill configuration in `hue.ini`.**

   The `hue.ini` file contains a `[[[drill]]]` section under which you can see configuration options needed for Hue to connect with Drill. You must uncomment an option (remove the # character) in the `hue.ini` file for the option to take effect.

   The following tables list and describe the Drill options with possible values and also provide examples:

Options	Descriptions	Examples
connection_type=	Tells Hue how to connect to Drill. Enter one of the following values: • direct • zookeeper A direct connection is a connection in which Hue connects directly to a Drillbit. A ZooKeeper connection is a connection in which Hue communicates with ZooKeeper and ZooKeeper provides Hue with a Drillbit to connect with.	connection_type=zookeeper

drillbits=	Enter the node IP address of the Drillbit that Hue connects with. Only enter a node address if using the “direct” connection_type.	drillbits=10.10.100.2:31010 To list multiple Drillbits, separate each IP address by a comma, as shown: drillbits=10.10.100.2:31010, 10.10.100.3:31010 **Note:** Port 31010 is the user port between nodes in a Drill cluster. This port is needed for an external client to connect into the cluster nodes and for the Drill Web Console.
zk_quorum=	Enter the list of ZooKeeper node IP addresses in the ZooKeeper quorum. Only enter the IP addresses for the ZooKeeper quorum if using the “zookeeper” connection_type.	zk_quorum=10.10.100.3:5181, 10.10.100.4:5181, 10.10.100.5:5181
zk_cluster_id=	Enter the name of the Drill cluster that you want Hue to connect to.	zk_cluster_id=dev-drillbits
mechanism=	The type of authentication enabled. Enter one of the following values: • None • GSSAPI • MapR-SECURITY Use None for Plain authentication. Use GSSAPI for Kerberos authentication. Use MAPR-SECURITY for maprsasl. If you set the mechanism to “none” and impersonation is enabled, you must set the username and password to the admin or proxy user that will impersonate Hue end users. You can set these with the user= and password= options. If you set the mechanism to GSSAPI, you must also include the ccache_path= option. For this option, enter the caching location for Kerberos credentials, for example: `ccache_path=/tmp/hue_krb5_ccache` **Note:** See Configure Hue to use Kerberos on page 3636 You can set the Drill Kerberos principal and/or Hue impersonation using the option named “`options=`.” See “options=” below.	mechanism=none

user=	If using Plain authentication, enter the username. If using another authentication mechanism, do not enter a value. Set the username to the admin or proxy user that will impersonate Hue end users. If impersonation is disabled, the you can set the user to any user. Hue will connect to Drill as the user specified.	user=mapr
password=	If using Plain authentication, enter the password. If using another authentication mechanism, do not enter a value. Set the password for the admin or proxy user that will impersonate Hue end users. If impersonation is disabled, set the password for the use specified.	password=mapr8

password_script=	Indicates which script to run for the database password when a password is required and the password= option is not set. Enter the location of the script.
	The following shell script is an example of a password script:
	`#!/bin/bash`
	`case $1 in`
	`drill) echo "password_1";;`
	`some-output) echo "password_2";;`
	`*) echo "wrong argument" >&2 exit 1;;`
	`esac`
		password_script='/root/hue_password_script/password_script.sh drill'

options=	Additional options related to impersonation and Kerberos authentication. This option takes the following values: • impersonation • principal Impersonation enables or disables outbound impersonation in Hue. Principal is the Drill service principal when Kerberos authentication is enabled.	options='{"impersonation": true, "principal": "mapr/localhost@REALM"}' options='{"impersonation": true}' options='{"impersonation": false}'

2. **Add the HTTPFS URL in `hue.ini`.**

   The `hui.ini` file contains a `[[[default]]]` section in the `[hadoop]` block under which you can see HDFS configuration options. You must uncomment an option (remove the # character) in the `hue.ini` file for the option to take effect.

   In the `[[[default]]]` section of the `[hadoop]` block, enter the IP address of the HTTPFS node as the value for the `webhdfs_url=` option, as shown:

   ```
   # Use WebHdfs/HttpFs as the communication mechanism.
   # Domain should be the NameNode or HttpFs host.
   # Default port is 14000 for HttpFs.
   webhdfs_url=https://<httpfs-node-ip-address>:14000/webhdfs/v1
   ```

3. **Start the services.**

   Start/Restart Hue, Drill, and HTTPS to apply the updated configurations, as shown in the following examples:

   ```
   maprcli node services -name hue -action start -nodes
   <hue-node-ip-address>

   maprcli node services -name drill-bits -action start -nodes
   <list-of-drill-node-ip-addresses>

   maprcli node services -name httpfs -action start -nodes
   <httpfs-node-ip-address>
   ```

Run Drill Queries in Hue

Once you have configured Hue and started the services, you can run Drill queries from Hue and visualize your data.

Complete the following steps to run Drill queries in Hue:

1. In your web browser, enter the Hue URL to navigate to the Hue web interface, as shown:

   ```
   http://hue-node-ip-address:8888
   ```

2. If prompted, enter your user credentials. The Hue interface opens.
3. In the **Query** drop-down, select **Editor > Drill**. The left navigation panel displays the list of schemas available in Drill.
4. Select a schema, for example `dfs.default`, and then enter a query in the text field.
5. Click the blue play button to execute the query. Query results display.
6. Optionally, you can use the buttons to the left of the query results to visualize the data.

Configure Hue to use Drill on a MapR-SASL-Secured Cluster
You can configure Hue to use Drill on a MapR-SASL cluster.

1. Configure Hue to use Drill:

a) In the `hue.ini`, go to the Drill section, and set the parameters. For example:

```
[librdbms]
  [[databases]]
    ...
    [[[drill]]]

    # Name to show in the UI.
    nice_name="Drill"

    # Database backend to use.
    engine=drill

    # Connection type. This can be:
    # 1. direct
    # 2. zookeeper
    connection_type=direct

    # Drillbit address for direct connection.
    drillbits=<node>:31010

    # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY.
    mechanism=MAPR-SECURITY

    # Available options:
    # "impersonation" to enable or disable outbound impersonation.
    # "principal" of Drill service. Used when Kerberos authentication
is enabled.
    options='{"impersonation": true, "principal": "mapr/
<node>@REALM"}'
```

2. Restart Hue to apply the updated configuration:

```
maprcli node services -name hue -action restart -nodes <node>
```

Configure Hue to use Drill on Kerberos-Secured Cluster
You can configure Hue to use Drill on a Kerberos-secured cluster.

1. Configure Hue to use Drill:

a) In the `hue.ini`, go to the Drill section, and set the parameters. For example:

```
[librdbms]
  [[databases]]
    ...
    [[[drill]]]

    # Name to show in the UI.
    nice_name=”Drill”

    # Database backend to use.
    engine=drill

    # Connection type. This can be:
    # 1. direct
    # 2. zookeeper
    connection_type=direct

    # Drillbit address for direct connection.
    drillbits=<node>:31010

    # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY.
    mechanism=GSSAPI

    # Available options:
    # "impersonation" to enable or disable outbound impersonation.
    # "principal" of Drill service. Used when Kerberos authentication
is enabled.
    options='{"impersonation": true, "principal": "mapr/
<node>@REALM"}'
```

2. Restart Hue to apply the updated configuration:

```
maprcli node services -name hue -action restart -nodes <node>
```

Configure Hue to use Drill on PAM-Secured Cluster
You can configure Hue to use Drill on a Pluggable Authentication Modules (PAM) secured cluster.

1. Configure Hue to use Drill:

a) In the `hue.ini`, go to the Drill section, and set the parameters. For example:

```
[librdbms]
  [[databases]]
    ...
    [[[drill]]]

    # Name to show in the UI.
    nice_name=”Drill”

    # Database backend to use.
    engine=drill

    # Connection type. This can be:
    # 1. direct
    # 2. zookeeper
    connection_type=direct

    # Drillbit address for direct connection.
    drillbits=<node>:31010

    # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY.
    mechanism=none

    # Username to authenticate with when connecting to the database.
    # Used with plain authentication (mechanism set to "none").
    user=<user>

    # Password matching the username to authenticate with when
    # connecting to the database.
    # Used with plain authentication (mechanism set to "none").
    password=<password>
```

2. Restart Hue to apply the updated configuration:

```
maprcli node services -name hue -action restart -nodes <node>
```

Configure Hue to use Drill on an Unsecured Cluster
You can configure Hue to use Drill on an unsecure cluster.

1. Configure Hue to use Drill:

a) In the `hue.ini`, go to the Drill section, and set the parameters. For example:

```
[librdbms]
  [[databases]]
    ...
    [[[drill]]]

    # Name to show in the UI.
    nice_name=”Drill”

    # Database backend to use.
    engine=drill

    # Connection type. This can be:
    # 1. direct
    # 2. zookeeper
    connection_type=direct

    # Drillbit address for direct connection.
    drillbits=<node>:31010

    # Security mechanism of authentication none/GSSAPI/MAPR-SECURITY.
    mechanism=none
```

2. Restart Hue to apply the updated configuration:

```
maprcli node services -name hue -action restart -nodes <node>
```

Integrate Hue 3.10+ With Spark 2

Important: Hue integration with Spark is an experimental feature. This topic describes how to integrate Hue 3.10 or later (expressed as "3.10+") with Spark 2 or later.

1. In the [spark] section of the `hue.ini` file, set the `livy_server_host` and `livy_server_port` parameters to the host and port where the Livy server is running:

```
[spark]
  # Host address of the Livy Server.
  livy_server_host=ubuntu500
  # Port of the Livy Server.
  livy_server_port=8998
```

2. To configure Hue to use Spark modes, modify `livy.conf` (`/opt/mapr/hue-livy/hue-livy-<version>/conf/livy.conf`):
   a) If Spark jobs run on local mode, set the `livy.spark.master` property:

```
…
# What spark master Livy sessions should use.
livy.spark.master = local[*]
….
```

b) If Spark jobs run on YARN mode, set the `livy.spark.master` and `livy.spark.deployMode` properties (client or cluster). For example:

```
….
# What spark master Livy sessions should use.
livy.spark.master = yarn
# What spark deploy mode Livy sessions should use.
livy.spark.deployMode = cluster
….
```

c) If Spark jobs run on Standalone mode, set the `livy.spark.master` property. For example:

```
# What spark master Livy sessions should use.
livy.spark.master = spark://ubuntu500:7077
```

d) If Spark jobs run on Mesos mode, set the `livy.spark.master` property. For example:

```
# What spark master Livy sessions should use.
livy.spark.master = mesos://<mesos-master-node-ip>:5050
```

Note: Integration of Spark on Mesos with Hue is not supported in cluster deployment mode.

3. If you want to be able to access Hive through Spark in Hue, you should configure Spark with Hive, and set `livy.repl.enableHiveContext` to `true` in `livy.conf`. For example:

```
...
# Whether to enable HiveContext in livy interpreter, if it is true
hive-site.xml will be detected
# on user request and then livy server classpath automatically.
livy.repl.enableHiveContext = true
...
```

4. If you are planning to use PySpark, you will need to set the PYTHONPATH environment variable in `livy-env.sh` (`/opt/mapr/livy/livy-<version>/conf/livy-env.sh`):

```
...
export PYTHONPATH=$SPARK_HOME/python/lib/py4j-<version>-
src.zip:$SPARK_HOME/python/:$PYTHONPATH
```

For example:

```
...
export PYTHONPATH=$SPARK_HOME/python/lib/py4j-0.10.7-
src.zip:$SPARK_HOME/python/:$PYTHONPATH
```

5. Make sure that R is installed on the node if you are planning to run SparkR. To install R to run SparkR jobs:

On Ubuntu	`sudo apt-get install r-base`
On Red Hat / CentOS	`sudo yum install R`

6. Restart the Spark REST Job Server (Livy).

```
maprcli node services -name livy -action restart -nodes <livy node>
```

7. Restart Hue:

```
maprcli node services -name hue -action restart -nodes <hue node>
```

Configure Hue with Sqoop2

Hue's Sqoop Transfer application is used to import data from MySQL to MapR File System and to export data from MapR File System to MySQL. Before you can run the Sqoop Transfer application, you need to:

- Install the Sqoop2 server and Sqoop2 client packages. For more information, see the Installing MapR and MapR Ecosystem Components on page 128.
- Edit the `hue.ini` file to configure Hue with Sqoop2.
- You can configure Hue to use Kerberos with Sqoop2. For more information, see Configure Kerberos Between Hue and Sqoop2.

Warning:

To disable the Sqoop2 application, see Disabling an Application.

Modifying the hue.ini File

1. Locate the `sqoop` section of the `hue.ini` file.

```
[sqoop]
  # For autocompletion, fill out the librdbms section.

  # Sqoop server URL
  ## server_url=https://localhost:12000/sqoop
```

2. Uncomment the `## server_url=https://localhost:12000/sqoop` statement and change `localhost` to the IP address or host name of the node where the Sqoop2 server is running.

```
server_url=https://<ip_address>:12000/sqoop
```

Warning: In Hue 3.x releases, Sqoop2 only supports files in comma-separated value (`.csv`) format.

Use Hue

This section provides information about using Hue, but it does not duplicate the Hue documentation.

You can also refer to the Hue documentation.

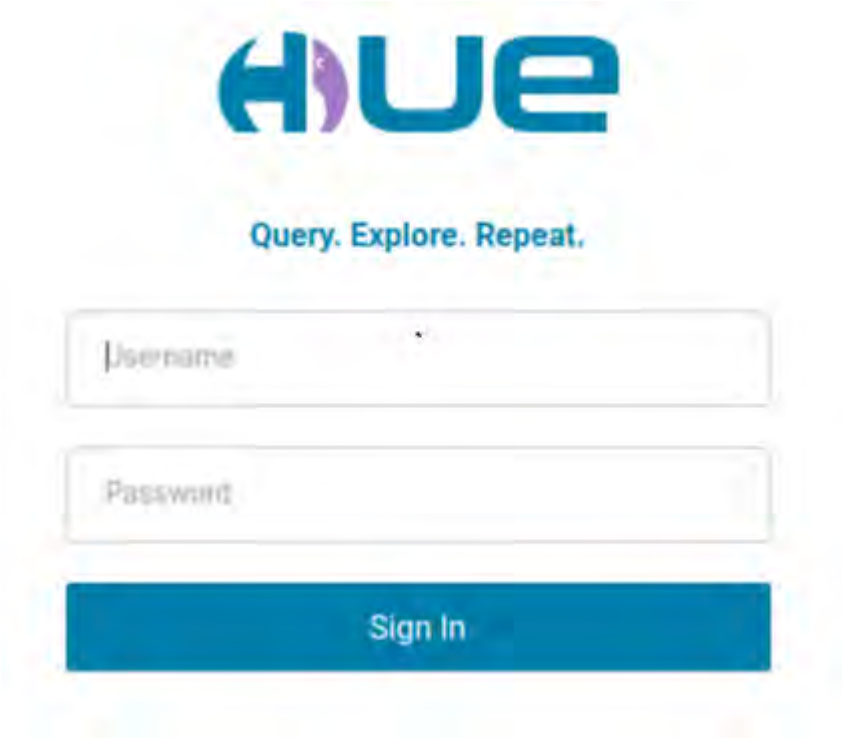
This section includes the following topics:

Logging in to Hue 4.X

Once Hue is installed and the configuration files have been edited, direct your browser to the IP address where you installed Hue.

- Open the Hue homepage: `ip_address>:8888`
  The following screen appears:

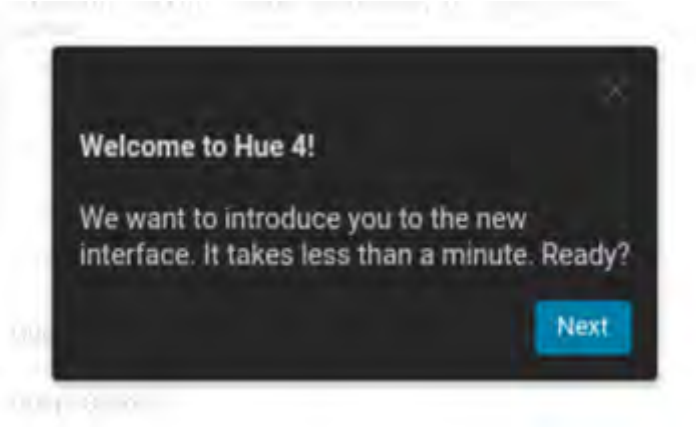
last-updated: Apr 11, 2022



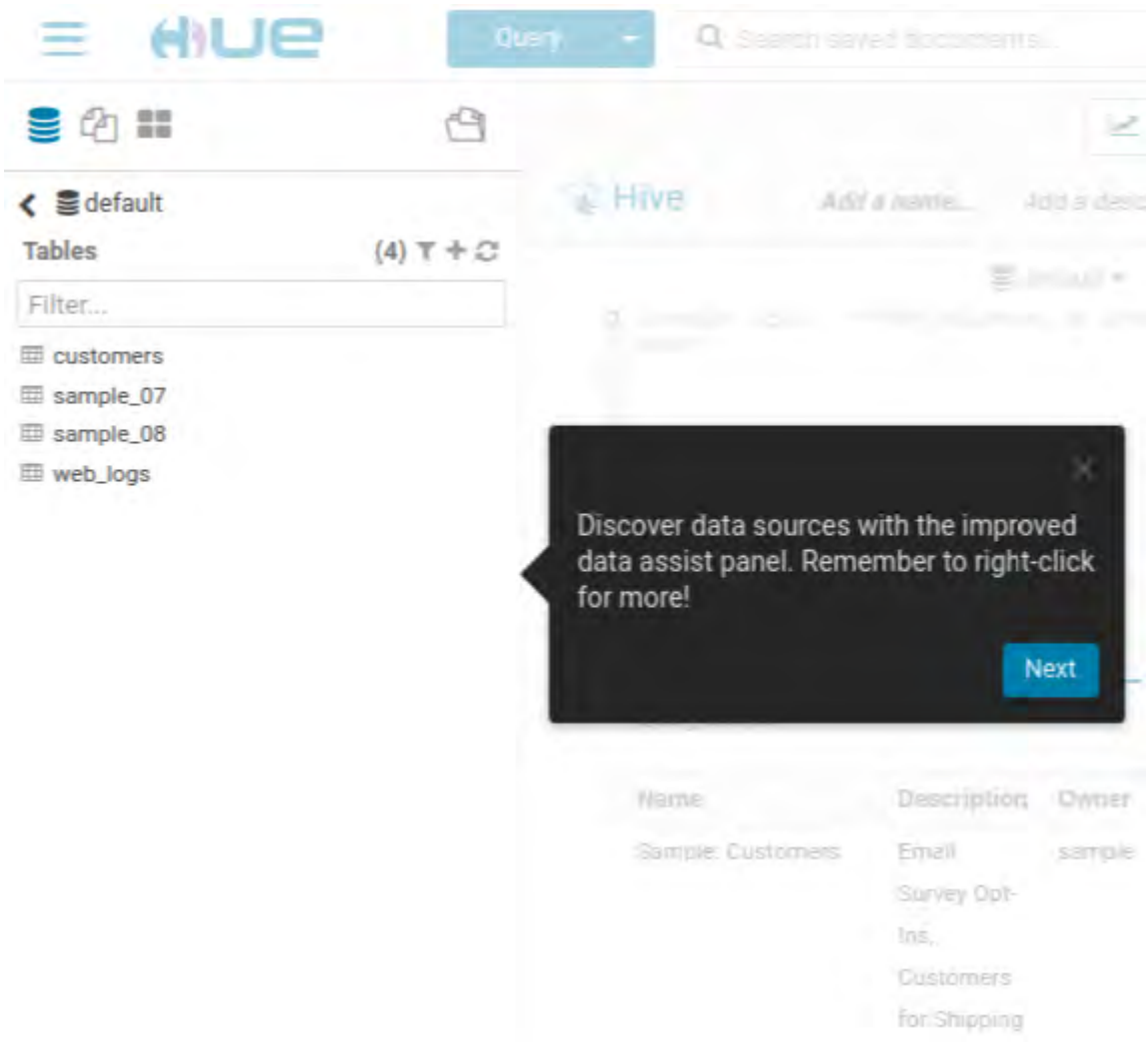
- Sign in with your username, enter the password `mapr` and click **Sign in**. You can find your username in the `/opt/mapr/conf/daemon.conf` file.

Using Hue Welcome Tour
After your first login, you will be introduced to the new features in Hue 4 by the "Welcome Hue 4!" tour.

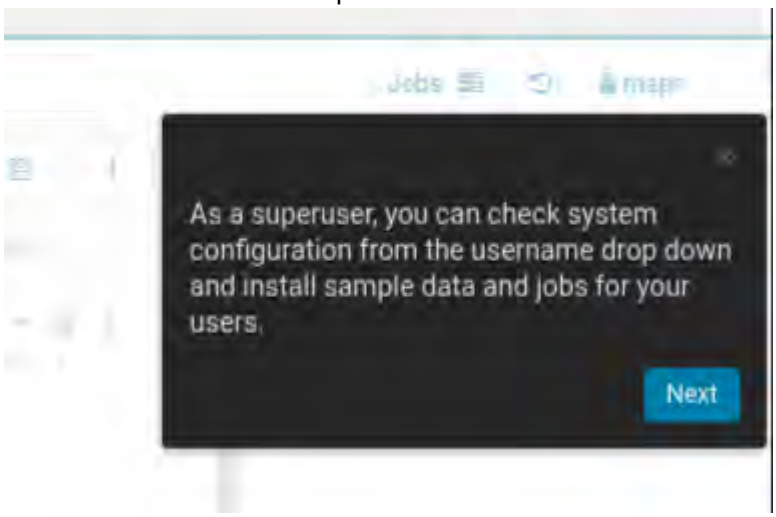
1. Start the welcome tour.
   The following screen appears:



2. You are first introduced to the navigation bar:



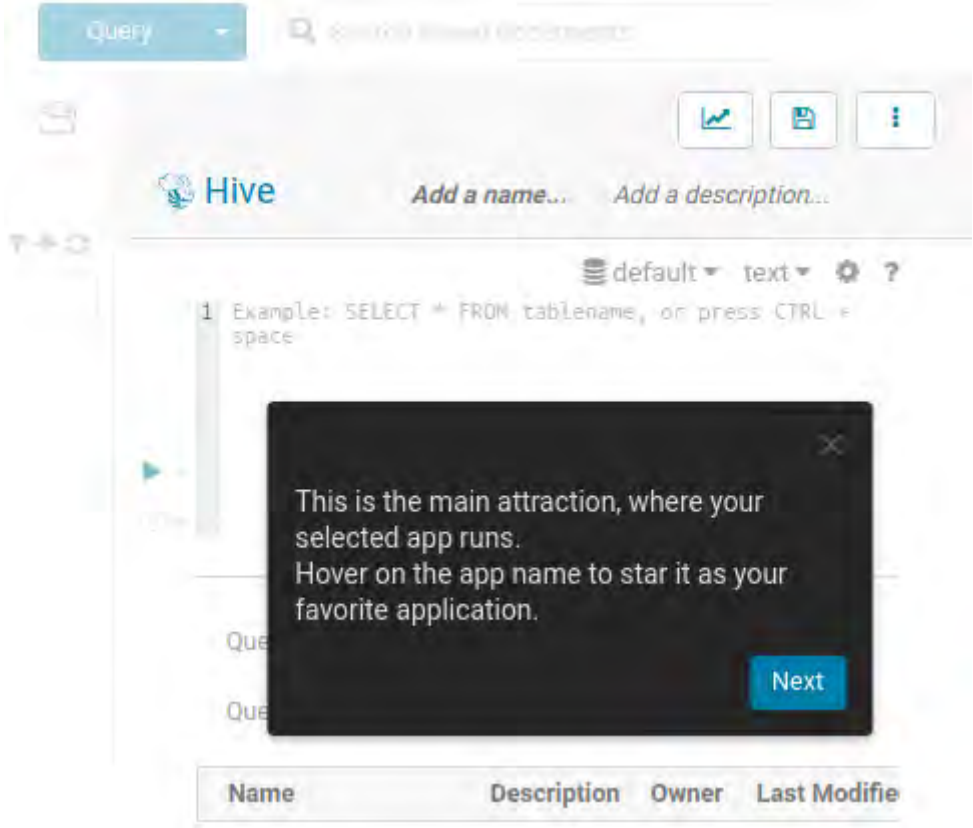
3. You are introduced to the superuser menu:



4. You are introduced to the left assist panel:

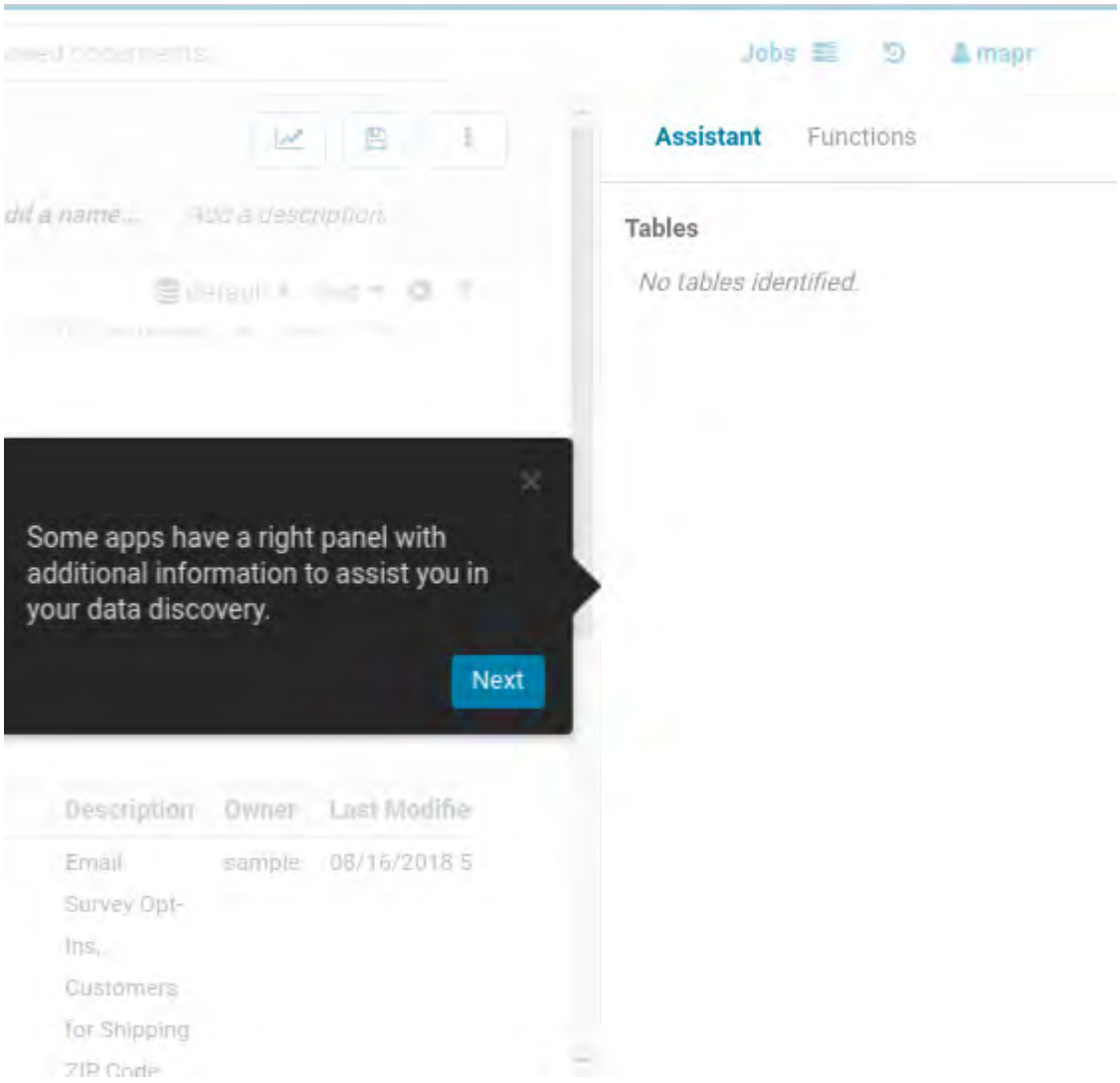
last-updated: Apr 11, 2022

5. You are introduced to the main working zone:

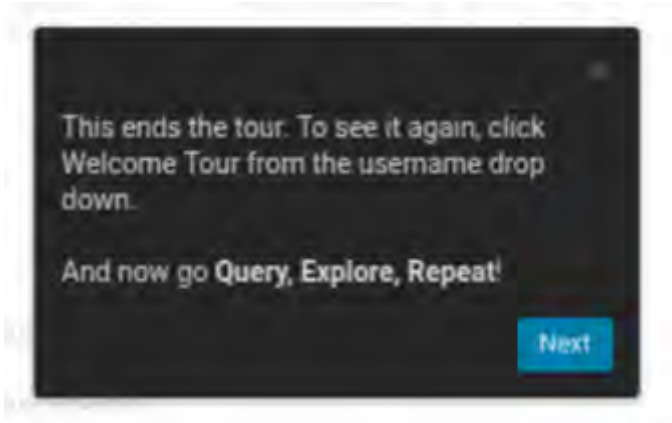


6. You are introduced to the right assist panel:

last-updated: Apr 11, 2022



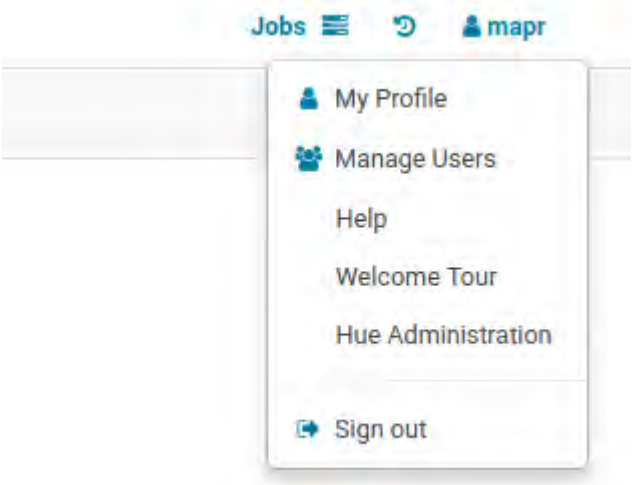
7. You then come to the end of the tour:



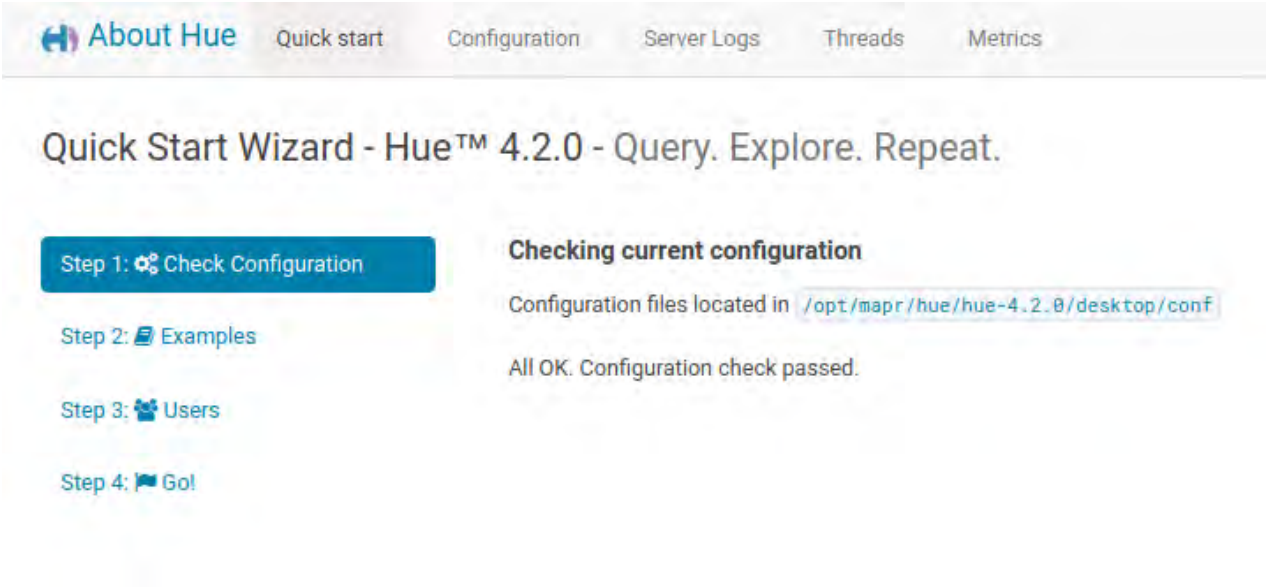
Getting Started with the Quick Start Wizard (Hue 4.X)
With the Quick Start Wizard you can check configuration, install examples, and create users.

Click on ***Hue Administration*** in the user menu:

last-updated: Apr 11, 2022

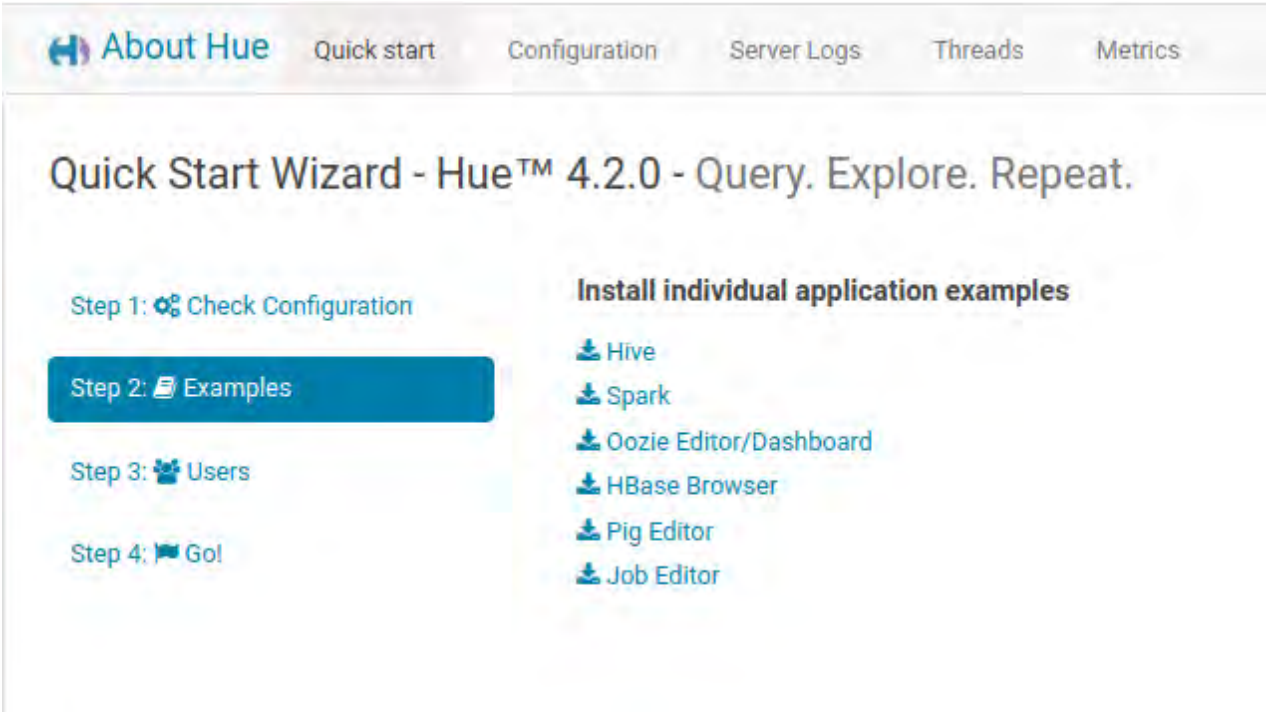


Step 1: Check Configuration

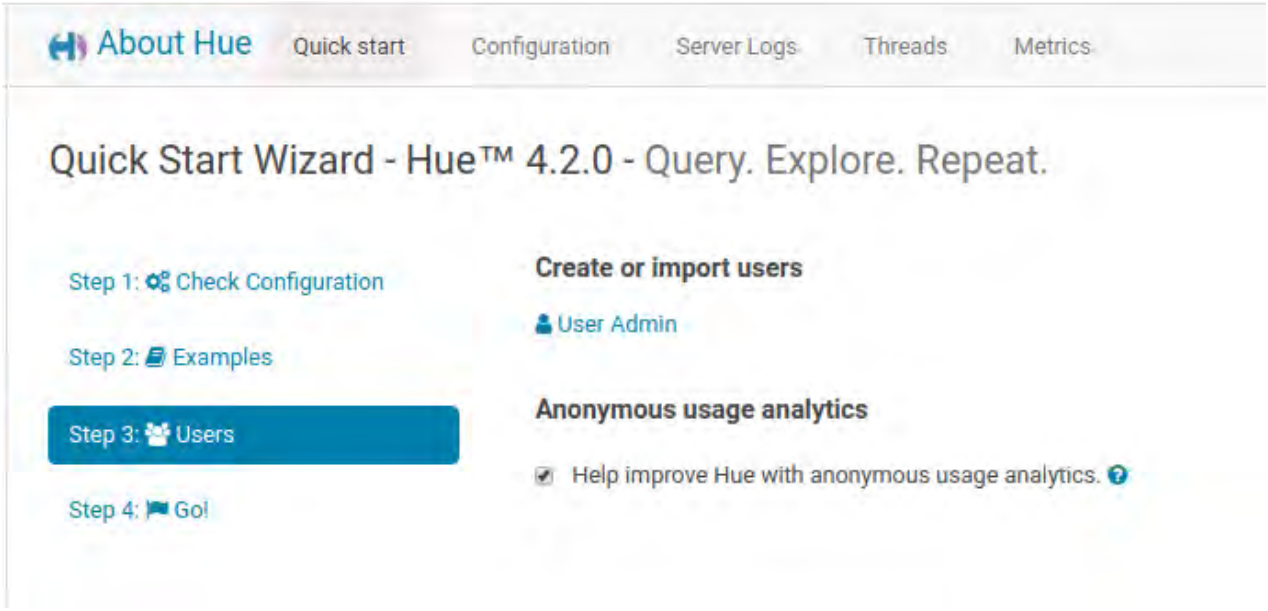


last-updated: Apr 11, 2022

Step 2: Examples



Step 3: Users



Note: By default, the Hue interface is configured to use PAM for authentication; so you cannot create or import users. For more information, see Configure Hue Interface Authentication.

User Administration in Hue 4.X

By default, the Hue interface is configured to use PAM for authentication; so you cannot use the Hue interface to create users or edit their passwords.

Note: For more information, see Configure Hue Interface Authentication.

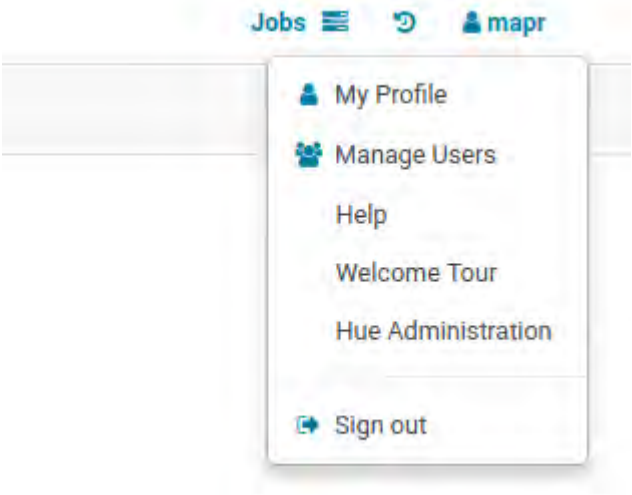
This section contains the following topics:

last-updated: Apr 11, 2022

Changing Your Password
Once you log in, you can change your Hue password.

To change your Hue password, perform the following steps:

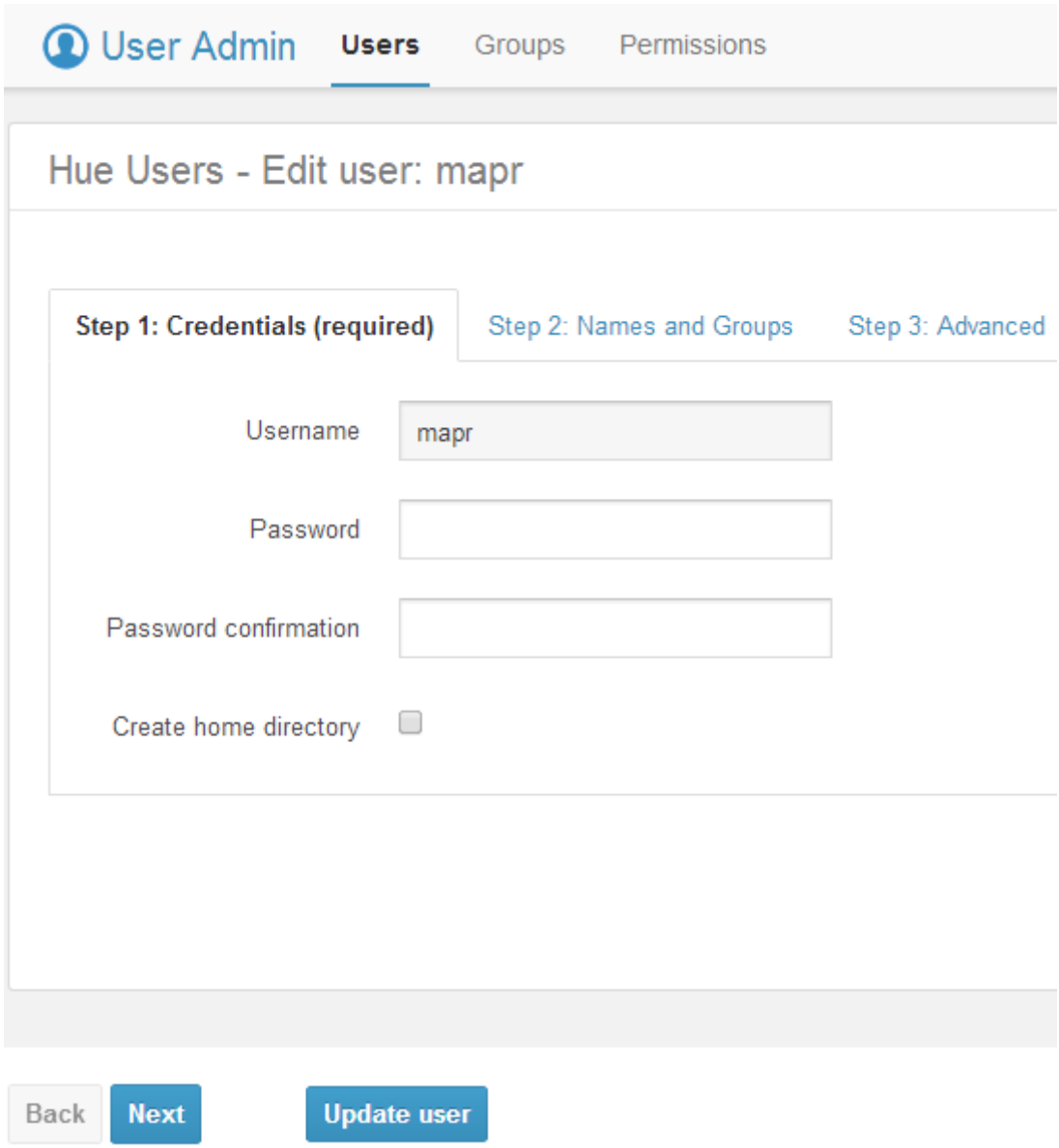
1. Click on your ***Username*** at the top right of the navigation menu bar and select **Edit Profile** from the drop-down menu. In this example, the username is **mapr**.



   The *Hue Users* dialog box opens.

2. Enter your current password in the *Current password* field and enter your new password in the *New Password* field. Retype the password in the *Password confirmation* field.

last-updated: Apr 11, 2022



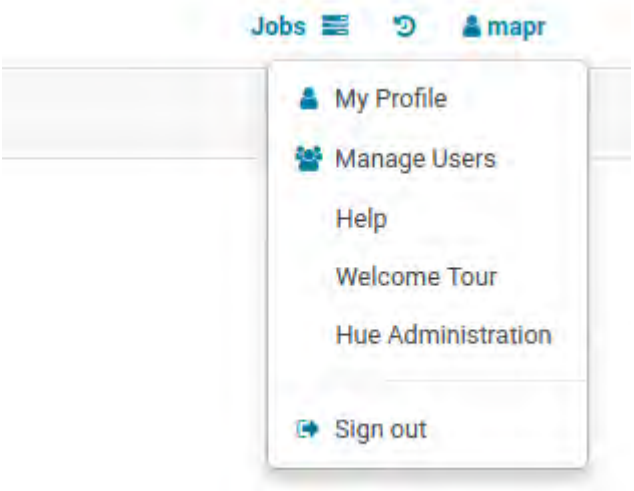
3. Click **Update user**.

Adding Users
When you click on your username, all users are displayed.

To create more users, follow these steps:

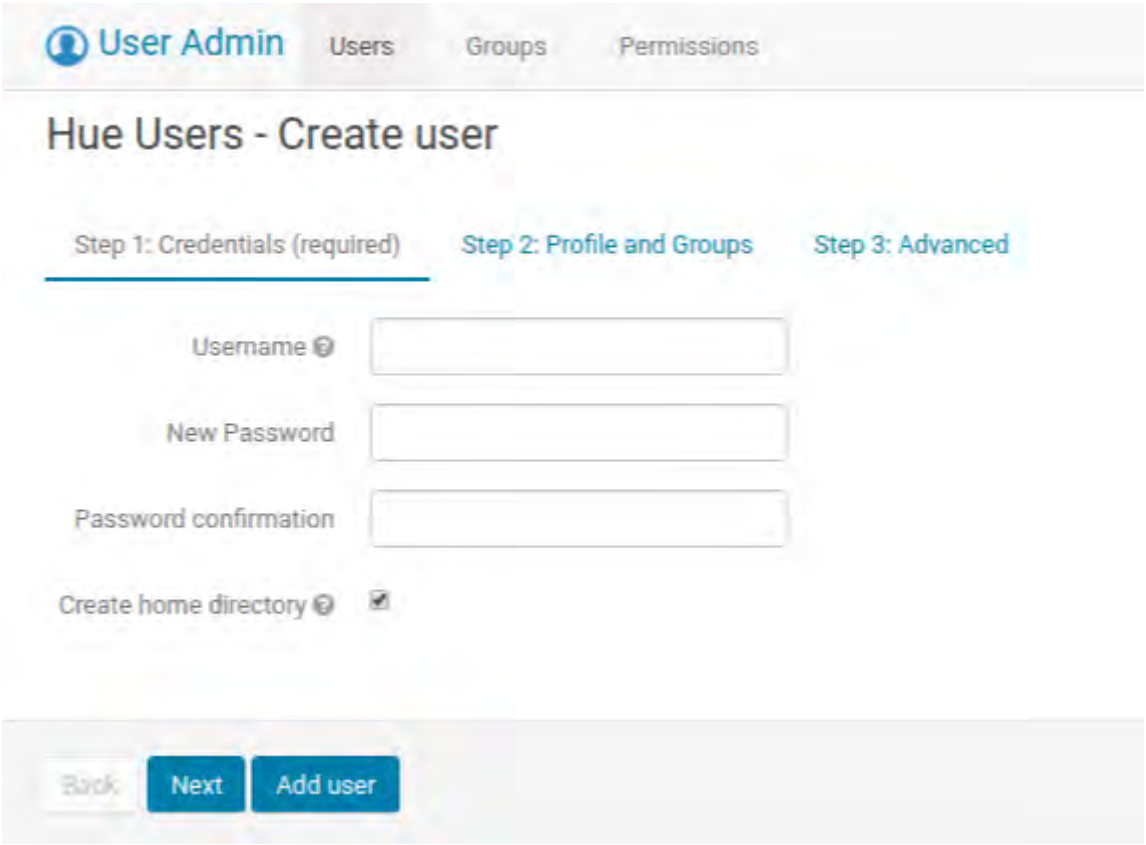
1. Click on your ***Username*** at the top right of the navigation menu bar and select **Manage Users** from the drop-down menu.

last-updated: Apr 11, 2022



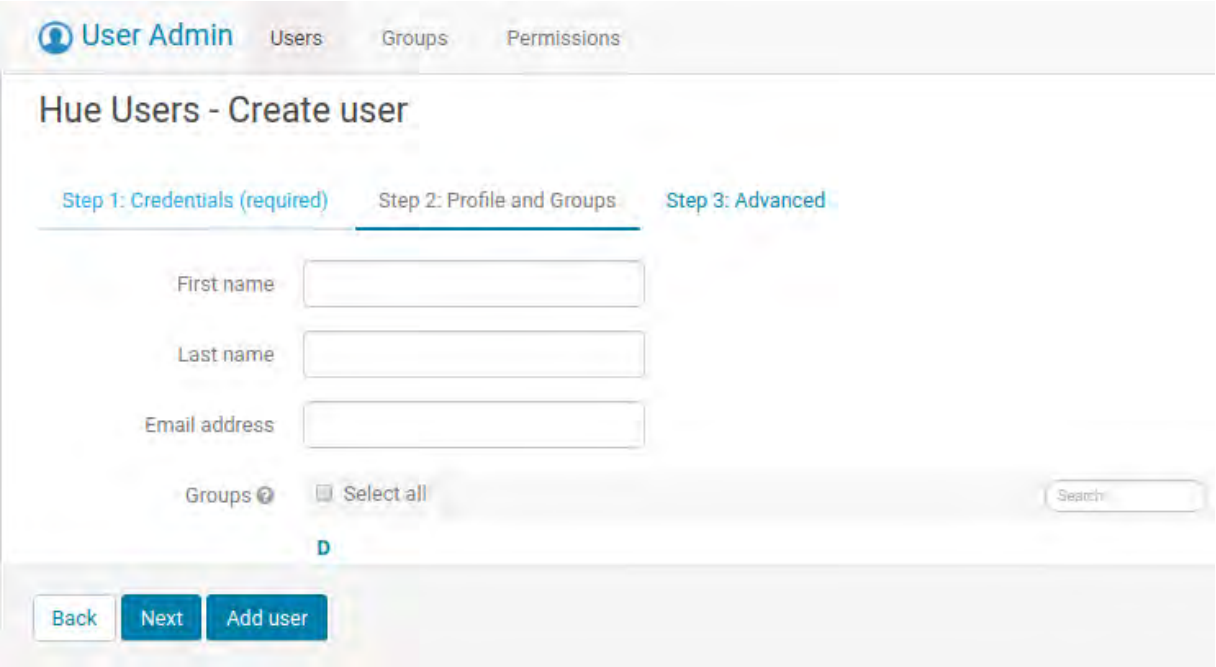
In this example, the username is **mapr**.

2. The *Hue Users - Create user* dialog box opens.

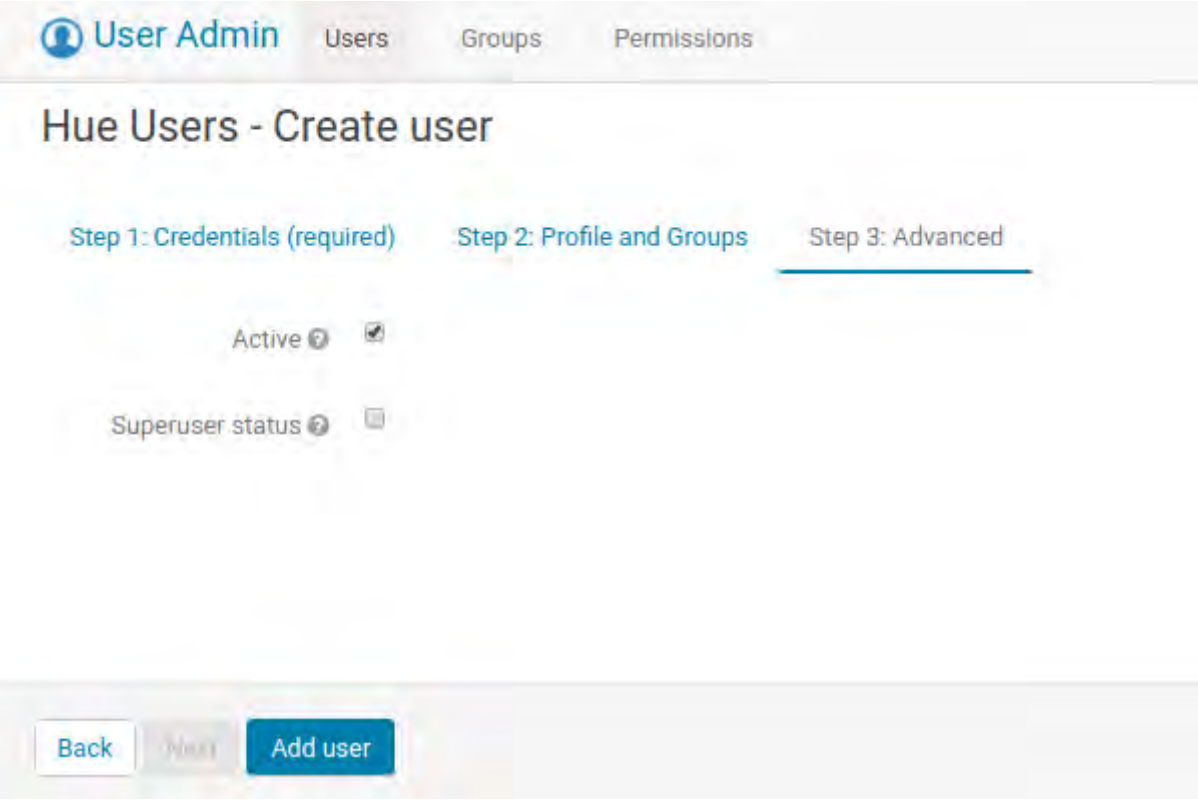


Click **Add user** to finish the process. If you want to assign superuser privileges to the user, click **Next** to proceed to the next screen.

3. Fill in the *Username*, *New Password*, and *Password confirmation* fields.
4. (Optional) Fill in the user's name and email address, and assign a group.



5. (Optional) Assign superuser privileges to the user that you just added by checking the *Superuser status* box.



6. Click **Add User**.

Managing MapR Database Binary Tables in Hue 4.X
You can create and manage MapR Database binary tables in the HBase Browser of the Hue interface.

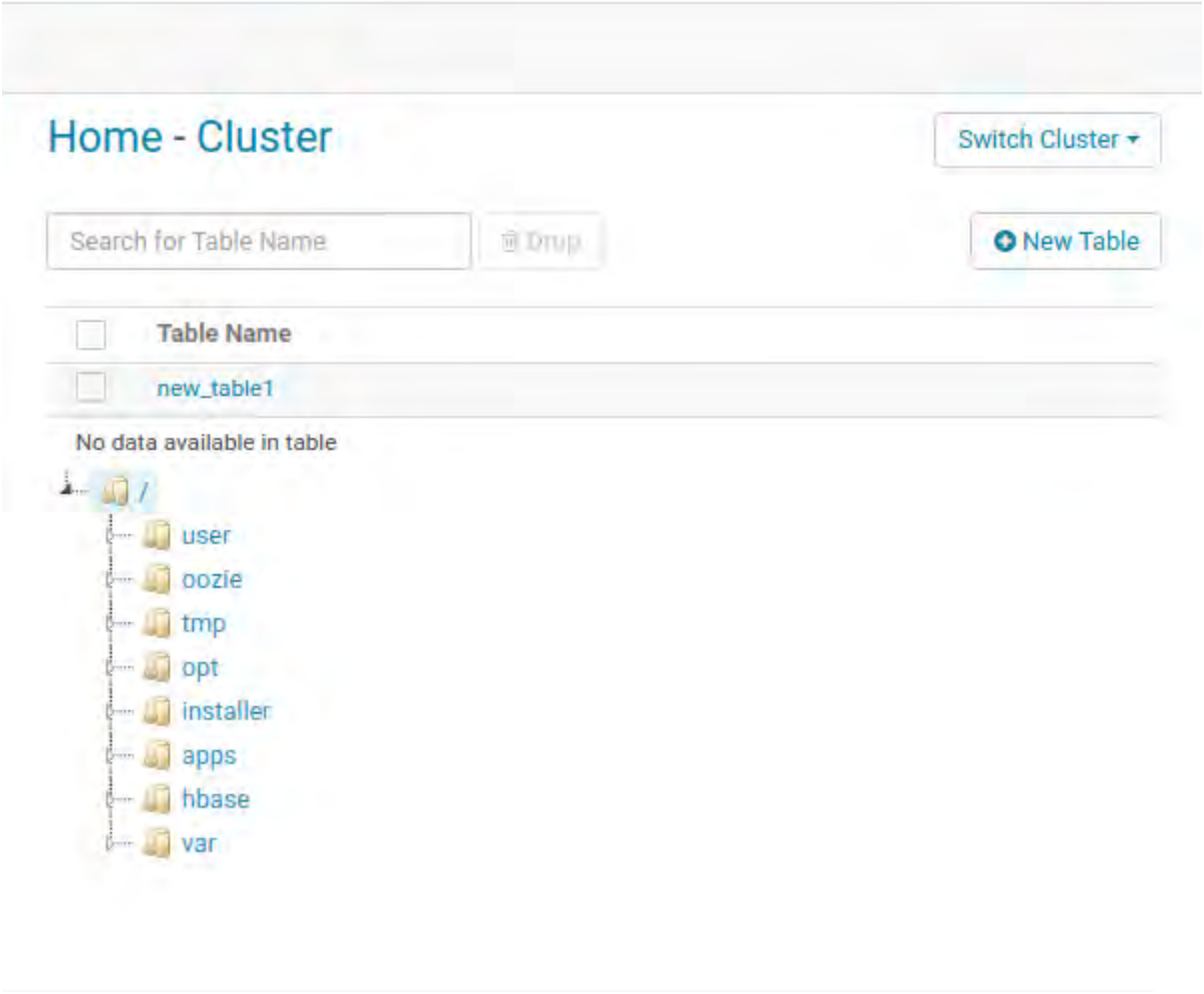
This section includes the following sections:

last-updated: Apr 11, 2022

Using the Hbase Browser
When you open the Hbase Browser, you can view the all the directories and MapR Database binary tables available in MapR Filesystem.

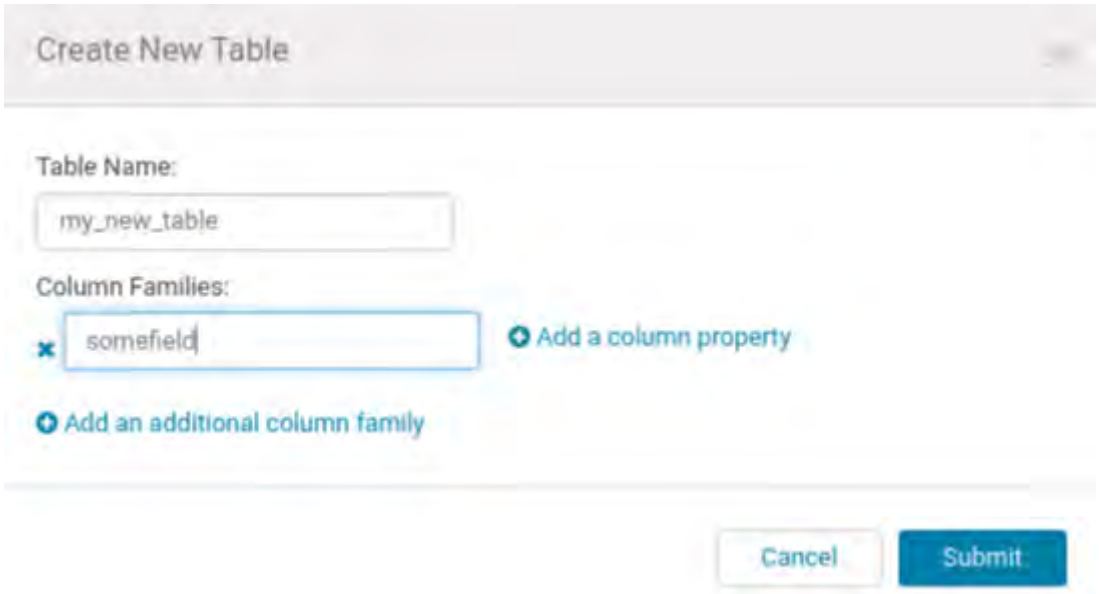
You can use the HBase Browser to create, edit, and search for MapR Database binary tables. However, you cannot enable, disable, or drop MapR Database binary tables.

Note: The browser also lists MapR Database JSON tables. However, their appearance is not different from that of MapR Database binary tables. You cannot edit JSON tables.

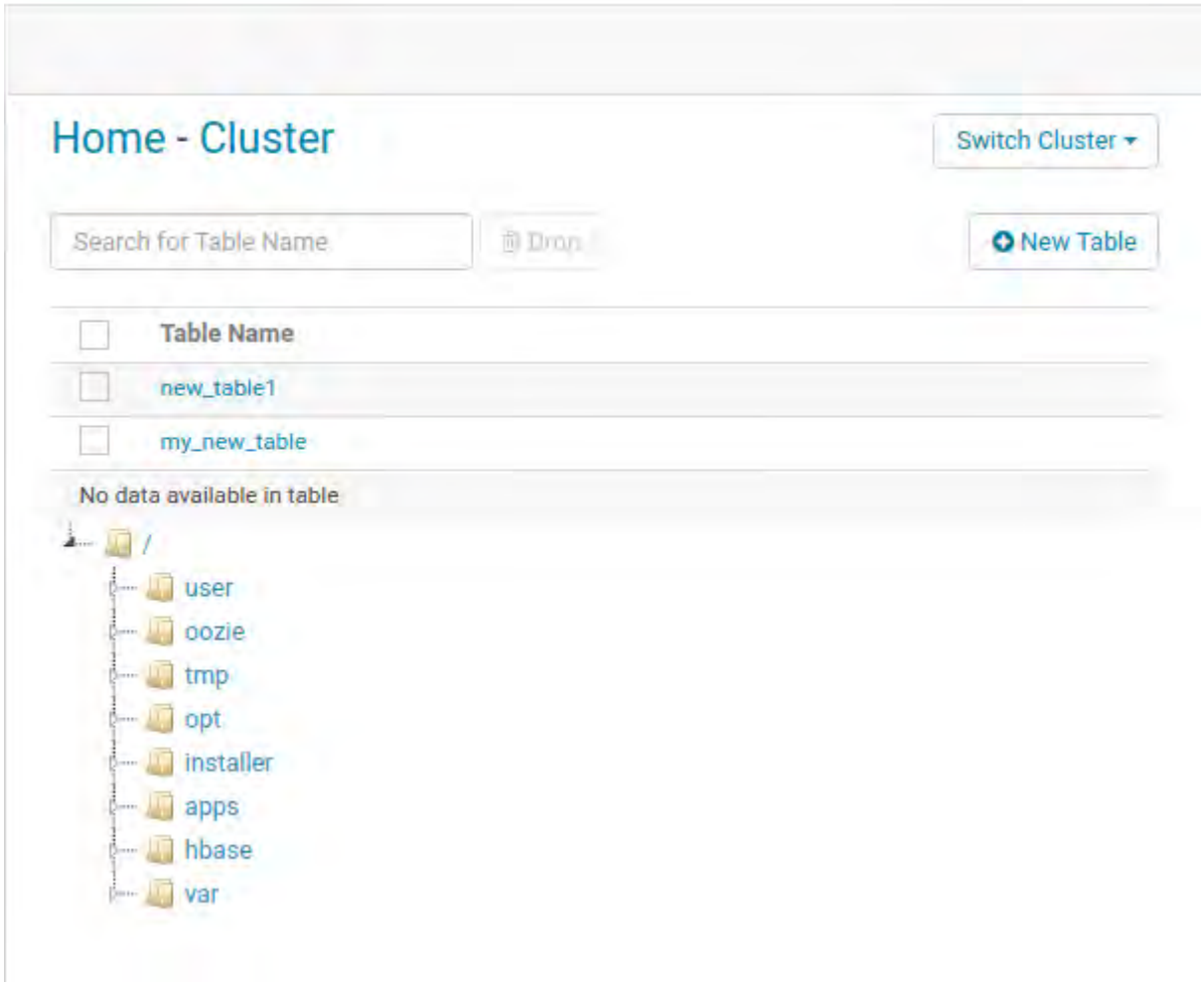


Creating a MapR Database Binary Table
You can create a new MapR Database Binary Table.

1. In the Hbase Browser, click **New Table**.
2. In the **Table Name** field, provide the full name of the table that you want to create. For example, `my_new_table`.



3. In the *Column Families* field, you can add column families and column properties.
4. Click **Submit.** The table that you created appears in the Hbase Browser:



last-updated: Apr 11, 2022

Starting the Hue Webserver
After you configure the `hue.ini`, you need to start the Hue Webserver and verify that it has started.

1. To start/restart the Hue Webserver, run the following command:
   - If Hue is installed on a cluster node (the common use case and recommended practice), run the following command to start the Hue webserver:

     ```
     maprcli node services -name hue -action start -nodes <ip_address>
     ```

   - If Hue is installed on a cluster node (the common use case and recommended practice), run the following command to restart the Hue webserver:

     ```
     maprcli node services -name hue -action restart -nodes <ip_address>
     ```

   - If Hue is installed on an edge node (not recommended), run the following command to start the Hue webserver:

     ```
     /opt/mapr/hue/hue-<version>/bin/hue-server start
     ```

2. To verify that the Hue webserver started, enter: `lsof -i:8888`

   The output from this command should look similar to this:

   ```
   COMMAND     PID USER   FD   TYPE   DEVICE SIZE/OFF NODE NAME
   python2.6 27688 mapr    3u  IPv4 69955314      0t0  TCP *:ddi-tcp-1
   (LISTEN)
   python2.6 27691 mapr    3u  IPv4 69955314      0t0  TCP *:ddi-tcp-1
   (LISTEN)
   ```

   You can also check Hue webserver logs to verify that Hue webserver started. If the Hue webserver was installed on a *cluster* node or an *edge* node, the log is found here:

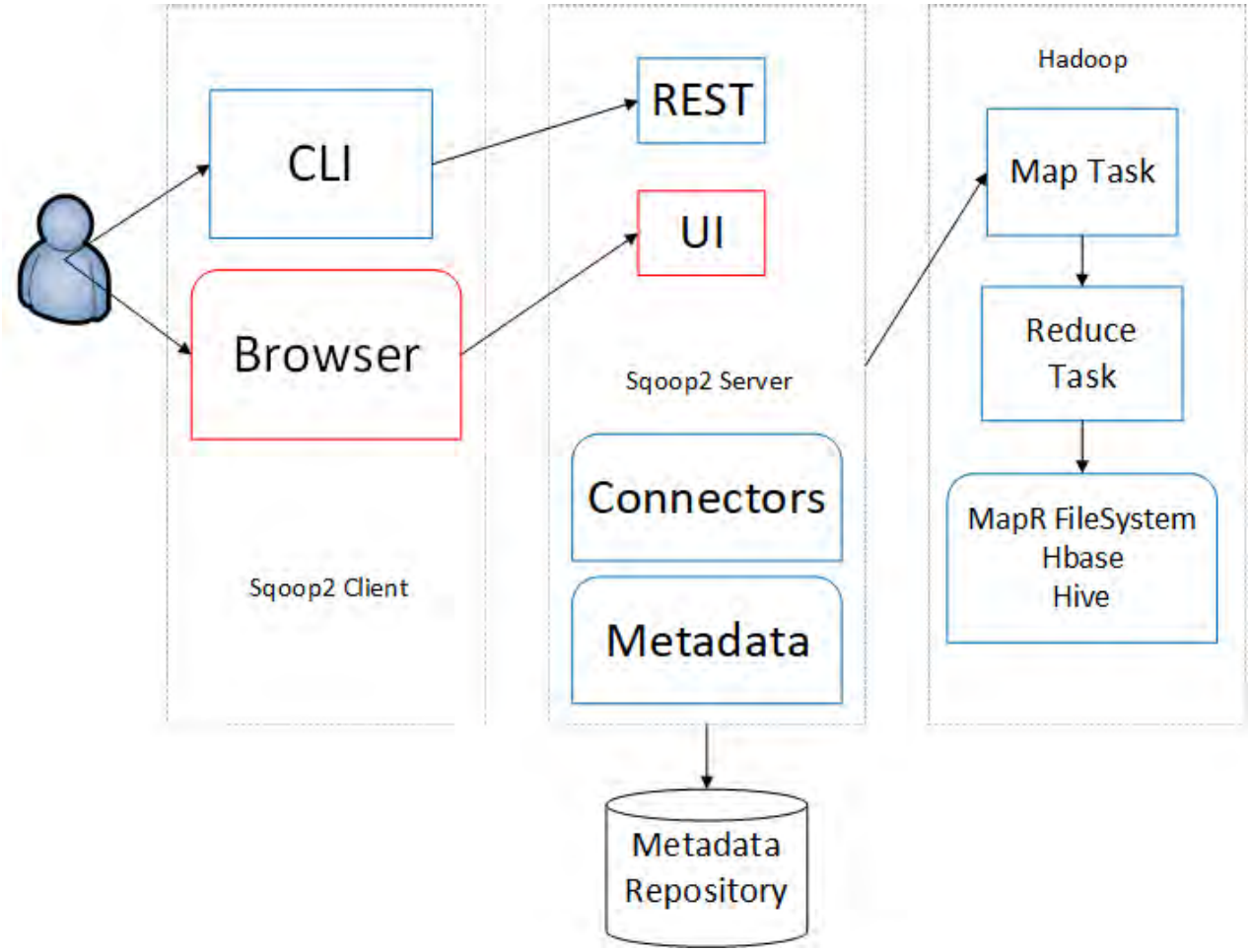
   `/opt/mapr/hue/hue-<version>/logs/runcpserver.log`

Sqoop2

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Sqoop2 transfers bulk data between Hadoop and various types of structured datastores, such as relational databases, enterprise data warehouses, and NoSQL systems. Sqoop2 is installed and configured server-side. When you install the Sqoop2 server package, the Sqoop2 client package is installed as well.

Sqoop2 can be used in conjunction with Hue, which is a web-based GUI that facilitates the importing and exporting of data from these structured datastores. This illustration shows how a user can access Sqoop2 from a browser, such as Hue.



This documentation does not duplicate Apache documentation. You can also refer to documentation available from the Apache Sqoop website.

Note: Sqoop2 does not officially support HA.

Sqoop2 Connector Support
Connector support differs for each version of Sqoop2.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Connector Name	Details
ftp-connector	Connects to FTP servers. **Supported Version:** 1.99.7 **Supported Direction:** To
generic-jdbc-connector	Connects to any data source that adheres to the JDBC 4 specification. **Supported Version:** 1.99.6 and 1.99.7 **Supported Direction:** To and From
hdfs-connector	Connects to MapR File System. **Supported Version:** 1.99.6 and 1.99.7

last-updated: Apr 11, 2022

Connector Name	Details
	Supported Direction: To and From **Additional Information:** • As of 1.99.7, the Parquet data format is supported. The sequence and text file formats continue to be supported.
kafka-connector	Not Supported
kite-connector	Connects to MapR File System and Hive Metastore. **Supported Versions** 1.99.7 **Supported Direction:** To and From **Additional Information:** • The kite connector cannot be used to connect to Hive on secure clusters. • Csv and avro formats are not supported. The parquet data format is supported.
oracle-jdbc-connector	Connects to Oracle databases. **Supported Versions** 1.99.6 **Supported Direction:** To and From
sftp-connector	Connects to Secure File Transfer Protocol (SFTP) servers. **Supported Version:** 1.99.7 **Supported Direction:** To

Configuring Sqoop2

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

This section provides information about configuring Sqoop2, but it does not duplicate Apache documentation.

Starting with the EEP 4.0 release, for secure clusters, Sqoop2 is configured to enable MapR-SASL and SSL. Prior to the EEP 4.0 release, by default, Sqoop2 is not configured with any type of security or encryption.

For additional configuration details, see the Apache Sqoop website.

Configure Kerberos Authentication for Sqoop2
You can configure Sqoop2 to use Kerberos authentication. When Sqoop2 uses Kerberos authentication, the cluster and other components that work with Sqoop2, such as Hue, must also use Kerberos authentication.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Note the following items when you complete the configuration steps:

- Replace `<FQDN>` with the FQDN of the server. To determine this value, run `hostname -f` in the command line.

last-updated: Apr 11, 2022

- Replace <REALM> with the realm name in the `krb5.conf` file, which is generated when you install the KDC server on the cluster.

Configuring Kerberos Authentication for Sqoop2:

1. Using the `kadmin` program, run the following commands to create principals for Sqoop 2:

   ```
   addprinc -randkey HTTP/<FQDN>@<REALM>
   addprinc -randkey mapr/<FQDN>@<REALM>
   ```

   Kerberos uses the principal `HTTP/<FQDN>@<REALM>` for communication between Sqoop2 client and Sqoop2 server. The principal `mapr/<FQDN>@<REALM>` is the Sqoop2 user that communicates between Sqoop2 server and MapR File System.

2. Using the `kadmin` program, run the following commands to create keytabs for the principals:

   ```
   xst -k /opt/mapr/conf/mapr.keytab HTTP/<FQDN>@<REALM>
   xst -k /opt/mapr/conf/mapr.keytab mapr/<FQDN>@<REALM>
   ```

3. Modify the following properties in Sqoop2 configuration file (sqoop.properties).

   In Sqoop 1.99.6, the sqoop.properties file is in the following directory: `/opt/mapr/sqoop/sqoop-<version>/server/conf/`. In Sqoop 1.99.7, the sqoop.properties file is in the following directory: `/opt/mapr/sqoop/sqoop-<version>/conf/`.

   ```
   org.apache.sqoop.security.authentication.type=KERBEROS
   org.apache.sqoop.security.authentication.handler=org.apache.sqoop.securit
   y.authentication.KerberosAuthenticationHandler
   org.apache.sqoop.security.authentication.kerberos.principal=mapr/
   <FQDN>@<REALM>
   org.apache.sqoop.security.authentication.kerberos.keytab=/opt/mapr/conf/
   mapr.keytab
   org.apache.sqoop.security.authentication.kerberos.http.principal=HTTP/
   <FQDN>@<REALM>
   org.apache.sqoop.security.authentication.kerberos.http.keytab=/opt/mapr/
   conf/mapr.keytab
   org.apache.sqoop.security.authentication.enable.doAs=true
   org.apache.sqoop.security.authentication.proxyuser.mapr.users=*
   ```

4. Start Sqoop2 server.

   ```
   maprcli node services -name sqoop2 -action start -nodes <space delimited
   list of nodes>
   ```

5. Using the `kinit` program, run the following command to generate a ticket:

   ```
   kinit HTTP/<FQDN>@<REALM> -kt /opt/mapr/conf/mapr.keytab
   ```

6. Start the Sqoop2 client.

   ```
   sudo -u mapr /opt/mapr/sqoop/sqoop-<version>/bin/sqoop.sh client
   ```

Configure MapR-SASL for Sqoop2

This section describes how to configure Sqoop2 to use MapR-SASL. Starting with the EEP 4.0 release, for secure clusters, MapR-SASL is automatically configured for Sqoop2 and you can skip the steps outlined in this section.

last-updated: Apr 11, 2022

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

When you configure MapR-SASL for Sqoop2, other components that communicate with Sqoop, such as Hue, must also be configured to use MapR-SASL. Complete the following steps on each node that runs the Sqoop2 server:

1. Remove the # before the following properties in the Authentication configuration section of the `sqoop.properties` file.

   In Sqoop 1.99.6, the sqoop.properties file is in the following directory: `/opt/mapr/sqoop/sqoop-<version>/server/conf/`. In Sqoop 1.99.7, the sqoop.properties file is in the following directory: `/opt/mapr/sqoop/sqoop-<version>/conf/`.

   ```
   #org.apache.sqoop.security.authentication.type=CUSTOM
   #org.apache.sqoop.security.authentication.custom_handler=org.apache.hadoop.security.authentication.server.MultiMechsAuthenticationHandler
   ```

2. Restart Sqoop2 server.

   ```
   maprcli node services -name sqoop2 -action restart -nodes <space delimited list of nodes>
   ```

Note: To start Sqoop2 client with MapR-SASL, run the following command:

```
sudo -u mapr /opt/mapr/sqoop/sqoop-<version>/bin/sqoop.sh client --custom
```

Configure SSL for Sqoop2

As of Sqoop 1.99.7, you can configure SSL to enable encrypted communications between the Sqoop2 server and its clients. Starting with the EEP 4.0 release, for secure clusters, SSL is automatically configured for Sqoop2 and you can skip the steps outlined in this section.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

1. Stop the Sqoop2 server:

   ```
   maprcli node services -name sqoop2 -action stop -nodes <space delimited list of nodes>
   ```

2. In the sqoop.properties file (/opt/mapr/sqoop/sqoop-<version>/conf/sqoop.properties), uncomment the SSL related properties. For example:

   ```
   #Enable Sqoop SSL
   org.apache.sqoop.security.tls.enabled=true
   #Change SSL protocol
   org.apache.sqoop.security.tls.protocol=TLSv1.2
   #Path to MapR ssk keystore
   org.apache.sqoop.security.tls.keystore=/opt/mapr/conf/ssl_keystore
   #Keystore password
   org.apache.sqoop.security.tls.keystore_password=<passwd>
   ```

   Note: You can use the default ssl_keystore and password. The password for the default ssl_keystore is `<ssl-keystore-password>`.

last-updated: Apr 11, 2022

3. Remove the Sqoop2 repository.

```
rm -rf /opt/mapr/sqoop/repository
```

4. Start the Sqoop2 server.

```
maprcli node services -name sqoop2 -action start -nodes <space delimited
list of nodes>
```

5. Start the Sqoop2 client:

```
sudo -u mapr /opt/mapr/sqoop/sqoop-<version>/bin/sqoop.sh client
```

Note: If you are using MapR-SASL, run the following command instead: `sudo -u mapr /opt/mapr/sqoop/sqoop-<version>/bin/sqoop.sh client --custom`.

6. Configure the Sqoop2 client to communicate the Sqoop2 server using SSL.

```
set server --host <sqoop_server_hosname> --port <sqoop_port> --webapp
<sqoop_webapp> --tls
```

For example:

```
sqoop:000> set server --host localhost --port 12000 --webapp sqoop --tls
```

7. Configure the Sqoop2 client truststore and truststore password.

```
set truststore --truststore /opt/mapr/conf/
ssl_truststore --truststore-password <passwd>
```

Each time you start the Sqoop2 client, you must reset the server and truststore configuration.

Configure Repository Encryption for Sqoop2

As of Sqoop 1.99.7, you can configure the Sqoop2 repository to encrypt password data.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

1. Stop the Sqoop2 server.

```
maprcli node services -name sqoop2 -action stop -nodes <space delimited
list of nodes>
```


last-updated: Apr 11, 2022

2. In the sqoop.properties file (`/opt/mapr/sqoop/sqoop-<version>/conf/sqoop.properties`), configure the repository encryption related properties. For example:

```
org.apache.sqoop.security.repo_encryption.enabled=true
org.apache.sqoop.security.repo_encryption.password=<ssl-keystore-password
>
org.apache.sqoop.security.repo_encryption.hmac_algorithm=HmacSHA256
org.apache.sqoop.security.repo_encryption.cipher_algorithm=AES
org.apache.sqoop.security.repo_encryption.cipher_key_size=16
org.apache.sqoop.security.repo_encryption.cipher_spec=AES/CBC/
PKCS5Padding
org.apache.sqoop.security.repo_encryption.initialization_vector_size=16
org.apache.sqoop.security.repo_encryption.pbkdf2_algorithm=PBKDF2WithHmac
SHA1
org.apache.sqoop.security.repo_encryption.pbkdf2_rounds=4000
```

3. Remove the Sqoop2 repository.

```
rm -rf /opt/mapr/sqoop/repository
```

4. Start the Sqoop2 server.

```
maprcli node services -name sqoop2 -action start -nodes <space delimited
list of nodes>
```

Using Sqoop2

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

This documentation provides all relevant details about using Sqoop2, but does not duplicate Apache documentation.

For more information, see the Apache Sqoop website.

Manage Sqoop2 Server and Client

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Starting the Sqoop2 Command Line Client

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The Sqoop install directory (`<SQOOP_INSTALL_DIR>`) is of the form `/opt/mapr/sqoop/sqoop-<version_number>`.

- To start the sqoop2 command line client without MapR-SASL authentication, enter:

```
$ cd <SQOOP_INSTALL_DIR>
$ ./bin/sqoop.sh client
```

When Sqoop2 does not use MapR-SASL, any user that has access to run jobs on the cluster can start the Sqoop2 client.

- To start the Sqoop2 command line client with MapR-SASL authentication, enter:

```
$ cd <SQOOP_INSTALL_DIR>
$ ./bin/sqoop.sh client --custom
```

When Sqoop2 uses MapR-SASL, any user with a MapR ticket can start the Sqoop2 client.

Starting and Stopping the Sqoop2 Server

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

To start the Sqoop2 server on a node called `node001`, enter the following command at the command line:

```
maprcli node services -name sqoop2 -action start -nodes node001
```

To stop the Sqoop2 server, enter the following command at the command line:

```
maprcli node services -name sqoop2 -action stop -nodes node001
```

Verifying Server Status

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The Sqoop2 server uses port 12000. To verify server status, enter:

```
lsof -i:12000
```

The output of this command should look similar to this:

```
COMMAND   PID USER   FD   TYPE  DEVICE SIZE/OFF NODE NAME
java    14444 mapr  103u  IPv6 2795065      0t0  TCP *:12000 (LISTEN)

Logs:
/opt/mapr/sqoop/sqoop-2.0.0/logs/sqoop.log
/opt/mapr/sqoop/sqoop-2.0.0/logs/server/logs/catalina.out
```

Create Links and Jobs

Use connectors to create links to data sources and then create jobs to execute the data transfer.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

1. Create a link to each data source using the connector name.

```
sqoop:000>  create link --connector <connector-name>
```

For example, to create a link with the `kite-connector`:

```
sqoop:000>  create link --connector kite-connector
```

2. Use link names to create a job.

```
sqoop:000>  create job --from <link_name> --to <link_name>
```


last-updated: Apr 11, 2022

For more information about using connectors, see the Sqoop2 connector documentation for Sqoop 1.99.6 and Sqoop 1.99.7.

Note: Prior to using an FTP or an SFTP connector, verify that the associated FTP or SFTP server is configured.

Example Kite Connector Link and Job Configuration Data

The following example shows the types of prompts that you can expect when setting up links and jobs. It also provides tips that may be helpful when you configure links and jobs with the Kite Connector.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Kite Connector Link Configuration for MapR File System

```
sqoop:000> create link --connector
kite-connector
Name: kite_maprfs

Global configuration
MAPRFS or Hive Metastore URI:
maprfs:///
Hadoop conf directory: /opt/mapr/
hadoop/hadoop-2.7.0/etc/hadoop/
```

Note: Set the Hadoop conf directory to /opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/.

Kite Connector Job Configuration for MapR File System

```
Dataset URI: dataset:maprfs:///
<somefolder>
File format:
  0 : CSV
  1 : AVRO
  2 : PARQUET
Choose: 2
```

Kite Connector Link Configuration for Hive

```
sqoop:000> create link --connector
kite-connector
Name: kite_hive

Global configuration
MAPRFS or Hive Metastore URI:
<hostname>:9083
Hadoop conf directory: /opt/mapr/hive/
hive-1.2/conf/
```

Note: Set the Hadoop conf directory to the Hive conf directory.

Kite Connector Job Configuration for Hive

```
Dataset URI:
dataset:hive:databasename/tablename
File format:
  0 : CSV
  1 : AVRO
  2 : PARQUET
Choose: 2
```


last-updated: Apr 11, 2022

Submit Sqoop2 Jobs

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Use one of the following methods to submit Sqoop2 jobs:

- Submit jobs with the Sqoop2 client. See the Sqoop2 documentation.
- Submit jobs from Hue. See the Hue documentation. See the Hue documentation.

Sqoop 1.99.7 API Changes
The following APIs have been deprecated in Sqoop 1.99.7.

Instead of this deprecated API...	Use...
/v1/connectors - [GET]	/v1/connector/all - [GET]
/v1/connector/[cid] - [GET]	/v1/connector/[cname] - [GET]
/v1/links/ - [GET]	/v1/link/all - [GET]
/v1/link/[lid] - [PUT]	/v1/link/[lname] - [PUT]
/v1/link/[lid] - [DELETE]	/v1/link/[lname] - [DELETE]
/v1/link/[lid]/enable - [PUT]	/v1/link/[lname]/enable - [PUT]
/v1/link/[lid]/disable - [PUT]	/v1/link/[lname]/disable - [PUT]
/v1/jobs/ - [GET]	/v1/job/all - [GET]
/v1/job/[jid] - [GET]	/v1/job/[jname] - [GET]
/v1/job/[jid] - [PUT]	/v1/job/[jname] - [PUT]
/v1/job/[jid] - [DELETE]	/v1/job/[jname] - [DELETE]
/v1/job/[jid]/enable - [PUT]	/v1/job/[jname]/enable - [PUT]
/v1/job/[jid]/disable - [PUT]	/v1/job/[jname]/disable - [PUT]
/v1/job/[jid]/start - [PUT]	/v1/job/[jname]/start - [PUT]
/v1/job/[jid]/stop - [PUT]	/v1/job/[jname]/stop - [PUT]
/v1/job/[jid]/status - [GET]	/v1/job/[jname]/status - [GET]

For more information, see the Sqoop 1.99.7 REST API documentation.

Impala

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Note:

SQL-on-Hadoop is an emerging space with several promising open-source technologies at various stages of maturity including Apache Drill, Apache Hive-on-Tez, Impala, Presto, Shark-on-Spark, Phoenix. MapR provides the broadest availability of these options to ensure that customers can pick the right tool for their use case.

MapR announces availability of Impala packages for MapR Converged Data Platform. Customers can easily install and run Impala on MapR clusters and evaluate its fit for their use cases. MapR will also provide build and installation support to help customers get started.

last-updated: Apr 11, 2022

Overview

Impala is an open source, interactive SQL engine for Hadoop that you can use to access data on MapR clusters. With Impala, you can use business intelligence (BI) tools to run ad-hoc queries directly on Hadoop. Impala provides the following benefits:

- Broad availability of Hadoop data to the business community
- Leverages familiar SQL tools and skill sets on Hadoop
- Full fidelity data analysis
- Cost savings by offloading from traditional DWH/Analytics platforms
- Time savings because you do not have to move around data

Impala uses the Apache Hive query language (HiveQL) and Hive metadata. You can use the most common SQL-92 features of HiveQL, including SELECT, joins, and aggregate functions to query data in your cluster. You can issue queries from the impala-shell command-line tool, or through an ODBC or JDBC client. When you submit a query, Impala works with other components in the cluster to process the query.

Impala uses the Hive metastore to store metadata. The Hive metastore is typically the same database that Hive uses to store metadata. Impala can access tables you create in Hive when they contain datatypes, file formats, and compression codecs that Impala supports.

If you have MapR Database configured to store table data, you can define the tables in Hive and then map them to equivalent tables in MapR Database. After you map the tables, you can use Impala to query the MapR Database tables and perform table joins between MapR Database and Hive tables.

The following table contains a list of components that work together to process a query issued to Impala:

Component	Description
Clients	The impala-shell, JDBC client, or ODBC client that you connect to Impala from. You issue a query to Impala from the client.
Hive Metastore	Stores information about the tables that Impala can access.
Impala (impalad, statestored, catalogd)	Impalad is a process that runs on designated nodes in the cluster. It coordinates and runs queries. Each node running the Impala process can receive, plan, and coordinate queries sent from a client. Statestored tracks the state of the Impalad processes in the cluster. Catalogd communicates new or updated metadata to all the other Impala nodes when Impala changes table data or metadata on a node.
MapR File System/MapR Database	MapR File System is the MapR filesystem that stores data files and tables. MapR Database stores MapR tables natively.

Impala Daemon

The Impala daemon (impalad) is the core Impala process. Install the impala server package on all nodes that you want to run the Impala service on. You can submit queries to any node configured with the Impala daemon. When you submit a query to a node running the Impala daemon process, the node becomes the central coordinator for the query. The coordinator distributes work to the other nodes running the Impala daemon process to complete the query request.

The coordinator node running the Impala daemon process completes the following tasks:

last-updated: Apr 11, 2022

- Accepts the query from the client
- Parallelizes queries and distributes pieces of the query to other impalad nodes
- Receives query results from other impalad nodes
- Builds the final result set for the query and passes it to the client
- Communicates with the statestore service

All nodes running the Impala daemon process perform the following tasks:

- Read and write data to files
- Transmit partial query results back to the coordinator
- Communicate with the statestore service

Impala Statestore

The Impala statestore service (statestored) verifies the health of nodes running the Impala daemon process. The node running the statestore service communicates node health to all nodes running the Impala daemon process. The statestore service ensures that impalad nodes only pass query work to active nodes running the Impala daemon process.

If the node running the statestore service goes offline, other Impala nodes continue the query work. If any of the other Impala nodes go offline while the statestore service is unavailable, query time may increase. The statestore service can automatically communicate with the Impala nodes and resume work when the node is back online.

Catalog Service

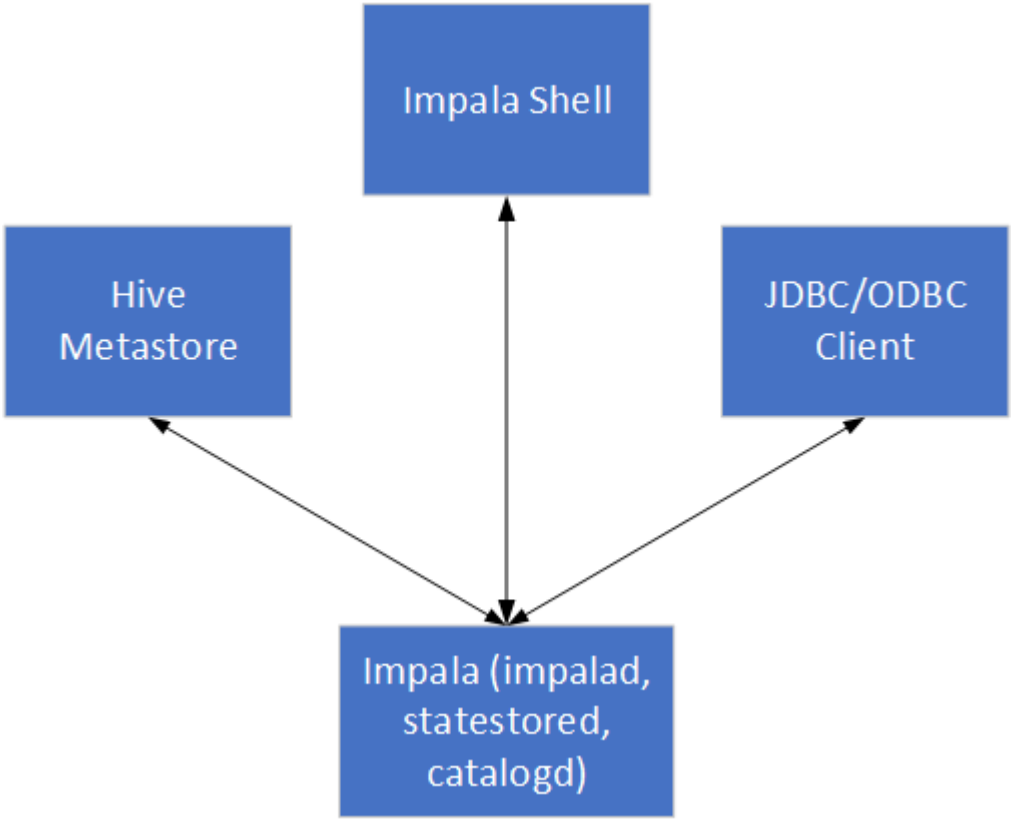
The catalog service (catalogd) resides on the same node as statestored. The catalog service communicates new or updated metadata to all of the other Impala nodes when Impala changes table data or metadata on one node. You no longer need to use the `REFRESH` or `INVALIDATE METADATA` statements after issuing the following statements to Impala:

- `CREATE TABLE`
- `ALTER TABLE`
- `DROP TABLE`
- `INSERT`
- `LOAD DATA`

The catalog service only works on operations performed through Impala. If you perform operations through the Hive shell or through MapR File System, you must issue the `REFRESH` and `INVALIDATE METADATA` statements. The catalog service broadcasts the results of the `REFRESH` and `INVALIDATE METADATA` operations to other Impala nodes so that you only have to issue the statements once.

The following image represents how the various components communicate:

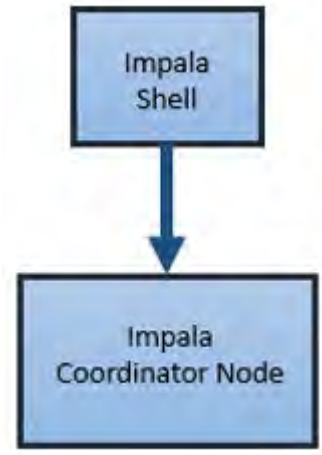
last-updated: Apr 11, 2022



Query Execution

Each node running the Impala service can receive, plan, and coordinate queries. The Impala daemon process on each node listens for requests from ports on each client. Requests from the impala-shell are routed to the impala daemons through one particular port. JDBC and ODBC requests are routed through other ports.

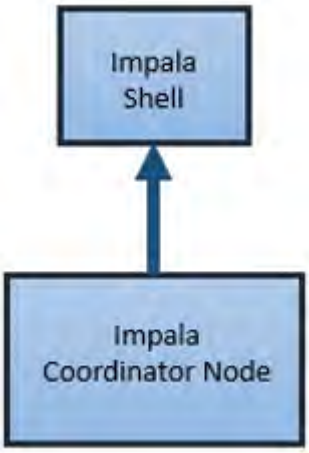
When you send a query to Impala, the client connects to a node running the Impala process. The node that the client connects to becomes the coordinator for the query.



The coordinator node parses the query into fragments and analyzes the query to determine what tasks the nodes running Impala must perform. The coordinator distributes the fragments across other nodes running the Impala daemon process. The nodes process the query fragments and return the data to the coordinator node.

last-updated: Apr 11, 2022

The coordinator node sends the result set back to the client.



New Features in Impala

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Each release of Impala for MapR has new features for improved performance and functionality. Refer to the following sections for a list of new features in each MapR release of Impala:

New Features in Impala 2.10 for MapR

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Impala 2.10.0 for MapR introduces some new features that enhance the behavior and performance of Impala.

Functions

The following sections provide information about new and improved Impala functions:

String function, replace()

Faster than `regexp_replace()` for simple string substitutions.

Description: `replace(string initial, string target, string replacement)`

Purpose: Returns the initial argument with all occurrences of the target string replaced by the replacement string.

Return type: string

Usage notes: Because this function does not use any regular expression patterns, it is typically faster than `regexp_replace()` for simple string substitutions. If any argument is NULL, the return value is NULL. The matching is case-sensitive. If the replacement string contains another instance of the target string, the expansion is only performed once, instead of applying it again to the newly constructed string.

Example:

```
[localhost:21000] > select replace('hello world','world','earth');
Query: select replace('hello world','world','earth')
Query submitted at: 2018-02-08 05:10:11 (Coordinator: http://
localhost.lab:25000)
Query progress can be monitored at: http://localhost:25000/query_plan?
query_id=5e4686bd9b870724:f281e86f00000000
+-----------------------------------------+
| replace('hello world', 'world', 'earth') |
+-----------------------------------------+
| hello earth |
+-----------------------------------------+
Fetched 1 row(s) in 0.02s
```

Date/time function, last_day()

For finding the date corresponding to the last day of a particular month.

Description: `last_day(timestamp t)`

Purpose: timestamp

Usage notes: If the input argument does not represent a valid Impala TIMESTAMP including both date and time portions, the function returns NULL. For example, if the input argument is a string that cannot be implicitly cast to TIMESTAMP, does not include a date portion, or is out of the allowed range for Impala TIMESTAMP values, the function returns NULL.

Example:

```
[localhost:21000] > select
> now() as right_now
> , dayofmonth(now()) as day
> , extract(day from now()) as also_day
> , dayofmonth(last_day(now())) as last_day
> , extract(day from last_day(now())) as also_last_day;
Query: select now() as right_now
, dayofmonth(now()) as day
, extract(day from now()) as also_day
, dayofmonth(last_day(now())) as last_day
, extract(day from last_day(now())) as also_last_day
Query submitted at: 2018-02-08 05:07:59 (Coordinator: http://
localhost:25000)
Query progress can be monitored at: http://localhost:25000/query_plan?
query_id=5147af5bc88d7632:1e42f5c100000000
+-------------------------------+-----+----------+----------+---------------+

| right_now | day | also_day | last_day | also_last_day |
+-------------------------------+-----+----------+----------+---------------+

| 2018-02-08 05:07:59.999909000 | 8 | 8 | 28 | 28 |
+-------------------------------+-----+----------+----------+---------------+
```

Date/time function, last_day()
Provides a simple way to get a stable, interoperable representation of a TIMESTAMP value without using a chain of functions to convert between representations and apply a specific timezone.

Other Features

Impala Query History

You can re-execute previously queries via `@query_number(from start)` or `@-query_number(from end)`:

```
[localhost:21000] >@1;
Rerunning invalidate metadata;
Query: invalidate metadata
[localhost:21000] > @-1;
Rerunning show tables in default;
Query: show tables in default
```

Drop partitions via special symbols (<, >, <>, !=, <=, >=)

ALTER TABLE query for DROP partition can use special symbols for drop range of the partitions:

```
ALTER TABLE db.table DROP PARTITION (partition<=partition_value);
```

CREATE TABLE with SORT BY

Introduced SORT BY clause in the CREATE TABLE statement:

```
CREATE TABLE sorted_tbl (id int, name char(22), age int) SORT BY (age);
```

TRUNK() can now apply to numeric types (FLOAT, DOUBLE, and DECIMAL) in addition to TIMESTAMP

Although this functionality was already available through the truncate() function, the new signatures for trunc() make it easier to port code from other popular database systems to Impala.

Support ENUM type for parquet tables

The CREATE TABLE LIKE PARQUET statement can now handle Parquet files produced outside of Impala and containing ENUM types. The ENUM columns become STRING columns in the target table, and the ENUM values are turned into corresponding STRING values.

New Features in Impala 2.7.0 for MapR

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Impala 2.7.0 for MapR introduces some new features that enhance the behavior and performance of Impala.

Performance Improvements

Impala 2.7.0 for MapR introduces the following performance improvements:

- Overall performance improvements for secure clusters.
- Overall performance improvements for join queries. A pre-fetching mechanism is used while building the in-memory hash table to evaluate join predicates.
- Improved query time for queries run against DECIMAL columns in Avro tables. The code that parses DECIMAL values from Avro now uses native code generation.
- Improved efficiency in LLVM code generation that can reduce codegen time, especially for short queries.
- Improved performance reading Parquet files.
- Improved performance for top-N queries, or queries that include both ORDER BY and LIMIT clauses.

last-updated: Apr 11, 2022

Support for the Amazon S3 File System

You can use Impala to query data in the Amazon S3 filesystem. Impala can query any file type supported by the S3 filesystem.

To run Impala on the Amazon S3 filesystem, add the properties below to the core-site.xml file, copy the core-site.xml file to the IMPALA_CONF_DIR (located in /opt/mapr/impala/impala-2.7.0/conf), and restart the MapR Warden service.

```
<property>
<name>fs.s3a.access.key</name>
<value><ACCESS_KEY></value>
</property>

<property>
<name>fs.s3a.secret.key</name>
<value><SECRET_KEY></value>
</property>
```

Note:

If you plan to run Impala on a secure cluster, create and add the root Certificate Authority (CA) certificate that signed the Amazon S3 certificate to the truststore. Alternatively, you can add the following property to core-site.xml:

```
<property>
  <name>fs.s3a.connection.ssl.enabled</name>
  <value>false</value>
</property>
```

Impala Web User Interface Improvements

In version 2.7.0, Impala provides the following Web User Interface improvements:

- Forced session expiry from the **/sessions** tab. You can force a session to end by clicking on the link provided in the tab.
- Additional Impala memory information on the **/memz** tab.
- A **Memory** tab on the **Details** page for a query.

Set Column Stats Clause

In version 2.7.0, you can use the SET COLUMN STATS clause with the ATLER TABLE statement to set a specific statistical value for a column. You can set a column to the number of distinct values, number of nulls, maximum size, or average size using the respective case-insensitive symbolic names:

`numDVs, numNulls, avgSize, maxSize`

Note: This operation applies to an entire table, not a specific partition. You must use single quotation marks around the symbolic name and value.

Run the ALTER TABLE command with the SET COLUMN STATS clause from the impala-shell, as shown in the following example:

```
$ impala-shell -i localhost
...
[localhost:21000] > create table example (id int);
[localhost:21000] > insert into example values (1), (2), (3);
[localhost:21000] > show column stats example;
+--------+--------+------------------+--------+----------+----------+
```

```
| Column | Type   | #Distinct Values | #Nulls | Max Size | Avg Size |
+--------+--------+------------------+--------+----------+----------+
| id     | INT    | -1               | -1     | 4        | 4        |
+--------+--------+------------------+--------+----------+----------+
[localhost:21000] > alter table example set column stats id
('numDvs'='3','numNulls'='0');
[localhost:21000] > show column stats example;
+--------+--------+------------------+--------+----------+----------+
| Column | Type   | #Distinct Values | #Nulls | Max Size | Avg Size |
+--------+--------+------------------+--------+----------+----------+
| id     | INT    | 3                | 0      | 4        | 4        |
+--------+--------+------------------+--------+----------+----------+
```

SOURCE Command

The SOURCE command runs SQL statements from an SQL source file. You issue the SOURCE command from the impala-shell. The source file can contain SQL statements and impala-shell commands, as well as additional SOURCE commands. Each statement or command must end with a semicolon (;), except for the last statement or command in the file.

Including additional SOURCE commands inside the source file allows you to set up flexible statement sequences for use cases, such as schema setup, ETL, or reporting.

Run the SOURCE command from the impala-shell, as shown in the following example:

```
$ cat example.sql
show databases;
show tables in customers;
$ impala-shell -i localhost
...
[localhost:21000] > source example.sql;
Query: show databases
+------------------+-------------------------------------------------+
| name             | comment                                         |
+------------------+-------------------------------------------------+
| customers        | Stores customer records                         |
| default          | Default Hive database                           |
+------------------+-------------------------------------------------+
Fetched 2 row(s) in 0.06s
Query: show tables in customers
+--------------+
| name         |
+--------------+
| customer_001 |
| customer_002 |
| customer_003 |
| customer_004 |
| customer_005 |
+--------------+
Fetched 5 row(s) in 0.03s
```

Additionally, you can run the SOURCE command on an SQL file that contains statements and calls a SOURCE command from another SLQ file, as shown in the following example:

```
$ cat example1.sql
show databases;
source example2.sql
$ cat example2.sql
show tables in customers;

$ impala-shell -i localhost
...
```

```
[localhost:21000] > source example1.sql;
Query: show databases
+------------------+-----------------------------------------------+
| name             | comment                                       |
+------------------+-----------------------------------------------+
| customers        | Stores customer records                       |
| default          | Default Hive database                         |
+------------------+-----------------------------------------------+
Fetched 2 row(s) in 0.06s
Query: show tables in customers
+--------------+
| name         |
+--------------+
| customer_001 |
| customer_002 |
| customer_003 |
| customer_004 |
| customer_005 |
+--------------+
Fetched 5 row(s) in 0.03s
```

New Runtime Filter Options

The Runtime Filtering feature was introduced in Impala 2.5.0 to optimize queries against partitioned tables or to evaluate join conditions when partial table data is needed. Impala 2.7.0 introduces two new filtering options that fine-tune the sizes of the Bloom filter structures used in runtime filtering:

The following table lists the new query options and their descriptions. Only modify these options when tuning long-running queries with a combination of large partitioned tables and joins on large tables.

Option	Description	Type	Default
RUNTIME_FILTER_MIN_SIZE	Adjusts runtime filtering settings. Defines the minimum size for a filter, regardless of the estimates produced by the planner. This setting overrides any smaller value set for the RUNTIME_BLOOM_FILTER_SIZE option. Impala rounds filter sizes up to the nearest power of two.	integer	0 (indicates that the value from the corresponding impalad startup option is used)
RUNTIME_FILTER_MAX_SIZE	Adjusts runtime filtering settings. Defines the maximum size for a filter, regardless of the estimates produced by the planner. This setting overrides any smaller value set for the RUNTIME_BLOOM_FILTER_SIZE option. Impala rounds filter sizes up to the nearest power of two.	integer	0 (indicates that the value from the corresponding impalad startup option is used)

New Features in Impala 2.5.0 for MapR

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Impala 2.5.0 for MapR introduces some new features that enhance the behavior and performance of Impala.

Performance Improvements

The following updates improve the performance of Impala:

- Impala caches file handles to avoid the overhead of repeatedly opening the same file which improves I/O performance.
- Basic query types, such as counting the elements of a complex column, use an optimized code path that improves the performance of queries involving nested complex types.
- Impala uses code generation in certain parts of queries, such as evaluating functions in the WHERE clause, even when code generation is not used in some phases of query execution.
- Using DECIMAL values in a GROUP BY clause triggers code generation optimization, which speeds up queries that group by values, such as prices, and improves performance and reliability of the DECIMAL data type.
- Improved coordination and parallelization between Impala nodes results in faster query startup time for queries that involve tables with many partitions or complex queries with many fragments.
- The coordinator node requires less memory, making it faster and less resource-intensive when performing joins that involve several tables with thousands of partitions.
- Impala only re-evaluates metadata for partitions that are affected by a DDL operation, not all partitions in the table. While a DDL or insert statement is in progress, other Impala statements that attempt to modify metadata for the same table wait until the first one finishes, improving performance and reliability of DDL and insert operations on partitioned tables with a large number of partitions.
- A new query option, OPTIMIZE_PARTITION_KEY_SCANS, speeds up aggregation operations that involve only the partition key columns of partitioned tables. This optimization can produce different results when files in a partition are manually deleted or are empty.

Security

This release of Impala provides the following new security features:

Column-Level Authorization

You can use column-level authorization to define access to particular columns within a table instead of the entire table. Creating views to set up authorization schemes for subsets of information is not required. This functionality requires Sentry 1.6

Column-level authorization has the following syntax:

```
GRANT SELECT(column_name) ON TABLE
table_name TO ROLE role_name;
REVOKE SELECT(column_name) ON TABLE
table_name FROM ROLE role_name;
```

LDAP Password Retrieval

You can use a new impala-shell command line option, `--ldap_password_cmd`, to retrieve the LDAP password. The resulting password is used to authenticate the impala-shell command with the LDAP server.

Scripting Capability Improvements

Scripting capability improvements for the impala-shell enable you to define substitution variables and use them in SQL statements that you execute through command-line options. The `--var` command-line option

last-updated: Apr 11, 2022

passes key-value pairs to the impala-shell. The shell substitutes the values into an SQL statement where it contains the notation `${var:varname}` before Impala executes the query.

You can use the SET and UNSET commands in a session to define or clear substitution variables with the `SET|UNSET VAR:variable_name=value` notation or a script file processed by the `-f` option. You cannot define your own substitution variables through the SET statement in a JDBC or ODBC application.

Dynamic Partition Pruning

Dynamic partition pruning is a technique that prevents Impala from reading data files from partitions that are not included in the result set. This occurs when a query references a partition key column in the WHERE clause and the column values are unknown until the query runs. Impala evaluates the predicate and skips the I/O for unnecessary partitions. This technique is useful for join queries that involve large partitioned tables.

You can control the level of dynamic partition pruning through the `RUNTIME_FILTER_MODE` query option. By default, this option is enabled and set at medium level: `RUNTIME_FILTER_MODE=LOCAL)`. The maximum setting uses more memory for queries than in previous releases. You can set `RUNTIME_FILTER_MODE=GLOBAL` to fully enable dynamic partition pruning. Before you run a query, check the EXPLAIN output to verify that partition pruning is applied.

Nested Loop Join Queries

Nested loop joins make additional non-equijoin queries possible and optimize queries that retrieve values from complex type columns. Some join queries that previously needed equality comparisons can use operators.

Live Progress Reporting

The live progress reporting feature enables you to monitor the status of queries through two command-line options, --live_progress and --live_summary. The live progress option provides an interactive progress bar that represents the percent completion for queries submitted through the impala-shell command. The live summary option summarizes the work performed in various stages of a query with the measurements updated in real time as the query progresses. These query options are disabled by default. To enable these options, start the impala-shell with the --live_progress and --live_summary command-line options, or use the SET command with LIVE_SUMMARY and LIVE_PROGRESS during a session:

```
set live_progress=true|false;
set live_summary=true|false;
```

Runtime Filtering

Runtime filtering is technique where Impala determines the filter conditions as a query runs and broadcasts the information to the Impala nodes reading a table. This technique is useful for optimizing queries against partitioned tables or to evaluate join conditions when only partial table data is needed. The technique eliminates the I/O required to read all of the partitioned data, as well as unnecessary network traffic.

The following table lists the new query options related to runtime filtering:

Option	Description	Type	Default
RUNTIME_FILTER_MODE	Adjusts the settings, turns the feature on and off, and controls how extensively the filters are transmitted between hosts.	numeric (0, 1, 2) or corresponding mnemonic strings (OFF, LOCAL, GLOBAL)	1 (same as LOCAL)
MAX_NUM_RUNTIME_FILTERS	Sets an upper limit on the number of runtime filters produced for each query.	integer	10

last-updated: Apr 11, 2022

RUNTIME_BLOOM_FILTER_SIZE	Size (in bytes) of Bloom filter data structure used by the runtime filtering feature.	integer	1048576 (1 MB) Max: 16 MB
RUNTIME_FILTER_WAIT_TIME_MS	Maximum filter wait time in milliseconds. By default, each scan node waits for up to 1 second (1000 milliseconds) for filters to arrive. If all filters have not arrived within the specified interval, the scan node proceeds, using whatever filters did arrive to help avoid reading unnecessary data. If a filter arrives after the scan node begins reading data, the scan node applies that filter to the data that is read after the filter arrives, but not to the data that was already read.	integer	1000 milliseconds
DISABLE_ROW_RUNTIME_FILTERING	Reduces the scope of the runtime filtering feature. Queries still dynamically prune partitions, but do not apply the filtering logic to individual rows within partitions. Only applies to queries against Parquet tables. For other file formats, Impala only prunes at the level of partitions, not individual rows.	Boolean; recognized values are 1 and 0, or true and false; any other value interpreted as false	false

Data Types

Impala now supports new complex data types that can encode multiple named fields, positional items, or key-value pairs within a single column. You can combine these types to produce nested types with arbitrarily deep nesting. Currently, complex data types are only supported for the Parquet file format.

The following table lists the complex data types and the syntax for each:

Data Type	**Syntax**
STRUCT	*column_name* STRUCT < *name* : *type* [COMMENT '*comment_string*'], ... > type ::= *primitive_type* | *complex_type*
ARRAY	*column_name* ARRAY < *type* > type ::= *primitive_type* | *complex_type*
MAP	*column_name* MAP < *primitive_type*, *type* > type ::= *primitive_type* | *complex_type*

Operators

The following table provides the new and improved operators with descriptions and syntax:

last-updated: Apr 11, 2022

Operator	Description	Syntax
ILIKE	Improvements enable the ILIKE operator to perform case-insensitive wildcard matches or regular expression matches, rather than explicitly converting column values with UPPER or LOWER.	*string_expression* ILIKE *wildcard_expression* *string_expression* NOT ILIKE *wildcard_expression*
IREGEXPR	Improvements enable the IREGEXPR operator to perform case-insensitive wildcard matches or regular expression matches, rather than explicitly converting column values with UPPER or LOWER.	*string_expression* IREGEXP *regular_expression*
IS [NOT] DISTINCT FROM	Compares values for a true or false result, even if one or both values are NULL. The IS NOT DISTINCT FROM operator, or its equivalent <=> notation, improves the efficiency of join queries that treat key values that are NULL in both tables as equal.	*expression1* IS DISTINCT FROM *expression2* *expression1* IS NOT DISTINCT FROM *expression2* *expression1* <=> *expression2*

Statements

The following table lists new or improved statements with their descriptions and syntax:

Statement	Description	Syntax
TRUNCATE TABLE	Removes all data from a table without removing the table itself. The statement accepts the IF EXISTS clause, making TRUNCATE TABLE easier to use in setup or ETL scripts where the table might or might not exist.	TRUNCATE TABLE [IF EXISTS] [*db_name*.]*table_name*
CREATE TABLE AS SELECT	The CREATE TABLE AS SELECT statement accepts a PARTITIONED BY clause. You can create a partitioned table and insert data into the table with a single statement.	CREATE [EXTERNAL] TABLE [IF NOT EXISTS] *db_name*.]*table_name* [PARTITIONED BY (*col_name*[, ...])] [COMMENT '*table_comment*'] [WITH SERDEPROPERTIES ('*key1*'='*value1*', '*key2*'='*value2*', ...)] [[ROW FORMAT *row_format*] [STORED AS *file_format*]] [LOCATION '*hdfs_path*'] [TBLPROPERTIES ('*key1*'='*value1*', '*key2*'='*value2*', ...)] [CACHED IN '*pool_name*' [WITH REPLICATION = *integer*] | UNCACHED] AS *select_statement*

last-updated: Apr 11, 2022

SHOW DATABASES	The SHOW DATABASES statement returns two columns rather than one. The second column includes the associated comment string, if it exists, for each database.	SHOW DATABASES;
DESCRIBE	The DESCRIBE statement displays metadata about a database.	DESCRIBE DATABASE db_name;
DROP DATABASE	The DROP DATABASE statement works for a nonempty database. When you specify the optional CASCADE clause, any tables in the database are dropped before the database itself is removed.	DROP (DATABASE|SCHEMA) [IF EXISTS] *database_name* [RESTRICT | CASCADE];
DROP TABLE/ALTER TABLE DROP PARTITION	The DROP TABLE, ALTER TABLE, and DROP PARTITION statements have an optional keyword, PURGE. PURGE causes Impala to immediately remove the relevant data files rather than sending them to the trashcan. This feature can help to avoid out-of-space errors on storage devices, and to avoid files being left behind in case of a problem with the trashcan. PURGE works when the trashcan is enabled.	DROP TABLE [IF EXISTS] [db_name.]table_name [PURGE] ALTER TABLE name { ADD [IF NOT EXISTS] | DROP [IF EXISTS] } PARTITION (partition_spec) [PURGE]

Functions

The following sections provide information about new and improved Impala functions.

User-Defined Functions

User-defined functions written in C++ persist when the catalog service is restarted. You no longer have to run the CREATE FUNCTION statements again after a restart. You must still reissue the CREATE FUNCTION statement for any Java-based user-defined functions. User-defined aggregate functions have more flexibility for intermediate data types.

Analytic (Window) Functions

The following table lists the new analytic functions:

Function	Description	Syntax

PERCENT_RANK	Calculates the rank, expressed as a percentage, of each row within a group of rows. If rank is the value for that same row from the RANK() function (from 1 to the total number of rows in the partition group), then the PERCENT_RANK() value is calculated as (*rank* - 1) / (*rows_in_group* - 1). If there is only a single item in the partition group, its PERCENT_RANK() value is 0. The ORDER BY clause is required. The PARTITION BY clause is optional. The window clause is not allowed.	PERCENT_RANK (*expr*) OVER ([*partition_by_clause*] *order_by_clause*)
NTILE	Returns the "bucket number" associated with each row, between 1 and the value of an expression. For example, creating 100 buckets puts the lowest 1% of values in the first bucket, while creating 10 buckets puts the lowest 10% of values in the first bucket. Each partition can have a different number of buckets. The ORDER BY clause is required. The PARTITION BY clause is optional. The window clause is not allowed.	NTILE (*expr* [, *offset* ...] OVER ([*partition_by_clause*] *order_by_clause*)

CUME_DIST	Returns the cumulative distribution of a value. The value for each row in the result set is greater than 0 and less than or equal to 1. The ORDER BY clause is required. The PARTITION BY clause is optional. The window clause is not allowed.	CUME_DIST (*expr*) OVER ([*partition_by_clause*] *order_by_clause*)

Math Functions

The following table lists the new math functions:

Function	Description	Return Type
cot(double a)	Returns the cotangent of the argument.	double
factorial(integer_type a)	Computes the factorial of an integer value and works with any integer type. You can use either the factorial() function or the ! operator. The factorial of 0 is 1. The factorial() function returns 1 for any negative value. The maximum positive value for the input argument is 20; a value of 21 or greater overflows the range for a BIGINT and causes an error.	bigint
radians(double a)	Converts the argument value from degrees to radians.	double

String Functions

The following table lists the new string functions:

Function	Description	Return Type

btrim(string a), btrim(string a, string chars_to_trim)	Removes all instances of one or more characters from the start and end of a STRING value. By default, removes only spaces. If a non-NULL optional second argument is specified, the function removes all occurrences of characters in that second argument from the beginning and end of the string.	string
chr(int character_code)	Returns a character specified by a decimal code point value. The interpretation and display of the resulting character depends on your system locale. Because consistent processing of Impala string values is only guaranteed for values within the ASCII range, only use this function for values corresponding to ASCII characters. In particular, parameter values greater than 255 return an empty string.	string

last-updated: Apr 11, 2022

regexp_like(string source, string pattern[, string options])	Returns true or false to indicate whether the source string contains anywhere inside it the regular expression given by the pattern. The optional third argument consists of letter flags that change how the match is performed, such as i for case-insensitive matching.	boolean
split_part(string source, string delimiter, bigint n)	Returns the nth field within a delimited string. The fields are numbered starting from 1. The delimiter can consist of multiple characters, not just a single character. All matching of the delimiter is done exactly, not using any regular expression patterns.	string

Date/Time Functions

The following table lists the new date and time functions:

Function	Description	Return Type
int_months_between(timestamp newer, timestamp older)	Returns the number of months between the date portions of two TIMESTAMP values, as an INT representing only the full months that passed.	int

months_between(timestamp newer, timestamp older)	Returns the number of months between the date portions of two TIMESTAMP values. Can include a fractional part representing extra days in addition to the full months between the dates. The fractional component is computed by dividing the difference in days by 31 (regardless of the month).	double
timeofday()	Returns a string representation of the current date and time, according to the time of the local system, including any time zone designation.	string
timestamp_cmp(timestamp t1, timestamp t2)	Tests if one TIMESTAMP value is newer than, older than, or identical to another TIMESTAMP.	int (either -1, 0, 1, or NULL)

Bit Manipulation Functions

The following table lists the new bit manipulation functions:

Function	Description	Return Type
bitand(integer_type a, same_type b)	Returns an integer value representing the bits that are set to 1 in both of the arguments. If the arguments are of different sizes, the smaller is promoted to the type of the larger. Equivalent to the & binary operator.	Same as the input value

bitnot(integer_type a)	Inverts all the bits of the input argument. Equivalent to the ~ unary operator.	Same as the input value
bitor(integer_type a, same_type b)	Returns an integer value representing the bits that are set to 1 in either of the arguments. If the arguments are of different sizes, the smaller is promoted to the type of the larger. Equivalent to the | binary operator.	Same as the input value
bitxor(integer_type a, same_type b)	Returns an integer value representing the bits that are set to 1 in one but not both of the arguments. If the arguments are of different sizes, the smaller is promoted to the type of the larger. Equivalent to the ^ binary operator.	Same as the input value
countset(integer_type a [, int zero_or_one])	By default, returns the number of 1 bits in the specified integer value. If the optional second argument is set to zero, it returns the number of 0 bits instead.	Same as the input value

last-updated: Apr 11, 2022

getbit(integer_type a, int position)	Returns a 0 or 1 representing the bit at a specified position. The positions are numbered right to left, starting at zero. The position argument cannot be negative. When you use a literal input value, it is treated as an 8-bit, 16-bit, and so on value, the smallest type that is appropriate. The type of the input value limits the range of the positions. Cast the input value to the appropriate type if you need to ensure it is treated as a 64-bit, 32-bit, and so on value.	Same as the input value

rotateleft(integer_type a, int positions)	Rotates an integer value left by a specified number of bits. As the most significant bit is taken out of the original value, if it is a 1 bit, it is "rotated" back to the least significant bit. Therefore, the final value has the same number of 1 bits as the original value, just in different positions. Specifying a second argument of zero leaves the original value unchanged. Rotating a -1 value by any number of positions still returns -1, because the original value has all 1 bits and all the 1 bits are preserved during rotation. Similarly, rotating a 0 value by any number of positions still returns 0. Rotating a value by the same number of bits as in the value returns the same value. Because this is a circular operation, the number of positions is not limited to the number of bits in the input value. For example, rotating an 8-bit value by 1, 9, 17, and so on positions returns an identical result in each case.	Same as the input value

last-updated: Apr 11, 2022

rotateright(integer_type a, int positions)	Rotates an integer value right by a specified number of bits. As the least significant bit is taken out of the original value, if it is a 1 bit, it is "rotated" back to the most significant bit. Therefore, the final value has the same number of 1 bits as the original value, just in different positions. Specifying a second argument of zero leaves the original value unchanged. Rotating a -1 value by any number of positions still returns -1, because the original value has all 1 bits and all the 1 bits are preserved during rotation. Similarly, rotating a 0 value by any number of positions still returns 0. Rotating a value by the same number of bits as in the value returns the same value. Because this is a circular operation, the number of positions is not limited to the number of bits in the input value. For example, rotating an 8-bit value by 1, 9, 17, and so on positions returns an identical result in each case.	Same as the input value

last-updated: Apr 11, 2022

setbit(integer_type a, int position [, int zero_or_one])	By default, changes a bit at a specified position to a 1, if it is not already. If the optional third argument is set to zero, the specified bit is set to 0 instead. If the bit at the specified position was already 1 (by default) or 0 (with a third argument of zero), the return value is the same as the first argument. The positions are numbered right to left, starting at zero. (Therefore, the return value could be different from the first argument even if the position argument is zero.) The position argument cannot be negative. When you use a literal input value, it is treated as an 8-bit, 16-bit, and so on value, the smallest type that is appropriate. The type of the input value limits the range of the positions. Cast the input value to the appropriate type if you need to ensure it is treated as a 64-bit, 32-bit, and so on value.	Same as the input value

last-updated: Apr 11, 2022

shiftleft(integer_type a, int positions)	Shifts an integer value left by a specified number of bits. As the most significant bit is taken out of the original value, it is discarded and the least significant bit becomes 0. The final value has either the same number of 1 bits as the original value, or fewer. Shifting an 8-bit value by 8 positions, a 16-bit value by 16 positions, and so on produces a result of zero. Specifying a second argument of zero leaves the original value unchanged. Shifting any value by 0 returns the original value. Shifting any value by 1 is the same as multiplying it by 2, as long as the value is small enough; larger values eventually become negative when shifted, as the sign bit is set. Starting with the value 1 and shifting it left by N positions gives the same result as 2 to the Nth power, or pow(2,N).	Same as the input value

shiftright(integer_type a, int positions)	Shifts an integer value right by a specified number of bits. As the least significant bit is taken out of the original value, it is discarded and the most significant bit becomes 0. Therefore, the final value has either the same number of 1 bits as the original value, or fewer. Shifting an 8-bit value by 8 positions, a 16-bit value by 16 positions, and so on produces a result of zero. Specifying a second argument of zero leaves the original value unchanged. Shifting any value by 0 returns the original value. Shifting any positive value right by 1 is the same as dividing it by 2. Negative values become positive when shifted right.	Same as the input value

Miscellaneous Functions

The following table lists a new miscellaneous function:

Function	Description	Syntax

last-updated: Apr 11, 2022

uuid()	The uuid() function generates an alphanumeric value that you can use as a guaranteed unique identifier. The uniqueness applies across tables in cases where an ascending numeric sequence is not suitable.	select uuid();
typeof(type value)	A type conversion function that returns the name of the data type corresponding to an expression. For types with extra attributes, such as length for CHAR and VARCHAR, or precision and scale for DECIMAL, includes the full specification of the type.	select typeof(type value); Returns the type For example, select typeof('xyz'); returns STRING For example, select typeof(5.30001 / 2342.1); returns DECIMAL(13,11)

New Features in Impala 2.2.0 for MapR

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Spill to Disk

Impala 2.2.0 for MapR introduces the "spill to disk" feature which prevents queries that use memory-intensive operations, such as large sort, join, aggregation, or analytic function operations, from failing with out-of-memory errors. Impala automatically writes data to disk when queries with memory-intensive operations exceed the set memory limit on an Impala node. You can enable or disable the spill to disk feature in `/opt/mapr/impala/impala-<version>/conf/env.sh`. You can also change the DISABLE_UNSAFE_SPILLS setting at the session level to allow or prevent data spills.

See Spill to Disk for more information about this feature.

Security

You can use SQL statements to create roles and grant/revoke privileges on objects.

Note: This security functionality requires Sentry 1.6 .

The following statements create roles and grant privileges:

- CREATE ROLE - creates roles for users and groups

last-updated: Apr 11, 2022

- GRANT - grants privileges on objects

The following statements revoke roles and privileges:

- DROP ROLE - removes roles from the metastore database
- REVOKE - revokes roles or privileges on a specified object from groups

The following statements display role related information:

- SHOW GRANT ROLE - list all the grants for the given role name
- SHOW ROLES - displays roles
- SHOW ROLE GRANT GROUP - lists all the roles assigned to a specified group

Smaller Parquet and MapR File System Block Size

The default setting for the `PARQUET_FILE_SIZE` query option has changed from 1 GB to 256 MB. This makes the file more accurate and reduces the amount of memory reserved during an INSERT into Parquet tables, potentially avoiding out-of-memory errors and improving scalability when inserting data into Parquet tables

.impalarc Configuration File

You can specify impala-shell options from the command line or in the `$HOME/.impalarc` configuration file. You can define a set of default options for your impala-shell environment in the`$HOME/.impalarc` configuration file. For every command line option, there is an equivalent setting in the `$HOME/.impalarc` configuration file. Options specified from the impala-shell command line override corresponding options in the configuration file. See Impala-Shell Command Line Options for more information. The configuration file must contain a header label `[impala]`, followed by the options specific to impala-shell. This is a standard convention for configuration files that enables a single file to hold configuration options for multiple applications. To specify a different filename or path for the configuration file, specify the argument

```
--config_file=path_to_config_file
```

on the impala-shell command line.

Files

Impala can read GZIP, BZIP, or Snappy compressed text files.The files do not require special table settings to work in an Impala text table. Impala ignores temporary files typically produced by ETL tools, such as those with the suffixes .copying and .tmp.

Log Rotation

Log rotation is the automatic removal of unneeded or old log files. By default, Impala switches out old log files every 5 seconds, based on the default interval specified in the `logbufsecs` setting. The `-max_log_files` configuration option specifies how many log files to keep at each severity level (INFO, WARNING, ERROR, and FATAL). You can configure the

`-max_log_files` option for each Impala daemon (impalad, statestored, and catalogd) in the `env.sh`configuration file. By default, Impala preserves the latest 10 log files for each severity level and removes old logs based on the logbufsecs setting. Setting `-max_log_files` to 0 preserves all of the log files. This setting requires manual log rotation. Setting`-max_log_files` to 1 preserves only the latest log file.

Impala Debug Web Interface

The Impala debug web interface displays a visual representation of the query plan. You can access the Impala debug web interface at

`http://<impala-node-hostname>:25000/`. To see visual representation of the query plan, select the **/queries** tab and click **Details** for a particular query. The Details page includes a Plan tab with a plan diagram for which you can zoom in or out using your mouse or trackpad.

Date and Time Improvements

Flexibility to interpret `TIMESTAMP` values using the UTC time zone or using the local time zone, for compatibility with `TIMESTAMP` values produced by Hive.

Startup flags for the impalad daemon enable a higher level of compatibility with `TIMESTAMP` values written by Hive and more flexibility for working with date and time data using the local time zone instead of UTC. To enable these features, set the impalad startup flags `-use_local_tz_for_unix_timestamp_conversions=true` and `-convert_legacy_hive_parquet_utc_timestamps=true`.

The `-use_local_tz_for_unix_timestamp_conversions` setting controls how the `unix_timestamp()`, `from_unixtime()`, and `now()` functions handle time zones. By default, this setting disabled, and Impala considers all `TIMESTAMP` values to be in the UTC time zone when converting to or from Unix time values. When this setting is enabled, Impala treats `TIMESTAMP` values passed to or returned from these functions to be in the local time zone. When this setting is enabled, verify that all hosts in the cluster have the same timezone settings to avoid inconsistent results depending on which host reads or writes `TIMESTAMP` data.

The `-convert_legacy_hive_parquet_utc_timestamps` setting causes Impala to convert `TIMESTAMP` values to the local time zone when it reads them from Parquet files written by Hive. This setting only applies to data using the Parquet file format where Impala can use metadata in the files to reliably determine that the files were written by Hive. If in the future Hive changes the way it writes `TIMESTAMP` data in Parquet, Impala will automatically handle that new `TIMESTAMP` encoding.

Built-in functions that accept or return integers representing TIMESTAMP values use the BIGINT type for parameters and return values rather than INT. This change prevents the date and time functions from having overflow errors that would otherwise occur on January 19th, 2038 (known as the "Year 2038 problem" or "Y2K38 problem"). This change affects the from_unixtime() and unix_timestamp() functions. You might need to change application code that interacts with these functions, change the types of columns that store the return values, or add CAST() calls to SQL statements that call these functions.

Analytic Functions

Analytic (window) functions operate on a set of rows and return a single value for each row from the underlying query. The term "window" describes the set of rows on which the function operates. A window function uses values from the rows in a window to calculate the returned values. When you use a window function in a query, you define the window using the OVER() clause. The OVER() clause (window clause) differentiates window functions from other analytical and reporting functions.

As of Impala 2.2.0, you can use the following analytic functions in queries:

- MAX()
- MIN()
- SUM()
- COUNT()
- AVG()

- RANK()
- LAG()
- LEAD()
- FIRST_VALUE()

The analytic functions support the following syntax:

```
function(args) OVER([partition_by_clause] [order_by_clause [window_clause]])
partition_by_clause ::= PARTITION BY expr [, expr ...]order_by_clause ::=
ORDER BY expr  [ASC | DESC] [NULLS FIRST | NULLS LAST] [, expr [ASC |DESC]
[NULLS FIRST | NULLS LAST] ...]
```

The window clause supports the following syntax:

```
ROWS BETWEEN [ { m | UNBOUNDED } PRECEDING | CURRENT ROW] [ AND [CURRENT
ROW | { UNBOUNDED | n } FOLLOWING] ]
```

```
RANGE BETWEEN [ {m | UNBOUNDED } PRECEDING | CURRENT ROW] [ AND [CURRENT
ROW | { UNBOUNDED | n } FOLLOWING] ]
```

Query Options

You issue query options to Impala using the SET command. SET has been promoted to an SQL statement and can be used in client applications through the JDBC and ODBC APIs.

The following table lists new query options that you can use with the SET statement:

APPX_COUNT_DISTINCT()	Allows multiple COUNT(DISTINCT) operations within a single query. When used in a query, Impala rewrites each COUNT(DISTINCT) to use the NDV() function, resulting in an approximate count rather than precise. Default is`false` (shown as 0 in output of `SET` statement). Syntax: `set APPX_COUNT_DISTINCT=true|false;`
SET EXEC_SINGLE_NODE_ROWS_THRESHOLD	This query option controls how many rows constitute a small query providing a guideline for when to turn optimizations on or off which prevents the unnecessary overhead of parallelizing and generating native code. Impala can complete small queries quickly and free-up YARN resources and admission control slots for data-intensive queries. Default is 100. Syntax: `SET EXEC_SINGLE_NODE_ROWS_THRESHOLD=number_of_rows;`
PARQUET_FILE_SIZE	Specifies the maximum size of each Parquet data file produced by Impala `INSERT` statements. Specify the size in bytes or with a trailing `m` or gcharacter to indicate megabytes or gigabytes. For example: `-- 128 megabytes.set PARQUET_FILE_SIZE=134217728 \ INSERT OVERWRITE parquet_table SELECT`

```
* FROM text_table;
-- 512 megabytes.set
PARQUET_FILE_SIZE=512m; \
INSERT OVERWRITE parquet_table SELECT
* FROM text_table;
-- 1 gigabyte.set
PARQUET_FILE_SIZE=1g; \
INSERT OVERWRITE parquet_table SELECT
* FROM text_table;
```

Syntax: `set PARQUET_FILE_SIZE=size INSERT OVERWRITE parquet_table SELECT * FROM text_table;`

QUERY_TIMEOUT_S

Sets the idle query timeout value, in seconds, for the session. Impala automatically cancels queries that sit idle for longer than the timeout value specified. QUERY_TIMEOUT_S must be smaller than or equal to the `--idle_query_timeout` value if the `--idle_query_timeoutstartup` option is set.

Syntax: `SET QUERY_TIMEOUT_S=seconds;`

Functions

The following table lists new functions with their descriptions and syntax:

APPX_MEDIAN()

An aggregate function that uses sampling to produce an estimate for the median value of a column. This function returns a value that is approximately the median (midpoint) of values in the set of input values. The input type must support less-than and greater-than comparison operators.

Return type: Same as the input value, except for CHAR and VARCHAR arguments which produce a STRING result

The return value is always the same as one of the input values, not an "in-between" value produced by averaging.

Syntax: `APPX_MEDIAN([DISTINCT | ALL] expression)`

CURRENT_DATABASE()

Utility function that returns the database that the session is currently using. Returns`default` if no database was selected or the database that the session switched to with the USE statement or the

```
impalad-d
```

option. Return type: string

Syntax: `CURRENT_DATABASE()`

DATE_PART()

A new date and time function, similar to EXTRACT(), but with the order of the arguments reversed. You can also call the EXTRACT() function using the SQL-99 syntax, EXTRACT(*unit* FROM *timestamp*). These enhancements simplify the porting process for date-related code from other systems. Return type: int

Syntax: `DATE_PART(string,timestamp)`

last-updated: Apr 11, 2022

DECODE()	A function that compares an expression to one or more possible values and returns a corresponding result when a match is found. This function works as a shorthand for a CASE() expression and improves compatibility with SQL code containing vendor extensions. Return type: Same as the initial argument value, except that integer values are promoted to BIGINT and floating-point values are promoted to DOUBLE; use CAST() when inserting into a smaller numeric column. Syntax: `decode(type expression, type search1, type result1 [, type search2, type result2...] [, type default] )`
isfalse(), isnotfalse(), isnottrue(), istrue(), nonnullvalue(), nullvalue()	These conditional functions provide enhanced compatibility when porting code that uses industry extensions.
mod()	This function returns the modulus of a number. MOD is equivalent to using the % arithmetic operator. It works with any size integer type, any size floating-point type, and DECIMAL with any precision and scale. Return type: Same as the input value. Syntax: `mod(numeric_type a, same_type b)`
NDV()	This aggregate function now returns DOUBLE results rather than STRING. Prior to 2.2.0, you had to CAST() the result to a numeric type before using the function in arithmetic operations. Syntax: `NDV([DISTINCT | ALL] expression)`
STDDEV(), STDDEV_POP(), STDDEV_SAMP()	These aggregate functions now return DOUBLE results rather than STRING. Prior to 2.2.0, you had to CAST() the result to a numeric type before using the function in arithmetic operations. Syntax: `{ STDDEV | STDDEV_SAMP | STDDEV_POP }([DISTINCT | ALL] expression)`
VARIANCE(), VARIANCE_POP(), VARIANCE_SAMP()	These aggregate functions now return DOUBLE results rather than STRING. Prior to 2.2.0, you had to CAST() the result to a numeric type before using the function in arithmetic operations.`VAR_SAMP()` and `VAR_POP()` are aliases for the existing `VARIANCE_SAMP()` and `VARIANCE_POP()` functions. Syntax: `{ VARIANCE | VAR[IANCE]_SAMP | VAR[IANCE]_POP } ([DISTINCT | ALL] expression)`

Statements

The following table lists new or improved statements with their descriptions and syntax:

COMPUTE|DROP INCREMENTAL STATS	This statement collects or drops statistics for individual partitions in a partitioned table instead of processing the entire table for each COMPUTE STATS statement. Use the COMPUTE STATS statement for non-partitioned tables or partitioned tables that are unchanging or have content that is entirely replaced. Syntax: `COMPUTE|DROP INCREMENTAL STATS [db_name.]table_name [PARTITION`

	`(partition_spec)] partition_spec ::= partition_col=constant_value`
CREATE ROLE	This statement creates a role. You can grant privileges to roles and then assign roles to users. A user can only exercise the privileges associated with a particular role. Only users that have administrative privileges can create/drop roles. By default, the `hive`, `impala` and `hue` users have administrative privileges in Sentry. Syntax: `CREATEROLErole_name`
DROP ROLE	Removes a role from the metastore database. When you drop a role, the role is revoked from all users to whom it was previously assigned, and all privileges granted to that role are revoked. In progress queries are not affected. Impala verifies the role information approximately every 60 seconds. Syntax: `DROPROLE role_name`

GRANT	This statement grants roles or privileges on specified objects to groups. The object name is typically an identifier. For URIs, it is a string literal.
	Only administrative users (initially, a predefined set of users specified in the Sentry service configuration file) can use this statement.
	The `WITH GRANT OPTION` clause allows members of the specified role to issue `GRANT` and `REVOKE` statements for those same privileges. If a role has the `ALL` privilege on a database and the `WITH GRANTOPTION` set, users granted that role can execute `GRANT`/`REVOKE` statements only for that database or child tables of the database. This means a user could revoke the privileges of the user that provided them the `GRANT OPTION`.
	You cannot revoke the `WITH GRANT OPTION` from a privilege that was previously granted to a role. To remove the `WITH GRANT OPTION`, revoke the privilege and grant it again without the `WITHGRANT OPTION` flag.
	Syntax:
	`GRANT ROLE role_name TO GROUP group_name`
	
	`GRANT privilege ON object_type object_name TO [ROLE] roleName [WITH GRANT OPTION]`
	
	`privilege ::= SELECT | INSERT | ALL object_type ::= TABLE | DATABASE | SERVER | URI`

REVOKE	This statement revokes roles or privileges on a specified object from groups. The object name is typically an identifier. For URIs, it is a string literal. Only administrative users (those with `ALL` privileges on the server, defined in the Sentry policy file) can use this statement. The revocation has a cascading effect.

last-updated: Apr 11, 2022

For example, revoking the `ALL` privilege on a database also revokes the same privilege for all the tables in that database.

Syntax:

```
REVOKE ROLE role_name FROM GROUP
group_name

REVOKE privilege ON object_type
object_name
FROM [ROLE] role_name
privilege ::= SELECT | INSERT | ALL
object_type ::= TABLE | DATABASE |
SERVER | URI
```

SHOW FILES

This statement displays the files that constitute a specified table or a partition within a partitioned table. Results include the names of the files, file sizes, and the applicable partition for a partitioned table. The size includes a suffix of B for bytes, MB for megabytes, and GB for gigabytes. SHOW FILES applies to tables and partitions stored on the MapR filesystem or S3. It does not apply to views or tables mapped to HBase. HBase does not use the same file-based storage layout.

Syntax: `SHOW FILES IN table`

SHOW GRANT ROLE

This statement lists all the grants for the given role name. This statement is only allowed for Sentry administrative users and other users that have been granted the specified role. This syntax is available when you are using the Sentry authorization framework along with the Sentry service. It does not apply when you use the Sentry framework with privileges defined in a policy file. When authorization is enabled, the output of the `SHOW` statement is limited to those objects for which you have some privilege, though there might be other concealed database and tables.

Syntax: `SHOWGRANTROLErole_name`

SHOW ROLES

This statement displays roles. This syntax is available when you are using Sentry authorization with the Sentry service. It does not apply when you use the Sentry framework with privileges defined in a policy file. When authorization is enabled, the output of the `SHOW` statement is limited to those objects for which you have some privilege, though there might be other concealed database and tables.

Syntax: `SHOW ROLES`

SHOW ROLEGRANT GROUP

This statement lists all the grants for the given group. This statement is only allowed for Sentry administrative users and other users that have been granted the specified role. This syntax is available when you are using the Sentry authorization framework along with the Sentry service. It does not apply when you use the Sentry framework with privileges defined in a policy file. When authorization is enabled, the output of the `SHOW` statement is limited to those objects for which you have some privilege, though there might be other concealed database and tables.

Syntax: `SHOWROLEGRANT GROUP group_name`

Data Types

The following table lists new or improved data types with their descriptions:

Data Type	Description
Complex	Impala can run queries against Parquet data containing columns with composite or nested data types.
CHAR	A fixed-length character type, padded with trailing spaces if necessary to achieve the specified length. If values are longer than the specified length, Impala truncates any trailing characters. The maximum length you can specify is 255. Provides enhanced support for `CHAR`in the COMPUTE STATS statement.
VARCHAR	A variable-length character type, truncated during processing if necessary to fit within the specified length. Short values are padded with spaces on the right. The maximum length you can specify is 65,535. Provides enhanced support for `VARCHAR` types in the COMPUTE STATS statement.

Clauses and Operators

The following table lists new or improved clauses and operators with descriptions and syntax:

[NOT] EXISTS

The EXISTS operator checks if a subquery returns any results. The NOT EXISTS operator finds values in a table that do not correspond to values in another table. You can use either of these operators in the WHERE clause of a subquery. Correlated subqueries used in the EXISTS operator cannot include a LIMIT clause.

Syntax: `[NOT]EXISTS (subquery)`

[NOT] IN

The IN operator compares an argument value to a set of values. If an argument value matches any value in a set, the result is TRUE. The NOT IN operator checks if the argument value is not part of a set of values. Correlated subqueries used in the IN operator cannot include a LIMIT clause.

Syntax: `expression [NOT] IN (expression [,expression])`

LEFT|RIGHT ANTI JOIN

These clauses return results from one table that has no match in another table. The LEFT ANTI JOIN clause returns those values from the left-hand table that have no matching value in the right-hand table. RIGHT ANTI JOIN reverses the comparison and returns values from the right-hand table.

Syntax:

```
SELECT select_list FROM
table_or_subquery1 {LEFT | RIGHT}
ANTI JOIN table_or_subquery2 |
[ ON col1 = col2 [AND col3 =
col4 ...] |
USING (col1 [, col2 ...]) ]
[other_join_clause ...]
[ WHERE where_clauses ]
```

Hint

The Impala SQL dialect supports query hints to fine-tune the inner workings of queries. You can specify hints as a temporary workaround for expensive queries where missing statistics or other factors cause inefficient performance.

Insert + immediately before a hint name to include hints inside comments that use `/* */` or `--` notation.

Example:

```
INSERT insert_clauses
/* +SHUFFLE|NOSHUFFLE */
SELECT remainder_of_query;
INSERT insert_clauses
-- +SHUFFLE|NOSHUFFLE
SELECT remainder_of_query;
```

New Features in Impala 1.4.1 for MapR

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Admission Control

A new feature that enforces limits on concurrent SQL queries and statements that run in an Impala cluster with heavy workloads. When admission control is enabled, Impala queues SQL queries and statements that it would otherwise cancel or re-run due to insufficient resources or performance bottlenecks.

You can configure admission control options that Impala uses to enforce limits on the following behaviors:

- Number of concurrent queries that run in the cluster
- Query queue size
- Number of delayed queries that the queue can hold
- Amount of memory that queries use
- Time that queries can exist in queue before Impala cancels them

Refer to Admission Control for more information.

Partition Pruning

Partition pruning now includes performance improvements that reduce the time spent on query planning for partitioned tables with thousands of partitions. Previously, Impala could query tables with up to approximately 3000 partitions. Now, Impala can comfortably query tables with tens of thousands of partitions.

Impala Daemon Options

The following table lists new Impala daemon start up options that you can add to the `env.sh` configuration file:

Option	Default	Description
load_catalog_in_background	true	New start up option to control the parallelism of metadata loading during start up for the catalogd daemon. Makes Impala use background threads after start up to load and cache metadata.

last-updated: Apr 11, 2022

num_metadata_loading_threads	16	New start up option for controlling the parallelism of metadata loading during start up for the catalogd daemon. Determines how much parallelism Impala devotes to loading metadata in the background. You can increase this value for systems with a huge number of databases, tables, or partitions. You can lower this value for busy systems that have CPU-constraints due to jobs from other components running in the cluster.
insert_inherit_permissions	Same permissions as parent directory	New start up option that causes Impala INSERT statements to create each new partition with the same MapR filesystem permissions as its parent directory. By default, INSERT statements create directories for new partitions using default MapR filesystem permissions.

Impala-Shell Command Line Options

The following table lists the new option with its description:

Option	Description
strict_unicode	The impala-shell interpreter now supports UTF-8 characters for input and output. You can control whetherimpala-shell ignores invalid Unicode code points through this option.

Security

The following table contains new security features and their descriptions:

Feature	Description
LDAP	LDAP connections can be secured through either SSL or TLS.
Sentry-based authorization	Impala can now use Sentry-based authorization based onthe original policy file.

Functions

The following table lists new functions and their descriptions:

Statement	Description
NULLIF() NULLIFZERO() ZEROIFNULL()	New conditional functions that simplify porting SQL containing vendor extensions to Impala.
CURRENT DATABASE()	New utility function that returns the database that the session is currently using.

Built-In Functions

The following table lists new built-in functions and their descriptions:

last-updated: Apr 11, 2022

Function	Description
ADD_MONTHS()	A built-in function that is an alias for the existing MONTHS_ADD() function.
EXTRACT()	A new built-in function that returns one date or time field from a TIMESTAMP value.
ROUND()	A new built-in function that rounds DECIMAL values to a specified number of fractional digits.
TRUNC()	A new built-in function that truncates date/time values to a particular granularity, such as year, month, day, hour, and so on.
STDDEV(), STDDEV_SAMP(), STDDEV_POP(), VARIANCE(), VARIANCE_SAMP(), VARIANCE_POP()	Built-in aggregate functions for computing properties for statistical distributions.
MAX_INT(), MIN_SMALLINT()	New built-in functions that you can use to check whether data values are in an expected range. You might be able to switch a column to a smaller type to save memory during processing.
IS_INF(), IS_NAN()	New built-in functions that check for special values infinity and “not a number”. These values could be specified as inf or nan in text data files. They may also be produced by certain arithmetic expressions.

Statements

The following table lists new or improved statements and their descriptions:

Statement	Description

last-updated: Apr 11, 2022

COMPUTE STATS	Use this statement to collect table and column statistics with a single statement. Useful for query planning, join queries, queries on partitioned tables, and any other data intensive operations. This release includes the following performance improvements for the statement: • The `NDV` function is sped up through native code generation. • Because the NULL count is not currently used by the Impala query planner, in Impala 1.4.0 and higher, `COMPUTE STATS` does not count the NULL values for each column. The #Nulls field of the stats table is left as -1, signifying that the value is unknown.
CREATE TABLE	SQL syntax for creating a table. This release includes the following improvements for the statement: • You can now specify the clause FIELDS TERMINATED BY '\0' to use text data files that use ASCII 0 (nul) characters as a delimiter. • For interoperability with Parquet files created through other Hadoop components, such as Pig or MapReduce applications, you can create an Impala table that automatically sets up the column definitions based on the layout of an existing Parquet data file. • Has a STORED AS AVRO clause, enabling you to create Avro tables through Impala.
EXPLAIN PLAN	When you run EXPLAIN PLAN followed by a SELECT query, it returns the query execution plan. EXPLAIN PLAN now provides more detail in a simpler format with the following verbosity levels: • 0 – Most concise • 1 • 2 • 3 – Most verbose
INVALIDATE METADATA table_name	Impala loads metadata for a Hive table when you issue the INVALIDATE METADATA table_name statement if a Hive table exists with the same name.

SHOW FUNCTIONS	Shows user-defined functions associated with a particular database. This release includes the following improvements for the statement: • Now displays the return type and argument type of each function. • You can now specify the clause FIELDS TERMINATED BY '\0' with a CREATE TABLE statement to use text data files that use ASCII 0 (nul) characters as a delimiter.
SHOW PARTITIONS	Displays information about each partition in a partitioned table. Run the COMPUTE STATS statement after creating all table partitions and then run SHOW PARTITIONS for more informative output.

Data Types

The following table lists new or improved data types with their descriptions:

Data Type	Description
DECIMAL	A numeric data type that you can use in CREATE TABLE and ALTER TABLE statements to store fixed-precision values when working with currency or other fractional values where it is important to represent values exactly and avoid rounding errors. Useful for calculations where the imprecise representation and rounding behavior of FLOAT and DOUBLE make them impractical to use. Includes enhancements to built-in functions, numeric literals, and arithmetic expressions.
TIMESTAMP	Accepts more input string formats through the UNIX TIMESTAMP function, and produces more string formats through the FROM UNIXTIME function.

Clauses and Operators

The following table lists new or improved clauses and operators with descriptions:

Feature	Description
ORDER BY	No longer requires a LIMIT clause. If the size of the result set to be sorted exceeds the memory available to Impala,Impala uses a temporary work space on disk to perform the sort operation.
REGEXP RLIKE	Improved compatibility with the regular expression support for popular database systems. There is no change to the behavior of the regexp_extract() and regexp_replace() built-in functions. These operators now match a regular expression string that occurs anywhere inside the target string; the same as if the regular expression was enclosed on each side by .*.

New Features in Impala 1.2.3 for MapR

last-updated: Apr 11, 2022

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Join Order Optimizations

A primary new feature of Impala 1.2.3 is join order optimizations that automatically distribute and parallelize the work for a join query to minimize disk I/O and network traffic. Automatic optimization reduces the need to use query hints or to rewrite join queries with the tables in a specific order based on size or importance.

Use the COMPUTE STATS statement when you want to gather critical, statistical information about each table when you enable join optimizations. If a join query is inefficient due to outdated statistics or unexpected data distribution, you can use the STRAIGHT_JOIN keyword immediately after the SELECT statement to prevent Impala from reordering the joined tables.

The STRAIGHT_JOIN keyword turns off the reordering of join clauses that Impala does internally, and produces a plan that relies on optimal ordering of the join clauses in the query text. For example, you rewrite the query so that the largest table is on the left, followed by the next largest, and so on until the smallest table is on the right.

Impala Catalog Service

When Impala changes table data or metadata on one node, the catalog service communicates the new or updated metadata to all the other Impala nodes. You no longer need to use the REFRESH or INVALIDATE METADATA statements after issuing the following statements to Impala:

- CREATE TABLE
- ALTER TABLE
- DROP TABLE
- INSERT
- LOAD DATA

Note:

The catalog service only works on operations performed through Impala. If you perform operations through the Hive shell or through the MapR filesystem, you must issue the REFRESH and INVALIDATE METADATA statements. The catalog service broadcasts the results of the REFRESH and INVALIDATE METADATA results to other Impala nodes so that you only have to issue the statements once.

Impala Daemon Options

The following table lists new Impala daemon startup options that you can add to the env.sh file:

Option	Description
--idle_query_timeout	Impala daemon option that controls how long a query can be idle before Impala cancels the query.
--idle_session_timeout	Impala daemon option that controls how long a session can be idle before the session expires.
--enable_ldap_auth	Enables LDAP authentication between Impala and the client. Example: -enable_ldap_auth=true \

last-updated: Apr 11, 2022

--ldap_uri	Sets the URI of the LDAP server to use. Typically, the URI is prefixed with ldap://<hostname>. Optionally, the URI can specify the port. For example, ldap://<hostname>:<port>. Example: -ldap_uri=ldap://10.250.1.5/ \
--ldap_manual_configextendingcolumn	Bypasses all the automatic configuration if you need to provide a custom SASL.
--ldap_tls	Tells Impala to start a TLS connection to the LDAP server and to fail authentication if Impala cannot start the TLS connection.

Security

The following table contains new security features and their descriptions:

Feature	Description
Impersonation support	Allows you to give users the permission to submit requests using the credentials of another user. Only available through the Hue interface.
LDAP authentication	You can configure LDAP authentication for client connections with Impala.

Functions

The following table lists new functions and their descriptions:

Function	Description
User-Defined Functions (UDFs)	Reusable SQL functions that you can create to encapsulate code that processes column values during an Impala query. You can run scalar UDFs and UDAs (user-defined aggregate functions). Impala accepts UDFs and UDAFs written in C++ or you can use Hive functions written in Java. For more information about user-defined functions, refer to Impala User-Defined Functions on page 3794.
NDV ()	An aggregate function that you can use to quickly return an approximate result instead of using COUNT (DISTINCT col).
GROUP_CONCAT()	An aggregate function that you can use to concatenate column values across all rows of a result set.

Built-In Functions

Several built-in functions and operators are now overloaded for more numeric data types to reduce CAST() function use with INSERT statements. Addition, subtraction, and multiplication functions now produce a result that is only one step bigger than their arguments. Numeric and conditional functions can return SMALLINT, FLOAT, and other smaller types instead of BIGINT or DOUBLE.

The following table lists new built-in functions and their descriptions:

Function	Description

last-updated: Apr 11, 2022

least ()	A mathematical function that returns the smallest value from a list of expressions. The return type is the same as the initial argument value, except that integer values are promoted to BIGINT and floating-point values are promoted to DOUBLE. Use CAST() when inserting into a smaller numeric column.
greatest ()	A mathematical function that returns the largest value from a list of expressions. The return type is the same as the initial argument value, except that integer values are promoted to BIGINT and floating-point values are promoted to DOUBLE. Use CAST() when inserting into a smaller numeric column.
initcap ()	Returns the input string with the first letter capitalized.
fnv_hash ()extending col	A mathematical function for constructing hashed values that returns a consistent 64-bit value derived from the input argument, for convenience of implementing hashing logic in an application. The return type is BIGINT.

Statements, Options, and Hints

The following table lists new statements, options, hints and their descriptions:

Statement	Description
COMPUTE STATS	You can use this statement to collect table and column statistics with a single statement. This is useful for query planning, join queries, queries on partitioned tables, and any other data intensive operations. Collect stats for each table involved in a join query.
CREATE TABLE AS SELECT	SQL syntax that you can use to create a table and copy data into the table in a single operation.
SHOW TABLE STATS	SQL syntax that you can use to verify that statistics are available and to see the values used during query planning.
SHOW COLUMN STATS	SQL syntax that you can use to verify that statistics are available and to see the values used during query planning.
SHOW CREATE TABLE	Summarizes the effects of the original CREATE TABLE statement and subsequent ALTER TABLE statements to provide a CREATE TABLE statement that recreates the current structure and layout for the table. You can use this statement to create a simplified setup script for a schema.
EXPLAIN_LEVEL	A query option that you can enable and use with the EXPLAIN and PROFILES statements to provide more verbose results. The verbose results provide estimated resource requirements and the available table and column statistics.
SYNC_DDL	A query option that you can enable before you issue a DDL, INSERT, or LOAD statement which causes the statement to wait and return only after the Catalog service broadcasts changes to all Impala nodes in the cluster.

last-updated: Apr 11, 2022

[SHUFFLE] [NOSHUFFLE]	You can use these hints in INSERT statements when inserting into Parquet tables to avoid problems due to memory consumption and open files in the filesystem, by collecting new data for each partition on a specific node.

Clauses and Operators

The following table lists new clauses, operators, and their descriptions:

Feature	Description
CROSS JOIN	A clause that you can use in the SELECT statement to allow joins without an equality comparison between columns in both tables.
NULLS FIRST	A clause that you can use to ensure consistent placement of NULL values in ORDER BY queries.
NULLS LAST	A clause that you can use to ensure consistent placement of NULL values in ORDER BY queries.
OFFSET	A clause that you can use with the ORDER BY and LIMIT clauses to produce paged result sets, like items 1-10, 11-20...
STORED AS PARQUET	You can use this clause instead of STORED AS PARQUETFILE for conciseness in new code.
TBLPROPERTIES	A clause that you can use with the CREATE TABLE and ALTER TABLE statements to associate random pieces of metadata with a table as key-value pairs.
WITH SERDEPROPERTIES	A clause that you can use with the CREATE TABLE and ALTER TABLE statements to specify the serializer and deserializer (SerDes) classes that read and write data for a table, which may be needed for Hive compatibility with Impala.
LIMIT	This clause accepts an arithmetic expression and numeric literals.
STRAIGHT_JOIN	You can use this operator to override the reordering of tables in a join query. Uses the original technique of ordering join tables in descending order of size enabling you to fine-tune the join query plan.

Additional Impala Configuration Options

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You can modify the `env.sh` file located in `/opt/mapr/impala/impala-<version>/conf/` to set certain Impala start up options.

Modifying Startup Options

The `env.sh` file contains values that the Impala server, statestore, and catalog services use during start up. The file also has information about resources allocated for Impala. Most of the default values in the `env.sh` file should work effectively, however there are some values that you can modify. You can check the current value of all the settings through the Impala web interface, available by default at `http://<impala-node-hostname>:25000/varz`.

You may want to modify the following content in the `env.sh` file:

- Statestore address

last-updated: Apr 11, 2022

- Amount of memory available to Impala
- Core dump enablement
- Session and query idle time

To modify the values, edit the `env.sh` file and then restart the Impala server, Impala statestore, and Impala catalog to implement the changes.

Example of some file content that you may want to modify:

```
HIVE_METASTORE_URI=thrift://localhost:9083
  # not needed if /opt/mapr/hive is configured
IMPALA_STATE_STORE_HOST=127.0.0.1
IMPALA_STATE_STORE_PORT=24000
IMPALA_BACKEND_PORT=22000
IMPALA_LOG_DIR=/opt/mapr/impala/impala-1.1.1/log
export IMPALA_STATE_STORE_ARGS=${IMPALA_STATE_STORE_ARGS:- \
-log_dir=${IMPALA_LOG_DIR}} \
-state_store_port=${IMPALA_STATE_STORE_PORT}
export IMPALA_SERVER_ARGS=${IMPALA_SERVER_ARGS:- -log_dir=${IMPALA_LOG_DIR}
\
-state_store_port=${IMPALA_STATE_STORE_PORT} \
-use_statestore
-state_store_host=${IMPALA_STATE_STORE_HOST} \
-be_port=${IMPALA_BACKEND_PORT}}
export ENABLE_CORE_DUMPS=false
```

The following table contains a list of settings that you can edit in `/opt/mapr/impala/impala-<version>/conf/env.sh` with descriptions for how to change them:

Setting	Description

Statestore address	You can modify this setting to change the statestore IP address or hostname.
	Example:
	If a machine with an IP address of 192.168.0.28 is hosting statestore, you can change
	`IMPALA_STATE_STORE_HOST=127.0.0.1`
	to
	`IMPALA_STATE_STORE_HOST=192.168.0.28.`

last-updated: Apr 11, 2022

Memory limits	If you run Impala on nodes that also run MapReduce, both frameworks may compete for memory. Configure memory based on your job requirements and SLAs to ensure that each framework has enough memory to avoid conflicts.
	
	You can include the `mem_limit` parameter in IMPALA_SERVER_ARGS to limit the amount of memory available to Impala. Use absolute notation, such as 500M or 2G, or a percentage of physical memory, such as 50% to specify the memory limit. Impala aborts queries that exceeds the specified memory limit. Percentage limits are based on the physical memory of the machine.
	
	You can also include the `num_threads_per_disk` parameter in IMPALA_SERVER_ARGS to limit the number of threads that each disk processes per impala server daemon. Limiting the number of threads per disk can reduce the overall amount of memory consumed.
	
	Example:
	
	To limit Impala to 50% of system memory, modify:
	
	`export IMPALA_SERVER_ARGS=${IMPALA_SERVER_ARGS:- \ -log_dir=${IMPALA_LOG_DIR} \ -state_store_port=${IMPALA_STATE_STORE_PORT} \ -use_statestore -state_store_host=${IMPALA_STATE_STORE_HOST} \ -be_port=${IMPALA_BACKEND_PORT}}`
	
	to
	
	`export IMPALA_SERVER_ARGS=${IMPALA_SERVER_ARGS:- \ -log_dir=${IMPALA_LOG_DIR} \ -state_store_port=${IMPALA_STATE_STORE_PORT} \ -use_statestore -state_store_host=${IMPALA_STATE_STORE_HOST} \ -be_port=${IMPALA_BACKEND_PORT} \ -mem_limit=50% \ -num_threads_per_disk=2}`

Core dump enablement	Core dump file locations can vary depending on your operating system configuration. Other security settings may prevent Impala from writing core dumps when you enable this option. To enable core dumps, change export `ENABLE_CORE_DUMPS=false` to `export ENABLE_CORE_DUMPS=true`.

last-updated: Apr 11, 2022

Session and query idle time	You can modify the time for which sessions and queries can remain idle by adding the following options to `env.sh`:
	`--idle_session_timeout`
	`--idle_query_timeout`
	Example:
	`IMPALA_SERVER_ARGS=" \ -log_dir=${IMPALA_LOG_DIR} \ -state_store_port=${IMPALA_STATE_STORE_PORT} \ -idle_query_timeout=<value in seconds> \ -idle_session_timeout=<value in seconds> \ -use_statestore \ -state_store_host=${IMPALA_STATE_STORE_HOST} \ -catalog_service_host=${CATALOG_SERVICE_HOST} \ -be_port=${IMPALA_BACKEND_PORT}"`

Use background threads to loadand cache metadata	The following option controls the parallelism of metadata loading during start up for the catalogd daemon and makes Impala use background threads after start up to load and cache metadata. `--load_catalog_in_background` The default setting is `true`.
Determine how much parallelism Impala devotes to loading metadata in background	The following option controls the parallelism of metadata loading during start up for the catalogd daemon and determines how much parallelism Impala devotes to loading metadata in the background: `--num_metadata_loading_threads` The default is 16, but you can increase this value for systems with a huge number of databases, tables, or partitions. You can lower this value for busy systems that have CPU-constraints due to jobs from other components running in the cluster.
INSERT to create new partition and inherit permissions	The following option causes Impala INSERT statements to create each new partition with the same MapR filesystem permissions as its parent directory: `--insert_inherit_permissions` By default, INSERT statements create directories for new partitions using default MapR filesystem permissions.

After you edit `/opt/mapr/impala/impala-<version>/conf/env.sh`, use the following commands to restart the Impala server and services:

last-updated: Apr 11, 2022

- Issue the following command to restart the Impala server:

```
$ sudo maprcli node services -name impalaserver -action restart -nodes
<IP address where impala server is installed>
```

- Issue the following command to restart the Impala statestore:

```
$ sudo maprcli node services -name impalastore -action restart -nodes <IP
address where impala statestore is installed>
```

- Issue the following command to restart the Impala catalog:

```
$ sudo maprcli node services -name impalacatalog -action restart -nodes
<IP address where impala catalog is installed>
```

Admission Control

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Admission control enforces limits on concurrent SQL queries and statements that run in an Impala cluster with heavy workloads. When admission control is enabled, Impala queues SQL queries and statements that it would otherwise cancel or re-run due to insufficient resources or performance bottlenecks.

You can configure admission control options that Impala uses to enforce limits on the following behaviors:

- Number of concurrent queries that run in the cluster
- Query queue size
- Number of delayed queries that the queue can hold
- Amount of memory that queries use
- Time that queries can exist in queue before Impala cancels them

Query Queuing

Admission control is embedded within each impalad daemon and communicates through the statestore service. The impalad daemon determines if a query runs immediately or if the query is queued. The queue can include queries submitted through multiple Impala nodes.

Impala executes queries in the order that they are submitted to allow for dependent statements, such as a CREATE TABLE statement followed by an INSERT INTO statement. Impala may not execute queries submitted through different nodes in the order that the queries were received. When Impala must execute a sequence of statements in order, you should submit the statements on the same Impala node within a single session. When statement order is not important, you can set up your table structures and submit statements through different Impala nodes.

If a sudden flow of requests causes more queries to run concurrently than expected, the overall Impala memory limit and the Linux cgroups mechanism serve as hard limits to prevent over allocation of memory. When queries hit these limits, Impala cancels the queries.

Configuring Admission Control

You can use a combination of Impala start-up options, and optionally edit configuration settings in `fair-scheduler.xml`, to configure admission control. Admission control uses the Fair Scheduler configuration settings to determine how to map users and groups to different resource pools.

Fair Scheduler Settings

You can configure settings in fair-scheduler.xml that admission control can use to determine how to map users and groups to different resource pools. The `<aclSubmitApps>` tag in fair-scheduler.xml contains users and groups that can submit Impala statements to a corresponding Impala resource pool. The user and group lists are separated by a space, as shown in the following example:

```
<aclSubmitApps>user1,user2,user3 dev,tech,admin</aclSubmitApps>
```

If the `<aclSubmitApps>` tag is empty for a pool, no one can submit directly to that pool, however child pools can have their own `<aclSubmitApps>` values to allow users and groups to submit to the child pools.

Impala does not use the `vcores` and `disks` values, however you must specify them to satisfy YARN requirements for the file content. For more information about Fair Scheduler configuration settings, refer to the Apache wiki.

The following fair-scheduler.xml shows examples of the `<aclSubmitApps>` tag and also shows the `vcores` and `disks` values:

```
<allocations>
    <queue name="root">
        <aclSubmitApps> </aclSubmitApps>
        <queue name="default">
            <maxResources>40000 mb, 0 vcores, 0 disks</maxResources>
            <aclSubmitApps>*</aclSubmitApps>
        </queue>
        <queue name="dev">
            <maxResources>100000 mb, 0 vcores, 0 disks</maxResources>
            <aclSubmitApps>user1,user2 dev,tech,admin</aclSubmitApps>
        </queue>
        <queue name="prod">
            <maxResources>2000000 mb, 0 vcores, 0 disks</maxResources>
            <aclSubmitApps> tech,admin</aclSubmitApps>
        </queue>
    </queue>
    <queuePlacementPolicy>
        <rule name="specified" create="false"/>
        <rule name="default" />
    </queuePlacementPolicy>
</allocations>
```

Impala Start-up Options

To configure admission control, modify Impala's start-up options. You can modify the start-up options in

```
/opt/mapr/impala/impala-<version>/conf/env.sh.
```

For more information about how to modify Impala start-up options, refer to Additional Impala Configuration Options.

The following table lists the admission control start-up options that you can configure:

last-updated: Apr 11, 2022

Option	Type	Default	Description
-default_pool_max_queued	int64	0	The maximum number of requests allowed in the queue. Impala rejects additional requests when the queue reaches this limit. This a “soft” limit that applies cluster-wide. Each Impala node decides independently whether to run queries immediately or to queue them. Impala allows for the overall number of queued queries to be slightly higher that the limit during times of heavy load. A negative value or 0 indicates that requests are always rejected once the maximum concurrent requests are executing. Ignored if fair_scheduler_config_path is set.
-default_pool_max_requests	int64	-1	The maximum number of concurrent requests allowed to run before incoming requests are queued. This a “soft” limit that applies cluster-wide. Each Impala node decides independently whether to run queries immediately or to queue them. The overall number of concurrent queries might be slightly higher during times of heavy load. A negative value indicates no limit. Ignored if fair_scheduler_config_path is set.

last-updated: Apr 11, 2022

Option	Type	Default	Description
-default_pool_mem_limit	string	"" (empty string)	The maximum amount of memory that all outstanding requests in this pool can use before new requests to this pool are queued. Specified in bytes, megabytes, or gigabytes by a number followed by the suffix b (optional), m, or g, either upper- or lowercase. You can specify floating-point values for megabytes and gigabytes, to represent fractional numbers such as 1.5. You can also specify it as a percentage of the physical memory by specifying the suffix %. 0 or no setting indicates no limit. Defaults to bytes if no unit is given. This is a soft limit applied cluster-wide. Each Impala node makes independent decisions to run queries immediately or queue them, so the overall memory used by concurrent queries might be slightly higher during times of heavy load. Ignored if fair_scheduler_config_path is set. **Note:** Impala relies on the statistics produced by the COMPUTE STATS statement to estimate memory usage for each query.
-disable_admission_control	Boolean	false	Turns off the admission control feature entirely, regardless of other configuration option settings.
-disable_pool_max_requests	Boolean	false	Disables all per-pool limits on the maximum number of running requests.
-disable_pool_mem_limits	Boolean	false	Disables all per-pool memory limits.
-fair_scheduler_allocation_path	String	"" (empty string)	Path to the Fair Scheduler allocation file, `fair-scheduler.xml`. Admission control can only use a small subset of the settings that can go in this file.

last-updated: Apr 11, 2022

Option	Type	Default	Description
-queue_wait_timeout_ms	int64	60000	The maximum amount of time (in milliseconds) that a request waits in queue to be executed before timing out.

Admission Control with Clients

Admission control works with JDBC and ODBC client interfaces, however you may experience the following scenarios due to limits enforced by this feature:

- The API call blocks SQL statements in the query queue instead of running them immediately. Query execution begins when the statement moves out of the query queue, at which time the client program can request the results, which may also block until they become available.
- If a SQL statement is canceled due to prolonged queue time or because it exceeded the memory limit during execution, an error occurs and the client program receives an error message.

You cannot set the following options from JDBC or ODBC applications:

- You must set the REQUEST_POOL option for a session through the impala-shell interpreter, or through the Impala start-up options if you want the setting to apply cluster-wide.
- You must set the MEM_LIMIT query option through the impala-shell interpreter. It cannot be used directly through JDBC or ODBC applications.

Admission Control Guidelines

The admission control system is not aware of other Hadoop workloads, such as MapReduce applications.

The following table lists some admission control guidelines to follow:

Guideline	Description
Examine query profile output	Examine the profile output for a query to see how admission control works for the query. The profile output provides details about the admission decision, such as whether the query was queued or not and which resource pool it was assigned to. It also includes the estimated and actual memory usage for the query, so you can fine-tune the configuration for the memory limits of the resource pools. In impala-shell, you can also specify which resource pool to direct queries to by setting the REQUEST_POOL query option. You can run the PROFILE statement in the impala-shell right after you run the query to see the query output, or you can review the Impala log file.
You cannot use admission control with Hue deployed	Unclosed Hue queries accumulate and exceed the queue size limit. To use admission control, you must explicitly enable it by specifying --disable_admission_control=false in the impalad command-line options safety valve field.
Set the MEM_LIMIT query option to override the query estimated memory usage	When a query cannot run due to high estimated memory usage, set the MEM_LIMIT query option in the impala-shell and issue the query through the shell in the same session to override the estimate. Impala treats the MEM_LIMIT value as the estimated amount of memory and overrides the estimate that Impala would generate based on table and column statistics. This value is used only for making admission control decisions, and is not pre-allocated by the query.

last-updated: Apr 11, 2022

Guideline	Description
Increase memory if needed when inserting into Parquet tables	Admission control affects query statements, as well as INSERT and CTAS. Inserting into a Parquet tables is memory intensive because 1GB of data is buffered before writing out each Parquet data block. When inserting into a partitioned Parquet table, Impala redistributes the data among the nodes to reduce memory consumption. You may need to temporarily increase the memory dedicated to Impala during the insert operation, or break up the load operation into several INSERT statements, or both.
Limits on queued queries affect subsequent statements in the same session	If Impala queues a query due to a limit on concurrent queries or memory usage, subsequent statements in the same session are also queued to ensure that the statements are processed in the correct order.
Reuse classifications and hierarchy developed for use with Sentry security	If you set up different resource pools for different users and groups, consider reusing any classifications and hierarchy you developed for use with Sentry security. See Enabling Sentry Authorization for Impala for details. For details about all the Fair Scheduler configuration settings, see the Apache wiki, in particular the tags such as <queue> and <aclSubmitApps> to map users and groups to particular resource pools (queues).
Use the COMPUTE STATS statement for large tables involved in join queries	Although COMPUTE STATS is an important statement to help optimize query performance, it is especially important when admission control is enabled. Admission control relies on COMPUTE STATS to generate accurate memory usage estimates for complex queries.

Spill to Disk

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Impala 2.2.0 for MapR introduces the "spill to disk" feature which prevents queries that use memory-intensive operations, such as large sort, join, aggregation, or analytic function operations, from failing with out-of-memory errors. Impala automatically writes data to disk when queries with memory-intensive operations exceed the set memory limit on an Impala node. You can enable or disable the spill to disk feature in `/opt/mapr/impala/impala-<version>/conf/env.sh`. You can also change the DISABLE_UNSAFE_SPILLS setting at the session level to allow or prevent data spills.

Impala reserves a certain amount of memory for memory-intensive operations that reach the set memory threshold. This memory reserve serves as a buffer that slows an operation's use of memory and in some instances prevents operations from exceeding the memory limit on a node. If the amount of memory used exceeds the set memory limit, Impala starts writing data to disk (spilling to disk).

Impala writes data to a temporary work area on disk. The default location of this work area is `/tmp/impala-scratch`. You can start the impalad daemon with the `--scratch_dirs="path_to_directory"` configuration option to change the directory location, however the scratch directory must reside on the local filesystem. You can specify a single directory, or a comma-separated list of directories. If there is less than 1 GB of available space on the filesystem where the scratch directory resides, Impala writes a warning message to its log when it runs. When an operation completes, the data is removed from the disk.

Prevent Data from Spilling to Disk

Spilling to disk can have significant performance implications due to the additional I/O required. Ideally, you want to optimize queries, system parameters, and the hardware configurations to prevent data from spilling to disk. The following points describe some techniques that can prevent data from spilling to disk:

last-updated: Apr 11, 2022

- View the query profile after you run queries to determine how often they spill to disk and how much temporary data is written to disk. You can run the PROFILE command from the impala-shell or you can view the Impala debug web user interface to see the profile for the query.
- Increase the number of Impala nodes in the cluster to increase the aggregate memory available.
- Increase the overall memory capacity of each node at the hardware level.
- On a multi-tenant cluster, use resource management features to allocate more memory for Impala.
- If you are running concurrent queries rather than a few memory-intensive ones, use the admission control feature to lower the limit on the number of concurrent queries. You can schedule the most resource-intensive queries to avoid spikes in memory usage and improve overall response times. See Admission Control.
- Use the following techniques to tune the queries with the highest memory requirements:
  - Run the COMPUTE STATS statement for all tables involved in large-scale joins and aggregation queries.
  - Minimize the use of STRING columns in join columns. Use numeric values instead.
  - Evaluate the EXPLAIN plan to see the execution plan used for the most resource-intensive queries.
  - Add hints to the most resource-intensive queries to select the right execution strategy.
  - If your queries experience substantial performance overhead due to spilling, enable the DISABLE_UNSAFE_SPILLS query option to prevent queries from spilling to disk.

DISABLE_UNSAFE_SPILLS

You can categorize queries that spill to disk as safe or unsafe. A spill that results from optimized queries may represent a safe spill, whereas a spill that results from running large ad-hoc queries with unknown performance implications may represent an unsafe spill. For example, if a query does not include a hint to set the most efficient mechanism for a join or an INSERT ... SELECT into a partitioned table, these tables will most likely result in suboptimal execution plans that could cause unnecessary spilling.

When DISABLE_UNSAFE_SPILLS is enabled (set to "`true`"), you can run the COMPUTE STATS statement on large tables involved in join queries and resource-intensive operations. The COMPUTE STATS statement collects table and column statistics to help estimate memory usage for each query before you allow data to spill to disk.

If you have determined that your queries will spill safely to disk, run the following command to turn the DISABLE_UNSAFE_SPILLS option off:

```
set DISABLE_UNSAFE_SPILLS=false
```

Disable the Spill to Disk Feature

You can disable the spill to disk feature if spilling to disk negatively affects performance or you do not know the performance implications or memory usage of queries in advance.

To disable the spill to disk feature, complete the following steps:

1. On each node running the Impala server, edit `/opt/mapr/impala/impala-<version>/conf/env.sh` , and set the following options to “`false`”:

   ```
   -enable_partitioned_aggregation=false
   -enable_partitioned_hash_join=false
   ```


last-updated: Apr 11, 2022

2. Save the `env.sh` file and then run the following commands to restart the Impala service and instances:
   a. Run the following command to restart the Impala server:

```
$ sudo maprcli node services -name impalaserver -action
restart -nodes <IP address where impala server is installed>
```

   b. Run the following command to restart the Impala statestore:

```
$ sudo maprcli node services -name impalastore -action restart -nodes
<IP address where impala statestore is installed>
```

   c. Run the following command to restart the Impala catalog:

```
$ sudo maprcli node services -name impalacatalog -action
restart -nodes <IP address where impala catalog is installed>
```

Working with Impala

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

After you start Impala, use the impala-shell or a JDBC or ODBC client to query data. You can query data stored in files, as well as data stored in MapR Database tables. Impala depends on the Hive metastore to track table metadata. The MapR filesystem tracks the metadata of other files. Impala supports Text and Parquet file formats. If you want to query data using SequenceFile, RCFile, and Avro file formats, use Hive to load the data. Impala supports Snappy, GZIP, Deflate, and BZIP compression codecs.

Impala File Formats

The following table summarizes the supported Impala text formats:

File Type	Format	Compression Codecs	Can Impala Create?	Can Impala INSERT?
Text	Unstructured	Snappy, GZIP, BZIP	Yes, for CREATE TABLE with no STORED AS clause;default file format is uncompressed text with values separated by ASCII 0x01 characters, typically represented a Ctrl-A	Yes. CREATE TABLE, INSERT, and query.
SequenceFile	Structured	Snappy, GZIP, deflate, BZIP2	Yes	No. Query only. Load data using Hive.
RCFile	Structured	Snappy, GZIP, deflate, BZIP2	Yes	No. Query only. Load data using Hive.
Parquet	Structured	Snappy (default), GZIP	Yes	Yes. CREATE TABLE, INSERT, and query.
Avro	Structured	Snappy, GZIP, deflate, BZIP2	No, create using Hive.	No. Query only. Load data using Hive.

Impala SQL Dialect

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

last-updated: Apr 11, 2022

Impala uses the SQL query language and is compatible with the Hive query language, HiveQL. You can use other languages, such as Java, to interact with Impala through ODBC and JDBC interfaces. The Impala SQL dialect supports a subset of SQL and HiveQL functions, statements, datatypes, operators, and built-in functions.

The Impala SQL dialect supports DML statements similar to the DML component of HiveQL. The Impala SQL dialect does not support UPDATE and DELETE statements, and does not support the INSERT… VALUES syntax to insert a single row.

Refer to Supported and Unsupported SQL/HiveQL Language Features for a list of supported and unsupported functions, statements, datatypes, operators, and features.

Example: Running an Impala SQL Query

In this example scenario, download a customer CSV file and use the Hive shell to create a table and import customer data into the table and then run an Impala query on the table.

1. Download the following CSV file to `/root/customers_sample_data.csv`:

   customers_sample_data.csv

2. Issue the following command from the hive-shell to import the CSV file and create a table:

```
hive> create table customers(FirstName string,
LastName string,Company string,Address string,
City string,County string,State string,Zip string,
Phone string,Fax string,Email string,Web string)
row format delimited fields terminated by ',' stored as textfile;
```

3. Issue the following command in the hive-shell to load the customer data into the customers table:

```
Hive> load data local inpath '/root/customers_sample_data.csv' overwrite
into table customers;
```

4. Issue the following command to start the Impala shell:

```
$ impala-shell
```

5. To connect to an instance of Impala, issue the following CONNECT command, replacing `impalad-host` with the host name you have configured on a node running Impala:

```
[Not connected] > connect impalad-host
[impalad-host:21000] >
```

6. Issue the following command to query the data to find the total number of customers:

```
select count(*) from customers
```

The query returns the following result:

```
+--------+
| count(*) |
+--------+
| 6 |
+--------+
```


last-updated: Apr 11, 2022

Query MapR Database Binary Tables with Impala

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You can use Impala to query data in MapR Database binary tables. Create an external table in the Hive shell and then map the Hive table to the corresponding MapR Database binary table. You can map a MapR Database binary table to a Hive table with or without string row keys. When you create an external table in Hive, use the HBaseStorageHandler clause in the Hive CREATE TABLE statement to allow Hive to access data stored in the MapR Database binary table. The HBaseStorageHandler has two important properties:

`hbase.columns.mapping`	Specifies the Hive column to column family mapping.
`hbase.table.name`	Absolute path of the table or just the table name, depending on whether the table path is mapped.

Example:

```
CREATE EXTERNAL TABLE students
(id string,
name string,
street string,
zipcode int,
state string)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler' WITH
SERDEPROPERTIES
('hbase.columns.mapping'=':key,account:name,address:street,address:zipcode,a
ddress:state')
TBLPROPERTIES ('hbase.table.name'='/user/userA/students');
```

Once you have mapped the MapR Database binary table to Hive, you can query or insert into the table from Impala because Hive and Impala share the metastore database. Impala nodes cache table metadata if a table contains a large amount of data or has many partitions. Caching the metadata reduces runtime for future queries on the table.

If you insert new data into a Hive or MapR Database binary table, use the Impala shell to issue the REFRESH statement to refresh the data location cache. The REFRESH command only applies to the node that the Impala shell is connected to. If you route all SQL statements to the same node, you do not have to issue regular REFRESH statements when table data is updated on other nodes.

If you create, drop, or alter any external tables or databases, use the Impala shell to issue the INVALIDATE METADATA statement to refresh table structure metadata.

To ensure Impala is quering MapR Database binary tables and not HBase, add the next property to `$HBASE_HOME/conf/hbase-site.xml`:

```
<property>
  <name>mapr.hbase.default.db</name>
  <value>maprdb</value>
</property>
```

Example: Running an Impala Query on MapR Database Binary Tables

In this example scenario, a professor wants to know how many times a student clicks on Google from his webpage. He wants to use Impala to query the data in MapR Database. One of his students offered to load the data into MapR Database so he can access it. In order to complete the professor's request, the student must use the HBase shell to create two MapR Database binary tables that contain the following schema and then put data in the tables:

- student
  - account – id, name

last-updated: Apr 11, 2022

  - address – street, zipcode, state
- clicks
  - clickinfo – clickid, studentid, url, time
  - iteminfo – itemid, quantity

Each bullet corresponds to a column family with a list of columns. In order to access the tables using Impala, the student must create external tables in Hive with mapped columns that match the MapR Database columns.

If you would like to be the student, you can perform the following steps to help the professor:

1. Use the HBase shell to create two tables in MapR Database: “student” and “clicks”. To create the tables, issue the following commands:

```
echo "create '/user/userA/students','account','address'" | hbase shell
echo "create '/user/userA/clicks','clickinfo','iteminfo'" | hbase shell
```

2. Issue the `hadoop fs -ls` command on the table location to verify that the tables exist.

```
hadoop fs -ls /user/userA
echo "describe '/user/usera/student'" | hbase shell
echo "describe '/user/usera/clicks'" | hbase shell
```

3. Create external tables in Hive with the appropriate column mapping for the “student” and “clicks” tables using a string row key. Remember the two important properties for HBaseStorageHandler:
   - `hbase.columns.mapping`, which specifies Hive column to column family mapping.
   - `hbase.table.name`, which can be the absolute path of the table or just the table name, depending on whether the table path is mapped.

   To create the external tables in Hive, run the following commands:

```
CREATE EXTERNAL TABLE students
(id string,
name string,
street string,
zipcode int,
state string)
STORED BY
'org.apache.hadoop.hive.hbase.HBaseStorageHandler' WITH SERDEPROPERTIES
('hbase.columns.mapping'=':key,account:name,address:street,address:zipcod
e,address:state')
TBLPROPERTIES
('hbase.table.name'='/user/userA/students');

CREATE EXTERNAL TABLE clicks
(clickid string,
studentid string,
url string,
time timestamp,
itemtype string,
quantity int)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ('hbase.columns.mapping'=':key, clickinfo:studentid,
clickinfo:url, clickinfo:time, iteminfo:itemtype, iteminfo:quantity')
TBLPROPERTIES
('hbase.table.name'='/user/userA/clicks');
```


last-updated: Apr 11, 2022

4. Create a `testdata.txt` file with the following content to add data into the “students” and “clicks” MapR Database binary tables.

```
cat > testdata.txt
put '/user/userA/students','student1','account:name','Alice'
put '/user/userA/students','student1','address:street','123 Ballmer Av'
put '/user/userA/students','student1','address:zipcode','12345'
put '/user/userA/students','student1','address:state','CA'
put '/user/userA/students','student2','account:name','Bob'
put '/user/userA/students','student2','address:street','1 Infinite Loop'
put '/user/userA/students','student2','address:zipcode','12345'
put '/user/userA/students','student2','address:state','CA'
put '/user/userA/students','student3','account:name','Frank'
put '/user/userA/students','student3','address:street','435 Walker Ct'
put '/user/userA/students','student3','address:zipcode','12345'
put '/user/userA/students','student3','address:state','CA'
put '/user/userA/students','student4','account:name','Mary'
put '/user/userA/students','student4','address:street','56 Southern Pkwy'
put '/user/userA/students','student4','address:zipcode','12345'
put '/user/userA/students','student4','address:state','CA'
put '/user/userA/clicks','click1','clickinfo:studentid','student1'
put '/user/userA/clicks','click1','clickinfo:url','http://www.google.com'
put '/user/userA/clicks','click1','clickinfo:time','2014-01-01
12:01:01.0001'
put '/user/userA/clicks','click1','iteminfo:itemtype','image'
put '/user/userA/clicks','click1','iteminfo:quantity','1'
put '/user/userA/clicks','click2','clickinfo:studentid','student1'
put '/user/userA/clicks','click2','clickinfo:url','http://www.amazon.com'
put '/user/userA/clicks','click2','clickinfo:time','2014-01-01
01:01:01.0001'
put '/user/userA/clicks','click2','iteminfo:itemtype','image'
put '/user/userA/clicks','click2','iteminfo:quantity','1'
put '/user/userA/clicks','click3','clickinfo:studentid','student2'
put '/user/userA/clicks','click3','clickinfo:url','http://www.google.com'
put '/user/userA/clicks','click3','clickinfo:time','2014-01-01
01:02:01.0001'
put '/user/userA/clicks','click3','iteminfo:itemtype','text'
put '/user/userA/clicks','click3','iteminfo:quantity','2'
put '/user/userA/clicks','click4','clickinfo:studentid','student2'
put '/user/userA/clicks','click4','clickinfo:url','http://www.ask.com'
put '/user/userA/clicks','click4','clickinfo:time','2013-02-01
12:01:01.0001'
put '/user/userA/clicks','click4','iteminfo:itemtype','text'
put '/user/userA/clicks','click4','iteminfo:quantity','5'
put '/user/userA/clicks','click5','clickinfo:studentid','student2'
put '/user/userA/clicks','click5','clickinfo:url','http://
www.reuters.com'
put '/user/userA/clicks','click5','clickinfo:time','2013-02-01
12:01:01.0001'
put '/user/userA/clicks','click5','iteminfo:itemtype','text'
put '/user/userA/clicks','click5','iteminfo:quantity','100'
put '/user/userA/clicks','click6','clickinfo:studentid','student3'
put '/user/userA/clicks','click6','clickinfo:url','http://www.google.com'
put '/user/userA/clicks','click6','clickinfo:time','2013-02-01
12:01:01.0001'
put '/user/userA/clicks','click6','iteminfo:itemtype','image'
put '/user/userA/clicks','click6','iteminfo:quantity','1'
put '/user/userA/clicks','click7','clickinfo:studentid','student3'
put '/user/userA/clicks','click7','clickinfo:url','http://www.ask.com'
put '/user/userA/clicks','click7','clickinfo:time','2013-02-01
12:45:01.0001'
put '/user/userA/clicks','click7','iteminfo:itemtype','image'
put '/user/userA/clicks','click7','iteminfo:quantity','10'
```


last-updated: Apr 11, 2022

```
put '/user/userA/clicks','click8','clickinfo:studentid','student4'
put '/user/userA/clicks','click8','clickinfo:url','http://www.amazon.com'
put '/user/userA/clicks','click8','clickinfo:time','2013-02-01
22:01:01.0001'
put '/user/userA/clicks','click8','iteminfo:itemtype','image'
put '/user/userA/clicks','click8','iteminfo:quantity','1'
put '/user/userA/clicks','click9','clickinfo:studentid','student4'
put '/user/userA/clicks','click9','clickinfo:url','http://www.amazon.com'
put '/user/userA/clicks','click9','clickinfo:time','2013-02-01
22:01:01.0001'
put '/user/userA/clicks','click9','iteminfo:itemtype','image'
put '/user/userA/clicks','click9','iteminfo:quantity','10'
```

5. Press Control + Z to finish editing the file and pipe these commands to the HBase shell to insert the test data:

```
cat testdata.txt | hbase shell
```

6. Scan the tables to verify that the data was inserted correctly. Run the following commands to perform the scan:

```
echo "scan '/user/userA/students'" | hbase shell
echo "scan '/user/userA/clicks'" | hbase shell
```

7. Use Hive to verify that the data was inserted into the tables. Issue the SELECT statement against students and clicks to verify the count in each table.

```
hive
hive> select * from students;
OK
student1        Alice   123 Ballmer Av  12345   CA
student2        Bob     1 Infinite Loop 12345   CA
student3        Frank   435 Walker Ct   12345   CA
student4        Mary    56 Southern Pkwy        12345   CA
hive> select * from clicks;
OK
click1  student1        http://www.google.com   2014-01-01
12:01:01.0001        image   1
click2  student1        http://www.amazon.com   2014-01-01
01:01:01.0001        image   1
click3  student2        http://www.google.com   2014-01-01
01:02:01.0001        text    2
click4  student2        http://www.ask.com      2013-02-01
12:01:01.0001        text    5
click5  student2        http://www.reuters.com  2013-02-01
12:01:01.0001        text    100
click6  student3        http://www.google.com   2013-02-01
12:01:01.0001        image   1
click7  student3        http://www.ask.com      2013-02-01
12:45:01.0001        image   10
click8  student4        http://www.amazon.com   2013-02-01
22:01:01.0001        image   1
click9  student4        http://www.amazon.com   2013-02-01
22:01:01.0001        image   10
```

Since the Impala shell was running when you inserted the data, verify that the metadata is refreshed to make sure that Impala is aware of the new tables created.

last-updated: Apr 11, 2022

8. From the Impala shell , issue the INVALIDATE METADATA statement to refresh the metadata.

```
> invalidate metadata;
> select * from students ;
Query: select * from students
Query finished, fetching results ...
+----------+-------+-------+------------------+---------+
| id       | name  | state | street           | zipcode |
+----------+-------+-------+------------------+---------+
| student1 | Alice | CA    | 123 Ballmer Av   | 12345   |
| student2 | Bob   | CA    | 1 Infinite Loop  | 12345   |
| student3 | Frank | CA    | 435 Walker Ct    | 12345   |
| student4 | Mary  | CA    | 56 Southern Pkwy | 12345   |
+----------+-------+-------+------------------+---------+
select count(*) from clicks;
> select * from clicks;
Query: select * from clicks
Query finished, fetching results ...
+---------+-----------+-------------------------------+------------------------+-----

| clickid | studentid | time                          |
url                    | itemtype | quantity |
+---------+-----------+-------------------------------+------------------------+-----

| click1  | student1  | 2014-01-01 12:01:01.000100000 | http://
www.google.com  | image    | 1        |
| click2  | student1  | 2014-01-01 01:01:01.000100000 | http://
www.amazon.com  | image    | 1        |
| click3  | student2  | 2014-01-01 01:02:01.000100000 | http://
www.google.com  | text     | 2        |
| click4  | student2  | 2013-02-01 12:01:01.000100000 | http://
www.ask.com     | text     | 5        |
| click5  | student2  | 2013-02-01 12:01:01.000100000 | http://
www.reuters.com | text     | 100      |
| click6  | student3  | 2013-02-01 12:01:01.000100000 | http://
www.google.com  | image    | 1        |
| click7  | student3  | 2013-02-01 12:45:01.000100000 | http://
www.ask.com     | image    | 10       |
| click8  | student4  | 2013-02-01 22:01:01.000100000 | http://
www.amazon.com  | image    | 1        |
| click9  | student4  | 2013-02-01 22:01:01.000100000 | http://
www.amazon.com  | image    | 10       |
+---------+-----------+-------------------------------+------------------------+-----
```


last-updated: Apr 11, 2022

9. To query the tables and to find out which students clicked on google.com, run the following command from the Impala shell:

```
> select * from clicks where url like '%google%';
Query: select * from clicks where url like '%google%'
Query finished, fetching results ...
+---------+-----------+-------------------------------+----------------------+------
| clickid | studentid | time                          |
url                  | itemtype | quantity |
+---------+-----------+-------------------------------+----------------------+------
| click1  | student1  | 2014-01-01 12:01:01.000100000 | http://
www.google.com | image    | 1        |
| click3  | student2  | 2014-01-01 01:02:01.000100000 | http://
www.google.com | text     | 2        |
| click6  | student3  | 2013-02-01 12:01:01.000100000 | http://
www.google.com | image    | 1        |
+---------+-----------+-------------------------------+----------------------+------
```

Managing Impala

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

To manage Impala, you can start and stop the Impala services, modify resource allocation, and use log files to identify and resolve issues.

Starting/Stopping Impala
From the Control System

You must start the statestore service before you start the catalog or the Impala service.

To start the Impala statestore and catalog services from the Control System, complete the following steps:

1. In the Navigation pane, expand the Cluster Views pane and click Dashboard.
2. In the Services pane, click Impala statestore. The Nodes screen appears and displays the node configured with the statestore service.
3. Click the hostname of the node with the statestore and catalog services configured to display the Node Properties screen.
4. Use the Stop/Start button in the Impala statestore row under Manage Services to start Impala statestore.
5. Use the Stop/Start button in the Impala catalog row under Manage Services to start Impala catalog.

To start Impala server from the Control System, complete the following steps:

1. In the Navigation pane, expand the Cluster Views pane and click Dashboard.
2. In the Services pane, click Impala server. The Nodes screen appears and displays the nodes configured with the Impala server.
3. Click the hostname of the a node with the Impala server configured to display its Node Properties screen.
4. Use the Stop/Start button in the Impala server row under Manage Services to start the Impala server.
5. Repeat steps 3 and 4 for the remaining nodes configured with the Impala server.

last-updated: Apr 11, 2022

From the CLI

You must start the statestore service before you start the catalog or the Impala service.

To start Impala from the command line, complete the following steps:

1. Issue the following command to start the statestore service on the node with statestored:

```
$ sudo maprcli node services -name impalastore -action start|stop -nodes
<node IP addresses separated by a space>
Example:
$ sudo maprcli node services -name impalastore -action start -nodes
10.10.30.166
```

2. Issue the following command to start the catalog service on the node with catalogd:

```
$ sudo maprcli node services -name impalacatalog -action start|
stop -nodes <node IP addresses separated by a space>
Example:
$ sudo maprcli node services -name impalacatalog -action start -nodes
10.10.30.166
```

3. Issue the following command to start the Impala service on the node(s) with impalad:

```
$ sudo maprcli node services -name impalaserver -action start|
stop -nodes < node IP addresses separated by a space >
Example:
$ sudo maprcli node services -name impalaserver -action start -nodes
10.10.30.166
```

4. Optionally, you can run the following command to launch the impala-shell if you want to issue queries from the command line:

```
${IMPALA_HOME}/bin/impala-shell.sh
```

Modify Resource Allocation

If you run Impala on nodes that also run MapReduce, both frameworks may suffer poor performance if they have to compete for resources. You can configure memory based on your job requirements and SLAs to ensure that each framework has enough resources to avoid conflicts. For information about modifying memory, refer to Additional Impala Configuration Options.

Impala Logs

Impala logs provide information errors, configuration, and completed jobs. You can review log files on each node. An Impala administrator should review the log files and set log levels.

Impala uses the glog_v logging system to store information. Some messages refer to C++ file names. The GLOG_v environment variable specifies which types of messages Impala logs.

Reviewing Logs

You can locate log files in the Impala installation directory (/opt/mapr/impala/impala-<version>/logs) on each node with Impala. Impala creates a new set of log files on each statestored or impalad restart.

The following table provides a list of the important log files for the impalad and statestored processes with descriptions:

Log Type	Impalad Filename	Statestored Filename	File Content

INFO	impalad.INFO	statestored.INFO	Shows configuration settings for the processes.
WARNING	impalad.WARNING	statestored.WARNING	Shows problem information, including such things as suboptimal settings and also serious runtime errors.
ERROR	impalad.ERROR	statestored.ERROR	Shows the most serious errors, such as process crashes, failed queries. The .WARNING file also shows these messages.

Setting Log Levels

The GLOG system has three logging levels that you can adjust by exporting variable settings. The logging levels are cumulative. Increasing logging levels may decrease performance and increases log size. Change logging settings before you start impalad.

The following table provides the logging levels and their descriptions:

Level	Description
GLOG_v=1	Default. Logs information about each connection and query that is initiated to an impalad instance, including runtime profiles
GLOG_v=2	Logs everything from GLOG_v=1 plus information for each RPC initiated. This level also records query execution progress information, including details on each file read.
GLOG_v=3	Logs everything from GLOG_v=2 plus it logs row read. This level is only applicable for the most serious troubleshooting and tuning scenarios, because it can produce exceptionally large and detailed large log files.

Use the following command to change logging settings:

```
export GLOG_v=1
```

For more information on how to configure GLOG, including how to set variable logging levels for different system components, see Fixed in 5.2 r1.0.

Log Rotation

Log rotation is the automatic removal of unneeded or old log files. By default, Impala switches out old log files every 5 seconds, based on the default interval specified in the `logbufsecs` setting. The `-max_log_files` configuration option specifies how many log files to keep at each severity level (INFO, WARNING, ERROR, and FATAL). You can configure the `-max_log_files` option for each Impala daemon (impalad, statestored, and catalogd) in the `env.sh` configuration file. By default, Impala preserves the latest 10 log files for each severity level and removes old logs based on the `logbufsecs` setting. Setting `-max_log_files` to 0 preserves all of the log files. This setting requires manual log rotation. Setting `max_log_files` to 1 preserves only the latest log file.

Impala Ports

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Before you deploy Impala, verify that the TCP ports are open on each system.

Impala uses the following ports:

last-updated: Apr 11, 2022

Component	Service	Port	Type	Description
Impala Daemon	Impala Daemon Frontend Port	21000	External	Impala uses this port to communicate commands and receive results through the impala-shell and the ODBC driver.
Impala Daemon	Impala Daemon Frontend Port	21050	External	Impala uses this port to communicate commands and receive results through applications.
Impala Daemon	Impala Daemon Backend Port	22000	Internal	The Impala daemons use this port to communicate with each other.
Impala Daemon	StateStoreSubscriber Service Port	23000	Internal	The Impala daemons listen for updates from the statestore on this port.
Impala Daemon	Impala Daemon HTTP Server Port	25000	External	The Impala web interface that administrators use for troubleshooting.
Impala Statestore Daemon	StateStore Service Port	24000	Internal	The statestore listens on this port for registration requests.
Impala Statestore Daemon	StateStore HTTP Server Port	25010	External	The statestore web interface that administrators use for troubleshooting.
Impala Catalog Daemon	Catalog HTTP Server Port	25020	External	The catalog service web interface that administrators use for troubleshooting.
Impala Catalog Daemon	Catalog Service Port	26000	Internal	The catalog service uses this port to communicate with the Impala daemons.

Programming Interfaces

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You can connect to Impala daemons through the impala-shell or through a JDBC or ODBC client. You can use the impala-shell to query data from a Linux environment. In a non-Linux environment, you can query data using JDBC and ODBC applications.

last-updated: Apr 11, 2022

Impala Shell

The Impala-shell is a command line tool that you can use for setting up databases and tables, inserting data, and issuing queries using SQL statements and some shell commands. The impala-shell can reside on any client machine.

You can connect to the same node and issue all SQL statements through that particular node to avoid issuing the REFRESH statement frequently. For load balancing, you can connect to different nodes for each impala-shell session. When you connect to different Impala nodes, you may need to issue the REFRESH and INVALIDATE METADATA statements to update table data and metadata.

When you run commands from within the impala-shell, use a semi-colon to terminate commands. Each command can span multiple lines. Refer to Impala-Shell Commands and Command Line Options for a list of commands and options that you can issue from within the impala-shell.

Connecting to impalad from impala-shell

You can connect to any node running the Impala daemon process from the impala-shell using the CONNECT command. Once connected, you can send queries to the Impala node from the command line.

To connect to impalad, complete the following steps:

1. Issue the following command to start the impala-shell without a connection:

   ```
   $ impala-shell
   ```

2. Issue the CONNECT command to connect to an instance of Impala, replacing impalad-host with the hostname of the node that you want to connect to.

   Example:

   ```
   [Not connected] > connect impalad-host
   [impalad-host:21000] >
   ```

JDBC and ODBC Clients

You can use a JDBC or ODBC client to connect to Impala and run queries against data stored in MapR File System and MapR Database. The client or application issues a query to an Impala daemon through an ODBC or JDBC connection. The ODBC and JDBC driver submits the query to Impala in the form of a SQL statement.

Install a JDBC or ODBC driver on the client machine and then configure the driver to enable communication between the client and Impala. For driver download, installation, and connection information, refer to the following documentation:

JDBC/ODBC

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

JDBC

The MapR Impala JDBC Driver is available in a JAR file. You can download the driver to access Impala from a Java application or any tool that uses a JDBC connection. You can download the Impala JDBC driver from `https://package.mapr.hpe.com/tools/MapR-JDBC/MapR_Impala/`. The documentation for the Impala JDBC 1.0.46 driver is available at Impala JDBC Driver.

last-updated: Apr 11, 2022

ODBC

The MapR Impala ODBC Driver is available for Microsoft Windows, Linux, and Mac OS X. The driver complies with the ODBC 3.52 data standard and adds Unicode and 32- and 64-bit support on all platforms. The ODBC driver connects to an Impala server regardless of the server's host operating system. You can download the Impala ODBC driver from https://package.mapr.com/tools/MapR-ODBC/MapR_Impala/. The documentation for the Impala ODBC 1.2.16 driver is available at Impala ODBC Driver.

The following sections provide JDBC and ODBC connection information:

JDBC Connections

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You can connect to Impala through a JDBC client tool, such as the SQuirreL client. You can also use the driver in a Maven application.

Using the JDBC Driver with a Client Application

You can download the Impala JDBC driver from https://package.mapr.hpe.com/tools/MapR-JDBC/MapR_Impala/MapR_Impala_jdbc_v1.0.46.1057/. After downloading the driver, refer to the documentation at Impala JDBC Driver to install and configure the JDBC driver.

Before you connect to Impala from a JDBC client tool, complete the following tasks:

1. Configuring the JDBC Port. The default JDBC port that Impala accepts connections through is port 21050. Verify that this port can communicate with other hosts on your network. If your JDBC application connects through a different port, use the - - hs2_port option when you start `impalad` to specify the port number.

2. Enable the JDBC driver on client machines. The JDBC driver is packaged in JAR files. From a system that has Hive installed, download the following JAR files to each client machine:

   - The JDBC driver is in /opt/mapr/hive/hive-2.1/lib/hive-jdbc-2.1.1-mapr-1710.jar.
   - The hadoop-common-2.7.0 jar file for MRv2 (YARN) is in /opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/common.
   - The other required JAR files are in /opt/mapr/hive/hive-2.1/lib/.

   Download the following JARs:

   ```
   commons-logging-1.1.3.jar
   guava-14.0.1.jar
   hive-exec-2.1.1-mapr-1710.jar
   hive-jdbc-2.1.1-mapr-1710.jar
   hive-metastore-2.1.1-mapr-1710.jar
   hive-service-2.1.1-mapr-1710.jar
   hive-shims-2.1.1-mapr-1710.jar
   httpclient-4.4.jar
   httpcore-4.4.jar
   libfb303-0.9.2.jar
   libthrift-0.9.2.jar
   ```

   Note: You may have a different Hive 2.1 version installed. This list pertains to the 1710 version of Hive 2.1.You also need the `slf4j-api-1.7.5.jar` and `slf4j-log4j12-1.7.5.jar` files, which are available in /opt/mapr/lib.

last-updated: Apr 11, 2022

3. Update the CLASSPATH. The client application needs to locate the JAR files. Set the CLASSPATH for the client process to include the JARs. Consult the documentation for your JDBC client application for more details on how to install new JDBC drivers.

   If the existing CLASSPATH on your client machine refers to an older version of the Hive JARs, verify that the new JARs are listed first. You can put the new JAR files ahead of the older JAR files, or delete the other references to Hive JAR files.

4. Specify a connection string to establish a connection between Impala and JDBC. The JDBC driver classpath is `org.apache.hive.jdbc.HiveDriver`.Use the following connection string to establish the connection:

   ```
   jdbc:hive2://host:port/;auth=noSasl
   ```

   Example:

   `jdbc:hive2://myhost.example.com:21050/;auth=noSasl`

Using the JDBC Driver in a Maven Application

To use the JDBC driver in a Maven application, complete the following steps:

1. Configure a MapR Maven repository. Refer to Maven Artifacts for MapR to configure the Maven repository in your Java application.

2. Add the following dependency in the `pom.xml`:

   ```
   <dependency>
       <groupId>org.apache.hive</groupId>
       <artifactId>hive-jdbc</artifactId>
       <version>0.12-mapr[[mapr-version-number]]</version>
   </dependency>
   ```


last-updated: Apr 11, 2022

3. Specify the connection string in the Java application. The following code shows a working example:

```
import java.sql.*;
public class ImpalaJdbcExample
{
       private static String connectionString =
"jdbc:hive2://10.10.80.231:21050/;auth=noSasl";
       private static String driverName =
"org.apache.hive.jdbc.HiveDriver";
       private static String queryString = "select count(*) from
customers";
       private static Connection con;
       private static ResultSet resultSet;
       private static Statement sqlStatement;
       public static void main(String[] args)
       {
               System.out.println("Loaded the driver successfully.
Trying to establish connection");
               try
               {
                       Class.forName(driverName);
                       con =
DriverManager.getConnection(connectionString);
                       System.out.println("Created connection. Preparing
statement");
                       sqlStatement = con.createStatement();
                       System.out.println("Executing "+queryString);
                       resultSet =
sqlStatement.executeQuery(queryString);
                       while(resultSet.next())
                       {
                               System.out.println("Result set
"+resultSet.getString(1));
                       }
                       con.close();
               }
               catch(SQLException sqle)
               {
                       System.out.println("Got sql exception");
                       sqle.printStackTrace();
               }
               catch(Exception e)
               {
                       System.out.println("Got exception");
                       e.printStackTrace();
               }
       }
}
```

ODBC Client Connections on Mac OS X

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You can use an ODBC client tool to connect to Impala from Mac OS X. Install the driver and configure it to pass a SQL query to Impala. After you install the driver, see Configure the MapR Impala ODBC Driver for Linux and Mac OS X.

Before you install the driver, verify that your system meets the following prerequisites:

- Mac OS X version 10.8 or later

- 100 MB of available disk space
- iODBC 3.52.7 or above

To install and configure the MapR Impala ODBC Driver, complete the following steps:

1. Download the MapR Impala ODBC Driver.
2. Install the MapR Impala ODBC Driver on the machine from which you connect to the Impala service. The driver supports 32- and 64-bit applications. To install the driver:
   a. Double-click to mount the `MapRImpalaODBC.dmg` disk image.
   b. Double-click `MapRImpalaODBC.pkg` to run the Installer.
   c. Follow the instructions in the installer to complete the installation process.
   d. When the installation completes, click **Close.**

   MapR Impala ODBC Driver files install in the following locations:
   - /opt/mapr/impalaodbc/ErrorMessages – Error messages files directory
   - /opt/mapr/impalaodbc/Setup – Sample configuration files directory
   - /opt/mapr/impalaodbc/lib/universal – Binaries directory
3. Update the DYLD_LIBRARY_PATH environment variable. The DYLD_LIBRARY_PATH environment variable must include the paths to the:
   - Installed ODBC driver manager libraries
   - Installed MapR Impala ODBC Driver for Impala shared libraries

   For example, if the ODBC driver manager libraries are installed in /usr/local/lib, then set DYLD_LIBRARY_PATH to the following:

```
export DYLD_LIBRARY_PATH=/usr/local/lib:/opt/mapr/impalaodbc/lib/
universal
```

Next Step: See Configure the MapR Impala ODBC Driver for Linux and Mac OS X.

ODBC Client Connections on Linux

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You can use an ODBC client tool to connect to Impala from Linux. Install the driver and configure it to pass a SQL query to Impala. After you install the driver, Configure the MapR Impala ODBC Driver for Linux and Mac OS X.

Before you install the driver, verify that your system meets the following system prerequisites:

- RedHat 6.x or CentOS 6.x
- 50 MB of available disk space
- An installed ODBC driver manager, such as unixODBC 2.3.0/2.3.1 or iODBC 3.52.7

To install and configure the MapR Impala ODBC Driver, complete the following steps:

1. Download the MapR Impala ODBC Driver. You can install the 32- or 64-bit driver on Linux. Download and install the version of the driver that matches the architecture of the client application that you use to access Impala. The 64-bit editions of Linux support 32- and 64-bit applications.

   Click on the link to download the driver appropriate for your system:

Operating System	Version	Drivers
RedHat	6.x	MapRImpalaODBC-1.2.1.1001-1.el6.x86_64.rpm
		MapRImpalaODBC-32bit-1.2.1.1001-1.el6.i686.rpm

2. Install the MapR Impala ODBC Driver on the machine that you connect to the Impala service from.

   a. Login as **root**.

   b. Navigate to the folder that contains the driver RPM packages to install.

   c. Enter the following command where *RPMFileName* is the file name of the RPM package containing the version of the driver that you want to install:

   ```
   yum localinstall --nogpgcheck RPMFileName
   ```

   MapR Impala ODBC Driver files install in the following locations:

   - /opt/mapr/impalaodbc/ErrorMessages – Error messages files directory
   - /opt/mapr/impalaodbc/Setup – Sample configuration files directory
   - /opt/mapr/impalaodbc/lib/32 – 32-bit shared libraries directory
   - /opt/mapr/impalaodbc/lib/64 – 64-bit shared libraries directory

   The MapR Impala ODBC Driver depends on the following resources:

   - cyrus-sasl-2.1.22-7 or above
   - cyrus-sasl-gssapi-2.1.22-7 or above
   - cyrus-sasl-plain-2.1.22-7 or above

   If the package manager in your Linux or Mac OS X distribution cannot resolve the dependencies automatically when installing the driver, download and manually install the packages.

3. Set the shared LD_LIBRARY_PATH environment variable. Update the shared library environment variable to include the paths to the following directories:

   - Installed ODBC driver manager libraries
   - Installed MapR Impala ODBC Driver for Impala shared libraries

   Note: You can have both 32- and 64-bit versions of the driver installed at the same time on the same computer, but do not include the paths to both 32- and 64-bit shared libraries in LD_LIBRARY PATH at the same time. Only include the path to the shared libraries that correspond to the driver matching the architecture of the client application used. For example, if you are using a 64-bit client application and ODBC driver manager libraries are installed in /usr/local/lib, then set LD_LIBRARY_PATH as follows:

   ```
   export LD_LIBRARY_PATH=/usr/local/lib:/opt/mapr/impalaodbc/lib/64
   ```


last-updated: Apr 11, 2022

Next Step: Configure the MapR Impala ODBC Driver for Linux and Mac OS X.

ODBC Client Connections on Windows

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You can connect to Impala through an ODBC client tool, such as Tableau, from a Windows operating system. Use the MapR Impala ODBC Driver for SQL access to files and tables. Install the driver and configure it to pass a SQL query to Impala. You can run 32- and 64-bit applications on 64-bit Windows operating systems. You can install both versions of the driver on the same computer, however you should use the version of the driver that matches the architecture of the client application running the query.

Before you install the driver, verify that your system meets the following system prerequisites:

- Administrator privileges on the machine.
- One of the following operating systems (32- and 64-bit editions are supported):
  - Windows 7
  - Windows Server 2008 R2
  - Windows Server 2012
- 25 MB of available disk space

To install and the configure the MapR Impala ODBC Driver, complete the following steps:

1. Download the MapR Impala ODBC Driver:
   - MapRImpalaODBC32.msi (32-bit)
   - MapRImpalaODBC64.msi (64-bit)
2. Install the MapR Impala ODBC Driver on the machine that you connect to the Impala service from. When you install the driver, the driver program updates the system path to include the JVM directory. To install the driver:
   a. Double-click the file.
   b. Click **Next**.
   c. Accept the terms of the License Agreement, and click **Next**.
   d. Select an installation location and then click **OK**. To accept the installation location, click **Next**.
   e. Click **Install**.
   f. When the installation completes, click **Finish**.
3. Configure the MapR Impala ODBC Driver. To configure the driver:
   a. Click the Windows **Start** button.
   b. Click **All Programs**.
   c. Select the MapR Impala ODBC Driver. If you installed both versions of the driver, two options appear. Select the version that matches the architecture of your application. For example, a DSN that is defined for the 32-bit driver will only be accessible from 32-bit applications.
   d. Click 64-bit ODBC Administrator or 32-bit ODBC Administrator. The ODBC Data Source Administrator window opens.

last-updated: Apr 11, 2022

e. Click the **Drivers** tab, and verify that the MapR Impala ODBC Driver appears in the list of ODBC drivers that are installed on your system.

f. Click the **System DSN**tab to create a system DSN or click the User DSN tab to create a user DSN. All users that login to a workstation can see a system DSN. A user DSN is specific to a user on the workstation. Only the user who creates a user DSN can see it.

g. Click **Add**. The Create New Data Source window opens.

h. Select **MapR Impala ODBC Driver** and then click **Finish**. The MapR Impala ODBC Driver DSN Setup dialog opens.

i. Enter the DSN information. Optionally, you can select **Advanced Options**, and enter the following information in the Advanced Options window:

Option	Description
Use Native Query	Select this option to disable translating ODBC SQL to Impala SQL. By default, the driver applies transformations to the queries produced by an application to convert the queries to an equivalent form in Impala SQL. If the application produces Impala SQL, you can turn off the translation to avoid additional overhead of query transformation.
Rows Fetched Per Block	Enter the number of rows to fetch per block. You can enter any positive 32-bit integer as a value.
Socket Timeout	Enter the number of seconds Impala waits to close an idle connection with the client application. If you want to disable this option, set the value to 0.
Allow Common Name Hostname Mismatch	Select this option to allow the common name of a CA-issued SSL certificate to mismatch the hostname of the Impala server. **Note:** This setting only applies to the User Name and Password (SSL) and No Authentication (SSL) authentication mechanisms. Other authentication mechanisms ignore this setting.
Trusted Certificates field	Enter the path to the file that contains the trusted certificates in this field to configure the driver to load trusted certificates (such as the certificate from the Impala server) from a specific file when authenticating the Impala server using SSL. **Note:** This setting only apples to the User Name and Password (SSL) and No Authentication (SSL) authentication mechanisms. Other authentication mechanisms ignore this setting. SSL certificates in the trusted certificates file must be in PEM format. If this setting is not set, then the driver defaults to using the trusted CA certificates PEM file installed by the driver.

j. Click **OK**.

k. Optionally, if the operations against Impala are to be done on behalf of a user that is different than the authenticated user for the connection, enter the user name of the user to be delegated in the **Delegation UID** text box.

l. Click **Test** to test the connection.

m. Review the connection test results, and click **OK** in the Test Results dialog.

n. In the MapR Impala ODBC Driver DSN Setup dialog, click **OK**.

last-updated: Apr 11, 2022

4. Configure authentication. The Impala server supports multiple authentication mechanisms. You must determine the authentication type your server is using and configure your DSN accordingly. The following table provides the authentication methods available with configuration instructions:

Method	Configuration
Kerberos	Kerberos must be configured before you can use this authentication mechanism. For details, see Configuring Kerberos Authentication for Windows on page 3763. After Kerberos is installed and configured, configure your DSN to use Kerberos authentication. To configure your DSN to use Kerberos authentication, complete the following steps: **a.** In the MapR Impala ODBC Driver DSN Setup dialog, click the drop-down arrow next to the Mechanism field, and then select **Kerberos** **b.** If your Kerberos setup does not define a default realm or if the realm of your Impala server is not the default, then type the Kerberos realm of the Impala server host in the Realm field. Otherwise, leave the field blank. **c.** In the Host FQDN field, type the fully qualified domain name of the Impala host. **d.** In the Service Name field, type the service name of the Impala server. For example, if the principle for the Impala server is `impala/fully.qualified.domain.name@your-realm.com` , then the value in the service name field is impala. If you are unsure of the correct service name to use for your particular Hadoop deployment, contact your Hadoop administrator. **e.** In the Transport Buffer Size field, type the number of bytes to reserve in memory for buffering unencrypted data from the network. **Note:** In most circumstances, the default value of 1000 bytes is optimal.
No Authentication	For this authentication mechanism, you do not need to configure any additional settings. To configure your DSN for connections that do not require authentication, click the drop-down arrow next to the Mechanism field, and then select **No Authentication** in the driver DSN Setup dialog.

last-updated: Apr 11, 2022

Method	Configuration
No Authentication (SSL)	This authentication mechanism uses SSL but does not require a user name or a password. The driver accepts self-signed SSL certificates. To configure your DSN to use No Authentication (SSL) authentication, complete the following steps: **a.** In the MapR Impala ODBC Driver DSN Setup dialog, click the drop-down arrow next to the Mechanism field, and then select **No Authentication (SSL)** **b.** Optionally, configure the driver to allow a mismatch between the common name of a CA-issued certificate and the host name of the Impala server by clicking **Advanced Options** and selecting the Allow Common Name Host Name Mismatch check box. **Note:** For self-signed certificates, the driver always allows the common name of the certificate to not match the host name. Optionally, configure the driver to load SSL certificates from a specific file by clicking **Advanced Options** and typing the path to the file in the Trusted Certificates field. **Note:** By default, the driver uses the trusted CA certificates PEM file that is installed with the driver.
User Name	This authentication mechanism requires a user name but does not require a password. The user name labels the session, facilitating database tracking. To configure your DSN to use User Name authentication, complete the following steps: **a.** In the MapR Impala ODBC Driver DSN Setup dialog, click the drop-down arrow next to the Mechanism field, and then select **User Name** **b.** In the User Name field, type a user name that is recognized by the Impala server. **c.** In the Transport Buffer Size field, type the number of bytes to reserve in memory for buffering unencrypted data from the network. **Note:** In most circumstances, the default value of 1000 bytes is optimal.

Method	Configuration
User Name and Password	This authentication mechanism requires a user name and a password. **Note:** This authentication mechanism should not be used with an Impala configuration that does not have LDAP enabled. To configure your DSN for User Name and Password authentication, complete the following steps: **a.** In the MapR Impala ODBC Driver DSN Setup dialog, click the drop-down arrow next to the Mechanism field, and then select **User Name and Password** **b.** In the User Name field, type a user name that is recognized by the Impala server. **c.** In the Password field, type the password corresponding to the user name you typed in step 2.
User Name and Password (SSL)	This authentication mechanism uses SSL and requires a user name and a password. The driver accepts self-signed SSL certificates. **Note:** This authentication mechanism should not be used with an Impala configuration that does not have LDAP enabled. To configure your DSN to use User Name and Password (SSL) authentication, complete the following steps: **a.** In the MapR Impala ODBC Driver DSN Setup dialog, click the drop-down arrow next to the Mechanism field, and then select **User Name and Password (SSL)** **b.** In the User Name field, type a user name that is recognized by the Impala server. **c.** In the Password field, type the password corresponding to the user name you typed in step 2. **d.** Optionally, configure the driver to allow a mismatch between the common name of a CA-issued certificate and the host name of the Impala server by clicking **Advanced Options** and selecting the Allow Common Name Host Name Mismatch check box. **Note:** For self-signed certificates, the driver always allows the common name of the certificate to not match the host name Optionally, configure the driver to load SSL certificates from a specific file by clicking **Advanced Options** and typing the path to the file in the Trusted Certificates field. **Note:** By default, the driver uses the trusted CA certificates PEM file that is installed with the driver.

Configuring Kerberos Authentication for Windows

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

last-updated: Apr 11, 2022

You can configure Kerberos Authentication for Windows through Active Directory or MIT Kerberos.

Active Directory

The MapR ODBC Driver for Impala supports Active Directory Kerberos on Windows. Before you can use Active Directory Kerberos on Windows, the following prerequisites must be met:

- MIT Kerberos is not installed on the client Windows machine.
- The MIT Kerberos Hadoop realm has been configured to trust the Active Directory realm, so that users in the Active Directory realm can access services in the MIT Kerberos Hadoop realm.

MIT Kerberos

To use Kerberos, you must download and install MIT Kerberos for Windows 4.0.1.

Complete the following steps to use MIT Kerberos:

1. Download MIT Kerberos for Windows 4.0.1.
   - **Download for 64-bit Computers**

     To download the Kerberos installer for 64-bit computers, use the following download link from the MIT Kerberos website:

     http://web.mit.edu/kerberos/dist/kfw/4.0/kfw-4.0.1-amd64.msi

     This installer includes both 32-bit and 64-bit libraries.

   - **Download for 32-bit Computers**

     To download the Kerberos installer for 32-bit computers, use the following download link from the MIT Kerberos website:

     http://web.mit.edu/kerberos/dist/kfw/4.0/kfw-4.0.1-i386.msi

     This installer includes 32-bit libraries only.
2. Run the MIT Kerberos for Windows 4.0.1 installer.
   a. To run the installer, double-click the .msi file that you downloaded.
   b. Follow the instructions in the installer to complete the installation process.
   c. When the installation completes, click **Finish**.
3. Setup the Kerberos configuration file.

   Settings for Kerberos are specified through a configuration file. You can set up the configuration file as a .ini file in the default location (the C:\ProgramData\MIT\Kerberos5directory) or as a .conf file in a custom location.

   Normally, the C:\ProgramData\MIT\Kerberos5 directory is hidden. Consult your Windows documentation if you want to view and use this hidden directory.

   Setup in the Default Location

   To set up the Kerberos configuration file in the default location, obtain the krb5.conf configuration file from your Kerberos administrator. Alternatively, you can obtain the /etc/krb5.conf configuration file on the machine that is hosting the Impala server and then complete the following steps:

   a. Rename the configuration file from krb5.conf to krb5.ini.
   b. Copy the krb5.ini file to the C:\ProgramData\MIT\Kerberos5 directory, and overwrite the empty sample file.

last-updated: Apr 11, 2022

Note: For more information on configuring Kerberos, consult the MIT Kerberos documentation.

Setup in a Custom Location

To set up the Kerberos configuration file in a custom location, obtain the krb5.conf configuration file from your Kerberos administrator. Alternatively, you can obtain the /etc/krb5.conf configuration file on the machine that is hosting the Impala server and then complete the following steps:

a. Place the krb5.conf file in an accessible directory and make note of the full path name.

b. Click Start, then right-click Computer, and then click Properties.

c. Click Advanced system settings. In the System Properties dialog, click the Advanced tab, and then click Environment Variables.

d. In the Environment Variables dialog, under the System variables list, click New.

e. In the New System Variable dialog, in the Variable Name field, type KRB5_CONFIG.

f. In the Variable Value field, type the absolute path to the krb5.conf file from step 2.

g. Click OK to save the new variable.

h. Ensure the variable is listed in the System variables list.

i. Click OK to close the Environment Variables dialog, and then click OK to close the System Properties dialog.

4. Setup the Kerberos Credential Cache File. Kerberos uses a credential cache to store and manage credentials. To set up the Kerberos credential cache file, complete the following steps:

a. Create a directory where you want to save the Kerberos credential cache file. For example, create the following directory: **C:\temp**

b. Click **Start**, then right-click **Computer**,and then click **Properties**.

c. Click **Advanced system settings**.

d. In the System Properties dialog, click the **Advanced** tab, and then click **Environment Variables.**

e. In the Environment Variables dialog, under the **System variables** list, click **New**.

f. In the New System Variable dialog, in the Variable Name field, type **KRB5CCNAME**.

g. In the Variable Value field, type the path to the folder you created in step 0, and then append the file name krb5cache For example, if you created the folder C:\temp in step 0, then type C:\temp\krb5cache.

Note: krb5cache is a file (not a directory) that is managed by the Kerberos software, and it should not be created by the user. If you receive a permission error when you first use Kerberos, ensure that krb5cache does not already exist as a file or a directory.

a. Click **OK** to save the new variable.

b. Ensure the variable appears in the **System variables** list.

c. Click **OK** to close the Environment Variables dialog, and then click **OK** to close the System Properties dialog.

d. To ensure that Kerberos uses the new settings, restart your computer.

5. Obtain a Ticket for a Kerberos Principal. A principal is a user or service that can authenticate to Kerberos. To authenticate to Kerberos, a principal must obtain a ticket by using a password or a keytab file. You can specify a keytab file to use, or use the default keytab file of your Kerberos configuration.

   Obtain a Ticket Using a Password

   a. Click the **Start** button, then click **All Programs**, and then click the **Kerberos for Windows (64-bit)** or the **Kerberos for Windows (32-bit)** program group.

   b. Click **MIT Kerberos Ticket Manager.**

   c. In the MIT Kerberos Ticket Manager, click **Get Ticket.**

   d. In the Get Ticket dialog, type your principal name and password, and then click **OK.**

   If the authentication succeeds, then your ticket information appears in the MIT Kerberos Ticket Manager.

   Obtain a Ticket Using a keytab File

   a. Click the **Start** button **> All Programs > Accessories > Command Prompt.**

   b. In the Command Prompt, type a command using the following syntax:

   ```
   kinit -k -t keytab_file principal
   ```

   keytab_file is the full path to the keytab file.

   For example:

   ```
   C:\mykeytabs\impalaserver.keytab
   ```

   principal is the Kerberos principal to use for authentication.

   For example:

   ```
    impala/impalaserver.example.com@EXAMPLE.COM
   ```

   If the cache location KRB5CCNAME is not set or not used, then use the -c option of the kinit command to specify the credential cache. In the command, the -c argument must appear last. For example:

   ```
   kinit -k -t C:\mykeytabs\impalaserver.keytab
          impala/fully.qualified.domain.name@your-realm.com -c
        C:\ProgramData\MIT\krbcache
   ```

   Note: krbcache is the Kerberos cache file, not a directory.

   Obtain a Ticket Using the Default keytab File

   Note: For instructions on configuring a default keytab file for your Kerberos configuration, consult the MIT Kerberos documentation.

   a. Click the **Start** button **> All Programs > Accessories > Command Prompt**

   b. In the Command Prompt, type a command using the following syntax:

   ```
   kinit -k principal
   ```

   principal is the principal to use for authentication.

For example:

```
impala/impalaserver.example.com@EXAMPLE.COM
```

If the cache location KRB5CCNAME is not set or not used, then use the -c option of the kinit command to specify the credential cache. In the command, the

-c argument must appear last.

For example:

```
kinit -k impala/fully.qualified.domain.name@your-realm.com -c
C:\ProgramData\MIT\krbcache
```

Note: krbcache is the Kerberos cache file, not a directory.

Configure the MapR Impala ODBC Driver for Linux and Mac OS X

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The configuration files reside in the the home directory. You can use the two environment variables, ODBCINI and ODBCSYSINI, to specify different locations for the odbc.ini and odbcinst.ini configuration files. Set ODBCINI to point to your odbc.ini file. Set ODBCSYSINI to point to the directory containing the odbcinst.ini file. For example, if your odbc.ini file is located in /etc and your odbcinst.ini file is located in /usr/local/odbc, set the environment variables as follows:

- `export ODBCINI=/etc/odbc.ini`
- `export ODBCSYSINI=/usr/local/odbc`

Sample Files

The driver installation contains the following sample configuration files in the Setup directory:

- `odbc.ini`
- `odbcinst.ini`
- `mapr.impalaodbc.ini`

The sample files appear in the directory listings by default because the filenames do not begin with a (.) Filenames that begin with a (.) are hidden. If the default location is used for odbc.ini and odbcinst.ini, filenames must begin with a period (.). For mapr.impalaodbc.ini, the filename must begin with a (.) and must reside in the user's home directory.

If the configuration files do not already exist in the home directory, you can copy the sample configuration files to that directory and rename them. If the configuration files already exist in the home directory, use the sample configuration files as a guide for modifying the existing configuration files.

Step 1: Configure the Environment

By default, the configuration files reside in the user's home directory. However, the ODBCINI, ODBCSYSINI, and SIMBAINI environment variables can be used to specify different locations for the odbc.ini, odbcinst.ini, and mapr.impalaodbc.ini configuration files. Set ODBCINI to point to your odbc.ini file.

last-updated: Apr 11, 2022

Set ODBCSYSINI to point to the directory containing the odbcinst.ini file. Set SIMBAINI to point to your mapr.impalaodbc.ini file. For example, if your odbc.ini and mapr.impalaodbc.ini files are located in /etc and your odbcinst.ini file is located in /usr/local/odbc, then set the environment variables as follows:

- `export ODBCINI=/etc/odbc.ini`
- `export ODBCSYSINI=/usr/local/odbc`
- `export SIMBAINI=/etc/mapr.impalaodbc.ini`

The following search order is used to locate the mapr.impalaodbc.ini file:

1. If the SIMBAINI environment variable is defined, then the driver searches for the file specified by the environment variable.

   Important: SIMBAINI must contain the full path, including the filename.

2. The current working directory of the application is searched for a file named mapr.impalaodbc.ini **not** beginning with a period.
3. The directory ~/ (that is, $HOME) is searched for a hidden file named .mapr.impalaodbc.ini.
4. The directory /etc is searched for a file named mapr.impalaodbc.ini **not** beginning with a period.

Step 2: Configure the odbc.ini File

Define the ODBC data sources in the odbc.ini configuration file. This file is divided into the following optional and required sections:

- (Optional) [ODBC]. Controls global ODBC configuration.
- (Required) [ODBC Data Sources]. Lists DSNs and associates them with a driver.
- (Required) A section with the same name as the data source specified in the [ODBC Data Sources] section. This is required to configure the data source.

Example odbc.ini sample file for Linux

```
[ODBC Data Sources]
Sample MapR Impala DSN 32=MapR Impala ODBC Driver 32-bit

[Sample MapR Impala DSN 32]
Driver=/opt/mapr/impalaodbc/lib/32/libmaprimpalaodbc32.so
HOST=MyImpalaServer
PORT=21050
```

Example odbc.ini file for Mac OS X

```
[ODBC Data Sources]
Sample MapR Impala DSN=MapR Impala ODBC Driver

[Sample MapR Impala DSN]
Driver=/opt/mapr/impalaodbc/lib/universal/libmaprimpalaodbc.dylib
HOST=MyImpalaServer
PORT=21050
```

Create a Data Source

To create a data source, complete the following steps:

1. Open the .odbc.ini configuration file in a text editor.
2. Add a new entry to the [ODBC Data Sources] section. Type the data source name (DSN) and the driver name.
3. To set configuration options, add a new section having a name matching the data source name (DSN) you specified in step 2. Specify configuration options as key-value pairs.
4. Save the .odbc.ini configuration file.

Refer to Driver Configuration Options for Linux and Mac OS X for available configuration options that control DSN behavior.

Step 3: Configure the odbcinst.ini File

The odbcinst.ini is an optional configuration file that defines the ODBC Drivers. This configuration file is optional because you can specify drivers directly in the odbc.ini configuration file. The odbcinst.ini file is divided into the following sections:

- [ODBC Drivers] lists the names of all the installed ODBC drivers.
- A section having the same name as the driver name specified in the [ODBC Drivers] section lists driver attributes and values.

Example odbcinst.ini sample file for Linux

```
[ODBC Drivers]
MapR Impala ODBC Driver 32-bit=Installed
MapR Impala ODBC Driver 64-bit=Installed

[MapR Impala ODBC Driver 32-bit]
Description= MapR Impala ODBC Driver (32-bit)
Driver=/opt/mapr/impalaodbc/lib/32/libmaprimpalaodbc32.so

[MapR Impala ODBC Driver 64-bit]
Description=MapR Impala ODBC Driver (64-bit)
Driver=/opt/mapr/impalaodbc/lib/64/libmaprimpalaodbc64.so
```

Example odbcinst.ini file for Mac OS X

```
[ODBC Drivers]
MapR Impala ODBC Driver=Installed

[MapR Impala ODBC Driver]
Description=MapR Impala ODBC Driver
Driver=/opt/mapr/impalaodbc/lib/universal/libmaprimpalaodbc.dylib
```

Define a Driver

To define a driver, complete the following steps:

1. Open the .odbcinst.ini configuration file in a text editor.
2. Add a new entry to the [ODBC Drivers] section. Type the driver name, and then type `=Installed`. Assign the driver name as the value of the Driver attribute in the data source definition instead of the driver shared library name.
3. In .odbcinst.ini, add a new section having a name matching the driver name you typed in step 2, and add configuration options to the section based on the sample odbcinst.ini file provided with MapR Impala ODBC Driver for Impala in the Setup directory. Specify configuration options as key-value pairs.
4. Save the .odbcinst.ini configuration file.

last-updated: Apr 11, 2022

Step 4: Configure the mapr.impalaodbc.ini File

Configure the MapR Impala ODBC Driverto work with your ODBC driver manager.

To configure mapr.impalaodbc.ini, complete the following steps:

1. Open the .mapr.impalaodbc.ini configuration file in a text editor.
2. Edit the DriverManagerEncoding setting. The value is typically UTF-16 or UTF-32, but depends on the driver manger used. iODBC uses UTF-32 and unixODBC uses UTF-16. Review your ODBC Driver Manager documentation for the correct setting.
3. Edit the ODBCInstLib setting. The value is the name of the ODBCInst shared library for the ODBC driver manager you use. The configuration file defaults to the shared library for iODBC. In Linux, the shared library name for iODBC is libiodbcinst.so. In Mac OS X, the shared library name for iODBC is libiodbcinst.dylib.

   Note: Review your ODBC Driver Manager documentation for the correct setting. Specify an absolute or relative filename for the library. If you use the relative filename, include the path to the library in the library path environment variable. In Linux, the library path environment variable is LD_LIBRARY_PATH. In Mac OS X, the library path environment variable is DYLD_LIBRARY_PATH.
4. Save the .mapr.impalaodbc.ini configuration file.

Step 5: Configure Authentication

The Impala server supports multiple authentication mechanisms. You must determine the authentication type your server is using and configure your DSN accordingly. The following table provides the authentication methods available with configuration instructions:

Note: For details on the keys involved in configuring authentication, see Driver Configuration Options for Linux and Mac OS X.

Method	Configuration
No Authentication	For this authentication mechanism, you do not need to configure any additional settings. To configure your DSN for connections that do not require authentication, set the AuthMech configuration key for the DSN to 0.

last-updated: Apr 11, 2022

Kerberos	For information on operating Kerberos, refer to the documentation for your operating system. To configure your DSN to use Kerberos authentication, complete the following steps: **1.** Set the AuthMech configuration key for the DSN to 1 **2.** If your Kerberos setup does not define a default realm or if the realm of your Impala server is not the default, then set the appropriate realm using the KrbRealm key. **3.** Set the KrbFQDN key to the fully qualified domain name of the Impala host. **4.** Set the KrbServiceName key to the service name of the Impala server. For example, if the principle for the Impala server is impala/fully.qualified.domain.name@your-realm.com, then the value in the service name field is impala. If you are unsure of the correct service name to use for your particular Hadoop deployment, contact your Hadoop administrator.
User Name	This authentication mechanism requires a user name but does not require a password. The user name labels the session, facilitating database tracking. To configure your DSN to use User Name authentication, complete the following steps: **1.** Set the AuthMech configuration key for the DSN to 2. **2.** Set the UID key to a user name that is recognized by the Impala server.
User Name and Password	This authentication mechanism requires a user name and a password. **Note:** This authentication mechanism should not be used with an Impala configuration that does not have LDAP enabled. To configure your DSN to use User Name and Password authentication, complete the following steps: **1.** Set the AuthMech configuration key for the DSN to 3. **2.** Set the UID key to a user name that is recognized by the Impala server. **3.** Set the PWD key to the password corresponding to the user name you provided in step 2.

last-updated: Apr 11, 2022

User Name and Password (SSL)	This authentication mechanism uses SSL and requires a user name and a password. The driver accepts self-signed SSL certificates. **Note:** This authentication mechanism should not be used with an Impala configuration that does not have LDAP enabled. To configure your DSN to use User Name and Password (SSL) authentication, complete the following steps: **1.** Set the AuthMech configuration key for the DSN to 4. **2.** Set the UID key to a user name that is recognized by the Impala server. **3.** Set the PWD key to the password corresponding to the user name you provided in step 2. **4.** Optionally, configure the driver to allow a mismatch between the common name of a CA-issued certificate and the host name of the Impala server by setting the CAIssuedCertNamesMismatch key to 1. **Note:** For self-signed certificates, the driver always allows the common name of the certificate to mismatch the host name. For more details, see Driver Configuration Options for Linux and Mac OS X. Optionally, configure the driver to load SSL certificates from a specific file by setting the TrustedCerts configuration key to the path of the file. **Note:** By default, the driver uses the trusted CA certificates PEM file that is installed with the driver. For more details, see Driver Configuration Options for Linux and Mac OS X.
No Authentication (SSL)	This authentication mechanism uses SSL but does not require a user name or a password. The driver accepts self-signed SSL certificates. **T**o configure your DSN to use No Authentication (SSL) authentication, complete the following steps: **1.** Set the AuthMech configuration key for the DSN to 5. **2.** Optionally, configure the driver to allow a mismatch between the common name of a CA-issued certificate and the host name of the Impala server by setting the CAIssuedCertNamesMismatch key to 1. **Note:** For self-signed certificates, the driver always allows the common name of the certificate to mismatch the host name. For more details, see Driver Configuration Options for Linux and Mac OS X. Optionally, configure the driver to load SSL certificates from a specific file by setting the TrustedCerts configuration key to the path of the file. **Note:** By default, the driver uses the trusted CA certificate PEM file that is installed with the driver. For more details, see Driver Configuration Options for Linux and Mac OS X.

Driver Configuration Options for Linux and Mac OS X

last-updated: Apr 11, 2022

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The configuration options that you can use to control the behavior of the MapR ODBC Driver for Impala are listed and described in the table below.

Note: You can set configuration options in your odbc.ini and mapr.impalaodbc.ini files. Configuration options set in the mapr.impalaodbc.ini file apply to all connections, whereas configuration options set in an odbc.ini file are specific to a connection. Configuration options set in odbc.ini take precedence over configuration options set in mapr.impalaodbc.ini.

Key	Default Value	Default
AuthMech	0	You can use the following authentication mechanism values: • **0** – No Authentication • **1** – Kerberos • **2** – User Name • **3** – User Name and Password • **4** – User Name and Password (SSL) • **5** – No Authentication (SSL)
CAIssuedCertNamesMismatch	0	Whether to allow the common name of a CA-issued SSL certificate to mismatch the host name of the Impala server. The following values are possible: • **0** – Do not allow the names to mismatch. • **1** – Allow the names to mismatch. **Note:** This setting is only applicable to the User Name and Password (SSL) and No Authentication (SSL) authentication mechanisms. It is ignored by other authentication mechanisms.
Driver		The location of the MapR ODBC Driver for Impala shared object file.
HOST		The IP address or hostname of the Impala server.
KrbFQDN		The fully qualified domain name of the Impala host used.
KrbRealm		If there is no default realm configured or if the realm of the Impala host is different from the default realm for your Kerberos setup, use this option to define the realm of the Impala host.
KrbServiceName		The Kerberos service principal name of the Impala server.
PORT	10000	The listening port for the service.

last-updated: Apr 11, 2022

PWD		The password of a user account on the host that is running Impala. PWD is required if AuthMech is set to User Name and Password or User Name and Password (SSL).
RowsFetchedPerBlock	10000	The maximum number of rows that a query returns at a time. Any positive 32-bit integer is a valid value, but testing has shown that performance gains are marginal beyond the default value of 10000 rows.
SocketTimeout	0	The number of seconds after which Impala closes the connection with the client application if the connection is idle.

TrustedCerts	For 32 bit driver:
	`/opt/mapr/impalaodbc/lib/32/cacerts.pem`
	For 64 bit driver:
	`/opt/mapr/impalaodbc/lib/64/cacerts.pem`
		Used to specify the location of the file containing trusted CA certificates for authenticating the Impala server when using SSL.
		If this setting is not set, then the driver defaults to using the trusted CA certificates file installed by the driver.
		Note: This setting is only applicable to User Name and Password (SSL) and No Authentication (SSL) authentication mechanisms, and it is ignored by other authentication mechanisms.

TSaslTransportBufSize	1000	The number of bytes to reserve in memory for buffering unencrypted data from the network. **Note:** In most circumstances, the default value of 1000 bytes is optimal.
UID	anonymous	The user name of an existing account on the host that is running Impala. UID is required if AuthMech is set to User Name and Password or User Name and Password (SSL). UID is optional if AuthMech is set to User Name.

UseNativeQuery	0	By default, the driver transforms the queries emitted by an application to convert the queries into an equivalent form in Impala SQL. Use this option to specify whether or not the driver transforms queries. The following values are possible: • **0** – Transform the queries into Impala SQL. • **1** – Do not transform the queries (use the native query instead). **Note:** If the application is Impala-aware and already emits Impala SQL, then set this option to 1 to avoid the extra overhead of query transformation.

Impala-Shell Commands

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You can specify `impala-shell` options from the command line or in the `$HOME/.impalarc` configuration file. You can define a set of default options for your `impala-shell` environment in the `$HOME/.impalarc` configuration file. For every command line option, there is an equivalent setting in the `$HOME/.impalarc` configuration file. Options specified from the `impala-shell` command line override corresponding options in the configuration file.

The configuration file must contain the header label `[impala]`, followed by the options specific to the `impala-shell`. This is a standard convention for configuration files that enables a single file to hold configuration options for multiple applications. To specify a different file name or path for the configuration file, specify the argument `--config_file=`*`path_to_config_file`* from the command line.

Impala-Shell Commands

After you connect to an instance of `impalad` from the `impala-shell`, you can issue certain commands from within the shell. Enter a command at the prompt or use the `-q` option to pass it as an argument. The `impala-shell` passes most of the commands to `impalad` as SQL statements.

Command	Description
alter	Changes table schema in an Impala table, or a table shared between Impala and Hive.
compute stats	Gathers critical, statistical information about each table when you enable join optimizations.
connect	Connects impala-shell to a specific instance of impalad. The default port is 21000 unless modified. You can connect to any host in your cluster running impalad. If you connect to an instance of impalad that was started with an alternate port specified by the --fe_port flag, provide the alternate port.
describe	Shows the columns, column data types, and column comments for the table specified. DESCRIBE FORMATTED shows additional information, such as the MapR filesystem data directory, partitions, and internal table properties. You can use desc as an abbreviated alternative for describe.

last-updated: Apr 11, 2022

drop	Removes a schema object and its associated data files.
explain	Provides the query execution plan and represents a query as a series of steps.
help	Provides a list of available commands and options.
history	Maintains an enumerated cross-session command history. The history is stored in ~/.impalahistory.
insert	Writes query results to the table specified. Overwrites table data or appends data to the existing table content.
invalidate metadata	Updates impalad metadata. Use this command after creating, dropping, or altering databases, tables, or partitions in Hive.
profile	Displays low-level information about the most recent query. Used for performance diagnosis and tuning.
quit	Exits the shell. Include the final semicolon to ensure that the shell recognizes the end of the command.
refresh	Refreshes impalad metadata. Issue the REFRESH command to refresh metadata after you make database changes, such as adding or removing a table.
select	Specifies the data set to complete an action upon. You can send the information returned from select to an output, such as the console or a file. You can also use it to complete another query element.
set	Manages query options for an impala-shell session. Issue query options to Impala using the SET command. Issue SET without arguments to see display a current list query options. Use these options for query tuning and troubleshooting. SET has been promoted to an SQL statement and can be used in client applications through the JDBC and ODBC APIs. To modify option values, issue commands with the syntax `SET option=value`. To restore an option to its default, use the UNSET command. Some options take Boolean values of true and false. Others take numeric arguments, or quoted string values.
shell	Executes commands specified in the operating system shell without exiting impala-shell. Use ! as an abbreviation for the shell command.
show	Displays metastore data for schema objects created and accessed through Impala and Hive. Use show to gather information about databases or tables.
summary	Summarizes the work performed in various stages of a query. It provides a high-level view of the information displayed by the `EXPLAIN` command.
unset	Removes user-specified values for a query option and returns the option to its default value.
use	Indicates the database against which to run subsequent commands. Avoid using fully qualified names when referring to tables in databases other than default. Do not use with the -q option.
version	Returns Impala version information.

last-updated: Apr 11, 2022

Impala-Shell Command Line Options

Specify the command-line options to change how shell commands run when you start the `impala-shell`. These options do not apply to `impalad` configuration.

Option	Configuration File Setting	Description
-B or --delimited	write_delimited=true	Prints query results in plain text format. Specify the delimiter character with the --output_delimiter option. Store query results in a file instead of printing to the screen with the -B option.
--print_header	print_header=true	Prints a header.
-o filename or --output_file filename	output_file=filename	Stores all query results in the file specified.
--output_delimiter=delimiter_character	output_delimiter=character	Defaults to tab ('\t'), and specifies the character to use as a delimiter between fields when query results are printed in plain text format by the -B option.
-p or --show_profiles	show_profiles=true	Displays the query execution plan. Provides the same output as the EXPLAIN statement with a detailed itemization of execution steps, for every query executed by the shell.
-h or --help	N/A	Displays help information.
-i hostname or --impalad=hostname	impalad=hostname[:portnum]	Connects to impalad on the host specified. The default port of 21000 is assumed unless you provide another value. If you connect to an instance of impalad that was started with an alternate port specified by the --fe_port flag, provide that port.
-q query or --query=query	query=query	Passes a query or shell command from the command line. The shell immediately exits after processing the statement. You can use the following statements with the command: • SELECT • CREATE TABLE • SHOW TABLES • Any other statement recognized in impala-shell Fully qualify the names for tables outside the default database. You can also issue the -f option to pass a file with a USE statement followed by other queries.
-f query_file or --query_file=query_file	query_file=path_to_query_file	Passes a SQL query from a file. The file must be semicolon (;) delimited.

last-updated: Apr 11, 2022

-k or --kerberos	use_kerberos=true	Uses Kerberos authentication when the shell connects to `impalad`. If Kerberos is not enabled on the `impalad` instance, an error displays.
-s kerberos_service_name or --kerberos_service_name=name	kerberos_service_name=name	Instructs the`impala-shell` to authenticate to a particular `impalad` service principal. If a kerberos_service_name is not specified, `impala` is used by default. Using this option in conjunction with a connection in which Kerberos is not supported returns erros.
-V or --verbose	verbose=true	Enables verbose output.
--quiet	verbose=false	Disables verbose output.
-v or --version	version=true	Displays version information.
-c	ignore_query_failure=true	Continues on query failure.
-r or --refresh_after_connect	refresh_after_connect=true	Refreshes Impala metadata upon connection. Equivalent to running the REFRESH statement after connecting.
-d default_db or --database=default_db	default_db=default_db	Specifies the database to use on startup. Equivalent to running the USE statement after connecting. If you do not specify a name, a database named default is used.
-ssl	ssl=true	Enables SSL for impala-shell.Default: ssl=true
--ca_cert=path_to_certificate	ca_cert=path_to_certificate	The local pathname that points to the third-party CA certificate or a copy of the server certificate for self-signed server certificates. If `--ca_cert` is not set, impala-shell enables SSL, but does not validate the server certificate. This is useful for connecting to a known-good Impala that is only running over SSL, when a copy of the certificate is not available (such as when debugging customer installations).
-l	use_ldap=true	Enables LDAP authentication.
-u	user=user_name	Sets the user. The impala-shell prompts you for the password. Per Active Directory, the user is the short username, not the full LDAP distinguished name.
--config_file=path_to_config_file	N/A	Specifies the path to the file that contains the impala-shell configuration settings. The default path is$HOME/.impalarc. You can only specify this setting on the command line.

last-updated: Apr 11, 2022

Note:

The `--strict_unicode` option no longer exists. To avoid problems with Unicode values in impala-shell, define the following setting before running impala-shell:

```
export LC_CTYPE=en_US.UTF-8
```

Impala Built-In Functions

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You can use built-in functions to transform data directly in SELECT statements to avoid post processing in another application. Built-in functions allow an SQL query to return result sets with formatting, calculating, and type conversions already applied.

Impala supports the following categories of functions:

- Aggregation Functions
- Type Conversion Functions
- Mathematical Functions
- Date and Time Functions
- String Functions
- Conditional Functions
- Bit Manipulation Functions
- Analytic Window Functions
- Miscellaneous Functions

The following table provides a few rules that apply to built-in functions:

Rule	Example

Call functions through the SELECT statement.
You can omit the FROM clause in most functions.
Supply literal values for any required arguments.
	`select abs(-1);`
	
	`select concat('The sly ', 'brown fox');`
	
	`select power(2,5);`

When you include a FROM clause and specify a column name as a function argument, the function is applied to each item in the result set.	`select concat('State = ',state_code) from all_states where population > 20000000;`
	
	`select round(price) as dollar_value from product_directory where price between 400.0 and 500.0;`

last-updated: Apr 11, 2022

If an argument to a built-in function is NULL, the result value NULL.	`select cos(null);`
	`select power(2,null);`
	`select concat('a',null,'b');`

Aggregate functions require a FROM clause in the query. They calculate a return value across all items in a result set. Aggregate functions ignore NULL values rather than returning a NULL result.	`select count(customer_id) from customer_directory;` `select max(weight), avg(weight) from census_data where age > 40;`

The following sections list the available functions:

Aggregation Functions

You can use the following aggregation functions:

- APPX_MEDIAN()
- AVG()
- COUNT()
- MAX()
- MIN()
- NDV()
- STDDEV()
- STDDEV_SAMP()
- STDDEV_POP()
- SUM()
- VARIANCE()
- VARIANCE_SAMP()
- VARIANCE_POP()

Type Conversion Functions

Use conversion functions in combination with other functions to explicitly pass the expected data types.

CAST	Use CAST when passing a column value or literal to a function that expects a parameter with a different type. Syntax `cast(expr as type)`

last-updated: Apr 11, 2022

Example

```
select concat('Here are the first
',10,' results.'); -- Fails
select concat('Here are the first
',cast(10 as string),' results.'); --
Succeeds
```

TYPEOF

Available as of Impala 2.5.0. Use `typeof` to reutrn the name of the data type corresponding to an expression. For types with extra attributes, such as length for CHAR and VARCHAR, or precision and scale for DECIMAL, includes the full specification of the type.

Syntax

```
typeof(type value)
```

Examples

```
select typeof(2);
select typeof('xyz');
select typeof(5.30001 / 2342.1);
```

The examples above return tinyint, string, and DECIMAL(13,11) respectively.

Mathematical Functions

The following functions were added in Impala 2.5.0:

- dceil(double a), dceil(decimal(p,s) a)
- dexp(double a)
- dfloor(double a), dfloor(decimal(p,s) a)
- dlog10(double a)
- dpow(double a, double p), fpow(double a, double p)
- dround(double a), dround(double a, int d), dround(decimal(p,s) a, int_type d)
- dsqrt(double a)
- dtrunc(double_or_decimal a[, digits_to_leave])
- factorial(integer_type a)
- radians(double a)
- random(), random(int seed)

The following table lists the mathematical functions with their descriptions and return types:

Function	Description	Return type
abs(double a)	Ensures all return values are positive. Returns the absolute value of the argument.	double

last-updated: Apr 11, 2022

acos(double a)	Returns the arccosine of the argument.	double
asin(double a)	Returns the arcsine of the argument.	double
bin(bigint a)	Returns the binary representation of an integer value.	string
ceil(double a), ceiling(double a),dceil(double a), dceil(decimal(p,s) a)	Returns the smallest integer > or = to the argument.	int or decimal(p,s) depending on the type of the input argument
conv(bigint num, int from_base, int to_base), conv(string num, int from_base, int to_base)	Returns a string representation of an integer value in a particular base. The input value can be a string. To use the return value as a number, use CAST() to convert to the appropriate type.	string
cos(double a)	Returns the cosine of the argument.	double
cot(double a)	Returns the cotangent of the argument.	double
degrees(double a)	Converts the argument value from radians to degrees.	double
e()	Returns the mathematical constant e.	double
exp(double a), dexp(double a)	Returns the mathematical constant e raised to the power of the argument.	double
factorial(integer_type a)	Computes the factorial of an integer value. It works with any integer type. You can use either the factorial() function or the ! operator. The factorial of 0 is 1. Likewise, the factorial() function returns 1 for any negative value. The maximum positive value for the input argument is 20; a value of 21 or greater overflows the range for a BIGINT and causes an error.	bigint
floor(double a), dfloor(double a), dfloor(decimal(p,s) a)	Returns the largest integer that is less than or equal to the argument.	int, bigint, or decimal(p,s) depending on the type of the input argument
fnv_hash(type v)	Returns a consistent 64-bit value, derived from the input argument. Use for implementing hashing logic in an application.	bigint
greatest()	Returns the largest value from a list of expressions.	The return type is the same as the initial argument value. Integer values are promoted to BIGINT. Floating-point values are promoted to DOUBLE. Use CAST() when inserting into a smaller numeric column.
hex(bigint a), hex(string a)	Returns the hexadecimal representation of an integer value, or of the characters in a string.	string
is_inf(double a)	Tests whether a value is equal to the special value "inf", signifying infinity.	boolean

last-updated: Apr 11, 2022

is_nan(double a)	Tests whether a value is equal to the special value "NaN", signifying "not a number".	boolean
least()	Returns the smallest value from a list of expressions.	The return type is the same as the initial argument value. Integer values are promoted to BIGINT. Floating-point values are promoted to DOUBLE. Use CAST() when inserting into a smaller numeric column.
ln(double a)	Returns the natural logarithm of the argument.	double
log(double base, double a)	Returns the logarithm of the second argument to the specified base.	double
log10(double a), dlog10(double a)	Returns the logarithm of the argument to the base 10.	double
log2(double a)	Returns the logarithm of the argument to the base 2.	double
mod()	This function returns the modulus of a number. MOD is equivalent to using the % arithmetic operator. It works with any size integer type, any size floating-point type, and DECIMAL with any precision and scale.	Same as the input value
negative(int a), negative(double a)	Returns the argument with the sign reversed; returns a positive value if the argument was already negative. If return values must be negative, use -abs(a) instead.	int or double
max_int(),min_smallint()	Checks whether data values are in an expected range. You might be able to switch a column to a smaller type to save memory during processing.	The same as the integral type being checked.
pi()	Returns the constant pi.	double
pmod(int a, int b), pmod(double a, double b)	Purpose: Returns the positive modulus of a number.	int or double
positive(int a), positive(double a)	Returns the original argument; applies to negative arguments also. If return values must be positive, use abs().	int or double
pow(double a, double p), power(double a, double p), dpow(double a, double p), fpow(double a, double p)	Returns the first argument raised to the power of the second argument.	double
quotient(int numerator, int denominator)	Returns the first argument divided by the second argument and discards any fractional part. Avoids promoting arguments to DOUBLE as happens with the / SQL operator.	int
radians(double a)	Converts argument value from degrees to radians.	double

last-updated: Apr 11, 2022

rand(), rand(int seed), random(), random(int seed)	Returns a random value between 0 and 1. After rand() is called with a seed argument, produces a consistent random sequence based on the seed value. Currently, the random sequence is reset after each query, and multiple calls to rand() within the same query return the same value each time. For different number sequences that are different for each query, pass a unique seed value to each call to rand(). For example, select rand(unix_timestamp()) from ...	double
round(double a), round(double a, int d), dround(double a), dround(double a, int d), dround(decimal(p,s) a, int_type d)	Rounds a floating-point value. By default (with a single argument), rounds to the nearest integer. Values ending in .5 are rounded up for positive numbers, down for negative numbers. The second argument is optional and specifies how many digits to leave after the decimal point. Values greater than zero produce a floating-point return value rounded to the requested number of digits to the right of the decimal point.	bigint for single and double argument; double for two-argument signature when second argument is greater than zero; for DECIMAL values, the smallest DECIMAL(p,s) type with appropriate precision and scale
sign(double a)	Returns -1, 0, or 1 to indicate the sign of the argument value.	int
sin(double a)	Returns the sine of the argument.	double
sqrt(double a), dsqrt(double a)	Returns the square root of the argument.	double
tan(double a)	Returns the tangent of the argument.	double
truncate(double_or_decimal a[, digits_to_leave]), dtrunc(double_or_decimal a[, digits_to_leave])	Removes some or all fractional digits from a numeric value. With no argument, removes all fractional digits, leaving an integer value. The optional argument specifies the number of fractional digits to include in the return value, and only applies with the argument type is DECIMAL. truncate() and dtrunc() are aliases for the same function.	decimal for DECIMAL arguments; bigint for DOUBLE arguments
unhex(string a)	Returns a string of characters with ASCII values corresponding to pairs of hexadecimal digits in the argument.	string

Date and Time Functions

TIMESTAMP is the underlying datatype for data and time data. Functions that extract a single field, such as hour() or minute(), typically return an integer value. Functions that format the date portion, such as date_add() or to_date(), typically return a string value.

The following table lists the date and time functions with their descriptions and return types:

Function	Description	Return Type
add_months()	Alias for the existing MONTHS_ADD() function.	timestamp

last-updated: Apr 11, 2022

date_add(string startdate, int days)	Adds a specified number of days to a date represented as a string.	string
date_part()	A new date and time function, similar to EXTRACT(), but with the order of the arguments reversed. You can also call the EXTRACT() function using the SQL-99 syntax, EXTRACT(*unit* FROM *timestamp*). These enhancements simplify the porting process for date-related code from other systems.	int
date_sub(string startdate, int days)	Subtracts a specified number of days from a date represented as a string.	string
datediff(string enddate, string startdate)	Returns the number of days between two dates represented as strings.	int
day(string date), dayofmonth(string date)	Returns the day field from a date represented as a string.	int
dayname(string date)	Returns the day field from a date represented as a string, converted to the string corresponding to that day name. The range of return values is 'Sunday' to 'Saturday'. Use in report-generating queries, instead of calling dayofweek() and turning that numeric return value into a string using a CASE expression.	string
dayofweek(string date)	Returns the day field from a date represented as a string, corresponding to the day of the week. The range of return values is 1 (Sunday) to 7 (Saturday).	int
extract()	Returns one date or time field from a TIMESTAMP value.	timestamp
from_unixtime(bigint unixtime[, string format])	Converts the number of seconds from the Unix epoch to the specified time into a string.	string
from_utc_timestamp(timestamp, string timezone)	Converts a specified UTC timestamp value into the appropriate value for a specified time zone.	timestamp
hour(string date)	Returns the hour field from a date represented as a string.	int
int_months_between(timestamp newer, timestamp older)	Available as of Impala 2.5.0. Returns the number of months between the date portions of two TIMESTAMP values, as an INT representing only the full months that passed.	int
minute(string date)	Returns the minute field from a date represented as a string.	int
month(string date)	Returns the month field from a date represented as a string.	int

last-updated: Apr 11, 2022

months_between(timestamp newer, timestamp older)	Available as of Impala 2.5.0. Returns the number of months between the date portions of two TIMESTAMP values. Can include a fractional part representing extra days in addition to the full months between the dates. The fractional component is computed by dividing the difference in days by 31 (regardless of the month).	double
now()	Returns the current date and time (in the UTC time zone) as a timestamp value.	timestamp
second(string date)	Returns the second field from a date represented as a string.	int
timeofday()	Available as of Impala 2.5.0. Returns a string representation of the current date and time, according to the time of the local system, including any time zone designation.	string
timestamp_cmp(timestamp t1, timestamp t2)	Available as of Impala 2.5.0. Tests if one TIMESTAMP value is newer than, older than, or identical to another TIMESTAMP	int (either -1, 0, 1, or NULL)
to_date(string timestamp)	Returns the date field from a timestamp represented as a string.	string
to_utc_timestamp(timestamp, string timezone)	Converts a specified timestamp value in a specified time zone into the corresponding value for the UTC time zone.	timestamp
trunc()	Truncates date/time values to a particular granularity, such as year, month, day, hour, and so on.	
unix_timestamp(), unix_timestamp(string date), unix_timestamp(string date, string pattern)	Returns a timestamp representing the current date and time, or converts from a specified date and time value represented as a string.	bigint
weekofyear(string date)	Returns the corresponding week (1-53) from a date represented as a string.	int
year(string date)	Returns the year field from a date represented as a string.	int

String Functions

The following table lists the string functions with their descriptions and return types:

Function	**Description**	**Return Type**
ascii(string str)	Returns the numeric ASCII code of the first character of the argument.	int

last-updated: Apr 11, 2022

btrim(string a), btrim(string a, string chars_to_trim)	Available as of Impala 2.5.0. Removes all instances of one or more characters from the start and end of a STRING value. By default, removes only spaces. If a non-NULL optional second argument is specified, the function removes all occurrences of characters in that second argument from the beginning and end of the string.	string
chr(int character_code)	Available as of Impala 2.5.0. Returns a character specified by a decimal code point value. The interpretation and display of the resulting character depends on your system locale. Because consistent processing of Impala string values is only guaranteed for values within the ASCII range, only use this function for values corresponding to ASCII characters. In particular, parameter values greater than 255 return an empty string.	string
concat(string a, string b...)	Returns a single string representing all the argument values joined together.	string
concat_ws(string sep, string a, string b...)	Returns a single string representing the second and following argument values joined together, delimited by a specified separator.	string
find_in_set(string str, string strList)	Returns the position (starting from 1) of the first occurrence of a specified string within a comma-separated string. Returns NULL if either argument is NULL, 0 if the search string is not found, or 0 if the search string contains a comma.	int
initcap(string str)	Returns the input string with the first letter capitalized.	string
instr(string str, string substr)	Returns the position (starting from 1) of the first occurrence of a substring within a longer string.	int
length(string a)	Returns the length in characters of the argument string.	int
locate(string substr, string str[, int pos])	Returns the position (starting from 1) of the first occurrence of a substring within a longer string, optionally after a particular position.	int
lower(string a), lcase(string a)	Returns the argument string converted to all-lowercase.	string
lpad(string str, int len, string pad)	Returns a string of a specified length, based on the first argument string. If the specified string is too short, it is padded on the left with a repeating sequence of the characters from the pad string. If the specified string is too long, it is truncated on the right.	string

last-updated: Apr 11, 2022

ltrim(string a)	Returns the argument string with any leading spaces removed from the left side.	string
parse_url(string urlString, string partToExtract [, string keyToExtract])	Returns the portion of a URL corresponding to a specified part. The part argument can be 'PROTOCOL', 'HOST', 'PATH', 'REF', 'AUTHORITY', 'FILE', 'USERINFO', or 'QUERY'. Literal values must be uppercase. You can specify a key to retrieve only the associated value from the key-value pairs in the query string when you request the query portion of the URL. Useful for importing web logs.	string
regexp_extract(string subject, string pattern, int index)	Returns the specified () group from a string based on a regular expression pattern.	string
regexp_like(string source, string pattern[, string options])	Available as of Impala 2.5.0. Returns true or false to indicate whether the source string contains anywhere inside it the regular expression given by the pattern. The optional third argument consists of letter flags that change how the match is performed, such as i for case-insensitive matching.	boolean
regexp_replace(string initial, string pattern, string replacement)	Returns the initial argument with the regular expression pattern replaced by the final argument string.	string
repeat(string str, int n)	Returns the argument string repeated a specified number of times.	string
reverse(string a)	Purpose: Returns the argument string with characters in reversed order.	string
rpad(string str, int len, string pad)	Returns a string of a specified length, based on the first argument string. If the specified string is too short, it is padded on the right with a repeating sequence of the characters from the pad string. If the specified string is too long, it is truncated on the right.	string
rtrim(string a)	Returns the argument string with any trailing spaces removed from the right side.	string
space(int n)	Returns a concatenated string of the specified number of spaces. Shorthand for repeat(' ',n).	string
split_part(string source, string delimiter, bigint n)	Available as of Impala 2.5.0. Returns the nth field within a delimited string. The fields are numbered starting from 1. The delimiter can consist of multiple characters, not just a single character. All matching of the delimiter is done exactly, not using any regular expression patterns.	string

last-updated: Apr 11, 2022

substr(string a, int start [, int len]), substring(string a, int start [, int len])	Returns the portion of the string starting at a specified point, optionally with a specified maximum length. The characters in the string are indexed starting at 1.	string
translate(string input, string from, string to)	Returns the input string with a set of characters replaced by another set of characters.	string
trim(string a)	Returns the input string with leading and trailing spaces removed. The same as passing the string through both ltrim() and rtrim().	string
upper(string a), ucase(string a)	Returns the argument string converted to all-uppercase.	string

Conditional Functions

Use the conditional functions to test equality, comparison operators, and nullity.

The following table lists the conditional functions with their descriptions and return types:

Function	Description	Return Type
CASE a WHEN b THEN c [WHEN d THEN e]... [ELSE f] END	Compares an expression to one or more possible values, and returns a corresponding result when a match is found.	Same as the initial argument value
CASE WHEN a THEN b [WHEN c THEN d]... [ELSE e] END	Tests whether any of a sequence of expressions is true, and returns a corresponding result for the first true expression.	Same as the initial argument value
coalesce(type v1, type v2, ...)	Returns the first specified argument that is not NULL, or NULL if all arguments are NULL.	Same as the initial argument value
decode()	Compares an expression to one or more possible values and returns a corresponding result when a match is found. This function works as a shorthand for a CASE() expression and improves compatibility with SQL code containing vendor extensions.	Same as the initial argument value, except that integer values are promoted to `BIGINT` and floating-point values are promoted to `DOUBLE`; use `CAST( )` when inserting into a smaller numeric column
if(boolean condition, type ifTrue, type ifFalseOrNull)	Tests an expression and returns a corresponding result depending on whether the result is true, false, or NULL.	Same as ifTrue argument value
isfalse(), isnotfalse(), isnottrue(), istrue(), notnullvalue(), nullvalue()	These conditional functions provide enhanced compatibility when porting code that uses industry extensions.	
isnull(type a, type ifNotNull)	Tests if an expression is NULL, and returns the expression result value if not. If the first argument is NULL, returns the second argument. Equivalent to the nvl() function from Oracle Database orifnull() from MySQL.	Same as the first argument value

last-updated: Apr 11, 2022

nullif(expr1,expr2)	Returns `NULL` if the two specified arguments are equal. If the specified arguments are not equal, returns the value of expr1. The data types of the expressions must be compatible. You cannot use an expression that evaluates to `NULL` for expr1, so you can distinguish a return value of `NULL` from an argument value of `NULL`, which would never match expr2.	Same as the initial argument value, except that integer values are promoted to `BIGINT` and floating-point values are promoted to `DOUBLE`; use `CAST( )` when inserting into a smaller numeric column
nullifzero(numeric_expr)	Returns `NULL` if the numeric expression evaluates to 0, otherwise returns the result of the expression.	Same as the initial argument value, except that integer values are promoted to `BIGINT` and floating-point values are promoted to `DOUBLE`; use `CAST( )` when inserting into a smaller numeric column
nvl(type a, type ifNotNull)	Alias for the isnull() function; added in Impala 1.1. Tests if an expression is NULL, and returns the expression result value if not. If the first argument is NULL, returns the second argument. Equivalent to the nvl() function from Oracle Database or ifnull() from MySQL.	Same as the first argument value
zeroifnull(numeric_expr)	Returns 0 if the numeric expression evaluates to `NULL`, otherwise returns the result of the expression.	Same as the initial argument value, except that integer values are promoted to `BIGINT` and floating-point values are promoted to `DOUBLE`; use `CAST( )` when inserting into a smaller numeric column

Bit Manipulation Functions

Impala 2.5.0 introduces bit manipulation functions.

The following table lists the bit manipulation functions with their descriptions and return types:

Function	Description	Return Type
bitand(integer_type a, same_type b)	Returns an integer value representing the bits that are set to 1 in both of the arguments. If the arguments are of different sizes, the smaller is promoted to the type of the larger. The bitand() function is equivalent to the & binary operator.	Same as the input value
bitnot(integer_type a)	Inverts all the bits of the input argument. The bitnot() function is equivalent to the ~ unary operator.	Same as the input value
bitor(integer_type a, same_type b)	Returns an integer value representing the bits that are set to 1 in either of the arguments. If the arguments are of different sizes, the smaller is promoted to the type of the larger. The bitor() function is equivalent to the | binary operator.	Same as the input value

bitxor(integer_type a, same_type b)	Returns an integer value representing the bits that are set to 1 in one but not both of the arguments. If the arguments are of different sizes, the smaller is promoted to the type of the larger. The bitxor() function is equivalent to the ^ binary operator.	Same as the input value
countset(integer_type a [, int zero_or_one])	By default, returns the number of 1 bits in the specified integer value. If the optional second argument is set to zero, it returns the number of 0 bits instead.	Same as the input value
getbit(integer_type a, int position)	Returns a 0 or 1 representing the bit at a specified position. The positions are numbered right to left, starting at zero. The position argument cannot be negative. When you use a literal input value, it is treated as an 8-bit, 16-bit, and so on value, the smallest type that is appropriate. The type of the input value limits the range of the positions. Cast the input value to the appropriate type if you need to ensure it is treated as a 64-bit, 32-bit, and so on value.	Same as the input value
rotateleft(integer_type a, int positions)	Rotates an integer value left by a specified number of bits. As the most significant bit is taken out of the original value, if it is a 1 bit, it is "rotated" back to the least significant bit. Therefore, the final value has the same number of 1 bits as the original value, just in different positions. Specifying a second argument of zero leaves the original value unchanged. Rotating a -1 value by any number of positions still returns -1, because the original value has all 1 bits and all the 1 bits are preserved during rotation. Similarly, rotating a 0 value by any number of positions still returns 0. Rotating a value by the same number of bits as in the value returns the same value. Because this is a circular operation, the number of positions is not limited to the number of bits in the input value. For example, rotating an 8-bit value by 1, 9, 17, and so on positions returns an identical result in each case.	Same as the input value

last-updated: Apr 11, 2022

rotateright(integer_type a, int positions)	Rotates an integer value right by a specified number of bits. As the least significant bit is taken out of the original value, if it is a 1 bit, it is "rotated" back to the most significant bit. Therefore, the final value has the same number of 1 bits as the original value, just in different positions. Specifying a second argument of zero leaves the original value unchanged. Rotating a -1 value by any number of positions still returns -1, because the original value has all 1 bits and all the 1 bits are preserved during rotation. Similarly, rotating a 0 value by any number of positions still returns 0. Rotating a value by the same number of bits as in the value returns the same value. Because this is a circular operation, the number of positions is not limited to the number of bits in the input value. For example, rotating an 8-bit value by 1, 9, 17, and so on positions returns an identical result in each case.	Same as the input value
setbit(integer_type a, int position [, int zero_or_one])	By default, changes a bit at a specified position to a 1, if it is not already. If the optional third argument is set to zero, the specified bit is set to 0 instead. If the bit at the specified position was already 1 (by default) or 0 (with a third argument of zero), the return value is the same as the first argument. The positions are numbered right to left, starting at zero. (Therefore, the return value could be different from the first argument even if the position argument is zero.) The position argument cannot be negative. When you use a literal input value, it is treated as an 8-bit, 16-bit, and so on value, the smallest type that is appropriate. The type of the input value limits the range of the positions. Cast the input value to the appropriate type if you need to ensure it is treated as a 64-bit, 32-bit, and so on value.	Same as the input value

shiftleft(integer_type a, int positions)	Shifts an integer value left by a specified number of bits. As the most significant bit is taken out of the original value, it is discarded and the least significant bit becomes 0. The final value has either the same number of 1 bits as the original value, or fewer. Shifting an 8-bit value by 8 positions, a 16-bit value by 16 positions, and so on produces a result of zero. Specifying a second argument of zero leaves the original value unchanged. Shifting any value by 0 returns the original value. Shifting any value by 1 is the same as multiplying it by 2, as long as the value is small enough; larger values eventually become negative when shifted, as the sign bit is set. Starting with the value 1 and shifting it left by N positions gives the same result as 2 to the Nth power, or pow(2,N).	Same as the input value
shiftright(integer_type a, int positions)	Shifts an integer value right by a specified number of bits. As the least significant bit is taken out of the original value, it is discarded and the most significant bit becomes 0. In computer science terms, this operation is a "logical shift". Usage notes: Therefore, the final value has either the same number of 1 bits as the original value, or fewer. Shifting an 8-bit value by 8 positions, a 16-bit value by 16 positions, and so on produces a result of zero. Specifying a second argument of zero leaves the original value unchanged. Shifting any value by 0 returns the original value. Shifting any positive value right by 1 is the same as dividing it by 2. Negative values become positive when shifted right.	Same as the input value

Analytic Functions

Analytic (window) functions operate on a set of rows and return a single value for each row from the underlying query. The term "window" describes the set of rows on which the function operates. A window function uses values from the rows in a window to calculate the returned values. When you use a window function in a query, you define the window using the OVER() clause. The OVER() clause (window clause) differentiates window functions from other analytical and reporting functions.

As of Impala 2.2.0, you can use the following analytic functions in queries:

- MAX()
- MIN()
- SUM()
- COUNT()

last-updated: Apr 11, 2022

- AVG()
- RANK()
- LAG()
- LEAD()
- FIRST_VALUE()

As of Impala 2.5.0, you can use the following analytic functions in queries:

- PERCENT_RANK
- NTILE
- CUME_DIST

The analytic functions support the following syntax:

```
function(args) OVER([partition_by_clause] [order_by_clause [window_clause]])
partition_by_clause ::= PARTITION BY expr [, expr ...]order_by_clause ::=
ORDER BY expr  [ASC | DESC] [NULLS FIRST | NULLS LAST] [, expr [ASC |DESC]
[NULLS FIRST | NULLS LAST] ...]
```

The window clause supports the following syntax:

```
ROWS BETWEEN [ { m | UNBOUNDED } PRECEDING | CURRENT ROW] [ AND [CURRENT
ROW | { UNBOUNDED | n } FOLLOWING] ]
```

```
RANGE BETWEEN [ {m | UNBOUNDED } PRECEDING | CURRENT ROW] [ AND [CURRENT
ROW | { UNBOUNDED | n } FOLLOWING] ]
```

Miscellaneous Functions

The following table lists miscellaneous functions with their descriptions and syntax:

Function	Description	Syntax
uuid()	The uuid() function generates an alphanumeric value that you can use as a guaranteed unique identifier. The uniqueness applies across tables in cases where an ascending numeric sequence is not suitable.	select uuid();

Impala User-Defined Functions

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

A user-defined function (UDF) is a SQL function that you create to encapsulate code that processes column values during an Impala query. UDFs are called from within a SQL statement, like a regular function, and return a single value. Starting in Impala 1.2.3 for MapR, you can create UDFs to perform custom calculations and transformations that built-in SQL operators and functions do not provide.

After you download and install the UDF development package, you can write your own UDFs and invoke them in the queries that you run on Impala. Impala can run scalar UDFs and UDAFs (user-defined aggregate functions).

last-updated: Apr 11, 2022

Note: User-defined functions written in C++ persist when the catalog service is restarted. You no longer have to run the CREATE FUNCTION statements again after a restart. You must still reissue the CREATE FUNCTION statement for any Java-based user-defined functions.

UDFs and UDAFs

UDFs and UDAFs differ in the number of rows that they accept as input. A UDF operates on a single row and produces a single row as the output. When you use a UDF in a query, the UDF is called once for each row in the result set. Mathematical and string functions are examples of UDFs. When you create a UDF, you issue the CREATE FUNCTION statement. After you create a UDF, you can use it in the expression of a SQL statement.

A UDAF operates on multiple input rows and produces a single row as output. The COUNT(), MAX(), SUM(), and AVG() aggregate functions are examples of UDAFs. You might use a UDAF in a query with a GROUP BY clause to produce a result set with a separate aggregate value for each combination of values from the GROUP BY clause. When you create UDAFs, use the CREATE AGGREGATE FUNCTION statement. After you create a UDAF, you can use it the expression of a SQL statement.

When you create UDFs and UDAFs, they cannot have the same name as any of the built-in functions.

UDFs in C++ and Java

Impala accepts UDFs and UDAFs written in C++, as well as Hive UDFs written in Java. A Java UDF may cause a query in Impala to run much slower than the equivalent native UDF written in C++. If you use Hive UDFs when you query Impala, the Hive UDFs must meet the following conditions:

- Parameters and return values must all use data types that Impala accepts. Impala does not accept nested and composite types.
- UDFs cannot accept or return the TIMESTAMP data type.
- The return type must be a writable type, like Text or IntWritable. UDFs return a NULL value for Java primitive types, like string or int.
- Impala does not accept Hive UDAFs or UDTFs.

For more information about Hive UDFs, refer to the Hive Language Manual UDF.

Get Started with UDFs

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Before you can write UDFs for your Impala queries, you must install the Impala development package that contains header and build configuration files and download the sample source code.

The development package provides a header file that is required to develop UDFs and UDAFs. The header contains the types that must be used and the FunctionContext object. This object is the interface object between the UDF or UDAF and the Impala process. You can access the header in /usr/include/impala_udf/udf.h

The sample source code provides sample files that you can view to see the declarations required to write UDFs and UDAFs, and sample source code used to create a simple UDF and UDAF example.

After you have installed the development package and sample source code, you can build the samples to create an environment that you can use to write your UDFs and UDAFs. Once you have completed your UDF or UDAF, you can deploy it.

Complete the following steps to create UDFs and UDAFs for use with Impala:

Install the Development Package

Use the appropriate commands for your operating system to install the development package.

last-updated: Apr 11, 2022

To install the development package, complete the following steps:

1. Install the gcc-c++ compiler and boost-devel using the command appropriate for your environment:

```
$ sudo yum install gcc-c++ cmake boost-devel
```

```
$ sudo apt-get install g++ libboost-devel cmake
```

2. Install the development package, using the command appropriate for your environment:

```
$ sudo yum install mapr-impala-udf
```

```
$ sudo apt-get install mapr-impala-udf
```

Warning: Impala is not required for UDF development. You can create UDFs on a minimal development system and then deploy the UDFs to a machine with Impala.

Download Sample Source Code

- To download the sample source code and build scripts, issue the following command:

```
git clone https://github.com/mapr/impala-udf-samples
```

The following table provides the source code file names and their descriptions:

File Name	Description
udf-sample.h	Header file that declares the signature for a scalar UDF.
udf-sample.cc	Sample source for a simple UDF that adds two integers.
udf-sample-test.cc	Basic unit tests for the sample UDF.
uda-sample.h	Header file that declares signature for sample aggregate functions.
uda-sample.cc	Sample source for simple UDAFs that show how to manage state transitions as underlying are called during various phases of query processing.
uda-sample-test.cc	Basic unit tests for the sample UDAFs.

Build the Samples

To build the samples, complete the following steps:

1. Issue the `cmake` configuration command. This command reads the file `CMakeLists.txt`, and generates a `Makefile` customized for your directory paths.

2. Issue the `make` command. The `make` command runs the build steps based on rules in the `Makefile`. Impala loads the shared library from a MapR filesystem location.

The samples get built to `build/`. This contains test executables that you can run locally, without the Impala service installed. It also contains shared object artifacts that you can run on Impala.

At this point, you can write your UDFs or UDAFs. When you write your functions, build a shared library to contain them. Use the `mapr` or `hadoop` commands to copy the binary files to a MapR filesystem location that Impala can read from.

Writing UDFs

When you write UDFs, use function-oriented programming best practices, and note data type differences when transferring values from high-level SQL to low-level UDF code.

The UDF code samples that you downloaded include the udf-sample.h and udf-sample.cc files that you can reference when you write your UDFs. The udf-sample.h file provides the basic declarations required to write a scalar UDF. This file defines a simple function named AddUdf() that you can reference. The udf-sample.cc file provides sample C++ code for a simple function named AddUdf().

Function Argument/Return Value Data Types

Every value that a UDF accepts as an argument or returns as a result, must map to a SQL data type that you can specify for a table column.

Every data type has a corresponding structure defined in the C++ and Java header files with two member fields and some predefined comparison operators and constructors:

- is_null indicates if a value is/is not NULL. When non-NULL, val holds the argument or return value.
- null () is a member function that constructs an instance of the struct with the is_null flag set.
- <,>=,BETWEEN,ORDER BY are built-in SQL comparison operators and clauses that work automatically based on the SQL return type of each UDF.
- Every struct within your UDF code defines == and != operators for comparisons with structs of the same type for typical C++ comparisons within your own code. Each kind of struct one or more constructors that define a filled-in instance of the struct.
- Every type of struct has a null() member function that returns an instance of the struct with is_null flag set.
- Impala cannot process UDFS that accept or return composite or nested types. This applies to UDFs written in C++ and Jave-based Hive UDFs.
- You can create multiple functions with the same SQL name and different argument types to overload functions, however you must use different C++ or Java entry point names in the underlying functions.

The following table lists the data types defined for C++ in `/usr/include/impala_udf/udf.h`:

Data Type	**Description**
IntVal	Represents an INT column.
BigIntVal	Represents a BIGINT column.
SmallIntVal	Represents a SMALLINT column.
TinyIntVal	Represents a TINYINT column.
StingVal	Represents a STRING column. It has a len field that represents the length of the string and a ptr field that points to the string data. It also has a constructor that creates a new StingVal struct based on a null-terminated C-style string or a pointer plus a length. It also has a constructor that takes a pointer to a FunctionContext struct and length, which does not allocate space for a new copy of the string data that you can use in UDFs that return string values.
BooleanVal	Represents a BOOLEAN column.
FloatVal	Represents a FLOAT column.

last-updated: Apr 11, 2022

DoubleVal	Represents a DOUBLE column.
TimestampVal	Represents a TIMESTAMP column. It has a 32-bit integer date field that represents the Gregorian date and a 64-bit integer time_of_day field that represents the current time of day in nanoseconds.

Variable-Length Argument List

Each argument is named explicitly in the signature of your C++ function. A UDF can take a fixed number of these named arguments, however a function can accept additional arguments if they are all of the same type.

You must code the signature of your function using the following format if you want your UDF to accept a variable-length argument list:

StringVal Concat(FunctionContext* context, const StringVal& separator, int num_var_args, const StringVal* args);

The SQL query call must pass at least one argument to the variable-length portion of the argument list. Impala calls the function and fills in the initial set of required arguments and then passes the extra arguments and a pointer to the first of the optional arguments.

NULL Values

Each UDF that you write must have the ability to handle NULL values. If a UDF receives a NULL value, it typically also returns a NULL value.

UDF Memory Allocation

Any memory allocated to a UDF is taken back by the system after the UDF exits. Input arguments remain allocated for the lifetime of a function. You can refer to the input arguments in expressions for return values. When using temporary variables to construct all new string values, use the StringValue() constructor, and copy the data into the newly allocated memory buffer. The StringValue() constructor takes an initial FunctionContext* argument followed by a length.

UDF Error Handling

Call functions that are members of the initial FunctionContext* argument passed to your function to handle UDF errors.

Use the following signature in the function to record warnings for conditions that indicate minor, recoverable problems that do not cause the query to stop:

```
bool AddWarning(const char* warning_msg);
```

Use the following signature if you want the UDF to set an error flag that prevents the query from returning any results in serious cases where the query cancellation must occur.

```
void SetError(const char* error_msg);
```

Writing UDAFs

A UDAF must maintain a state value across subsequent calls in order to accumulate a result across a set of calls, instead of deriving it purely from one set of arguments.

The underlying functions represent a UDAF:

Initialization function	Sets counters to zero, creates empty buffers, and performs any other initial setup for a query.
Update function	Processes the arguments for each row in the result set and accumulates an intermediate result for each node.

last-updated: Apr 11, 2022

Merge function	Combines the intermediate results from different nodes.
Finalize function	Passes through the combined result unchanged, or does one final transformation.

Deploying UDFs and UDAFs

To deploy your UDFs or UDAFs, complete the following steps from the impala-shell:

1. Issue a USE statement to the database that you want to associate your new function with.
2. Issue a CREATE FUNCTION statement in the impala-shell to make Impala aware of the new function.
   - To deploy a UDF, issue the CREATE FUNCTION statement.
   - To deploy a UDAF, issue the CREATE AGGREGATE FUNCTION statement. Specify the entry points of the underlying C++ functions using the clauses INIT_FN, UPDATE_FN, and FINALIZE_FN.

Warning: You can also issue the CREATE FUNCTION with a fully qualified database name to skip the USE statement step. Example: `CREATE FUNCTION <db_name.function_name>`

Hive UDFs

To use Hive UDFs in Impala, get the applicable JAR files from the Hive UDFs, and use the Impala UDF deployment process to create new UDFs with new names.

Data types of arguments must match the function signature exactly when reusing Hive Java code for built-in functions.

To use a Hive UDF with Impala, complete the following steps:

1. Get a copy of the Hive JAR file with the UDFs that you want to use with Impala.
2. Issue the following command to see a list of classes inside the JAR file: `jar -tf <jar_filename>`
3. Copy the JAR file to a MapR filesystem location that Impala can read.
4. From the impala-shell, create a database to use with the UDF.
5. To identify the data base that you want to query using the UDF, issue the USE statement to through the impala-shell for that particular database, or specify the SQL function name as `db_name.function_name`.
6. Issue a CREATE FUNCTION statement for each UDF that you want to use with Impala. The CREATE FUNCTION statement should contain a LOCATION clause with the full MapR filesystem path to the JAR file and a SYMBOL clause with a fully qualified name of the class. Use dots as separators. Do not use the `.classpath` extension.
7. Issue a query and call the function. Pass the correct type of arguments to the function.

UDF Limitations

Impala UDFs have the following limitations:

- Impala cannot work with UDFs that accept or return composite, nested, or types not available in Impala tables.
- UDFs must produce the same output each time the same argument value is passed.
- UDFs cannot spawn other threads or processes.

last-updated: Apr 11, 2022

- UDFs become undefined when you restart the catalogd process. In this scenario, you must reload the UDFs.
- You currently cannot include user-defined table functions in Impala queries.

UDF Security

If you enable the Impala authorization feature, consider the following:

- You must have the required read privilege for database and tables used in the query to call a UDF in the query.
- Only an administrative user can create UDFs because improperly coded UDFs can cause performance and capacity issues. Issuing the CREATE FUNCTION statement requires the ALL privilege on the server.

Supported SQL Language Features

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The Impala SQL dialect supports many standard SQL language features and some extensions for Hadoop related to data loading and data warehousing. The impala-shell interpreter supports multi-line commands. A semicolon at the end of each statement is required.

SQL Features

Impala supports the following SQL features:

- Comments
- External/Internal tables
- Hints
- Joins
- Views

Statements

Impala supports the following SQL statements:

- ALTER TABLE
- CASCADE
- CREATE DATABASE
- CREATE TABLE
- CREATE TABLE AS SELECT
- CREATE VIEW
- DESCRIBE
- DROP DATABASE
- DROP TABLE/ALTER TABLE DROP PARTITION
- EXPLAIN

- INSERT
- INVALIDATE METADATA
- LOAD DATA
- PURGE
- REFRESH
- SELECT
- SHOW
- SHOW DATABASES
- TRUNCATE TABLE
- USE

Clauses

Impala supports the following SQL clauses:

- IF EXISTS
- WHRRE
- VALUES
- WITH
- GROUP BY
- HAVING
- LIMIT
- ORDER BY

Operators

Impala supports the following SQL operators:

- BETWEEN
- DISTINCT
- ILIKE
- IREGEXPR
- IS [NOT] DISTINCT FROM
- LIKE
- REGEXP
- RLIKE

Functions

Impala supports the following SQL functions:

- AVG
- BITAND
- BITNOT
- BITOR
- BITXOR
- BTRIM
- CHR
- COT
- COUNT
- COUNTSET
- CUME_DIST
- DCEIL
- DEXP
- DFLOOR
- DLOG10
- DPOW
- DROUND
- DSQRT
- DTRUNC
- FACTORIAL
- FPOW
- GETBIT
- GROUP_CONCAT
- INT_MONTHS_BETWEEN
- MAX
- MIN
- MONTHS_BETWEEN
- NDV
- NTILE

last-updated: Apr 11, 2022

- PERCENT_RANK
- REGEXP_LIKE
- ROTATELEFT
- ROTATERIGHT
- SETBIT
- SHIFTLEFT
- SHIFTRIGHT
- SPLIT_PART
- SUM
- TIMEOFDAY
- TIMESTAMP_CMP
- TYPEOF

Data Types

Impala supports the following SQL data types:

- BIGINT
- BOOLEAN
- DECIMAL
- DOUBLE
- FLOAT
- INT
- SMALLINT
- NULL
- STRING
- TIMESTAMP
- TINYINT

Impala supports the following complex data types:

- STRUCT
- ARRAY
- MAP

Supported and Unsupported SQL/HiveQL Language Features

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

last-updated: Apr 11, 2022

Supported HiveQL Statements and Clauses

- JOIN
- AGGREGATE
- DISTINCT
- UNION ALL
- ORDER BY (Must use a LIMIT clause with ORDER BY.)
- LIMIT (Accepts arithmetic expressions and numeric literals. You can include the OFFSET clause with the LIMIT clause to produce paged result sets, like 11-20.)
- FROM (You can include FROM in an uncorrelated subquery.)
- INSERT INTO
- INSERT OVERWRITE
- LEFT|RIGHT ANTI JOIN

Supported SQL Statements

When you issue an SQL statement, use a semicolon at the end of each statement.

- COMPUTE INCREMENTAL STATS
- COMPUTE STATS
- CREATE ROLE
- CREATE TABLE
  - Use the STORED AS PARQUET or STORED AS TEXTFILE clause with CREATE TABLE to identify the format of the underlying data files.
  - Use the TBLPROPERTIES clause with CREATE TABLE to associate random metadata with a table as key-value pairs.
  - Use SERDEPROPERTIES with CREATE TABLE to set up metadata that defines how tables are read and written. This may be required for Hive compatibility.
  - USE WITH SERDEPROPERTIES with CREATE TABLE to specify the SerDe classes that read and write data for a table. This may be required for Hive compatibility.
- CREATE TABLE AS SELECT
- CREATE VIEW
- CURRENT DATABASE(Returns the database that the session is currently using.)
- DESCRIBE
- DROP INCREMENTAL STATS
- DROP ROLE
- EXPLAIN (You can issue the command SET EXPLAIN_LEVEL=verbose for more detailed EXPLAIN output. To revert, issue the command SET EXPLAIN_LEVEL=normal.)

- EXPLAIN PLAN
- GRANT
- INVALIDATE METADATA
- LOAD DATA
- REFRESH
- REVOKE
- SELECT
  - You can use the STRAIGHT_JOIN keyword immediately after SELECT to override the reordering of join clauses that Impala does internally.
- SHOW CREATE TABLE
- SHOW DATABASES
- SHOW FILES
- SHOW FUNCTIONS
- SHOW GRANT ROLE
- SHOW ROLE GRANT GROUP
- SHOW ROLES
- SHOW SCHEMAS
- SHOW TABLES
- SHOW COLUMN STATS
- SHOW TABLE STATS
- USE

Supported DDL Statements

DDL statements typically change the table schema. Issue an INVALIDATE METADATA statement manually on the other nodes to update metadata. You can use the SET command to enable the SYNC_DDL option with any DDL statements to return after the Impala catalog service has propagated changes on all Impala nodes.

- ALTER TABLE
  - Use the TBLPROPERTIES clause with ALTER TABLE to associate random metadata with a table as key-value pairs.
  - Use SERDEPROPERTIES with ALTER TABLE to set up metadata that defines how tables are read and written. This may be required for Hive compatibility.
- ALTER VIEW
- COMPUTE STATS
- CREATE DATABASE

- CREATE FUNCTION
- CREATE ROLE
- CREATE TABLE
- CREATE VIEW
- DROP DATABASE
- DROP FUNCTION
- DROP ROLE
- DROP TABLE
- DROP VIEW
- GRANT
- REVOKE

Supported DML Statements

Impala only supports the INSERT and LOAD DATA statements which modify data stored in tables.

- Issue the REFRESH statement on other nodes to refresh the data location cache.
- If the SYNC_DDL statement is enabled, INSERT statements complete after the catalog service propagates data and metadata changes to all Impala nodes.
- You can use the [SHUFFLE] and [NOSHUFFLE] hints with INSERT to redistribute work during INSERT...SELECT operations. You may want to use these hints when inserting into partitioned Parquet tables to avoid memory consumption problems and simultaneous open files, by collecting new data for each partition on a specific node.

Supported Datatypes

Impala supports datatypes with the same names and semantics equivalent to those in Hive.

- char
- varchar
- complex
- string
- tinyint
- smallint
- int
- bigint
- float
- double
- boolean

- string
- timestamp
- asSTRING

Supported Operators

SQL operators are a class of comparison functions used within the WHERE clauses of a SELECT statement.

- BETWEEN
- COMPARISON
- CROSS JOIN (Use as the join operator for Cartesian joins; does not use any ON clause.)
- IN
- IS NULL
- LIKE
- [NOT] EXISTS
- [NOT] IN
- REGEXP
- RLIKE

Supported Literals

Each of the Impala data types has corresponding notation for literal values of that type. Specify literal values in SQL statements, such as in the SELECT list or WHERE clause of a query, or as an argument to a function call.

- Numeric
- String
- Boolean
- Timestamp
- NULL (You can add the NULLS FIRST or NULLS LAST clause at the end of the ORDER BY clause to override or specify the sorting behavior for NULL.)

Supported Schema Objects and Object Names

- Aliases
- Identifiers
- Databases
- Tables
- Views
- Functions

Analytic Functions

Analytic (window) functions operate on a set of rows and return a single value for each row from the underlying query. The term "window" describes the set of rows on which the function operates. A window function uses values from the rows in a window to calculate the returned values. When you use a window function in a query, you define the window using the OVER() clause. The OVER() clause (window clause) differentiates window functions from other analytical and reporting functions.

As of Impala 2.2.0, you can use the following analytic functions in queries:

- MAX()
- MIN()
- COUNT()
- AVG()
- RANK()
- LAG()
- LEAD()
- FIRST_VALUE()
- MAX(), SUM()
- COUNT()

The analytic functions support the following syntax:

```
function(args) OVER([partition_by_clause] [order_by_clause [window_clause]])
partition_by_clause ::= PARTITION BY expr [, expr ...]
order_by_clause ::= ORDER BY expr  [ASC | DESC] [NULLS FIRST | NULLS LAST] [, expr
[ASC | DESC] [NULLS FIRST | NULLS LAST] ...]
```

The window clause supports the following syntax:

```
ROWS BETWEEN [ { m | UNBOUNDED } PRECEDING | CURRENT ROW] [ AND [CURRENT ROW |
{ UNBOUNDED | n } FOLLOWING] ]
RANGE BETWEEN [ {m | UNBOUNDED } PRECEDING | CURRENT ROW] [ AND [CURRENT ROW |
{ UNBOUNDED | n } FOLLOWING] ]
```

Supported Aggregate Functions

Aggregate functions calculate a return value across all the items in a result set. When you issues a function, you must include a FROM clause in the query. Aggregate functions ignore NULL values rather than returning a NULL result.

- APPX_MEDIAN()
- AVG()
- COUNT()
- GROUP_CONCAT()
- MAX()
- MIN()
- NDV()

- SUM()

Built-in Functions

Impala supports lots of built-in functions that you can use directly with a SELECT statement to perform data transformation operations, such as mathematical calculations, string manipulations, and data calculations.

Unsupported SQL Features

- Non-scalar data types, such as maps, arrays, structs
- LOAD DATA to load raw files
- Restricted set of literal formats for the TIMESTAMP data type and the from_unixtime() format string
- Extensibility mechanisms such as TRANSFORM, custom file formats, or custom SerDes
- XML and JSON functions
- The following HiveQL aggregate functions: variance, var_pop, var_samp, stddev_pop, stddev_samp, covar_pop, covar_samp, corr, percentile, percentile_approx,histogram_numeric, collect_set
- User Defined Table Generating Functions (UDTFs)
- User Defined Aggregate Functions (UDAFs)
- User Defined Funtions
- Sampling
- Lateral views
- Authorization features such as roles
- Multiple DISTINCT clauses per query

Unsupported HiveQL Statements

- ANALYZE TABLE (the Impala equivalent is COMPUTE STATS)
- DESCRIBE COLUMN
- DESCRIBE DATABASE
- EXPORT TABLE
- IMPORT TABLE
- SHOW PARTITIONS
- SHOW TABLE EXTENDED
- SHOW TBLPROPERTIES
- SHOW FUNCTIONS
- SHOW COLUMNS
- SHOW CREATE TABLE
- SHOW INDEXES

last-updated: Apr 11, 2022

Semantic Differences in Impala Statements vs HiveQL

- Different syntax and names for query hints.
- MapReduce specific features of SORT BY, DISTRIBUTE BY, or CLUSTER BY are not exposed.
- Queries do not need a FROM clause.
- Impala does not allow:
  - Implicit cast between string and numeric or Boolean types
  - Implicit casts among the numeric types or from string to timestamp
  - Storing timestamps using the local timezone; Timestamps are stored relative to GMT
  - Return column overflows as NULL; Impala returns the largest or smallest value in the range for the type
  - Virtual columns
- Impala does not:
  - Expose locking
  - Expose some configuration properties

Impala Limitations

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

What Impala Does Not Provide

Impala does not replace Hive or other frameworks built on MapReduce for long-running batch-oriented queries.

Impala is not fit as a query layer to support operational/OLTP applications (No update/deletes, not optimized for point look-ups).

Known Limitations in Impala

Impala has the following known limitations:

- The LOAD DATA statement does not work when the source directory and destination table are in different encryption zones.
- The Impala configuration option, `--disk_spill_encryption`, is not supported to secure sensitive data from being observed or tampered with when temporarily stored on disk.
- Redaction of sensitive data from Impala log files is not supported.
- You cannot use the lineage information feature to track who has accessed data through Impala SQL statements.

Impala UDFs (user-defined functions) have the following known limitations:

- Impala does not work with UDFs that accept or return composite, nested, or types not available in Impala tables.
- UDFs must produce the same output each time the same argument value is passed.

- UDFs cannot spawn other threads or processes.
- Prior to Impala 2.5.0, UDFs become undefined when you restart the catalog service. You must reload the UDFs.
- You currently cannot include user-defined table functions in Impala queries.

Impala Security

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Features

Warning: While Impala is compatible with the platform, using Impala security is not encouraged for several reasons:

- By design, Impala security secures data outside the underlying filesystem, which creates the potential of backdoor access. When you enable Impala authorization using Sentry, the MapR platform security is bypassed and the MapR converged data platform can no longer fully secure your data.
- If you enable this Impala security, impersonation will be disabled and data ownership will be shifted to the Impala user, which makes the "Impala" data inaccessible by regular users through means other than Hive or Impala (because the users no longer own the data).

You can configure Impala to use the security features listed in the next table on either a secure or a non-secure MapR cluster. If you use the MapR Installer and select **Enable Secure Cluster**, Impala will not be automatically secured.

Feature	Description
LDAP	You can configure LDAP authentication for client connections with Impala. You can use LDAP authentication with Sentry to authenticate users and provide precise levels of access to users. See LDAP Authentication for Impala.
Kerberos	You can configure Impala to use Kerberos for authentication. You can also use Sentry authorization in conjunction with Kerberos if you want to configure user-level access to databases, tables, columns, and partitions. See Enable Kerberos Authentication for Impala.
MapR Security	You can configure MapR security between Impala and Hive. See Configure Hive Metastore to use MapR-SASL on page 3437. **Note:** MapR security is not present between the Impala client and the Impala server. To avoid security holes, you must configure the Impala client on Kerberos or LDAP.
SSL	You can enable SSL network encryption for communication between Impala and client programs and between Impala nodes in a cluster. See Enable SSL for Impala.

last-updated: Apr 11, 2022

Important: The Impala client does not support MapR ticket security, but you can authentication connections as follows:

- Between the Impala server and client (JDBC, Impala-shell) - Kerberos or LDAP. However, you might encounter issues with Impala on Kerberos using the JDBC connector.
- Between Impala (the Impala catalog) and Hive metastore - MapR ticket security or Kerberos.

To avoid security holes, configure Impala on Kerberos or LDAP. If Impala is not secure or only has LDAP authentication enabled, only the client connection to Impala is authenticated and there is no wire level encryption or server-to-server authentication.

You can enable MapR SASL for the Hive metastore. When the Hive metastore is SASL enabled, Impala can run in any security mode (none, LDAP, or Kerberos).

Component Compatibility

You can configure Impala to use the components and/or features listed below on a secure MapR cluster. The following table assumes that each component is configured with Kerberos on Impala. Hive and Hue can be configured with MapR security for authentication.

Note: MapR security is not present between the Impala client and the Impala server. To avoid security holes, you must configure the Impala client on Kerberos or LDAP. Hive and Hue use security.

Component	Version	Impala 1.4.1	Impala 2.2.0	Impala 2.5.0	Impala 2.7.0	Impala 2.10
MapR	6.0.x and later	Yes	Yes	Yes	Yes	Yes
	5.1.x and later	Yes	Yes	Yes	Yes	Yes
	5.0.x	Yes	Yes	No	No	No
	4.0.1	Yes	No	No	No	No
LDAP	N/A	Yes	Yes	Yes	Yes	Yes
Kerberos	N/A	Yes	Yes	Yes	Yes	Yes
Sentry	1.7	No	No	No	Yes	Yes
	1.6	No	Yes	Yes	No	No
Hue	4.2	No	No	No	No	Yes
	3.12	No	No	No	Yes	Yes
	3.9	Yes	Yes	Yes	No	No
	3.6	Yes	No	No	No	No
	1.4	Yes	No	No	No	No
Hive	2.3	No	No	No	No	Yes
	2.1.x	No	No	No	Yes	Yes
	1.2.1	No	Yes	Yes	No	No
	0.13	Yes	No	No	No	No

The following table lists the supported and unsupported component and security combinations necessary to access the Hive metastore:

last-updated: Apr 11, 2022

Impala Client Security Mode	Hive + MapR SASL	Hive + Kerberos
	Note: The Impala Catalog will access Hive Metastore using MapR security.	
None	Supported	Not supported
LDAP	Supported	Not supported
Kerberos	Supported **Note:** Issues with JDBC might exist.	Supported

Related concepts

Using the Enable MapR Secure Cluster Option on page 5427
You use the Enable MapR Secure Cluster option to control whether or not the cluster is configured as a secure cluster.

Enable Kerberos Authentication

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You can enable Kerberos authentication for Impala on a secure and non-secure MapR cluster.

Once you have configured Impala to use Kerberos for authentication, restart Impala and then start the `impala-shell` with the `-s mapr -k` flags to enable Kerberos.

To enable Kerberos authentication for Impala, complete the following steps:

1. Enable encryption between RPC services on Kerberos in core-site.xml (change "hadoop.rpc.protection" value to "privacy") and copy core-site.xml to Impala-conf directory:

   ```
   <property>
     <name>hadoop.rpc.protection</name>
     <value>privacy</value>
   </property>
   ```

   And restart the Warden service to apply changes in core-site.xml.

2. Copy the following files to the `$IMPALA_HOME/conf/` directory:

   - `$HIVE_HOME/conf/hive-site.xml`
   - `$HADOOP_HOME/etc/hadoop/core-site.xml`

   Note: Any time the `hive-site.xml` file is modified, copy the file to the `$IMPALA_HOME/conf/` directory.

3. Create service principals for each host that runs impalad, catalogd, or statestored and for the HTTP service. Principal names take the following form:

   ```
   mapr/<fully.qualified.domain.name>@<KERBEROS.REALM>
   ```

   a. Create an Impala service principal and specify the following information:

      - Name “mapr”

last-updated: Apr 11, 2022

- Fully qualified domain name of each node running impalad
- Realm name

```
kadmin: addprinc -requires_preauth -randkey -allow_renewable mapr/
impala_host.example.com@TEST.EXAMPLE.COM
```

b. Create an HTTP service principal.

```
kadmin: addprinc -randkey HTTP/
impala_host.example.com@TEST.EXAMPLE.COM
```

4. Create, merge, and distribute keytab files for the principals.

a. Create keytab files with both principals.

```
kadmin: xst -k /opt/mapr/conf/mapr.keytab mapr/impala_host.example.com
```

b. Use the keytab utility to read the content of the keytab files and then write the content to a new file.

```
ktutil
ktutil: rkt /opt/mapr/conf/mapr.keytab
ktutil: rkt /opt/mapr/conf/http.keytab
ktutil: wkt /opt/mapr/conf/mapr-http.keytab
ktutil: quit
```

c. Optionally, test the credentials in the merged keytab file to verify their validity and to verify that “renew until” data is set to a future time.

```
klist -e -k -t /opt/mapr/conf/mapr-http.keytab
```

d. Change the file owner to the `mapr` user to make `mapr` the only user authorized to read the file content.

```
chmod 400 /opt/mapr/conf/mapr-http.keytab
```

5. Edit `/opt/mapr/impala/impala-<version>/conf/env.sh` to include the fully qualified domain name for the IMPALA_STATE_STORE_HOST, IMPALA_STATE_STORE_HOST variables, and Kerberos options.

a. Set the IMPALA_STATE_STORE_HOST and CATALOG_SERVICE_HOST variables to point to the fully qualified domain name.

```
IMPALA_STATE_STORE_HOST=impala_host.example.com
IMPALA_STATE_STORE_PORT=24000
CATALOG_SERVICE_HOST=impala_host.example.com
```

b. Add the following Kerberos options for impalad, catalogd, and statestored daemons using the IMPALA_SERVER_ARGS, IMPALA_CATALOG_ARGS, and IMPALA_STATE_STORE_ARGS variables:

```
-kerberos_reinit_interval=60
-principal=mapr/impala_host.example.com@TEST.EXAMPLE.COM
-keytab_file=/opt/mapr/conf/mapr-http.keytab
```

```
IMPALA_SERVER_ARGS=" \
     -log_dir=${IMPALA_LOG_DIR} \
     -state_store_port=${IMPALA_STATE_STORE_PORT} \
     -use_statestore \
     -authorized_proxy_user_config=mapr=* \
     -state_store_host=${IMPALA_STATE_STORE_HOST} \
     -catalog_service_host=${CATALOG_SERVICE_HOST} \
     -be_port=${IMPALA_BACKEND_PORT} \
     -disable_admission_control=true \
     -kerberos_reinit_interval=60 \
     -principal=mapr/impala_host.example.com@TEST.EXAMPLE.COM \
     -keytab_file=/opt/mapr/conf/mapr-http.keytab "
```

c. Restart Impala and the catalog and statestore services. See Managing Impala.

d. To enable Kerberos from the impala-shell, start the impala-shell with the `-s mapr -k` flags.

```
impala-shell -s mapr -k
```

For more information on changing the Impala defaults specified in `env.sh`, see Impala-Shell Commands.

Enable SSL for Impala

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Impala 2.5.0 supports SSL encryption for internal Impala connections.

Complete the following steps to configure SSL for Impala:

1. Configure encryption in Hive. See Hive Encryption.
2. Configure client-server encryption only or configure client-server and Impala internal encryption.
   - To configure client-server encryption only, add the following start-up options for the Impala Server to `/opt/mapr/impala/impala-<version>/conf/env.sh`:

`-ssl_server_certificate`	Full path to the server certificate on the local filesystem.
`-ssl_private_key`	Full path to the server private key on the local filesystem.

   - To configure client-server and Impala internal encryption, add the following start-up options for the Impala server, catalog, and statestore to `/opt/mapr/impala/impala-<version>/conf/env.sh`:

`-ssl_server_certificate`	Full path to the server certificate on the local filesystem.
`-ssl_private_key`	Full path to the server private key on the local filesystem.

last-updated: Apr 11, 2022

`-ssl_client_ca_certificate`	Full path to the certificate on the local filesystem required for client/server encryption.

Note: When you add the SSL flags to Impala start-up options, Impala listens for HiveServer2 on the SSL-secured ports. A client program usually has equivalent options to verify a connection to the correct server.

After you enable SSL, you can issue the following options when you start the impala-shell:

Option	Description
`--ssl`	Enables SSL for the impala-shell.
`--ca_cert`	Local path name that points to the third-party CA certificate, or to a copy of the server certificate for self-signed server certificates. If --ca_cert is not set, impala-shell enables SSL, but does not validate the server certificate. This is useful for connecting to an Impala node that you know is only running over SSL when a copy of the certificate is not available.

For more information about the impala-shell, refer to Impala-Shell Commands.

For more information about configuring Impala start-up options, see Additional Impala Configuration Options.

LDAP Authentication

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

As of Impala 1.2.3, client connections with Impala can be authenticated against LDAP servers. You can configure LDAP authentication for client connections with Impala on a non-secure MapR cluster. When you configure LDAP authentication, you must configure SSL between the client and Impala, and between Impala and the LDAP server to avoid sending credentials over the wire in clear text. Configuration requirements apply to the server side when configuring and starting Impala.

If the Hive metastore has MapR-SASL enabled, copy `$HIVE_HOME/conf/hive-site.xml` to `$IMPALA_HOME/conf/`. Repeat this step any time the `hive-site.xml` file is modified.

Modify `/opt/mapr/impala/impala-<version>/conf/env.sh` and include the following options to configure Impala server LDAP authentication:

Options	Description
--enable_ldap_auth	Set this option to "`true`" in all environments to enable LDAP authentication between Impala and the client.
--ldap_uri	Sets the URI of the LDAP server to use. Typically, the URI is prefixed with `ldap://<hostname>`. Optionally, the URI can specify the port. For example, `ldap://<hostname>:<port>`.
--ldap_manual_config extend col	Bypasses all of the automatic configuration if you need to provide a custom SASL.
--ldap_tls	Tells Impala to start a TLS connection to the LDAP server and to fail authentication if Impala cannot start the TLS connection.

last-updated: Apr 11, 2022

Example:

```
IMPALA_SERVER_ARGS=" \
    -log_dir=${IMPALA_LOG_DIR} \
    -state_store_port=${IMPALA_STATE_STORE_PORT} \
    -enable_ldap_auth=true \
    -ldap_uri=ldap://10.250.1.5/ \
    -ldap_tls=true \
    -use_statestore \
    -state_store_host=${IMPALA_STATE_STORE_HOST} \
    -catalog_service_host=${CATALOG_SERVICE_HOST} \
    -be_port=${IMPALA_BACKEND_PORT}"
```

After you restart the Impala server, statestore, and catalog, you can connect to Impala using LDAP authentication. To connect to Impala, launch the impala-shell from a client node and issue the following commands:

Command	Description
`-l`	Enables LDAP authentication.
`-u`	Sets the user. The impala-shell prompts you for the password. Per Active Directory, the user is the short username, not the full LDAP distinguished name.
`--ldap_password_cmd`	(Impala 2.5.0 only) Sets the bash command to retrieve the LDAP user password.

Example:

```
impala-shell -l -u
uid=qa-user1,ou=People,dc=maprtech,dc=com --ldap_password_cmd="echo -n
      secret"
```

Sentry

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Important: MapR officially supports Sentry with Impala, Hive, Sqoop2, and Hue if you purchase Impala support and configure Impala to use Sentry.

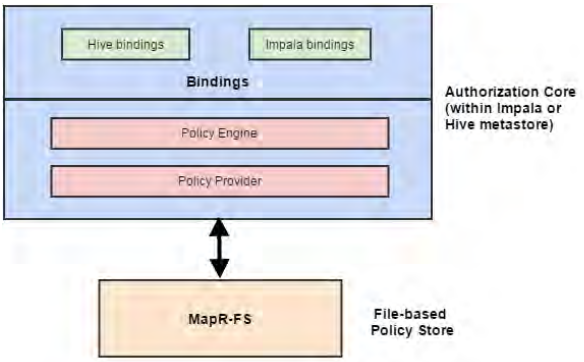
Apache Sentry is an authorization module for Hadoop that provides the granular, role-based authorization required to provide precise levels of access to authenticated users and applications. Sentry allows users to see only those objects for which they have privileges.

Storage Models

Sentry supports two models for storing policy rules:

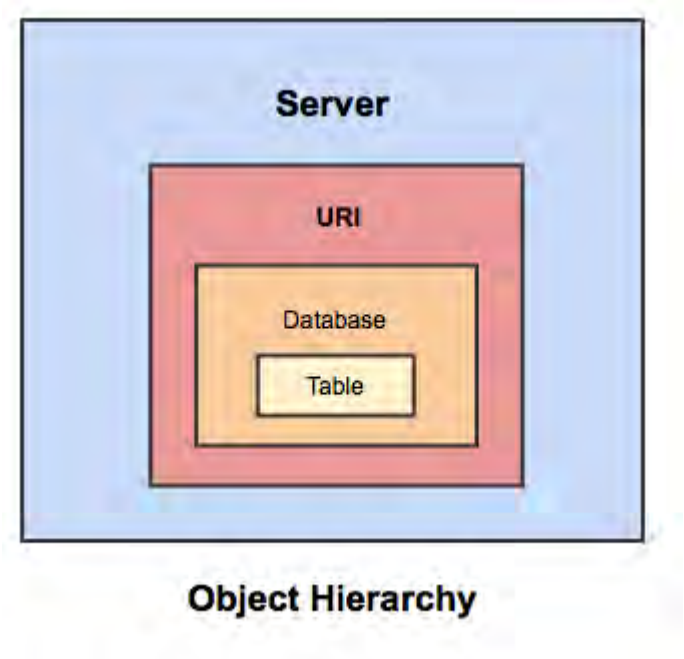
Database storage (preferred)	As of Sentry 1.6-1602, you can configure Sentry to use the database storage mode. With this mode, the Sentry service provides access to read and maintain privileges and roles from a database.
File-based storage	Privileges and roles are accessed from and maintained in a policy file (`global-policy.ini`) which you can store on the MapR filesystem. The following diagram illustrates the architecture of the file-based storage model:

last-updated: Apr 11, 2022



Privileges

Privileges are granted on different objects in the schema, including tables, databases, URIs and servers. The object hierarchy is set up like this, where objects inherit privileges from objects above them in the hierarchy:



Privileges and roles are specified in either a relational database (for database rule storage) or a `global-policy.ini` file (for file-based rule storage). The following examples show a `global-policy.ini` file along with the equivalent set of privileges and roles expressed in HiveSQL syntax.

Sample global-policy.ini File

```
[groups]
        # Assigns each Hadoop group to its set of roles
        manager = analyst_role, junior_analyst_role
        analyst = analyst_role
        jranalyst = junior_analyst_role
        customers_admin = customers_admin_role
        admin = admin_role

        [roles]
```

```
# The uris below define a landing pad which
# the user can use to import or export data from the system.
# Since the server runs as the user "hive," files in that directory
# must either have read/write permissions set for the group hive
# or have read/write permissions set for world.
analyst_role = server=HS2->db=analyst1, \
server=HS2->db=jranalyst1->table=*->action=select
server=HS2->uri=maprfs:/landing/analyst1
junior_analyst_role = server=HS2->db=jranalyst1, \
server=HS2->uri=maprfs:/landing/jranalyst1

# Implies everything on HS2.
admin_role = server=HS2
```

Known Issues

Upgrading Sentry

If you are upgrading from Sentry in EEP 6.3.1 or EEP 7.0.0 to Sentry in the latest EEP version, manually back up the `/conf` and `/logs` directories located in SENTRY_HOME. After the upgrade completes, add those directories back into the SENTRY_HOME directory.

You can see the `/conf` and `/logs` directories listed in the Sentry installation directory, as shown:

```
ll /opt/mapr/sentry/sentry-1.7.0
total 76
drwxr-xr-x 2 mapr mapr  4096 Jan  5
13:34 bin
-rw-r--r-- 1 mapr mapr 15211 Jan  5
10:26 CHANGELOG.txt
drwxr-xr-x 2 mapr mapr  4096 Jan  5
13:34 conf
drwxr-xr-x 2 mapr mapr  4096 Jan  5
13:34 conf.d
drwxr-xr-x 2 mapr mapr  4096 Jan  5
13:34 conf.new
drwxr-xr-x 4 mapr mapr 12288 Jan  5
13:34 lib
-rw-r--r-- 1 mapr mapr 16000 Jan  5
10:26 LICENSE.txt
drwxr-xr-x 2 mapr mapr  4096 Jan  5
13:34 logs
-rw-r--r-- 1 mapr mapr   388 Jan  5
10:26 NOTICE.txt
-rw-r--r-- 1 mapr mapr  1580 Jan  5
10:26 README.md
drwxr-xr-x 3 mapr mapr  4096 Jan  5
13:34 scripts
```

See Pre-Upgrade Steps for Sentry on page 346 and Upgrading Sentry on page 368.

Removing Sentry

This issue applies to Sentry in EEP 6.3.1 and EEP 7.0.0.

If you try to remove Sentry on an Ubuntu OS before the Sentry process has started (the Sentry process

does not exist), the system may return the following error message:

```
...
dpkg: error processing package
mapr-sentry (--purge):
 subprocess installed pre-removal
script returned error exit status 1
Errors were encountered while
processing:
 mapr-sentry
E: Sub-process /usr/bin/dpkg returned
an error code (1)
...
```

As a workaround, to completely remove Sentry on an Ubuntu OS, run:

```
sudo mv /var/lib/dpkg/info/
mapr-sentry.* /tmp/
sudo
dpkg --remove --force-remove-reinstreq
 mapr-sentry
sudo rm -rf /opt/mapr/sentry/
```

Sentry Feature Support

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Important: MapR support for Sentry is limited to Impala users.

- HDFS ACLs are not supported.
- As of MapR Sentry 1.6-1602, the database storage mode is supported. It was not supported in the Sentry 1.4.0-1412 release.
- Sentry 1.7.0-1703 supports Sentry Simple Shell and Hive Authorization V2.

The global-policy.ini File

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The default `global-policy.ini` file defines the `admin_role`, which gives full access to the Hiveserver2 server for the `mapr` user. The file is located in `/opt/mapr/sentry/sentry-<version>/conf` in your local file system. You can relocate the file to the MapR filesystem if you prefer. By default, the file contains these sections:

```
[groups]
      mapr = admin_role

      [roles]
      admin_role = server=HS2
```

Sample sentry-provider.ini File

```
[databases]
        # Defines the location of the per-DB policy file for the customer's
```

```
DB or schema
        customers = /etc/sentry/customers.ini

        [groups]
        customers_admin = customers_admin_role

        [roles]
        customers_admin_role = server=HS2->db=customers
```

Sample customers.ini File

```
[groups]
        manager = customers_insert_role, customers_select_role
        analyst = customers_select_role

        [roles]
        customers_insert_role =
server=HS2->db=customers->table=*->action=insert
        customers_select_role =
server=HS2->db=customers->table=*->action=select
```

Configuring Sentry
The steps to configure Sentry are determined by the storage model, the type of security used by the cluster, and the components that will use Sentry authorization.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

1. Configure Sentry to use one of the following storage models: file-based storage or database storage.
2. If the cluster is secure, you can configure Sentry to use kerberos authentication.
3. Based on your requirements, complete the steps to integrate Sentry with one or more ecosystem components.

Configure Sentry to use Database Storage

As of Sentry 1.6, the database storage model is the preferred method for storing privileges and roles. When you configure Sentry to use the database storage model, it also includes a service that is managed by Warden.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following databases can be used to store privileges and roles: MySQL, Oracle, Postgres, DB2 and Derby. Examples in the following procedure use MySQL as the database type.

Note: When you install Sentry with the MapR Installer and you specify MySQL as the database for Sentry, the MapR Installer performs the following configurations.

1. Create a database for Sentry.

   For example, run the following commands to create a MySQL database:

   ```
   mysql> create database sentry_store;
   mysql> use sentry_store;
   mysql> create user 'sentry'@'<hostname>' identified by 'sentry';
   mysql> grant all on *.* to 'sentry'@'<hostname>' identified by 'sentry';
   mysql> flush privileges;
   ```


last-updated: Apr 11, 2022

2. In `/opt/mapr/sentry/sentry-<version>/conf/sentry-site.xml`, update the value of the following property:

Property	Configuration
sentry.hive.provider.backend	Set the value to `org.apache.sentry.provider.db.SimpleDBProviderBackend`

For example:

```
<property>
  <name>sentry.hive.provider.backend</name>
  <value>org.apache.sentry.provider.db.SimpleDBProviderBackend</value>
  <description> Options:
{org.apache.sentry.provider.db.SimpleDBProviderBackend,
org.apache.sentry.provider.file.SimpleFileProviderBackend}Privilege
provider to be used, we support file based or db based</description>
</property>
```

3. In the `sentry-site.xml` file (`/opt/mapr/sentry/sentry-<version>/conf/sentry-site.xml`), add the following properties:

Property	Configuration
sentry.store.jdbc.url	Set the value to the JDBC connection URL.
sentry.store.jdbc.driver	Set the value to the Backend JDBC driver.
sentry.store.jdbc.user	Set the value to the JDBC user name.
sentry.store.jdbc.password	Set the value to the JDBC password.

For example:

```
<property>
  <name>sentry.store.jdbc.url</name>
  <value>jdbc:mysql://localhost/sentry_store</value>
</property>

<property>
  <name>sentry.store.jdbc.driver</name>
  <value>com.mysql.jdbc.Driver</value>
</property>

<property>
  <name>sentry.store.jdbc.user</name>
  <value>sentry</value>
</property>

<property>
  <name>sentry.store.jdbc.password</name>
  <value>sentry</value>
</property>
```

4. Initialize the database schema.

```
/opt/mapr/sentry/sentry-<version>/bin/sentry --command
schema-tool --conffile /opt/mapr/sentry/sentry-<version>/conf/
sentry-site.xml --dbType mysql --initSchema
```

5. To add Sentry to the list of services that Warden monitors, copy `/opt/mapr/sentry/sentry-<version>/conf.d/warden.sentry.conf` to `/opt/mapr/conf/conf.d`.

Configure Sentry to use File-Based Storage

You can configure Sentry to use the file-based storage model where permissions are stored in the global-policy.ini.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

As of Sentry 1.6.0-1602, the properties in step 1 are configured by default.

1. In the sentry-xml file (/opt/mapr/sentry/sentry-<version>/conf/sentry-site.xml), edit or verify the configuration of the following verify properties:

Property	Configuration
sentry.hive.provider.backend	Set the value to `org.apache.sentry.provider.file.SimpleFileProviderBackend`
sentry.hive.provider.resource	Set the value to the location of the global-policy.ini.

For example:

```
<property>
                        <name> sentry.hive.provider.backend </name>
                        <value>
org.apache.sentry.provider.file.SimpleFileProviderBackend </value>
                        <description> The privilege provider to be used
(either file-based or db-based). </description>
                        </property>

                        <property>
                        <name> sentry.hive.provider.resource </name>
                        <value> file:///opt/mapr/sentry/sentry-<version>/
conf/global-policy.ini </value>
                        <description> Provides location of the policy
file. If the policy file is in the filesystem, then the URL should start
from next schema: 'maprfs:///'. </description>
                        </property>
```

2. Complete the steps to integrate Hive with Sentry. This includes the step to configure the global-policy.ini file. See The global-policy.ini File on page 3820

Configure Sentry to use Kerberos Authentication

You can configure Sentry to run in a secure cluster that uses Kerberos authentication.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

last-updated: Apr 11, 2022

The same settings are valid for both the file-based and database storage modes.

1. Configure the following properties in the `/opt/mapr/sentry/sentry-<version>/conf/sentry-site.xml` file:

```
<property>
    <name>sentry.service.security.mode</name>
    <value>kerberos</value>
    <description>Options: kerberos, other, none. Authentication mode for
Sentry service.</description>
</property>
<property>
    <name>sentry.hive.testing.mode</name>
    <value>false</value>
</property>
```

2. Add the following properties to `/opt/mapr/sentry/sentry-<version>/conf/sentry-site.xml`:

```
<property>
    <name>sentry.service.server.principal</name>
    <value>mapr/<FQDN@REALM></value>
</property>
<property>
    <name>sentry.service.server.keytab</name>
    <value>/opt/mapr/conf/mapr.keytab</value>
</property>
<property>
    <name>sentry.service.allow.connect</name>
    <value>mapr,hive,impala</value>
</property>
```

3. Before starting Sentry, use the kinit tool:

```
kinit -kt /opt/mapr/conf/mapr.keytab -p mapr/<CLUSTER_NAME@REALM>
```

Example

```
kinit -kt /opt/mapr/conf/mapr.keytab -p mapr/my.cluster.com@NODE1
```

Integrating Sentry
Integrate Sentry with other ecosystem components to enable those components to use Sentry authorization. You can also integrate Sentry with Hue in order to use the Hue Security application to manage Sentry privileges and roles.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Important: MapR officially supports Sentry with Impala, Hive, Sqoop2, and Hue only if you buy Impala support and configure Impala to use Sentry.

You can configure Sentry to provide authorization in the following scenarios:

Sentry Version	Sentry Storage Mode	Component Integration
1.6.0	Database	Impala 2.2.0 and Hive 1.2.1
1.6.0	Database	Sqoop2

last-updated: Apr 11, 2022

Sentry Version	Sentry Storage Mode	Component Integration
1.6.0	File-Based	Impala 2.2.0 and Hive 1.2.1
1.4.0	File-Based	Impala 1.4.1 and Hive 0.13

Note:

- When you configure Impala to use Sentry authorization, both Impala and Hive must use Sentry for authorization management. Also, impersonation cannot be enabled for Hive.
- For clusters with Hive but not Impala, authorization can be implemented using Hive's native authorization mechanisms;Sentry is not needed.

Configure Impala to Use Sentry Authorization

You can configure Impala to work with Sentry for authorization. When you configure Sentry authorization for Impala, Impala uses operating system IDs to associate privileges with each user that runs the impala-shell or client program.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Before you can configure Impala for Sentry authorization, you must have Hive and Sentry installed and configured to work together.

The following table provides the MapR and component versions required when you want to configure Impala to use Sentry authorization:

Component	Versions
MapR	6.1.0
Impala	Impala 2.12.0.x
Sentry	Sentry 1.7.0
Hive	Hive 2.3.6

Configure Impala to Use Sentry Authorization With the Database Storage Model

As of Sentry 1.6, Sentry can be configured to use the Database storage model. Complete the following steps to configure Impala to use Sentry authentication when Sentry uses the DB storage model:

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

1. Set the following properties in env.sh (/opt/mapr/impala/impala-<version>/conf/env.sh):

```
IMPALA_SERVER_ARGS=" -server_name=HS2 -sentry_config=<SENTRY-HOME>/conf/
sentry-site.xml
IMPALA_CATALOG_ARGS=" -sentry_config=<SENTRY-HOME>/conf/sentry-site.xml
IMPALA_STATE_STORE_ARGS=" -sentry_config=<SENTRY-HOME>/conf/
sentry-site.xml
```

Note: SENTRY-HOME is a variable used in place of the ecosystem component home directory. Typically, the default ecosystem component home directory for a component is `/opt/mapr/<component>/<component>-<version>/`.

last-updated: Apr 11, 2022

2. Run the following commands to restart Impala:

```
sudo -u mapr maprcli node services -name impalaserver -action
restart -nodes <nodename>
```

```
sudo -u mapr maprcli node services -name impalastore -action
restart -nodes <nodename>
```

```
sudo -u mapr maprcli node services -name impalacatalog -action
restart -nodes <nodename>
```

3. When Impala is running, you can issue the following command to start the impala-shell as a particular user:

```
impala-shell -u <user_name>
```

Configure Impala to Use Sentry Authorization in File-Based Storage Model

Impala reads the file and controls which objects users can access and what operations they can perform on the objects. Impala caches the security information in the policy file every five minutes. If you make significant changes to security policies, restart Impala. Changes immediately take effect.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

1. Edit `env.sh` located in `/opt/mapr/impala/impala-<version>/conf/`.
2. In the IMPALA_SERVER_ARGS declaration, add the following options:

Option	Description
-server_name	This option turns on Sentry authorization for Impala. Specify the symbolic server name to use as the argument for this option. You must also specify this server name as the value for the sentry.hive.server property in the sentry-site.xml configuration file for Hive. For example: `-server_name=<hive_server_2>`
-authorization_policy_file	You can store privileges in an authorization policy file. When you specify this option, in addition to the server_name option, Impala reads privilege information from the policy file instead of a database. Specify the MapR filesystem path to the policy file that contains the privilege information. For example: `- authorization_policy_file=file:///opt/mapr/sentry/sentry-/conf/.ini \` **Note:** If the policy file is stored in MapR filesystem, indicate the MapR filesystem location using the following format: `-authorization_policy_file=maprfs:///<path_to_policy_file>`

last-updated: Apr 11, 2022

3. Restart the Impala server, statestore service, and catalog service.

```
sudo -u mapr maprcli node services -name impalaserver -action
restart -nodes <nodename>
```

```
sudo -u mapr maprcli node services -name impalastore -action
restart -nodes <nodename>
```

```
sudo -u mapr maprcli node services -name impalacatalog -action
restart -nodes <nodename>
```

Note: Impala does not start if it detects any issues in the authorization settings or the policy file.

4. When Impala is running, you can issue the following command to start the impala-shell as a particular user:

```
impala-shell -u <user_name>
```

Configure Sqoop2 to use Sentry Authorization

As of Sentry 1.6.0, you can configure Sqoop2 to use Sentry authentication when Sentry uses the database storage model, the cluster is secure, and the cluster uses Kerberos authentication. Use these steps:

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

1. Add the following properties to the `sentry-site.xml` file (`/opt/mapr/sentry/sentry-<version>/conf/sentry-site.xml`):

```
<property>
                        <name>sentry.sqoop.provider.backend</name>

<value>org.apache.sentry.provider.db.generic.SentryGenericProviderBackend
</value>
                        </property>

                        <property>
                        <name>sentry.service.allow.connect</name>
                        <value>mapr,sqoop</value>
                        <description>comma separated list of users -
List of users that are allowed to connect to the service (eg Hive,
Impala)</description>
                        </property>
```


last-updated: Apr 11, 2022

2. Configure the following properties in the `sqoop.properties` file (`/opt/mapr/sqoop/sqoop-2.0.0/server/conf/sqoop.properties`):

```
# Authentication configuration

org.apache.sqoop.security.authentication.type=KERBEROS

org.apache.sqoop.security.authentication.handler=org.apache.sqoop.security.authentication.KerberosAuthenticationHandler

org.apache.sqoop.security.authentication.kerberos.principal=mapr/<FQDN>@<REALM>

org.apache.sqoop.security.authentication.kerberos.keytab=/opt/mapr/conf/mapr.keytab

org.apache.sqoop.security.authentication.kerberos.http.principal=HTTP/<FQDN>@<REALM>

org.apache.sqoop.security.authentication.kerberos.http.keytab=/opt/mapr/conf/mapr.keytab

org.apache.sqoop.security.authentication.enable.doAs=true

org.apache.sqoop.security.authentication.proxyuser.mapr.users=*

org.apache.sqoop.security.authentication.proxyuser.mapr.groups=*

org.apache.sqoop.security.authentication.proxyuser.mapr.hosts=*

                        # Authorization configuration

org.apache.sqoop.security.authorization.handler=org.apache.sentry.sqoop.authz.SentryAuthorizationHander

org.apache.sqoop.security.authorization.access_controller=org.apache.sentry.sqoop.authz.SentryAccessController

org.apache.sqoop.security.authorization.validator=org.apache.sentry.sqoop.authz.SentryAuthorizationValidator

org.apache.sqoop.security.authorization.server_name=SqoopServer1
                        sentry.sqoop.site.url=file:///opt/mapr/sqoop/sqoop-2.0.0/server/conf/sqoop-sentry-site.xml
```

3. Copy the following JAR files from `/opt/mapr/sentry/sentry-<version>/lib` to `/opt/mapr/sqoop/sqoop-2.0.0/server/webapps/sqoop/WEB-INF/lib/`. (For Sqoop 1.99.7, use `/opt/mapr/sqoop/sqoop-2.0.0/server/lib/`):

   For Sentry Version 1.6.0

   - `sentry-provider-db-1.6.0-incubating-mapr-1606.jar`
   - `shiro-core-1.2.1.jar`
   - `sentry-core-common-1.6.0-incubating-mapr-1606.jar`
   - `sentry-core-model-db-1.6.0-incubating-mapr-1606.jar`
   - `sentry-core-model-search-1.6.0-incubating-mapr-1606.jar`

- `sentry-core-model-sqoop-1.6.0-incubating-mapr-1606.jar`
- `sentry-provider-common-1.6.0-incubating-mapr-1606.jar`
- `sentry-policy-common-1.6.0-incubating-mapr-1606.jar`
- `libthrift-0.9.2.jar`
- `sentry-provider-file-1.6.0-incubating-mapr-1606.jar`
- `sentry-binding-sqoop-1.6.0-incubating-mapr-1606.jar`
- `sentry-policy-sqoop-1.6.0-incubating-mapr-1606.jar`

For Sentry Version 1.7.0

- `sentry-provider-db-1.7.0-mapr-1703.jar`
- `shiro-core-1.2.3.jar`
- `sentry-core-common-1.7.0-mapr-1703.jar`
- `sentry-core-model-db-1.7.0-mapr-1703.jar`
- `sentry-core-model-search-1.7.0-mapr-1703.jar`
- `sentry-core-model-sqoop-1.7.0-mapr-1703.jar`
- `sentry-provider-common-1.7.0-mapr-1703.jar`
- `sentry-policy-common-1.7.0-mapr-1703.jar`
- `libthrift-0.9.2.jar`
- `sentry-provider-file-1.7.0-mapr-1703.jar`
- `sentry-binding-sqoop-1.7.0-mapr-1703.jar`
- `sentry-policy-sqoop-1.7.0-mapr-1703.jar`

4. Create `sqoop-sentry-site.xml` in the `/opt/mapr/sqoop/sqoop-2.0.0/server/conf/` directory. (If you use Sqoop 1.99.7, create `sqoop-sentry-site.xml` in the `/opt/mapr/sqoop/sqoop-2.0.0/conf` directory.)

last-updated: Apr 11, 2022

5. Add the following properties to the `sqoop-sentry-site.xml`:

```
<property>
                        <name>sentry.service.security.mode</name>
                        <value>kerberos</value>
                        </property>


                        <property>
                        <name>sentry.service.server.principal</name>
                        <value>mapr/<FQDN>@<REALM></value>
                        </property>

                        <property>
                        <name>sentry.service.server.keytab</name>
                        <value>/opt/mapr/conf/mapr.keytab</value>
                        </property>

                        <property>
                        <name>sentry.service.client.server.rpc-address</
name>
                        <value>localhost</value>
                        </property>

                        <property>
                        <name>sentry.service.client.server.rpc-port</
name>
                        <value>8038</value>
                        </property>

                        <property>
                        <name>sentry.sqoop.provider.backend</name>

<value>org.apache.sentry.provider.db.generic.SentryGenericProviderBackend
</value>
                        </property>

                        <property>
                        <name>sentry.service.admin.group</name>
                        <value>sqoop2,sqoop,hive,impala,solr,mapr</value>
                        </property>
```

6. Start the Sqoop2 server:

```
maprcli node services -name sqoop2 -action start -nodes <space delimited
list of nodes>
```

Configure Hive to use Sentry Authorization

Configure Hive to use Sentry when you want to use Sentry authorization with Impala.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

last-updated: Apr 11, 2022

Complete the following steps to configure Hive to use Sentry authorization, and create an `admin` role for the `mapr` user:

1. For Sentry 1.6.0 and Hive 1.2, add the following properties to `hive-site.xml`:

```
<property>
    <name>hive.server2.session.hook</name>
    <value>org.apache.sentry.binding.hive.HiveAuthzBindingSessionHook</
value>
</property>
<property>
    <name>hive.sentry.conf.url</name>
    <value>file:///opt/mapr/sentry/sentry-<version>/conf/
sentry-site.xml</value>
    <description>sentry-site.xml file location</description>
</property>
<property>
    <name>hive.metastore.rawstore.impl</name>
    <value>org.apache.sentry.binding.metastore.AuthorizingObjectStore</
value>
</property>
<property>
    <name>hive.metastore.filter.hook</name>

<value>org.apache.sentry.binding.metastore.SentryMetaStoreFilterHook</
value>
</property>
<property>
    <name>hive.server2.enable.doAs</name>
    <value>false</value>
    <description>Set this property to enable impersonation in
Hive Server 2</description>
</property>
<property>
    <name>hive.metastore.execute.setugi</name>
    <value>true</value>
</property>
<property>
    <name>hive.sentry.subject.name</name>
    <value>mapr</value>
</property>
<property>
    <name>hive.stats.collect.scancols</name>
    <value>true</value>
    <description>Property for use column level privileges in
Hive and Sentry Integration</description>
</property>
```

Note: The `hive.metastore.rawstore.impl` property is optional, but is recommended for metadata read protection.

last-updated: Apr 11, 2022

2. If Sentry uses the database storage model for rules, add the following properties to `hive-site.xml`:

```
<property>
    <name>hive.support.concurrency</name>
    <description>Enable Hive's Table Lock Manager Service</description>
    <value>true</value>
</property>
<property>
    <name>hive.zookeeper.quorum</name>
    <description>ZooKeeper quorum used by Hive's Table Lock Manager</
description>
    <value>hostname</value>
</property>
<property>
    <name>hive.zookeeper.client.port</name>
    <description>The port where the clients connect</description>
    <value>5181</value>
</property>
<property>
    <name>hive.security.authorization.task.factory</name>

<value>org.apache.sentry.binding.hive.SentryHiveAuthorizationTaskFactoryI
mpl</value>
</property>
<property>
    <name>hive.metastore.rawstore.impl</name>
    <value>org.apache.sentry.binding.metastore.AuthorizingObjectStore</
value>
</property>
<property>
    <name>hive.metastore.pre.event.listeners</name>
    <value>org.apache.sentry.binding.metastore.MetastoreAuthzBinding</
value>
    <description>list of comma separated listeners for metastore
events.</description>
</property>
```

3. To configure Sentry with Hive 2.1 or later, add the following properties to `HIVE_HOME/conf/hive-site.xml`:

```
<property>
    <name>hive.server2.session.hook</name>

<value>org.apache.sentry.binding.hive.v2.HiveAuthzBindingSessionHookV2</
value>
</property>
<property>
    <name>hive.sentry.subject.name</name>
    <value>mapr</value>
    <description>sentry-site.xml file location</description>
</property>
<property>
    <name>hive.sentry.conf.url</name>
    <value>file:///opt/mapr/sentry/sentry-1.7.0/conf/sentry-site.xml</
value>
    <description>sentry-site.xml file location</description>
</property>
<property>
    <name>hive.security.authorization.task.factory</name>

<value>org.apache.sentry.binding.hive.v2.SentryHiveAuthorizationTaskFacto
ryImplV2</value>
</property>
<property>
    <name>hive.metastore.rawstore.impl</name>

<value>org.apache.sentry.binding.hive.v2.metastore.AuthorizingObjectStore
V2</value>
</property>
<property>
    <name>hive.metastore.filter.hook</name>

<value>org.apache.sentry.binding.metastore.SentryMetaStoreFilterHook</
value>
</property>
<property>
    <name>hive.server2.enable.doAs</name>
    <value>false</value>
    <description>Set this property to enable impersonation in Hive
Server 2</description>
</property>
<property>
    <name>hive.metastore.execute.setugi</name>
    <value>true</value>
</property>
<property>
    <name>hive.internal.ss.authz.settings.applied.marker</name>
    <value>true</value>
</property>
<property>
    <name>hive.security.authorization.manager</name>
    <value>org.apache.sentry.binding.hive.v2.SentryAuthorizerFactory</
value>
</property>
<property>
    <name>hive.security.authenticator.manager</name>

<value>org.apache.hadoop.hive.ql.security.SessionStateUserAuthenticator</
value>
</property>
```


last-updated: Apr 11, 2022

```
<property>
    <name>hive.security.authorization.enabled</name>
    <value>true</value>
</property>
<property>
    <name>hive.metastore.pre.event.listeners</name>

<value>org.apache.sentry.binding.hive.v2.metastore.MetastoreAuthzBindingV
2</value>
    <description>list of comma separated listeners for metastore
events.</description>
</property>
<property>
<name>hive.metastore.event.listeners</name>

<value>org.apache.sentry.binding.hive.v2.metastore.SentryMetastorePostEve
ntListenerV2</value>
    <description>list of comma separated listeners for metastore, post
events.</description>
</property>
<property>
    <name>hive.zookeeper.client.port</name>
    <value>5181</value>
    <description>The Zookeeper client port. The MapR default clientPort
is 5181.</description>
</property>
<property>
    <name>hive.zookeeper.quorum</name>
    <description>Zookeeper quorum used by Hive's Table Lock Manager</
description>
    <value><!--host with Zookeper--></value>
</property>
```

4. Restart HiveServer2 and the Hive Metastore:

```
sudo -u mapr maprcli node services -name hs2 -action restart -nodes
<nodename>
sudo -u mapr maprcli node services -name hivemeta -action restart -nodes
<nodename>
```

5. If Sentry was configured to use the database storage model, issue the following command to restart Sentry:

```
sudo -u mapr maprcli node services -name sentry -action restart -nodes
<nodename>
```

6. Create the `admin` role.
   - For the database storage model, run the following commands from the Hive beeline to create the admin role for the mapr user:

```
>create role admin_role;
>grant all on server HS2 to role admin_role;
>grant role admin_role to group mapr;
```


last-updated: Apr 11, 2022

- For the file-based model, update the The global-policy.ini File on page 3820 in `/opt/mapr/sentry/sentry-<version>/conf`. For example:

```
[groups]
mapr = admin_role
testuser = test_role
[roles]
admin_role = server=HS2
test_role = server=HS2->db=test_db1->table=test_table->action=all
```

Note: If you include a non-existent mapping or path to a JAR file that represents a UDF (user-defined function) in any section of the `global-policy.ini` file, Sentry silently fails and cannot control access to Hive. For example, if you include a mapping to a role that does not exist in the `[groups]` section, Sentry fails. For more information, see Getting Started with Sentry in Hive.

Integrate Hue with Sentry

You can integrate Hue with Sentry 1.6 on a secure cluster that uses Kerberos authentication; however, the integration is not supported on a secure cluster using MapR-SASL authentication. Integrate Hue with Sentry when you want to use the Hue Security application to manage Sentry roles and privileges. When you integrate Hue with Sentry, Sentry must use the database storage model.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Note: The Hue user that manages Sentry roles and privileges in Hue must belong to a Hue group that is also part of the Sentry admin group.

1. In the [libsentry] section of the hue.ini (/opt/mapr/hue/hue-<version>/desktop/conf/hue.ini), configure the following properties:

```
[libsentry]
                        # Hostname or IP of server.
                        hostname=localhost

                        # Port the sentry service is running on.
                        port=8038

                        # Sentry configuration directory, where
sentry-site.xml is located.
                        sentry_conf_dir=/opt/mapr/sentry/
sentry-<version>/conf/sentry-site.xml
```

2. If Sentry uses Kerberos authentication, edit the following property in sentry-site.xml (/opt/mapr/sentry/sentry-<version>/conf/sentry-site.xml) to enable the Hue admin user (mapr) to connect to Sentry:

```
<property>
                        <name>sentry.service.allow.connect</name>
                        <value>impala,hive,mapr</value>
                        </property>
```

3. Restart Hue:

```
maprcli node services -name hue -action restart -nodes <ip_address>
```

Sentry 1.7.0 API Changes

last-updated: Apr 11, 2022

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

This page lists the API changes that occurred between Sentry 1.6.0 and Sentry 1.7.0:

Added Classes

```
org.apache.sentry.binding.metastore.AuthorizingObjectStoreBase

org.apache.sentry.binding.metastore.SentryMetastorePostEventListenerBase
          org.apache.sentry.binding.hive.v2.HiveAuthzBindingHookV2
          org.apache.sentry.binding.hive.v2.HiveAuthzBindingSessionHookV2
          org.apache.sentry.binding.hive.v2.HiveAuthzPrivilegesMapV2
          org.apache.sentry.binding.hive.v2.SentryAuthorizerFactory

org.apache.sentry.binding.hive.v2.SentryHiveAuthorizationTaskFactoryImplV2
          org.apache.sentry.binding.hive.v2.SentryHivePrivilegeObject

org.apache.sentry.binding.hive.v2.authorizer.DefaultSentryAccessController

org.apache.sentry.binding.hive.v2.authorizer.DefaultSentryValidator
          org.apache.sentry.binding.hive.v2.authorizer.SentryHiveAuthorizer

org.apache.sentry.binding.hive.v2.metastore.AuthorizingObjectStoreV2

org.apache.sentry.binding.hive.v2.metastore.MetastoreAuthzBindingV2

org.apache.sentry.binding.hive.v2.metastore.SentryMetastorePostEventListener
V2
          org.apache.sentry.binding.hive.v2.util.SentryAuthorizerUtil
          org.apache.sentry.binding.hive.v2.util.SimpleSemanticAnalyzer
          org.apache.sentry.binding.hive.v2.DummyHiveAuthenticationProvider
          org.apache.sentry.binding.hive.v2.HiveAuthzBindingHook
          org.apache.sentry.binding.hive.v2.MetastoreAuthzBinding
          org.apache.sentry.kafka.ConvertUtil
          org.apache.sentry.kafka.authorizer.SentryKafkaAuthorizer
          org.apache.sentry.kafka.binding.KafkaAuthBinding
          org.apache.sentry.kafka.binding.KafkaAuthBindingSingleton
          org.apache.sentry.kafka.conf.KafkaAuthConf
          org.apache.sentry.kafka.MockGroupMappingServiceProvider
          org.apache.sentry.core.model.kafka.Cluster
          org.apache.sentry.core.model.kafka.ConsumerGroup
          org.apache.sentry.core.model.kafka.Host
          org.apache.sentry.core.model.kafka.KafkaActionConstant
          org.apache.sentry.core.model.kafka.KafkaActionFactory
          org.apache.sentry.core.model.kafka.Topic
          org.apache.sentry.policy.common.KeyValue
          org.apache.sentry.policy.common.PolicyConstants
          org.apache.sentry.policy.kafka.KafkaModelAuthorizables
          org.apache.sentry.policy.kafka.KafkaPrivilegeValidator
          org.apache.sentry.policy.kafka.KafkaWildcardPrivilege
          org.apache.sentry.policy.kafka.SimpleKafkaPolicyEngine
          org.apache.sentry.policy.kafka.KafkaPolicyFileProviderBackend
          org.apache.sentry.policy.kafka.MockGroupMappingServiceProvider
          org.apache.sentry.provider.cache.SimplePrivilegeCache
          org.apache.sentry.provider.common.SentryGroupNotFoundException

org.apache.sentry.provider.db.generic.service.thrift.TListSentryPrivilegesBy
AuthRequest

org.apache.sentry.provider.db.generic.service.thrift.TListSentryPrivilegesBy
AuthResponse
```


last-updated: Apr 11, 2022

```
org.apache.sentry.provider.db.generic.service.thrift.TSentryPrivilegeMap

org.apache.sentry.provider.db.generic.service.thrift.SentryGenericPolicyProc
essorWrapper
          org.apache.sentry.provider.db.generic.tools.SentryShellKafka
          org.apache.sentry.provider.db.generic.tools.SentryShellSolr

org.apache.sentry.provider.db.generic.tools.command.AddRoleToGroupCmd
          org.apache.sentry.provider.db.generic.tools.command.CreateRoleCmd

org.apache.sentry.provider.db.generic.tools.command.DeleteRoleFromGroupCmd
          org.apache.sentry.provider.db.generic.tools.command.DropRoleCmd

org.apache.sentry.provider.db.generic.tools.command.GrantPrivilegeToRoleCmd

org.apache.sentry.provider.db.generic.tools.command.ListPrivilegesByRoleCmd
          org.apache.sentry.provider.db.generic.tools.command.ListRolesCmd

org.apache.sentry.provider.db.generic.tools.command.RevokePrivilegeFromRoleC
md
          org.apache.sentry.provider.db.log.entity.DBAuditMetadataLogEntity
          org.apache.sentry.provider.db.log.entity.GMAuditMetadataLogEntity
          org.apache.sentry.provider.db.service.thrift.ConfServlet
          org.apache.sentry.provider.db.tools.SentryShellHive
          org.apache.sentry.provider.db.tools.command.hive.CommandUtil
          org.apache.sentry.provider.db.tools.command.hive.CreateRoleCmd
          org.apache.sentry.provider.db.tools.command.hive.DropRoleCmd

org.apache.sentry.provider.db.tools.command.hive.GrantPrivilegeToRoleCmd

org.apache.sentry.provider.db.tools.command.hive.GrantRoleToGroupsCmd
          org.apache.sentry.provider.db.tools.command.hive.ListPrivilegesCmd
          org.apache.sentry.provider.db.tools.command.hive.ListRolesCmd

org.apache.sentry.provider.db.tools.command.hive.RevokePrivilegeFromRoleCmd

org.apache.sentry.provider.db.tools.command.hive.RevokeRoleFromGroupsCmd

org.apache.sentry.provider.db.generic.service.thrift.SentryGenericServiceInt
egrationBase
```

Added Interfaces

```
org.apache.sentry.core.model.kafka.KafkaAuthorizable extends Authorizable
          org.apache.sentry.provider.db.generic.tools.command.Command

org.apache.sentry.provider.db.generic.tools.command.TSentryPrivilegeConverto
r
          org.apache.sentry.provider.db.tools.command.hive.Command
```

Changed API

```
org.apache.hadoop.hive.SentryHiveConstants
          org.apache.hadoop.hive.ql.exec.SentryFilterDDLTask
          org.apache.hadoop.hive.ql.exec.SentryHivePrivilegeObjectDesc
          org.apache.sentry.binding.hive.HiveAuthzBindingHook

org.apache.sentry.binding.hive.SentryHiveAuthorizationTaskFactoryImpl
```

```
           org.apache.sentry.binding.hive.SentryIniPolicyFileFormatter
           org.apache.sentry.binding.hive.SentryOnFailureHookContext
           org.apache.sentry.binding.hive.SentryOnFailureHookContextImpl
           org.apache.sentry.binding.hive.SentryPolicyFileFormatFactory
           org.apache.sentry.binding.hive.SentryPolicyFileFormatter
           org.apache.sentry.binding.hive.authz.HiveAuthzBinding
           org.apache.sentry.binding.hive.authz.HiveAuthzPrivileges
           org.apache.sentry.binding.hive.authz.SentryConfigTool
           org.apache.sentry.binding.hive.conf.HiveAuthzConf
           org.apache.sentry.binding.hive.conf.InvalidConfigurationException
           org.apache.sentry.binding.metastore.MetastoreAuthzBinding
           org.apache.sentry.binding.metastore.SentryHiveMetaStoreClient
           org.apache.sentry.binding.metastore.SentryMetaStoreFilterHook
           org.apache.sentry.Command
           org.apache.sentry.core.common.Action
           org.apache.sentry.core.common.Authorizable
           org.apache.sentry.core.model.db.DBModelAuthorizable
           org.apache.sentry.core.model.indexer.IndexerModelAuthorizable
           org.apache.sentry.core.model.search.SearchConstants
           org.apache.sentry.core.model.search.SearchModelAuthorizable
           org.apache.sentry.core.model.sqoop.SqoopAuthorizable
           org.apache.sentry.policy.common.PolicyEngine
           org.apache.sentry.policy.common.Privilege
           org.apache.sentry.policy.common.PrivilegeValidator
           org.apache.sentry.provider.cache.PrivilegeCache
           org.apache.sentry.provider.cache.SimpleCacheProviderBackend
           org.apache.sentry.provider.common.AuthorizationProvider
           org.apache.sentry.provider.common.GroupMappingService

org.apache.sentry.provider.common.HadoopGroupResourceAuthorizationProvider
           org.apache.sentry.provider.common.KeyValue
           org.apache.sentry.provider.common.ProviderBackend
           org.apache.sentry.provider.common.ProviderConstants
           org.apache.sentry.provider.db.SentryPolicyStorePlugin
           org.apache.sentry.provider.db.SimpleDBProviderBackend
           org.apache.sentry.provider.db.generic.SentryGenericProviderBackend

org.apache.sentry.provider.db.generic.service.persistent.PrivilegeOperatePer
sistence

org.apache.sentry.provider.db.generic.service.persistent.SentryStoreLayer
           org.apache.sentry.provider.db.log.entity.AuditMetadataLogEntity
           org.apache.sentry.provider.db.log.entity.JsonLogEntity
           org.apache.sentry.provider.db.log.util.CommandUtil
           org.apache.sentry.provider.db.service.model.MSentryPrivilege
           org.apache.sentry.provider.file.PolicyFile
           org.apache.sentry.core.model.db.Column
           org.apache.sentry.hdfs.SentryHDFSServiceProcessor
           org.apache.sentry.provider.common.AuthorizationComponent
           org.apache.sentry.provider.common.NoAuthorizationProvider
           org.apache.sentry.provider.common.ResourceAuthorizationProvider

org.apache.sentry.provider.db.generic.service.persistent.DelegateSentryStore
           org.apache.sentry.provider.db.log.entity.JsonLogEntityFactory
           org.apache.sentry.provider.db.log.util.Constants
           org.apache.sentry.provider.db.service.persistent.SentryStore
           org.apache.sentry.provider.db.service.persistent.ServiceRegister
           org.apache.sentry.provider.db.tools.SentrySchemaTool
```

Removed Classes

```
org.apache.sentry.provider.common.ProviderConstants
```

Deprecated Classes

```
org.apache.sentry.provider.file.HadoopGroupResourceAuthorizationProvider
```

Deprecated Methods

```
org.apache.sentry.provider.db.service.thrift.SentryPolicyServiceClientDefaul
tImpl#grantServerPrivilege
```

Livy

Apache Livy is primarily used to provide integration between Hue and Spark.

Beginning with EEP 4.0.0, Livy is included as its own package in EEP repositories. Before EEP 4.0.0, Livy was included as `mapr-hue-livy` and released only as a part of Hue. For more information about Livy, see Apache Livy.

This documentation set covers the following topics for Livy:

- Installing Livy on page 204
- Configure Livy on page 3839
- Pre-Upgrade Steps for Livy on page 341
- Upgrading Livy on page 361
- Post-Upgrade Steps for Livy on page 378

Livy Limitations

This page describes some limitations of the MapR Data Platform implementation of Apache Livy.

Current limitations are as follows:

- The Livy programmatic Java/Scala/Python API is not supported.
- Livy artifacts are not published in the public Maven repository with other MapR Data Platform artifacts.

Configure Livy

This topic describes how to configure Livy.

For information about the required package names to configure the Livy server, see Livy on page 3839.

Configure Livy with Security

Starting from EEP 6.0.0, MapR SASL authentication, encryption, and impersonation for Livy are enabled by default on secure clusters.

The Livy user interface is available on port 8998. To start the user interface, open a browser, and navigate to the following address:

```
https://<hostname>:8998
```

Configure Livy on Kerberos

This topic describes how to configure Livy on Kerberos.

last-updated: Apr 11, 2022

To configure Livy on Kerberos, add the following properties to the `livy.conf` file:

```
livy.server.auth.type = kerberos
livy.server.auth.kerberos.principal = HTTP/_HOST@HADOOP.LOCALDOMAIN
livy.server.auth.kerberos.keytab = $KEYTAB
livy.server.launch.kerberos.principal = $USER/_HOST@HADOOP.LOCALDOMAIN
livy.server.launch.kerberos.keytab = $KEYTAB
```

For example:

```
livy.server.auth.type = kerberos
livy.server.auth.kerberos.principal = HTTP/node2.cluster@NODE1
livy.server.auth.kerberos.keytab = /opt/mapr/conf/mapr.keytab
livy.server.launch.kerberos.principal = mapr/node2.cluster@NODE1
livy.server.launch.kerberos.keytab = /opt/mapr/conf/mapr.keytab
```

Livy UI on Kerberos

Note: You can login to the Livy UI on a Kerberos setup only using a web browser configured with SPNEGO.

The other option is to configure multiauth authentication on Kerberized configurations to allow you to login to Livy UI not only with `SPNEGO/Mapr-Negotiation` mechanisms but also with PAM credentials.

Configure Livy with Custom SSL Encryption
This topic describes how to configure Livy with custom SSL encryption.

1. By default, on a secure cluster, Livy reads the `ssl-server.xml` file and configures SSL from this file.
2. If you want to use custom SSL configuration, add the following properties to the `livy.conf` file:

```
## Use this keystore for the SSL certificate and key.
 livy.keystore = <path-to-ssl_keystore>

# Specify the keystore password.
livy.keystore.password = <password>

# Specify the key password.
livy.key-password = <password>
```

Configure Livy with Spark Modes
This topic describes how to configure Livy with different Spark modes.

Use these steps to configure Livy:

1. Modify the `livy.conf` file (`/opt/mapr/livy/livy-<version>/conf/livy.conf`):
   a. If Spark jobs run in local mode, set the `livy.spark.master` property:

```
...
# What spark master Livy sessions should use.
livy.spark.master = local[*]
...
```

b. If Spark jobs run in YARN mode, set the `livy.spark.master` and `livy.spark.deployMode` properties (client or cluster). For example:

```
...
# What spark master Livy sessions should use.
livy.spark.master = yarn
# What spark deploy mode Livy sessions should use.
livy.spark.deployMode = client
...
```

or

```
...
# What spark master Livy sessions should use.
livy.spark.master = yarn
# What spark deploy mode Livy sessions should use.
livy.spark.deployMode = cluster
...
```

c. If Spark jobs run in Standalone mode, set the `livy.spark.master` and `livy.spark.deployMode` properties (client or cluster). For example:

```
…
# What spark master Livy sessions should use.
livy.spark.master = spark://node:7077
# What spark deploy mode Livy sessions should use.
livy.spark.deployMode = client
…
```

or

```
…
# What spark master Livy sessions should use.
livy.spark.master = spark://node:7077
# What spark deploy mode Livy sessions should use.
livy.spark.deployMode = cluster
…
```

d. If Spark jobs run in Mesos mode, set the `livy.spark.master` property. For example:

```
# What spark master Livy sessions should use.
livy.spark.master = mesos://<mesos-master-node-ip>:5050
```

2. To you want to use impersonation with Livy, set `livy.impersonation.enabled` to `true` in `livy.conf`. For example:

```
# If livy should impersonate the requesting users when creating a new
session.
livy.impersonation.enabled = true
```

3. If you want to be able to access Hive through Spark for Livy, you should configure Spark with Hive, and set `livy.repl.enableHiveContext` to `true` in `livy.conf`. For example:

```
...
# Whether to enable HiveContext in livy interpreter, if it is true
hive-site.xml will be detected
# on user request and then livy server classpath automatically.
livy.repl.enableHiveContext = true
...
```

Note: If Hive is installed on a cluster and if Spark is configured on Hive, this property is set to true by default: `livy.repl.enableHiveContext = true`.

4. To apply the needed changes, restart the Livy service:

```
maprcli node services -name livy -action restart -nodes <livy node>
```

MapR Event Store For Apache Kafka Clients and Tools

Describes the supported MapR Event Store For Apache Kafka tools and clients.

MapR Event Store For Apache Kafka Tools

The following Kafka tools are supported:

Table

MapR Event Store For Apache Kafka Tool	EEP Release	MapR version	Kafka version
Kafka Streams	6.0 EEP release	6.1	1.1
KSQL	6.0 EEP release	6.1	1.1
Kafka REST 4.1	6.0 EEP release	6.1	1.1
Kafka Connect 4.1	6.0 EEP release	6.1	1.1
Kafka Schema Registry 4.1.1	6.1 EEP release	6.1	1.1
Spark Streaming	6.0 EEP release	6.1	1.1

Starting in EEP 8.0.0 and Core 6.2, Kafka 2.6.1.0 supports the following tools and components:

- Kafka Streams API 1.1
- KSQL 6.0.0.0
- Kafka REST 6.0.0.0
- Kafka Connect 10.0.0.0
- Kafka Schema Registry 6.0.0.0
- Spark Streaming

For a complete list of supported versions in each EEP, see Component Versions for Released EEPs on page 5586.

The following points describe the Kafka tools and provide links to additional information:

- Kafka Streams on page 3854: This tool is a programming library used for creating Java or Scala streaming applications.

- KSQL on page 3843: This tool is an open source streaming SQL engine that implements continuous, interactive queries.
- Kafka Schema Registry: This tool provides a RESTful interface for storing and retrieving Avro schemas.
- Kafka REST Proxy on page 3865: This tool is used as a RESTful interface to MapR Event Store For Apache Kafka.
- Kafka Connect on page 3903: This tool is used to stream data between MapR Event Store For Apache Kafka and other storage systems.

MapR Event Store For Apache Kafka Clients

MapR Event Store For Apache Kafka client applications can be developed for MapR Event Store For Apache Kafka (as of MapR 5.2.1 with EEP 3.0). The MapR Event Store For Apache Kafka clients are based on distributions of librdkafka that works with MapR Event Store For Apache Kafka.

- MapR Event Store For Apache Kafka C Client - Used to develop MapR Event Store For Apache Kafka applications in C. See MapR Event Store For Apache Kafka C Applications on page 2795
- MapR Event Store For Apache Kafka Java Client - Used to develop MapR Event Store For Apache Kafka applications in Java. See MapR Event Store For Apache Kafka Java Applications on page 2754
- MapR Event Store For Apache Kafka Python Client - Used to develop MapR Event Store For Apache Kafka applications in Python. This client is available as of MapR 5.2.1 with EEP 3.0. See MapR Event Store For Apache Kafka Python Applications on page 2998

Table

MapR release	EEP Release	Kafka librdkafka version
As of MapR 6.0.1	As of 5.0	0.11.3

KSQL

KSQL is an open-source streaming SQL engine that implements continuous, interactive queries.

Use KSQL to query, read, write, and process data in real-time, at scale, through SQL commands. KSQL interacts directly with the Kafka Streams API, eliminating the need for a Java application.

The KSQL data flow architecture is designed such that the user interacts with the KSQL server and the KSQL server interacts with the MapR Event Store For Apache Kafka server.

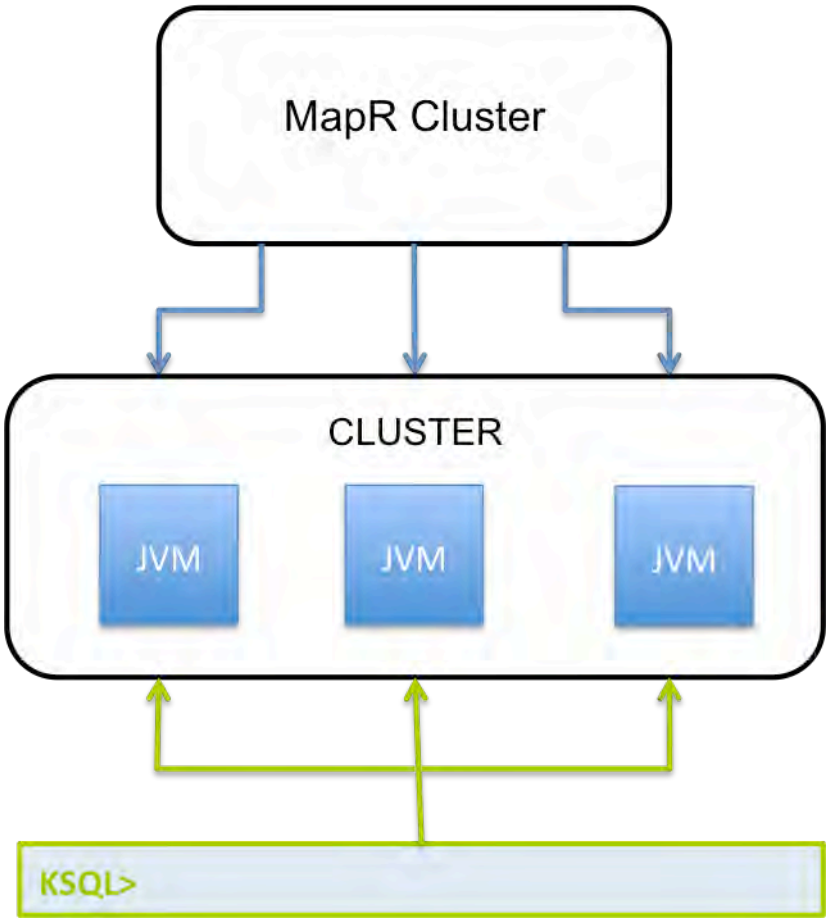
Use Cases

Common use cases include fraud detection, personalization, notifications, real-time analytics, and sensor data and IoT.

Architecture

A set of KSQL processes run as a cluster, and the KSQL server process completes queries. You can dynamically add more processing capacity by starting more instances of the KSQL server. These instances are fault-tolerant: if one fails, the others continue the work. Queries are launched using the interactive KSQL command line client, which sends commands to the cluster over a REST API. The command line allows you to inspect the available streams and tables, issue new queries, check the status of and terminate running queries.

last-updated: Apr 11, 2022



KSQL Server	The KSQL server runs the engine that completes KSQL queries. This includes processing, reading, and writing data to and from the target Kafka cluster. KSQL servers form KSQL clusters and can run in containers, virtual machines, and bare-metal machines. You can add and remove servers in a KSQL cluster during live operations to elastically scale processing capacity. You can deploy different KSQL clusters to achieve workload isolation.
KSQL CLI	You can interactively write KSQL queries through the KSQL command line interface (CLI). The KSQL CLI acts as a client to the KSQL server.

KSQL Deployment Modes

You can deploy KSQL queries through Interactive or Non-Interactive mode.

Interactive Mode	In Interactive mode, users interact with the KSQL server through a REST API, such as the KSQL CLI. Interactive mode is useful when users need to write and verify their queries interactively on a shared KSQL cluster. In interactive KSQL clusters, the authenticated KSQL user must have open access to create, read, write, delete topics, and use of any consumer group. Interactive KSQL clusters are not supported in a production environment. When you deploy queries to the production environment, lock-down access to KSQL servers by deploying a non-interactive (headless) environment.

last-updated: Apr 11, 2022

Non-Interactive Mode (Headless Mode)	Non-interactive mode supports locked-down or “headless” deployment scenarios where interactive use of the KSQL cluster is disabled, thereby preventing KSQL CLI access. In this mode, you can write queries to an SQL file, which allows for version control, and lock down access to KSQL servers to prevent users from interacting directly with the KSQL cluster. Non-interactive KSQL clusters (headless mode) is the main deployment model for KSQL queries on MapR (similar to the Apache Kafka model). Use non-interactive mode for production scenarios.

For More Information

- Apache Kafka KSQL
- Configuring Apache Kafka KSQL Server
- Apache Kafka Streams

KSQL Security
Discusses KSQL security topics.

KSQL COMMANDS

The KSQL COMMANDS internal topic is used to backup information about KSQL streams, KSQL tables, KSQL persistent queries, and so on. KSQL uses KSQL COMMANDS to restore the KSQL server state in case there is a fault or server restart.

Each KSQL Server cluster has a unique service ID which is provided through the `ksql.service.id` property. By default, the `kslq.service.id` is `_default`. To provide additional security, `ksql.service.id`-specific folders are created in the `ksql-internal-stream` stream.

Note: The `/apps` directory has only write access to `mapr` user. Therefore, the `/apps/ksql` directory cannot be modified or deleted by any user other than `mapr` user.

KSQL `ksql.service.id`-specific folders are created in the `/apps/ksql/` directory for every KSQL server cluster (represented by `ksql.service.id`).

Default Stream

KSQL Server provides a default stream for topics when they are being processed. When KSQL Server is not impersonated (non-interactive or interactive+no-impersonation), the KSQL Server default stream is used.

KSQL Cleanup

The KSQL cleanup feature is integrated to ensure that the underlying KSQL state (such as internal topics) are cleaned up correctly. See Application Reset Tool on page 3860 for more information.

Deployment

Note: A service ID (`ksql.service.id`) is uniquely created for the KSQL implementation; this means that the user associated with the `service ID` cannot grant permissions to other users to use the same service ID.

MapR KSQL deployment model is the same as Apache Kafka's deployment model. The KSQL Servers are not managed as part of the MapR cluster (for example, mapr-warden); you are required to run (or manage) your own KSQL Servers.

For More Information

- Apache Kafka KSQL
- Configuring Apache Kafka KSQL Server

KSQL Configuration
Set KSQL configuration and security parameters in the `ksql-server.properties` file. The default port for KSQL is 8084.

Configuring KSQL to Run in Non-Interactive (Headless) Mode

To run KSQL in non-interactive mode, set the parameters shown and then start KSQL.

Note: Note that the `ksql.default.stream` parameter is optional, but recommended. This parameter sets the default stream to consume from and send the messages to. The default stream is used if the topic name does not include the stream name. For example, if a message is sent to `exampleTopic` and this parameter is set to `/exampleStream`, then the message is sent to `/exampleStream:exampleTopic`.

Set the following properties in the `/opt/mapr/ksql/ksql-<version>/etc/ksql/ksql-server.properties` file:

```
ksql.command.topic.suffix=commands
ksql.service.id=app2
listeners=http://localhost:8084
ksql.default.stream=/sample-stream
```

Start KSQL:

```
$KSQL_INSTALL_DIR/bin/ksql-server-start
$KSQL_INSTALL_DIR/etc/ksql/ksql-server.properties --queries-file
some-queries-file.sql
```

Configuring KSQL to Run in Interactive Mode

The following example shows the configuration parameters that you must set in the `ksql-server.properties` file to run KSQL in interactive (distributed) mode:

```
ksql.command.topic.suffix=commands
            ksql.service.id=app2
            listeners=http://192.168.121.73:8084
            ksql.default.stream=/sample-stream
```

Note: You must set the `listeners` parameter to an actual IP address.

For more information

- For installation information, see Installing KSQL on page 195.
- For Apache Kafka information, see the Apache Kafka Streams API, Apache Kafka Producer Clients, and the Apache Kafka Consumer Clients.

KSQL Configuration Parameters

Set the KSQL configuration parameters in the `/opt/mapr/ksql/ksql-<version>/config/ksql-server.properties` file. For more information about configuration parameters, see KSQL Configuration Parameter Reference.

Note: The default KSQL configuration parameters are stored in `$KSQL_INSTALL_DIR/etc/ksql`. The default value for KSQL_INSTALL_DIR is `/opt/mapr/ksql/ksql-<version>/`.

The following table describes some KSQL configuration parameters:

Parameters	Description
`listeners`	Set this parameter to your localhost when using KSQL in non-interactive (headless) mode. Set this parameter to an actual IP address when using KSQL in interactive (distributed) mode. Default: 8084
`ksql.default.stream`	(Optional, but Recommended) The default stream to consume from and send the messages to. The default stream is used if the topic name does not include the stream name. For example, if a message is sent to `exampleTopic` and this parameter is set to `/exampleStream`, then the message will be sent to `/exampleStream:exampleTopic`.

KSQL Reference
KSQL-specific commands are commands for setting your KSQL configuration, exiting the CLI, and so on.

Run the KSQL with --help to see the available options.

```
./bin/ksql --help
```

CLI commands include:

- help
- clear
- output
- output <format>
- history
- version
- exit

KSQL is started by issuing the `./bin/ksql` command and the KSQL statements are run in the KSQL command line once it starts.

KSQL Statements

General Syntax

KSQL is started by issuing the `./bin/ksql` command and the KSQL statements are run in the KSQL command line once it starts.

KSQL statements must be terminated with a semicolon (;).

For multi-line statements:

last-updated: Apr 11, 2022

- In the CLI, you use a back-slash (\) to indicate continuation of a statement on the next line.
- Do not use backslashes (\) for multi-line statements in .sql files.

CREATE STREAM

CREATE STREAM WITH clause

Creates a new stream with the specified columns and properties.

```
CREATE STREAM stream_name ( { column_name data_type } [, ...] )
  WITH ( property_name = expression [, ...] );
```

CREATE STREAM WITH clause and AS SELECT

Creates a new stream with the specified columns and properties along with the corresponding MapR Event Store For Apache Kafka topic.

```
CREATE STREAM stream_name
        [WITH ( property_name = expression [, ...] )]
        AS SELECT  select_expr [, ...]
        FROM from_item [, ...]
        [ WHERE condition ]
        [PARTITION BY column_name]
```

CREATE TABLE

CREATE TABLE WITH clause

Creates a new KSQL table with the specified columns and properties.

```
CREATE TABLE table_name ( { column_name data_type } [, ...] )
        WITH ( property_name = expression [, ...] );
```

CREATE TABLE WITH clause and AS SELECT

Creates a new stream with the specified columns and properties along with the corresponding MapR Event Store For Apache Kafka topic and stream.

```
CREATE TABLE table_name
        [WITH ( property_name = expression [, ...] )]
        AS SELECT  select_expr [, ...]
        FROM from_item [, ...]
        [ WINDOW window_expression ]
  [ WHERE condition ]
  [ GROUP BY grouping_expression ]
  [ HAVING having_expression ];
```

DESCRIBE

DESCRIBE

Lists the columns in a stream or table along with their data type and other attributes.

```
DESCRIBE (stream_name|table_name);
```

DESCRIBE EXTENDED

Displays DESCRIBE information with additional runtime statistics, MapR Event Store For Apache Kafka topic details, and the set of queries that populate the table or stream.

```
DESCRIBE [EXTENDED] (stream_name|table_name);
```

EXPLAIN

EXPLAIN

Shows the execution plan for a SQL expression or, given the ID of a running query, shows the execution plan plus additional runtime information and metrics.

```
EXPLAIN (sql_expression|query_id);
```

DROP STREAM

DROP STREAM

Drops an existing stream

```
DROP STREAM stream_name;
```

DROP TABLE

DROP TABLE

Drops an existing table.

```
DROP TABLE table_name;
```

PRINT

PRINT

Prints topic contents to the KSQL CLI.

Note: SQL grammar defaults to uppercase formatting. To print topics containing lower-case characters, use quotations.

```
PRINT qualified_name (FROM BEGINNING)? ((INTERVAL | SAMPLE) number)?
```

Print Example

```
ksql> print '/sample-stream:streams-pipe-input' FROM BEGINNING;
Format:STRING
3/16/18 1:04:39 AM EET , 1 , record1
3/16/18 1:04:39 AM EET , 5 , record5
3/16/18 1:04:39 AM EET , 6 , record6
3/19/18 4:22:51 PM EET , null , Hello
3/19/18 4:23:05 PM EET , null , Hello2
```

SELECT

last-updated: Apr 11, 2022

SELECT

Selects rows from a KSQL stream or table. The result of this statement is not persisted in a topic and is only printed out in the console. To stop the continuous query in the CLI press Ctrl-C.

```
SELECT select_expr [, ...]
        FROM from_item [, ...]
        [ WINDOW window_expression ]
        [ WHERE condition ]
        [ GROUP BY grouping_expression ]
        [ HAVING having_expression ];
```

SELECT CAST expression type

Casts an expression's type to a new type.

```
CAST (expression AS data_type);
```

For example, to convert BIGINT to VARCHAR type:

```
SELECT page_id, CONCAT(CAST(COUNT(*) AS VARCHAR), '_HELLO')
  FROM pageviews_enriched
  WINDOW TUMBLING (SIZE 20 SECONDS)
  GROUP BY page_id;
```

SELECT LIKE operator

The LIKE operator is used for prefix or suffix matching. Currently KSQL supports %, which represents zero or more characters.

```
column_name LIKE pattern;
```

For example:

```
SELECT user_id
  FROM users
  WHERE user_id LIKE 'santa%';
```

SHOW TOPICS

Note: Information about active consumers and consumer groups is not available.

SHOW TOPICS

Prints topic information for all topics for the default stream (specified by ksql.default.stream). If the default stream is not specified, then an exception is thrown.

```
SHOW TOPICS;
```

SHOW TOPICS <stream_name>

Prints topic information for all topics from the specified stream. For example: /sample-stream

```
SHOW TOPICS '/sample-stream';
```

SHOW STREAMS

SHOW STREAMS

List the defined streams.

```
SHOW | LIST STREAMS;
```

SHOW TABLES

SHOW TABLES

List the defined TABLES.

```
SHOW | LIST TABLES;
```

SHOW QUERIES

SHOW QUERIES

List the running persistent queries.

```
SHOW | LIST QUERIES;
```

SHOW PROPERTIES

SHOW PROPERTIES

Lists the configuration setting that are currently in effect.

```
SHOW PROPERTIES;
```

TERMINATE

TERMINATE

Terminate a persistent query. Persistent queries run continuously until they are explicitly terminated

```
TERMINATE query_id;
```

Scalar Functions

The following are scalar functions for KSQL.

Table

Function	Example	Description
ABS	ABS(col1)	Absolute value of a value.
CEIL	CEIL(col1)	Ceiling of a value.
CONCAT	CONCAT(col1, '_hello')	Concatenate two strings.
EXTRACTJSONFILELD	EXTRACTJSONFIELD(message, '$.log.cloud')	Given a sring column in JSON format, extract the field that matches.
ARRAYCONTAINS	ARRAYCONTAINS('[1, 2, 3]', 3	Given a JSON or AVRO array, checks if a search value is contained in it.
FLOOR	FLOOR(col1)	Floor of a value.
LCASE	LCASE(col1)	Convert a string to lowercase.
LEN	LEN(col1)	Length of a string.

last-updated: Apr 11, 2022

Table (Continued)

Function	Example	Description
RANDOM	RANDOM()	Returns a random DOUBLE value between 0 and 1.0
ROUND	ROUND(col1	Round a value to the nearest BIGINT value.
STRINGTOTIMESTAMP	STRINGTOTIMESTAMP(col1, 'yyyy-MM-dd HH:mm:ss.SSS')	Converts a string value in the given format into the BIGINT value representing the timestamp.
SUBSTRING	SUBSTRING(col1, 2, 5)	Returns the substring with the start and end indices.
TIMESTAMPTOSTRING	TIMESTAMPTOSTRING(ROWTIME, 'yyyy-MM-dd HH:mm:ss.SSS')	Converts a BIGINT timestamp value into the string representation of the timestamp in the given format.
TRIM	TRIM(col1)	Trim the spaces from the beginning and the end of the string.
CASE	CASE(col1)	Convert a string to uppercase.

Aggregate Functions

The following are aggregate functions for KSQL.

Table

Function	Example	Description
COUNT	COUNT(col1)	Counts the number of rows.
MAX	MAX(col1)	Returns the maximum value for a given column and window.
MIN	MIN(col1)	Returns the minimum value for a given column and window.
SUM	SUM(col1)	Sums the column values.
TOPK	TOPK(col1, K)	Returns the TopK values for the given column and window.
TOPKDISTINCT	TOPKDISTINCT(col1, K)	Returns the distinct TopK values for the given column and window.

Pipe Code Sample
Provides sample code for a Pipe example.

The following is a Pipe code sample that moves an inputTopic to an outputTopic:

```
ksql> CREATE STREAM stream3 (message varchar) WITH (kafka_topic='/
sample-stream:inputTopic', value_format='DELIMITED');

ksql> CREATE STREAM stream4  WITH (kafka_topic='/
sample-stream:streams-pipe-output1', value_format='DELIMITED') AS SELECT *
FROM stream3;
```

KSQL Demo
The following demo example creates a stream, performs a non-persistent query, and a persistent query.

Setup

Complete the following steps to prepare your environment for querying:

1. Create a default stream using `/sample-stream`:

```
maprcli stream create -path /sample-stream
    -produceperm p -consumeperm p -topicperm p
```

2. Run the following script to generate test data that writes to an MapR Event Store For Apache Kafka topic:

```
./bin/ksql-datagen quickstart=pageviews format=delimited
topic=/sample-stream:pageviews maxInterval=10000
```

3. Run KSQL CLI and create a KSQL table:

```
> ./bin/ksql
ksql> CREATE TABLE pageviews_original_table
    (viewtime bigint, userid varchar, pageid varchar)
    WITH (kafka_topic='/sample-stream:pageviews',
value_format='DELIMITED', key='viewtime');
```

4. Run the SHOW TABLES command to list your KSQL tables:

```
ksql> SHOW TABLES;
```

Run a Non-persistent Query

For a non-persistent query, run:

```
ksql> SELECT * FROM pageviews_original_table;
```

For a non-persistent query in KSQL 6.0, run:

Run a Persistent Query

For a persistent query, do the following:

1. Create the topic, `/sample-stream:input-topic`:

```
maprcli stream topic create -path /sample-stream -topic input-topic
```

2. Create a KSQL input stream:

```
ksql> CREATE STREAM stream1 (message varchar) WITH
    (kafka_topic='/sample-stream:input-topic' ,
value_format='DELIMITED');
```

3. Create persistent query with filtering:

```
ksql> CREATE STREAM stream2
    WITH (kafka_topic='/sample-stream:output-topic' ,
value_format='DELIMITED')
    AS SELECT * FROM stream1 WHERE LEN(message) > 2;
```


last-updated: Apr 11, 2022

4. List your queries:

```
ksql> SHOW QUERIES;
```

5. Run the provided sample code for the console producer:

```
/opt/mapr/kafka/kafka-<version>/bin/kafka-console-producer.sh
    --broker-list fake.server.id:9092 --topic /sample-stream:input-topic
```

6. Run the provided sample code for the console consumer:

```
/opt/mapr/kafka/kafka-<version>/bin/kafka-console-consumer.sh
    --bootstrap-server fake.server.id:9092
    --topic /sample-stream:output-topic
```

7. Produce some data:

```
>Hi
>Hello
>No
>Yes
```

8. Get the next results:

```
Hello
Yes
```

Auxiliary Scripts Location

The sample code for `kafka-console-producer.sh` and `kafka-console-consumer.sh` is packaged with MapR Kafka. Once MapR Kafka is installed, you can find them at:

```
/opt/mapr/kafka/kafka-<version>/bin/
```

Kafka Streams

Kafka Streams is a programming library used for creating Java or Scala streaming applications and, specifically, building streaming applications that transform input topics into output topics.

Kafka Streams allows you to build moderately complex operational streaming applications faster by offloading common functions such as failure recovery, joins and enrichment, and aggregations and windowing.

Kafka Streams application is a distributed Java application that is launched with one or more Kafka Streams application instances. Kafka Streams applications can be built using the KStream library. A KStream application instance is required to be provided with an `application.id` property. The `application.id` property uniquely identifies the Kafka Streams distributed application.

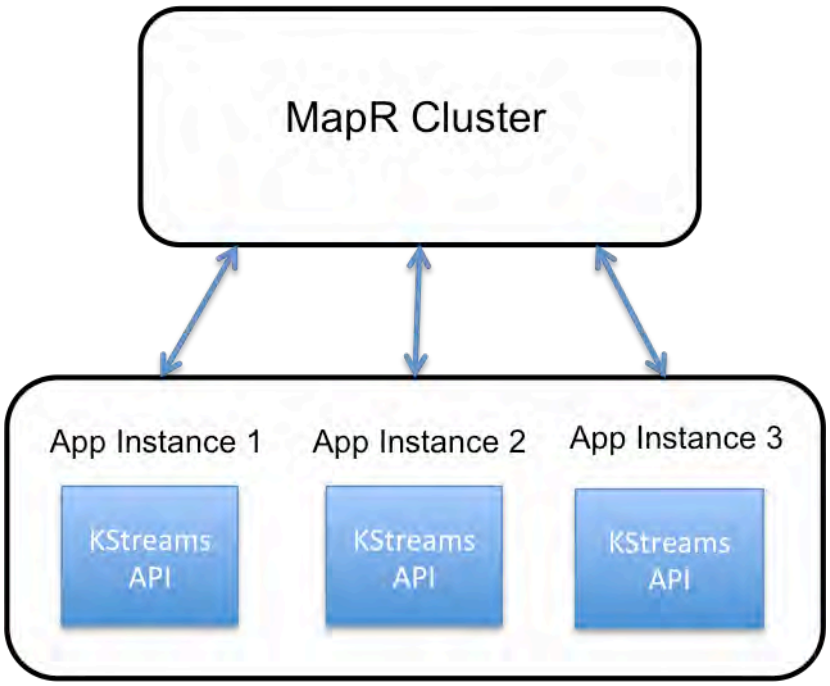
Attention: The Kafka Streams application must always be launched as the same user.

Architecture

An application that uses the Kafka Streams API is a normal Java application. Package, deploy, and monitor it like you would do for any other Java application. There is no need to install separate processing clusters or similar special-purpose and expensive infrastructure.

Note: You can run one or more instances of your application. They run independently but will automatically discover each other and collaborate. In addition, you can elastically add and remove application instances during live operations. If one instance dies, another instance continues where that instance left off.

The following diagram shows an application that is running three (3) application instances.



For More Information

Apache Kafka Streams

Kafka Streams Configuration
Describes how to configure Kafka Streams.

Kafka Streams Configuration

To configure Kafka Streams, set the following parameters in the Java API StreamsConfig instance:

- (Optional) Set the MapR Event Store For Apache Kafka `streams.default.stream` configuration parameter. See Apache Kafka Streams: Configuring a Streams Application for more information about all of the configuration parameters, required and optional.

  The default stream is used to consume from and send the messages to, if the topic name does not include the stream name. For example, if a message is sent to exampleTopic and this parameter is set to /exampleStream, then the message will be sent to /exampleStream:exampleTopic.

  Note: If the default stream option is not set and the topic name is specified without a stream name, an exception is thrown.

- Set the Apache Kafka Streams `application.id` configuration parameter. See Apache Kafka Streams: Configuring a Streams Application for more information about all of the configuration parameters, required and optional.

For more information

- For installation information, see Installing Kafka Streams on page 196.

last-updated: Apr 11, 2022

- For Apache Kafka Streams information, see Apache Kafka Streams: Configuring a Streams Application.

Supported Apache Kafka Streams APIs
Specifies the supported and not supported Apache Kafka Streams APIs.

Supported APIs

MapR Kafka Streams uses the same APIs as Apache Kafka Streams. Behavior for MapR Event Store For Apache Kafka is the same as for Apache Kafka Streams. See Apache Kafka Streams documentation .

Not Supported APIs

The following `stream` methods in the `StreamsBuilder` class are not supported:

- `<K,V> KStream<K,V> stream(java.util.regex.Pattern topicPattern)`

- ```
  <K,V> KStream<K,V> stream(java.util.regex.Pattern topicPattern,
  Consumed<K,V> consumed)
  ```

Use the following `stream` method instead:

- `<K,V> KStream<K,V>stream(java.util.Collection<java.lang.String> topics)`

Running a Kafka Streams Java App
Describes how to set up and run a Kafka Streams Java application.

Setup

To set up your project, add the required dependencies to the `pom.xml` file, as shown in the example. Note that the versions you use may differ from the versions shown in the example. Version numbers typically change with new releases.

Maven artifacts are published to https://repository.mapr.com/maven/. You can also refer to Maven Artifacts for MapR on page 4155 for dependency versions.

Example pom.xml
The following example shows the dependencies you must add to `pom.xml`.

```
<repository>
    <id>mapr-releases</id>
<url>https://repository.mapr.com/maven/</url>
</repository>
<dependency>
    <groupId>org.apache.kafka</groupId>
    <artifactId>kafka-clients</artifactId>
    <version>2.1.1.200-mapr-710</version>
</dependency>
<dependency>
    <groupId>org.apache.kafka</groupId>
    <artifactId>kafka-streams</artifactId>
    <version>2.1.1.200-mapr-710</version>
</dependency>
<dependency>
    <groupId>com.mapr.streams</groupId>
    <artifactId>mapr-streams</artifactId>
    <version>6.2.0.0-mapr</version>
</dependency>
```
```

```
<dependency>
     <groupId>org.rocksdb</groupId>
     <artifactId>rocksdbjni</artifactId>
     <version>5.7.3</version>
</dependency>
<dependency>
     <groupId>commons-logging</groupId>
     <artifactId>commons-logging</artifactId>
     <version>1.1.1</version>
</dependency>
```

Running a Kafka Streams App on a Cluster

To run a Kafka Streams Java application on a cluster:

1. Copy the <Kafka Streams Java application>.jar file to an arbitrary folder on your cluster.
2. Run the following shell command on your cluster:
   - For Kafka 2.1.1, run:

     ```
     java -cp "$(mapr clientclasspath):<Kafka Streams Java
     application>.jar" <Kafka Streams Application Main Class Name>
     ```

   - For Kafka 2.6.1, run:

     ```
     java cp "$(mapr clientclasspath):/opt/mapr/kafka/kafka<version>/libs/
     kafka-eventstreams<version>.jar:<Kafka Streams Java application>.jar"
     <Kafka Streams Application Main Class Name>
     ```

Pipe Code Sample

Provides sample code for a Pipe example.

The following is a Pipe code sample that moves records from an inputTopic to an outputTopic:

```
...
final StreamsBuilder builder = new StreamsBuilder();
KStream<String.String> source = builder stream(inputTopic);
source.to(outputTopic);
final Topology topology= builder.build();
final KafkaStreams streams=new KafkaStreams(topology, props);
...
streams.start();
...
```

Kafka Streams Demo

Provides a Kafka Streams demo example that creates a stream and topics and runs the WordCountDemo class code. The sample code produces and consumes messages.

1. Create the a stream named /sample-stream:

   ```
   maprcli stream create -path /sample-stream -produceperm p -consumeperm
   p -topicperm p
   ```

2. Create word-count-input and word-count-output topics:

```
maprcli stream topic create -path /sample-stream -topic word-count-input
maprcli stream topic create -path /sample-stream -topic word-count-output
```

3. Build the word count application and copy its JAR file to your cluster.

4. Run the WordCountDemo class.
   - For Kafka 2.1.1 and earlier, run:

```
java -cp "$(mapr clientclasspath):<Word Count Application Name>.jar"
WordCountDemo
```

5. Run the console producer:

```
/opt/mapr/kafka/kafka-<version>/bin/kafka-console-producer.sh
--broker-list fake.server.id:9092
--topic /sample-stream:word-count-input
```

6. Run the console consumer:

```
/opt/mapr/kafka/kafka-<version>/bin/kafka-console-consumer.sh
--bootstrap-server fake.server.id:9092
--topic /sample-stream:word-count-output
--property print.key=true
```

7. Produce some input with the console producer:

```
>word27 word28 word27 word29
```

8. Get the following output:

```
word28 1
word27 2
Word29 1
```

WordCountDemo Class Code

```
import org.apache.kafka.common.serialization.Serdes.StringSerde;
import org.apache.kafka.common.serialization.Serdes;
import org.apache.kafka.common.utils.Bytes;
import org.apache.kafka.streams.KafkaStreams;
import org.apache.kafka.streams.StreamsBuilder;
import org.apache.kafka.streams.StreamsConfig;
import org.apache.kafka.streams.Topology;
import org.apache.kafka.streams.kstream.*;
import org.apache.kafka.streams.state.KeyValueStore;

import java.util.Arrays;
import java.util.Locale;
import java.util.Properties;
import java.util.concurrent.CountDownLatch;

public class WordCountDemo {
```

```
    public static final String INPUT_TOPIC = "/
sample-stream:word-count-input";
    public static final String OUTPUT_TOPIC = "word-count-output"; //
Default stream will be used

    public static final String DEFAULT_STREAM = "/sample-stream";

    public static final String APP_ID = "app-id";

    public static void main(String[] args) {
        Properties props = new Properties();
        props.put(StreamsConfig.APPLICATION_ID_CONFIG, APP_ID);
        props.put(StreamsConfig.DEFAULT_KEY_SERDE_CLASS_CONFIG,
StringSerde.class);
        props.put(StreamsConfig.DEFAULT_VALUE_SERDE_CLASS_CONFIG,
StringSerde.class);


        props.put(StreamsConfig.COMMIT_INTERVAL_MS_CONFIG, 500); // Put
attention to this property
        props.put(StreamsConfig.STREAMS_DEFAULT_STREAM_CONFIG,
DEFAULT_STREAM);

        final StreamsBuilder builder = new StreamsBuilder();


        KStream<String, String> wordCountStream = builder.<String,
String>stream(INPUT_TOPIC)
                .flatMapValues(value ->
Arrays.asList(value.toLowerCase(Locale.getDefault()).split("\\W+")))
                .groupBy((key, value) -> value)
                .count(Materialized.<String, Long, KeyValueStore<Bytes,
byte[]>>as("counts-store"))
                .mapValues(x -> x.toString())

                .toStream();

        wordCountStream.to(OUTPUT_TOPIC, Produced.with(Serdes.String(),
Serdes.String()));

        final Topology topology = builder.build();
        final KafkaStreams streams = new KafkaStreams(topology, props);
        final CountDownLatch latch = new CountDownLatch(1);

        // attach shutdown handler to catch control-c
        Runtime.getRuntime().addShutdownHook(new
Thread("streams-shutdown-hook") {
            @Override
            public void run() {
                streams.close();
                latch.countDown();
            }
        });

        try {
            streams.start();
            latch.await();
        } catch (Throwable e) {
            e.printStackTrace();
            System.exit(1);
        }
        System.exit(0);
```


last-updated: Apr 11, 2022

```
    }
}
```

Note: The `kafka-console-producer.sh` and `kafka-console-consumer.sh` scripts are part of the **mapr-kafka** package.

Application Reset Tool
This tool allows your to reset an application and force it to reprocess its data from scratch by using the application reset tool. This tool can be useful for development and testing, or when fixing bugs.

Description

The application reset tool (ART) handles the Kafka Streams user topics (input, output, and intermediate topics) and internal topics differently when resetting the application.

The application reset tool does the following for each topic type:

- Input topics: Reset to the beginning of the topic. This means that it sets the application's committed consumer offsets for all partitions to each partition's `earliest` offset (for consumer group `application.id`).
- Intermediate topics: Skip to the end of the topic, i.e., set the application's committed consumer offsets for all partitions to each partition's `logSize` (for consumer group `application.id`).
- Internal topics: Delete the internal topic (this automatically deletes any committed offsets).

The application reset tool does not do the following:

- Reset output topics of an application. If any output (or intermediate) topics are consumed by downstream applications, it is your responsibility to adjust those downstream applications as appropriate when you reset the upstream application.
- Reset the local environment of your application instances. It is your responsibility to delete the local state on any machine on which an application instance was run.

See Confluent Application Reset Tool for additional reference information.

Running the Application Reset Tool

Invoke the application reset tool from the command line:

```
/opt/mapr/bin/kafka-streams-application-reset.sh
```

The tool accepts the following parameters:

Note: Parameters can be combined as needed. For example, if you want to restart an application from an empty internal state, but not reprocess previous data, simply omit the `--input-topics` and `--intermediate-topics` parameters.

Option	Description
--application-id <String: id>	(Required) The Kafka Streams application ID (application.id).
--default-stream	The default stream that is used when the topic name is specified but the stream name is not.
--config-file <String: file name>	Property file containing configs to be passed to admin clients and embedded consumer.
--dry-run	Display the actions that would be performed without executing the reset commands.

last-updated: Apr 11, 2022

Option	Description
--input-topics <String: list>	Comma-separated list of user input topics. For these topics, the tool will reset the offset to the earliest available offset.
--intermediate-topics <String: list>	Comma-separated list of intermediate user topics (topics used in the through() method). For these topics, the tool will skip to the end.

Resetting your Local Environments

To reset the local environments of your application instances, you must delete your application's local state directory on any machines where the application instance was run. You must do this before restarting an application instance on the same machine. You can use either of these methods:

Note: This a a complete application reset

The API method KafkaStreams#cleanUp() in your application code. Manually delete the corresponding local state directory (default location: /tmp/kafka-streams/<application.id>). For more information, see state.dir StreamsConfig class.

Example

In this example you are developing and testing an application locally and you want to iteratively improve your application via run-reset-modify cycles.

```
package mapr.examples.streams;

import ...;

public class ResetDemo {

  public static void main(String[] args) throws Exception {
  // Kafka Streams configuration
    Properties streamsConfiguration = new Properties();
    streamsConfiguration.put(StreamsConfig.APPLICATION_ID_CONFIG,
"my-streams-app");
    // ...and so on...

    // Define the processing topology
    StreamsBuilder builder = new StreamsBuilder();
    builder.stream("my-input-topic")
        .selectKey(...)
        .through("rekeyed-topic")
        .countByKey("global-count")
        .to("my-output-topic");

    KStreams app = new KafkaStreams(builder.build(), streamsConfiguration);

    // Delete the application's local state.
    // Note: In real application you'd call `cleanUp()` only under
    // certain conditions.  See tip on `cleanUp()` below.
    app.cleanUp();

    app.start();

    // Note: In real applications you would register a shutdown hook
    // that would trigger the call to `app.close()` rather than
    // using the sleep-then-close example we show here.
    Thread.sleep(30 * 1000L);
    app.close();
```


last-updated: Apr 11, 2022

```
  }

}
```

You can then perform run-reset-modify cycles as follows:

```
# Run your application
$ bin/kafka-run-class mapr.examples.streams.ResetDemo

# After stopping all application instances, reset the application
$ bin/kafka-streams-application-reset.sh --application-id my-streams-app \
                                         --input-topics my-input-topic \
                                         --intermediate-topics rekeyed-topic

# Now you can modify/recompile as needed and then re-run the application
again.
# You can also experiment, for example, with different input data without
# modifying the application.
```

Kafka Streams Security
Discusses Kafka Streams security topics.

Internal Topics

All Kafka Streams application's internal topics are grouped in the Kafka Streams application directory: **/apps/kafka-streams**.

- The **/apps** directory has only write access to **mapr user**. The **/apps/kafka-streams** directory is not modifiable/deletable by any user other than **mapr user**.
- All users can create sub-directories inside the **/apps/kafka-streams** directory. Only the following users have read/write/delete permission for sub-directories or files created in this directory.
  - **mapr user**
  - Current user of the sub-directory:
    - If security is enabled, the current user is the MapR ticket identity. See Managing Tickets on page 1424 for more information.
    - If security is ***not*** enabled, the current MapR identity.

Kafka Streams Application Specific Folders

Some Kafka Streams applications need to create internal topics. These topics are created in the **/apps/kafka-streams/<application.id>** directory.

Important: This directory is created at runtime by the Kafka Streams application and can only be modified by the current user or super users. This directory can only be deleted by the Application Reset Tool on page 3860 (ART) and, again, by only the current user or super users.

Application Reset Tool and Cleanup APIs

The application reset tool allows to reset a Kafka Streams application's internal state, such that it can re-process its input data from scratch. Kafka Streams internal topics can be cleaned using application reset tool.

Only the current user of the Kafka Streams application or **mapr user** has permissions to clean up a Kafka Streams application using Application Reset Tool. The Application Reset Tool is integrated with the cleanup APIs so that the application's internal topics are prefixed with the same directory.

The application reset tool takes `application.id` as the input for cleaning up Kafka Streams application. As part of this process, all internal-topics are deleted for the application user under the **/apps/kafka-streams/<application.id>** directory, including the **/apps/kafka-streams/<application.id>** directory. See Application Reset Tool on page 3860 for more information.

Changes in Kafka 2.6.1

Describes several differences to note when upgrading from Kafka 2.1.1 to 2.6.1.

Classpath change	• Kafka 2.6.1 uses classes from `kafka-eventstreams.jar` instead of `mapr-streams.jar` to access the cluster. • If an application fails with a `ClassNotFoundException` or `NoClassDefFoundError` for classes in packages under `com.mapr.kafka.eventstreams.*`, verify that `kafka-eventstreams.jar` is in the Java classpath. You can find `kafka-eventstreams.jar` in the `/opt/mapr/lib/` directory, or you can download it from the Maven repository.
Scala changes	• Scala version 2.11 is no longer supported. Scala versions 2.12 and 2.13 are supported. • Scala code leveraging the `NewTopic(String, int, short)` constructor with literal values must explicitly call `toShort` on the second literal.
RocksDBs change	• Kafka Streams version 2.6.1 requires RocksDB version 5.18.4.
Default consumer group id	• The default consumer group id has been changed from the empty string ("") to `null`. Consumers that use the new default group id will not be able to subscribe to topics and fetch or commit offsets. The empty string as consumer group id is deprecated but will be supported until a future major release. Old clients that rely on the empty string group id will now have to explicitly provide it as part of the consumer configuration. For more information, see KIP-289.
client.dns.lookup	The default value for the `client.dns.lookup` configuration has been changed from default to `use_all_dns_ips`. If a hostname resolves to multiple IP addresses, clients and brokers will now attempt to connect to each IP in sequence until the connection is successfully established. For more information, see KIP-602.
DSL	• Use the DSL operator, `cogroup()`, to aggregate multiple streams together at once. • Kafka Streams DSL switches its used store types. While this change is mainly transparent to users, there are some corner cases that may require code changes.
KStream.toTable() API	Use the KStream.toTable() API to translate an input event stream into a KTable.

last-updated: Apr 11, 2022

Serde type Void	Use the Serde type, Void, to represent null keys or null values from an input topic.
Sticky partitioning	The DefaultPartitioner now uses a sticky partitioning strategy. This means that records for a specific topic with null keys and no assigned partition will be sent to the same partition until the batch is ready to be sent. When a new batch is created, a new partition is chosen. This decreases latency to produce, but it may result in uneven distribution of records across partitions in edge cases. Generally, users will not be impacted, but this difference may be noticeable in tests and other situations producing records for a very short amount of time.
Rebalancing	• We are introducing incremental cooperative rebalancing to the clients' group protocol, which allows consumers to keep all of their assigned partitions during a rebalance and in the end revoke only those which must be migrated to another consumer for the overall cluster balance. The ConsumerCoordinator will choose the latest RebalanceProtocol that is commonly supported by all of the consumer's supported assignors. • We are introducing a new rebalancing protocol for Kafka Connect based on incremental cooperative rebalancing. The new protocol does not require stopping all the tasks during a rebalancing phase between Connect workers. Instead, only the tasks that need to be exchanged between workers are stopped and they are started in a follow-up rebalance. The new Connect protocol is enabled by default. For more details on how it works and how to enable the old behavior of eager rebalancing, checkout incremental cooperative rebalancing design.
Deprecated APIs	• Deprecated UsePreviousTimeOnInvalidTimestamp and replaced with UsePartitionTimeOnInvalidTimeStamp. • Provided support to query stale stores (for high availability) and the stores belonging to a specific partition by deprecating KafkaStreams.store(String, QueryableStoreType) and replacing it with KafkaStreams.store(StoreQueryParameters). • The internal PartitionAssignor interface has been deprecated and replaced with a new ConsumerPartitionAssignor in the public API. Some methods/signatures are slightly different between the two interfaces. Users implementing a custom PartitionAssignor should migrate to the new interface as soon as possible. • The blocking KafkaConsumer#committed methods have been extended to allow a list of partitions as input parameters rather than a single partition. It enables fewer request/response iterations between clients and brokers fetching for the committed offsets for the consumer group. The old overloaded functions are deprecated and we would recommend users making their code changes to leverage the new methods

last-updated: Apr 11, 2022

- The default consumer group id has been changed from the empty string ("") to `null`. Consumers who use the new default group id will not be able to subscribe to topics and fetch or commit offsets. The empty string as consumer group id is deprecated but will be supported until a future major release. Old clients that rely on the empty string group id will now have to explicitly provide it as part of their consumer configuration.

Kafka REST Proxy
The Kafka REST Proxy provides a RESTful interface to MapR Event Store For Apache Kafka clusters to consume and produce messages and to perform administrative operations.

It allows you to:

- Consume messages from topics or concrete topic partitions.
- Produce messages to topics or partitions.
- View the state of the cluster.

Use cases include ingesting messages into a stream-processing framework and scripting administrative operations.

Configuration
This section describes how to configure the Kafka REST Proxy for MapR Event Store For Apache Kafka.

You can set these configuration parameters in the `kafka-rest.properties` file. The Control System displays information about the Kafka REST Proxy for the MapR Event Store For Apache Kafka service. By default, the service runs on port **8082**.

To install the Kafka REST Proxy, see Installing MapR Event Store For Apache Kafka Tools on page 202.

To configure the Kafka REST Proxy for MapR Event Store For Apache Kafka, edit the following file:

```
/opt/mapr/kafka-rest/kafka-rest-<version>/config/kafka-rest.properties
```

To view the Kafka REST Proxy for MapR Event Store For Apache Kafka log files, see the following location:

```
/opt/mapr/kafka-rest/kafka-rest-<version>/logs/kafka-rest.log
```

Note: After installation, Warden automatically detects the configuration and starts the service. To configure the Kafka REST Proxy for MapR Event Store For Apache Kafka, stop the service, configure the parameters, and restart the service. To stop and restart services, see maprcli node services. For example:

```
maprcli node services -name kafka-rest -action stop
```

```
https://<host>:8443/rest/node/services?
name=kafka-rest&action=stop&nodes=<node_names>
```

where `node_names` is the node on which to perform the action; either a list of nodes, or a filter that matches a set of nodes .

Configuration Parameters
This section provides the Kafka REST Proxy for MapR Event Store For Apache Kafka parameters.

last-updated: Apr 11, 2022

These parameters are configurable in the `kafka-rest.properties` file.

```
/opt/mapr/kafka-rest/kafka-rest-<version>/config/kafka-rest.properties
```

Table

Parameter	Description
streams.default.stream	The default stream the consumer should poll messages from and the producer should send messages to. If the topic name does not specify the stream path, and the property has a valid value, then this topic name is found in the default stream.
id	Unique ID for this REST server instance. This is used in generating unique IDs for consumers that do not specify their ID. The ID is empty by default, which makes a single server setup easier to get up and running, but is not safe for multi-server deployments where automatic consumer IDs are used. Type: string. Default: empty
consumer.threads	The number of threads to run consumer requests on. Type: int. Default: 1
simpleconsumer.cache.max.records	Maximum number of records that can be stored in a single cache. Records with higher offsets replace records with lower ones. The value must be greater than 0. Type: int. Default: 1000.
simpleconsumer.max.caches.num	Maximum number topic-partition combinations for which records are cached. If this parameter is set to 0, then caching is disabled and extra records are thrown away. Cache improves performance if records are fetched sequentially thus increasing offsets. A pool of caches are available to store extra fetch records by a KafkaConsumer for a particular TopicPartition. The cache increases performance when records are fetched from a particular topic partition in a sequential manner. For example, every next request will start with the following offset after the offset of the latest fetched record in the previous request. Type: int. Default: 0
simpleconsumer.max.poll.time	Specifies the maximum number of milliseconds that are spent for polling records by a simpleconsumer. The greater the value means greater latency but higher throughput. Type: int. Default: 1000
simpleconsumer.pool.size.max	Maximum number of SimpleConsumers that can be instantiated. If 0, then the pool size is not limited. Type: int. Default: 25
simpleconsumer.pool.timeout.ms	Amount of time to wait for an available SimpleConsumer from the pool before failing. Use 0 for no timeout. Type: int. Default: 1000
consumer.instance.timeout.ms	Amount of idle time (in milliseconds) before a consumer instance is automatically destroyed. Type: int. Default: 300000 (5 minutes)
consumer.iterator.backoff.ms	Amount of time (in milliseconds) to backoff when an iterator runs out of data. If a consumer has a dedicated worker thread, this is effectively the maximum error for the entire request timeout. This parameter should be small enough to closely target the timeout, but large enough to avoid busy waiting. Type: int. Default: 50
consumer.request.max.bytes	Maximum number of bytes in unencoded message keys and values returned by a single request. This can be used by administrators to limit the memory used by a single consumer and to control the memory usage required to decode responses on clients that cannot perform a streaming decode. Note that the actual payload will be larger due to overhead from base64 encoding the response data and from JSON encoding the entire response. Type: long. Default: 6710884

Table (Continued)

Parameter	Description
consumer.request.timeout.ms	The maximum total time (in milliseconds) to wait for messages for a request if the maximum number of messages has not yet been reached. Type: int. Default: 1
producer.threads	Number of threads to run producer requests on. Type: int. Default: 5
request.logger.name	Name of the SLF4J logger to write the NCSA Common Log Format request log. Type: string. Default: io.confluent.rest-utils.requests.
response.mediatype.default	The default response media type that should be used if no specify types are requested in an Accept header. Type: string. Default: application/vnd.kafka.v1+json
response.mediatype.preferred	An ordered list of the server's preferred media types used for responses, from most preferred to least. Type: list. Default: application/vnd.kafka.v1+json, application/vnd.kafka+json, application/json
access.control.allow.methods	Sets the value to the Jetty Access-Control-Allow-Origin header for specified methods. Type: string. Default: empty
access.control.allow.origin	Sets the value for the Jetty Access-Control-Allow-Origin header. Type: string. Default: empty
host.name	The host name used to generate absolute URLs in responses. If empty, the default canonical hostname is used. Type: string. Default: empty
debug	Boolean indicating whether extra debugging information is generated in some error response entities. Type: Boolean. Default: false
shutdown.graceful.ms	Amount of time to wait after a shutdown request for outstanding requests to complete. Type: int. Default: 1000
metric.reporters	A list of classes to use as metrics reporters. Implementing the MetricReporterinterface allows plugging in classes that will be notified of new metric creation. The JmxReporter is always included to register JMX statistics. Type: list. Default: empty
metrics.jmx.prefix	Prefix to apply to metric names for the default JMX reporter. Type: string. Default: kafka.rest
metrics.num.samples	The number of samples maintained to compute metrics. Type: int. Default: 2
metrics.sample.window.ms	The metrics system maintains a configurable number of samples over a fixed window size. This configuration controls the size of the window. For example, used to maintain two samples each measured over a 30 second period. When a window expires, the oldest window is erased and overwritten. Type: long. Default: 30000

Security Parameters
Describes Kafka REST security parameters.

Configure security for Kafka REST through the security parameters in the `kafka-rest.properties` file.

```
/opt/mapr/kafka-rest/kafka-rest-<version>/config/kafka-rest.properties
```

Note: Ensure that both a `ssl_keystore` and a `ssl_truststore` file have been created.

last-updated: Apr 11, 2022

Table

Parameter	Description	Type	Default
listeners	Comma-separated list of listeners that listen for API requests over either HTTP or HTTPS. Each listener must include the protocol, hostname, and port. For example: http://localhost:8082	list	none
`rest.proxy.enable.doAs`	Specifies whether or not to enable impersonation for MapR Event Store For Apache Kafka topics. For this to take effect, PAM authentication must be enabled.	boolean	true
`authentication.method`	Specifies whether or not to enable PAM authentication. Set to NONE to disable.	string	BASIC
`authentication.realm`	Specifies realm for PAM authentication. Set to an empty string ("") to disable PAM. Set to jpamLogin to enable authentication	string	jpam
ssl.cipher.suites	A list of SSL cipher suites. This list is a comma-separated list. Leave blank to use Jetty's default.	list	none
ssl.cipher.suites.exclude	A list of disabled SSL cipher suites. This is a comma-separated list. Leave blank to use Jetty's default.	list	• TLS_DHE.* • TLS_EDH.* • .DES. • .MD5. • .RC4.
ssl.client.auth	Specifies whether or not to acquire the HTTPS client to authenticate via the server's trust store.	boolean	false
ssl.disabled.protocols	The list of SSL protocols that will not be accepted by clients. This is a comma-separated list.	list	• SSLv3 • TLSv1.0
ssl.enabled.protocols	The list of SSL protocols that can be accepted from clients. The list is a comma-separated list. Leave blank to use Jetty's defaults.	list	empty
ssl.endpoint.identification.algorithm	The endpoint identification algorithm to validate the server hostname using the server certificate. IMPORTANT: Jetty requires that the key's CN, stored in the keystore, must match the FQDN if ssl_endpoint_identification_algorithm=https. Leave blank to use Jetty's default.	string	none

last-updated: Apr 11, 2022

Table (Continued)

Parameter	Description	Type	Default
ssl.key.password	The password of the private key in the keystore file. This parameter should be taken from the /opt/mapr/conf/ssl-client.xml file. If this parameter is not set, the property value is obtained from the ssl-client.xml file. **Note:** If the ssl-client.xml file is changed, Kafka REST must be restarted.	string	empty
ssl.keymanager.algorithm	The algorithm used by the key manager factory for SSL connections. Leave blank to use Jetty's default.	string	empty
ssl.keystore.location	Location of the keystore file. This parameter should be taken from the /opt/mapr/conf/ssl-client.xml file. If this parameter is not set, the property value is obtained from the ssl-client.xml file. **Note:** If the ssl-client.xml file is changed, Kafka REST must be restarted.	string	empty
ssl.keystore.password	The store password for the keystore file. This parameter should be taken from the /opt/mapr/conf/ssl-client.xml file. If this parameter is not set, the property value is obtained from the ssl-client.xml file. **Note:** If the ssl-client.xml file is changed, Kafka REST must be restarted.	string	empty
ssl.keystore.type	The type of keystore file.	string	JKS
ssl.protocol	The SSL protocol used to generate the SslContextFactory.	string	TLS-v1.2-
ssl.provider	The SSL security provider name. Leave blank to use Jetty's default.	string	none
ssl.trustmanager.algorithm	The algorithm used by the trust manager factory for SSL connections. Leave blank to use Jetty's default.	string	none
ssl.truststore.location	Location of the trust store. Required only to authenticate HTTPS clients.	string	empty
ssl.truststore.password	The store password for the trust store file.	string	empty

last-updated: Apr 11, 2022

Table (Continued)

Parameter	Description	Type	Default
ssl.truststore.type	The type of trust store file.	string	JKS
ssl.trustallcerts.enable	Set to true if you want to disable certificates verification.	boolean	false

SSL Security Configuration
Describes how to configure Kafka REST security.

Secure by Default

As of MapR 6.0, the MapR Installer performs the Kafka REST configuration for new installations. This means that:

- If MapR core is installed as *secure*, then Kafka REST is also installed as *secure*.
- If MapR core is installed as *insecure*, then Kafka REST is also installed as *insecure*.

Manually Securing Kafka REST Only

CAUTION: This configuration is ***not*** a typical configuration.

If you have an *insecure* MapR cluster, and you want to *secure* Kafka REST, do the following:

1. Generate the server and client certificates.
2. Add any necessary property configurations to the `kafka-rest.properties` configuration file. For example:

```
listeners=http://0.0.0.0:8082,https://0.0.0.0:8085
            ssl.keystore.location=<ssl-keystore-path>
            ssl.keystore.password=<ssl-keystore-password>
            ssl.key.password=<ssl-keystore-password>
```

3. Restart Kafka REST.

```
maprcli node services -name kafka-rest -action restart -nodes <space
delimited list of nodes>
```

4. Run a curl command to ensure that HTTPS is enabled.

```
curl -X GET https://node1:8085/streams/%2Ftesting/topics --cacert
<certificate-path>
```

Manually Unsecuring Kafka REST

Warning: This scenario is ***NOT*** recommended or supported.

If you have an *secure* MapR cluster, and you want to *insecure* Kafka REST, do the following:

1. In the `kafka-rest.properties` configuration file, change **https://** to **http://** for the listeners and remove the **ssl.*** properties. For example:

```
listeners=http://0.0.0.0:8082
```


last-updated: Apr 11, 2022

2. Restart Kafka REST.

```
maprcli node services -name kafka-rest -action restart -nodes <space
delimited list of nodes>
```

User Impersonation
Describes how to disable, enable, and use impersonation with Kafka REST.

User impersonation enables Kafka REST jobs to be submitted as a particular user. Without impersonation, Kafka REST submits jobs as the user that started Kafka REST server.

On an MapR Data Platform cluster, the impersonated user is typically the `mapr` user or the user specified in the MAPR_USER environment variable. By default, impersonation and PAM authentication in Kafka REST are enabled on all types of security.

Disabling User Impersonation

To disable user impersonation, you need to first disable the PAM authentication properties in the kafka-rest.properties file and then disable the rest.proxy.enable.doAs property.

1. Disable PAM authentication. Set the following properties in `opt/mapr/kafka-rest/kafka-rest-<version>/config/kafka-rest.properties:`
   - authentication.method=NONE
   - authentication.realm=""
2. Once authentication is disabled, set the following property in `/opt/mapr/kafka-rest/kafka-rest-<version>/config/kafka-rest.properties:`
   - rest.proxy.enable.doAs=false

Example: Verify that a list of topics is owned by an impersonated user

This example demonstrates how to get a list of topics from a particular stream and then verifies that the list of topics is owned by a particular user. Depending on whether or not impersonation is enabled (the default), you may need to use a different `curl` command.

```
$  sudo maprcli stream info -json  -path /stream1
{
 "timestamp":1505471089855,
  "timeofday":"2017-09-15 10:24:49.855 GMT+0000",
  "status":"OK",
  "total":1,
  "data":[
       {
       "path":"/stream1",
       "physicalsize":245760,
       "logicalsize":114688,
       "numtopics":2,
       "defaultpartitions":1,
       "ttl":604800,
       "compression":"lz4",
       "autocreate":true,
       "produceperm":"u:root",
       "consumeperm":"u:root",
       "topicperm":"u:root",
       "copyperm":"u:root",
       "adminperm":"u:root",
       "ischangelog":false
       }
```


last-updated: Apr 11, 2022

```
    ]
 }
```

If impersonation is enabled (the default), use the following query, where the query is submitted as the root user.

```
$ curl -u root  -X GET http://localhost:8082/topics/%2Fstream%3Atopic1
Enter host password for user 'root':
{"name":"/stream:topic1","configs":null,"partitions":
  [{"partition":0,"leader":0,"replicas":
[{"broker":0,"leader":true,"in_sync":true},
   {"broker":0,"leader":false,"in_sync":true}]}]}
```

If impersonation is disabled, use the following query, where the query is submitted as the `mapr` user.

```
$ curl  -X GET http://localhost:8082/topics/%2Fstream%3Atopic1
{"error_code":40401,"message":"Topic not found."}
```

Saving Kafka REST Configurations
Describes how Kafka REST configurations are saved during an upgrade.

Starting in EEP 6.0.0, the configuration for a previously installed version of Kafka REST is stored in a folder with a timestamp.

- The configuration files are saved *and* overwritten by new configuration files when upgrading from:
  - 4.1.0 to 5.1.2
  - 5.1.2 to 6.0.0.0
- The configuration files are saved only (not overwritten) when upgrading from:
  - 5.1.2 to 5.1.2

Example

The following example shows the list of configuration files that are saved after upgrading from EEP 6.3.1 to EEP 6.3.2:

```
ls /opt/mapr/kafka-rest/
kafka-rest-4.1.0  kafka-rest-4.1.0.202009150334  kafka-restversion

ls /opt/mapr/kafka-rest/kafka-rest-4.1.0.202009150334/config/
kafka-rest.properties  log4j.properties  warden.kafka-rest.conf
```

Services Management
The Kafka REST Proxy for MapR Event Store For Apache Kafka service can be started, restarted, and stopped via the maprcli nodes services command or using the REST API equivalent.

The following `maprcli nodes services` commands summarize the commands. For more information, see node services on page 1730.

CLI commands

```
maprcli node services -name kafka-rest -action start -nodes <node_list>
```

```
maprcli node services -name kafka-rest -action stop -nodes <node_list>
```

```
maprcli node services -name kafka-rest -action restart -nodes <node_list>
```

REST

```
https://<host>:8443/rest/node/services?
name=kafka-rest&action=stop&nodes=<node_names>
```

where `node_names` is the node on which to perform the action; either a list of nodes, or a filter that matches a set of nodes .

HTTP Methods and URI Summary
This section provides HTTP method and URI summaries for multiple Kafka REST Proxy API versions for MapR Event Store For Apache Kafka.

API v2: Kafka REST Proxy Summary

Availability of Kafka REST Proxy API v2 started in Kafka REST 4.0.0 on Core 6.0.x.

The following table lists the HTTP methods, URIs (with links to examples), and descriptions:

HTTP Method	URI	Description
GET	/topics	Retrieves a list of topic names.
GET	/topics/{string: topic_name}	Retrieves metadata about a specific topic.
POST	/topics/{string: topic_name}	Produces messages to a topic.
GET	/topics/{string: topic_name}/partitions	Retrieves a list of partitions for the topic.
GET	/topics/{string: topic_name}/partitions/{string: partition_id}	Retrieves metadata about a specific partition within a topic.
POST	/topics/{string: topic_name}/partitions/{string: partition_id}	Produces messages into a partition of a topic.
POST	/consumers/{string: group_name}	Creates a new consumer instance in the consumer group.
DELETE	/consumers/{string: group_name}/instances/{string: instance_id}	Destroys the consumer instance.
POST	/consumers/{string: group_name}/instances/{string: consumer_instance_id}/offsets	Commits a list of offsets for the consumer. When the post body is empty, it commits all the records that have been fetched by the consumer instance.
GET	/consumers/{string: group_name)/instances/{string: instance_id}/offsets	Gets the last committed offsets for the given partitions (whether the commit happened by this process or another).
POST	/consumers/{string: group_name}/instances/{string: instance_id}/subscription	Subscribes to the given list of topics or a topic pattern to get dynamically assigned partitions. If a prior subscription exists, it would be replaced by the latest subscription.
GET	/consumers/{string: group_name}/instances/{string: instance_id}/subscription	Gets the current subscribed list of topics.
DELETE	/consumers/{string: group_name}/instances/{string: instance_id}/subscription	Unsubscribes from topics currently subscribed to.
POST	/consumers/{string: group_name}/instances/{string: instance_id}/assignments	Manually assigns a list of partitions to a consumer.
GET	/consumers/{string: group_name}/instances/{string: instance_id}/assignments	Retrieves the list of partitions manually assigned to this consumer.

last-updated: Apr 11, 2022

HTTP Method	URI	Description
POST	/consumers/{string: group_name}/instances/{string: instance_id}/positions	Overrides the fetch offsets that the consumer will use for the next set of records to fetch.
POST	/consumers/{string: group_name}/instances/{string: instance_id}/positions/beginning	Seek to the first offset for each of the given partitions.
POST	/consumers/{string: group_name}/instances/{string: instance_id}/positions/end	Seek to the last offset for each of the given partitions.
GET	GET /consumers/{string: group_name}/instances/{string: instance_id}/records	Fetches data for the topics or partitions specified using one of the subscribe/ assign APIs.
GET	/streams/{string: stream_name}/topics	Retrieves a list of topics in a given stream.

API v1: Kafka REST Proxy Summary

The following table lists the HTTP methods, URIs (with links to examples), and descriptions:

HTTP Method	URI	Description
GET	/topics	Retrieves a list of topic names.
GET	/topics/{topic: string}	Retrieves metadata about a specific topic.
POST	/topics/{topic: string}	Produces a message into a topic.
GET	/topics/{topic: string}/partitions	Retrieves a list of partitions for the topic.
GET	/topics/{topic: string}/partitions/{partition_id: string}	Retrieves metadata about specific partition in a topic.
POST	/topics/{topic: string}/partitions/{partition_id: string}	Produces messages to one partition of the topic.
GET	/topics/{topic: string}/partition/{partition_id: string}/ messages?offset={int}[&count={int}]	Consumes messages from one partition of the topic.
GET	/stream/{stream: string}/topics	Retrieves a list of topics in a given stream.
POST	/consumers/{group: string}	Creates a new consumer instance in the consumer group.
POST	/consumers/{group: string}/instances/{instance: string}/ offsets	Commits offsets for the consumer. Returns a list of the partitions with the committed offsets.
DELETE	/consumers/{group: string}/instances/{instance: string}	Destroys the consumer instance.
GET	/consumers/{group: string}/instances/{instance: string}/ topics/{topic: string}	Consumes messages from a topic.

API v2 HTTP Methods and URIs

GET /topics

Retrieves a list of topic names.

last-updated: Apr 11, 2022

Description

Depending on the configuration, the type of information retrieved has different behavior. See the `streams.default.stream` in Configuration Parameters on page 3865

Table

Parameters Defined	Response
streams.default.stream is defined	Returns a list of topic names in the default stream. Returns topic names that contain a stream path.
streams.default.stream is not defined	Returns {"error_code":80001,"message":"MapR Event Store For Apache Kafka does not currently support this API. Set the streams.default.stream parameter to return topics for the default stream"}

Syntax

```
http://<host>:8082/topics
```

Request Example

```
$ curl "Content-Type: application/vnd.kafka.v2+json" "http://localhost:8082/
topics"
```

Response Example

```
[
        "streaming_data/stream:testtopic1",
        "streaming_data/stream:testtopic2"
        ]
```

GET /topics/{string: topic_name}
Retrieves metadata about a specific topic.

Description

Depending on the configuration, the type of information retrieved has different behavior. See the `streams.default.stream` in Configuration Parameters on page 3865

Table

Parameters Defined	Response
streams.default.stream is defined	Gets metadata about a specific MapR Event Store For Apache Kafka topic. A fully qualified topic name can be passed or not. If the topic name is not fully qualified, the metadata is retrieved and appended to the default stream path. For example, topic1 is equivalent to default_stream:topic1
streams.default.stream is not defined	Gets metadata about a specific MapR Event Store For Apache Kafka topic. A fully qualified topic name is passed that contains the stream path.

Note: The full name for the MapR Event Store For Apache Kafka topic contains characters such as a forward slash (/) and a colon (:), therefore, it should be encoded. For example, `/streaming_data/stream:topic-1` is equivalent to `%2Fstreaming_data%2Fstream%3Atopic-1`.

last-updated: Apr 11, 2022

Table

Parameters	Description
topic_name (string)	Name of the topic to get metadata about.

Syntax

Syntax for a topic in a default stream where the default stream is configured:

```
http://<host>:8082/topics/<topic_string>
```

Syntax for a topic where the fully qualified topic name is specified:

```
http://<host>:8082/topics/%2F<streaming_data>%2F<stream>%3A<topic1>
```

Request Example

```
curl "http://localhost:8082/topics/test"
```

Response Example

```
{
  "name":"test",
  "configs":null,
  "partitions":
    [
      {
        "partition":0,
        "leader":0,
        "replicas":
          [
            {"broker":0,"leader":true, "in_sync":true},
            {"broker":0,"leader":false, "in_sync":true}
          ]
      }
    ]
}
```

POST /topics/{string: topic_name}

Produces messages to a topic.

Description

Depending on the configuration, the type of information retrieved has different behavior. See the `streams.default.stream` in Configuration Parameters on page 3865

Table

Parameters Defined	Response
streams.default.stream is defined	Produces messages into specific MapR Event Store For Apache Kafka topics. If the topic name does not contain a stream path, then the default stream path is used.
streams.default.stream is not defined	Produces messages into a MapR Event Store For Apache Kafka topic. The topic name should contain a stream path and be encoded.

last-updated: Apr 11, 2022

Note: If the topic does not exist, the following error results: [“error_code”:40401, “message”: ”Topic not found.”}. New topics are not created by the POST operation.

Table

Parameters	Description
topic_name (*string*)	Name of the topic to produce the messages to.

Syntax

```
http://<host>:8082/topics/<topic_string>
```

Request Example

This example produces a message using binary embedded data with the value, Kafka, to the topic, test.

```
curl -X POST -H "Content-Type: application/vnd.kafka.binary.v2" --data
'{"records":[{"value":"S2Fma2E="}]}' "http://localhost:8082/topics/test"
```

Response Example

```
{
   "offsets":
    [{
        "partition":0,
        "offset": 1,
        "error_code":null,
        "error":null
      }],
  "key_schema_id":null,
  "value_schema_id":null
}
```

GET /topics/{string: topic_name}/partitions
Retrieves a list of partitions for the topic.

Description

Depending on the configuration, the type of information retrieved has different behavior. See the `streams.default.stream` in Configuration Parameters on page 3865.

Table

Parameters Defined	Response
streams.default.stream is defined	Gets metadata about specific MapR Event Store For Apache Kafka partitions within a topic. The user could pass fully qualified topic name or not. If a fully qualified topic name is not used, metadata is retrieved and appended to the default stream path.
streams.default.stream is not defined	Gets metadata about specific MapR Event Store For Apache Kafka partitions within topic. The user could only pass fully qualified topic name that contains stream path.

Table

Parameters	Description
topic_name (*string*)	Name of the topic.

last-updated: Apr 11, 2022

Syntax

```
http://<host>:8082/topics/<topic_name>/partitions
```

Request Example

```
curl -X GET -H "Content-Type: application/vnd.kafka.v2"
"http://localhost:8082/topics/testtopic1/partitions"
```

Response Example

```
[
  {
    "partition":0,
    "leader":0,
    "replicas":
      [{
          "broker":0,
          "leader":true,
          "in_sync":true
        }]
  },
  {
    "partition":1,
    "leader":0,
    "replicas":
      [{
          "broker":0,
          "leader":true,
          "in_sync":true
         }]
  }
]
```

GET /topics/{string: topic_name}/partitions/{string: partition_id}
Retrieves metadata about a specific partition within a topic.

Description

Depending on the configuration, the type of information retrieved has different behavior. See the `streams.default.stream` in Configuration Parameters on page 3865.

Table

Parameters Defined	Response
streams.default.stream is defined	Gets metadata about a specific partition within a MapR Event Store For Apache Kafka topic. The user could pass fully qualified topic name or not. If a fully qualified topic name is not used, metadata is retrieved and appended to the default stream path.
streams.default.stream is not defined	Gets metadata about specific MapR Event Store For Apache Kafka partitions within a topic. The user could only pass fully qualified topic names that contains stream path.

Table

Parameters	Description
topic_name (*string*)	Name of the topic.
partition_id (*int*)	ID of the partition to inspect.

last-updated: Apr 11, 2022

Syntax

```
http://<host>:8082/topics/<topic_name>/partitions/<partition_id>
```

Request Example

```
curl -X GET -H "Content-Type: application/vnd.kafka.v2"
http://localhost:8082/topics/%2Fstreaming_data%2Fstream%3Atesttopic1/
partitions/0
```

Response Example

```
{
  "partition":0,
  "leader":0,
  "replicas":
    [{
      "broker":0,
      "leader":true,
      "in_sync":true
      }]
}
```

GET /topics/{string: topic_name}/partitions/{string: partition_id}/offsets
Returns a summary with beginning and end offsets for the given topic and specific partition.

Description

Information retrieved varies depending on the configuration. See the `streams.default.stream` in Configuration Parameters on page 3865.

Table

Parameters Defined	Response
streams.default.stream is defined	Gets summary with beginning and end offsets for the specific partition of the MapR Event Store topic. You can pass a fully qualified topic name or not. If a fully qualified topic name is not used, metadata is retrieved and appended to the default stream path.
streams.default.stream is not defined	Gets summary with beginning and end offsets for the specific MapR Event Store partition within a topic. You can only pass the fully qualified topic names that contain the stream path.

Table

Parameter	Description
topic_name (*string*)	Name of the topic.
partition_id (*int*)	ID of the partition to inspect.

Syntax

```
http://<host>:8082/topics/<topic_name>/partitions/<partition_id>/offsets
```

Request Example

```
curl -X GET -H "Content-Type:
application/vnd.kafka.v2" http://localhost:8082/topics/
%2Fstreaming_data%2Fstream%3Atesttopic1/partitions/0/offsets
```

Response Example

```
{
  "beginning_offset":0,
  "end_offset":0
}
```

POST /topics/{string: topic_name}/partitions/{string: partition_id}
Produces messages into a partition of a topic.

Description

Depending on the configuration, the type of information retrieved has different behavior. See the `streams.default.stream` in Configuration Parameters on page 3865.

Table

Parameters Defined	Response
streams.default.stream is defined	Produces messages into a partition of MapR Event Store For Apache Kafka topic. The user could pass fully qualified topic name or not. If a fully qualified topic name is not used, messages are produced into topics in the default stream path.
streams.default.stream is not defined	Produces messages into a partition within MapR Event Store For Apache Kafka topic. The user could only pass fully qualified topic name that contains stream path.

Table

Parameters	Description
topic_name (*string*)	Topic to produce the messages to.
partition_id (*int*)	Partition to produce the messages to.

Syntax

```
http://<host>:8082/topics/<topic_name>/partitions/<partition_id>
```

Request Example

```
curl -X POST -H "Content-Type: application/vnd.kafka.binary.v2+json" --data
'{"records":[{"key":"a2v5","value":"Y29uZmx1ZW50"}]}'
"http://localhost:8082/topics/testtopic1/partitions/0"
```

Response Example

```
{
  "offsets":
    [{
       "partition":0,
       "offset":1,
       "error_code":null,"error":null}
```

```
    ],
  "key_schema_id":null,
  "value_schema_id":null
}
```

POST /consumers/{string: group_name}

Creates a new consumer instance in the consumer group.

Description

Table

Parameters	Description
group_name (*string*)	The name of the consumer group to join.
name (*string*)	Name for the consumer instance, which will be used in URLs for the consumer. This must be unique, at least within the proxy process handling the request. If omitted, falls back on the automatically generated ID. Using automatically generated names is recommended for most use cases.
format (*string*)	The format of consumed messages, which is used to convert messages into a JSON-compatible form. Valid values: “binary”, “avro”, “json”. If unspecified, defaults to “binary”.
auto.offset.reset (*string*)	Sets the **auto.offset.reset** setting for the consumer. Values: latest, earliest, none
auto.commit.enable (*string*)	Sets the **auto.commit.enable** setting for the consumer.

Note: You cannot set the time-to-live (TTL) for consumer instances or consumer groups. However, consumers can be configured to be deleted after some idle time. The amount of idle time before a consumer instance is automatically destroyed is set by the `consumer.instance.timeout.ms` property in the **kafka-rest.properties** file. See Configuration Parameters on page 3865.

Syntax

```
http://<host>:8082/consumers/<group_name>
```

Request Example

```
curl -X POST -H "Content-Type: application/vnd.kafka.v2+json"
 --data '{"name":"user","format": "binary", "auto.offset.reset":
"earliest"}'
 http://localhost:8082/consumers/grouptest
```

Response Example

The response JSON object is in the following form:

- **instance_id** (*string*) – Unique ID for the consumer instance in this group. The `instance_id` is automatically generated if the `name` parameter is not specified.
- **base_uri** (*string*) – Base URI used to construct URIs for subsequent requests against this consumer instance. This will be of the form `http://hostname:port/consumers/consumer_group/instances/instance_id`.

```
{
        "instance_id":"user",
```


last-updated: Apr 11, 2022

```
        "base_uri":"http://localhost:8082/consumers/grouptest/instances/
user"
}
```

DELETE /consumers/{string: group_name}/instances/{string: instance_id}
Destroys the consumer instance.

Description

The request must be made to the specific REST proxy instance holding the consumer instance.

Table

Parameters	Description
group_name (*string*)	The name of the consumer group.
instance (*string*)	The ID of the consumer instance

Syntax

```
http://<host>:8082/topics/<group_name>/instances/<instance_string>
```

Request Example

```
curl -X DELETE -H "Content-Type: application/vnd.kafka.v2+json"
http://localhost:8082/consumers/my_binary_consumer/instances/
rest-consumer-11561681-
8ba5-4b46-bed0-905ae1769bc6
```

Response Example

```
HTTP/1.1 204 No Content
```

POST /consumers/{string: group_name}/instances/{string: consumer_instance_id}/offsets
Commits a list of offsets for the consumer. When the post body is empty, it commits all the records that have been fetched by the consumer instance.

Parameters

Table

Parameters	Description
group_name (*string*)	The name of the consumer group.
instance_id (*string*)	The ID of the consumer instance.
offsets	A list of offsets to commit for partitions.
offsets[i].topic (*string*)	Name of the topic
offsets[i].partition (*int*)	Partition ID
offset	The offset to commit.

Syntax

```
http://<host>:8082/consumers/<group_name>/instances/<consumer_instance_id>/
offsets
```

Request Example

```
curl -X POST -H "Content-Type: application/vnd.kafka.v2+json" --data
'{"offsets": [{"topic":
"/mystream:first","partition": 0,"offset":5}]}'
https://node2:8082/consumers/grouptest/instances/user/offsets
```

Response Example

```
HTTP/1.1 200 OK
```

GET /consumers/{string: group_name)/instances/{string: instance_id}/offsets
Gets the last committed offsets for the given partitions (whether the commit happened by this process or another).

Parameters

Table

Parameter	Description
group_name (*string*)	Name of the consumer group.
instance (*string*)	ID of the consumer instance.
partitions	A list of partitions to find the last committed offsets.
partitions[i].topic (*string*)	Name of the topic.
partitions[i].partition (*int*)	Partition ID

Syntax

```
http://localhost:8082/consumers/<group_name>/instances/<consumer_name>/
offsets
```

Request Example

```
curl -X GET -H "Content-Type: application/vnd.kafka.binary.v2+json" --data
'{"partitions":[{"topic":"/stream:topic","partition":0}]}'
https://node2:8082/consumers/grouptest/instances/user/offsets
```

Response Example

The response JSON object is in the following form:

- offsets - A list of committed offsets.
- offsets[i].topic (string) – Name of the topic for which an offset was committed
- offsets[i].partition (int) – Partition ID for which an offset was committed
- offsets[i].offset (int) – Committed offset
- offsets[i].metadata (string) – Metadata for the committed offset

```
{"offsets":
  [
    {
```

```
        "topic": "/stream:topic",
        "partition": 0,
        "offset": 21,
        "metadata":""
    }
  ]
}
```

POST /consumers/{string: group_name}/instances/{string: instance_id}/subscription
Subscribes to the given list of topics or a topic pattern to get dynamically assigned partitions. If a prior subscription exists, it would be replaced by the latest subscription.

Parameters

Table

Parameter	Description
group_name (*string*)	Name of the consumer group.
instance (*string*)	ID of the consumer instance.

Syntax

```
http://<host>:8082/consumers/<group_name>/instances/<consumer_name>/
subscription
```

Request Example

```
curl -X POST -H "Content-Type: application/vnd.kafka.v2+json" --data
'{"topics":["/stream:first","/stream:second"]}'
https://localhost:8082/consumers/grouptest/instances/user/subscription
```

Response Example

```
HTTP/1.1 204 No Content
```

GET /consumers/{string: group_name}/instances/{string: instance_id}/subscription
Gets the current subscribed list of topics.

Parameters

Table

Parameter	Description
group_name (*string*)	Name of the consumer group.
instance (*string*)	ID of the consumer instance.

Syntax

```
http://<host>:8082/consumers/<group_name>/instances/<consumer_name>/
subscription
```

Request Example

```
curl -X GET -H "Content-Type: application/vnd.kafka.v2+json"
https://localhost:8082/consumers/grouptest/instances/user/subscription
```

Response Example

The response JSON object is in the following form:

- topics – A list of subscribed topics
- topics[i] (string) – Name of the topic

```
{
  "topics":[
            "/stream:first",
            "/stream:second"
           ]
}
```

DELETE /consumers/{string: group_name}/instances/{string: instance_id}/subscription
Unsubscribes from topics currently subscribed to.

Parameters

Table

Parameter	Description
group_name (*string*)	Name of the consumer group.
instance (*string*)	ID of the consumer instance.

Syntax

```
http://<host>:8082/consumers/<group_name>/instances/<consumer_name>/
subscription
```

Request Example

```
curl -X DELETE -H "Content-Type: application/vnd.kafka.v2+json"
https:/localhost:8082/consumers/grouptest/instances/user/subscription
```

Response Example

```
HTTP/1.1 204 No Content
```

POST /consumers/{string: group_name}/instances/{string: instance_id}/assignments
Manually assigns a list of partitions to a consumer.

Parameters

Table

Parameter	Description
group_name (*string*)	Name of the consumer group.
instance (*string*)	ID of the consumer instance.

last-updated: Apr 11, 2022

Syntax

```
http://<host>:8082/consumers/<group_name>/instances/<consumer_name>/
assignments
```

Request Example

```
curl -X POST -H "Content-Type: application/vnd.kafka.v2+json" --data
'{"partitions":[{"topic":"first","partition":0}]}'
https://localhost:8082/consumers/grouptest/instances/user/assignments
```

Response Example

```
HTTP/1.1 204 No Content
```

GET /consumers/{string: group_name}/instances/{string: instance_id}/assignments
Retrieves the list of partitions currently assigned to this consumer.

Parameters

Table

Parameter	Description
group_name (*string*)	Name of the consumer group.
instance (*string*)	ID of the consumer instance.

Syntax

```
http://<host>:8082/consumers/<group_name>/instances/<consumer_name>/
assignments
```

Request Example

```
curl -X GET -H "Content-Type: application/vnd.kafka.v2+json"
https://localhost:8082/consumers/grouptest/instances/user/assignments
```

Response Example

The response JSON object is in the following form:

- partitions – A list of partitions assigned to this consumer.
- partitions[i].topic (*string*) – Name of the topic.
- partitions[i].partition (*int*) – Partition ID

```
{
  "partitions": [
    {
      "topic": "test",
      "partition": 0
    },
    {
      "topic": "test",
      "partition": 1
    }
```

```
  ]
}
```

POST /consumers/{string: group_name}/instances/{string: instance_id}/positions
Overrides the fetch offsets that the consumer will use for the next set of records to fetch.

Parameters

Table

Parameter	Description
group_name (*string*)	Name of the consumer group.
instance (*string*)	ID of the consumer instance.
offsets	A list of offsets
offsets[i].topic (*string*)	Name of the topic
offsets[i].partition (*int*)	Partition ID
offsets[i].offset (*int*)	Seek to offset for the next set of records to fetch.

Syntax

```
http://<host>:8082/consumers/<group_name>/instances/<consumer_name>/
positions
```

Request Example

```
curl  -X POST -H "Content-Type: application/vnd.kafka.v2+json" --data
'{"offsets": [{"topic": "/stream:first","partition": 0,"offset":3}]}'
https://localhost:8082/consumers/grouptest/instances/user/positions
```

Response Example

```
HTTP/1.1 204 No Content
```

POST /consumers/{string: group_name}/instances/{string: instance_id}/positions/beginning
Seek to the first offset for each of the given partitions.

Parameters

Table

Parameter	Description
group_name (*string*)	Name of the consumer group.
instance (*string*)	ID of the consumer instance.
partitions	A list of partitions.
partitions[i].topic (*string*)	Name of the topic
partitions[i].partition (*int*)	Partition ID

Syntax

```
http://<host>:8082/consumers/<group_name>/instances/<consumer_name>/
positions/beginning
```

Request Example

```
curl  -X POST -H "Content-Type: application/vnd.kafka.v2+json" --data
'{"partitions": [{"topic": "/stream:first","partition": 0}]}'
https://localhost:8082/consumers/grouptest/instances/user/positions/
beginning
```

Response Example

```
HTTP/1.1 204 No Content
```

POST /consumers/{string: group_name}/instances/{string: instance_id}/positions/end
Seek to the last offset for each of the given partitions.

Parameters

Table

Parameter	Description
group_name (*string*)	Name of the consumer group.
instance (*string*)	ID of the consumer instance.
partitions	A list of partitions.
partitions[i].topic (*string*)	Name of the topic
partitions[i].partition (*int*)	Partition ID

Syntax

```
http://<host>:8082/consumers/<group_name>/instances/<consumer_name>/
positions/end
```

Request Example

```
curl  -X POST -H "Content-Type: application/vnd.kafka.v2+json" --data
'{"partitions": [{"topic": "/stream:first","partition": 0}]}'
https://localhost:8082/consumers/grouptest/instances/user/positions/end
```

Response Example

```
HTTP/1.1 204 No Content
```

GET /consumers/{string: group_name}/instances/{string: instance_id}/records
Fetches data for the topics or partitions specified using one of the subscribe/assign APIs.

Parameters

The format of the embedded data returned by this request is determined by the format specified in the initial consumer instance creation request and must match the format of the Accept header.

Note: This request *must* be made to the specific REST proxy instance holding the consumer instance.

Table

Parameter	Description
group_name (*string*)	Name of the consumer group.
instance (*string*)	ID of the consumer instance.
timeout	The number of milliseconds for the underlying client library poll(timeout) request to fetch the records. Default: 5000ms.
max_bytes	The maximum number of bytes of unencoded keys and values that should be included in the response. This provides approximate control over the size of responses and the amount of memory required to store the decoded response. The actual limit is the minimum of this setting and the server-side configuration consumer.request.max.bytes. Default: unlimited

Syntax

```
http://<host>:8082/consumers/<group_name>/instances/<consumer_name>/records
```

Request Example

```
curl -X GET -H "Content-Type: application/vnd.kafka.v2+json"
https://localhost:8082/consumers/grouptest/instances/user/records
```

Response Example

```
[
  {
    "topic": "test",
    "key": "a2V5",
    "value": "Y29uZmx1ZW50",
    "partition": 1,
    "offset": 100,
  },
  {
    "topic": "test",
    "key": "a2V5",
    "value": "a2Fma2E=",
    "partition": 2,
    "offset": 101,
  }
]
```

GET /streams/{string: stream_name}/topics

Retrieves a list of topics in a given stream.

Description

Stream names contain characters such as backslashs (/) and colons (:) and, therefore, should be encoded.

Table

Parameters	Description
stream_name *(string)*	The name of the stream.

last-updated: Apr 11, 2022

Syntax

```
http://<host>:8082/streams/<stream_name>/topics
```

Request Example

```
curl -X GET -H "Content-Type: application/vnd.kafka.v2+json"
http://localhost:8082/streams/%2Fstreaming_data%2Fstream/topics
```

Response Example

```
[
  "/streaming_data/stream:testtopic1",
  "/streaming_data/stream:testtopic2"
]
```

API v3 HTTP Methods and URIs

GET /v3/clusters
Retrieves a list of metadata about the cluster.

Description

Retrieves one cluster only. Always retrieves the current MapR Data Platform cluster.

Syntax

```
http://<host>:8082/v3/clusters
```

Request Example

```
curl -X GET -H "Content-Type: application/json" http://localhost:8082/v3/
clusters
```

Response Example

```
{
  "kind":"KafkaClusterList",
  "metadata":
    {
      "self":"http://node1.cluster.com:8082/v3/clusters",
      "next":null
    },
  "data":
    [
      {
        "kind":"KafkaCluster",
        "metadata":
          {
            "self":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619",
            "resource_name":"crn:///kafka=682798077049224619"
          },
        "cluster_id":"682798077049224619",
        "controller":
          {
            "related":"http://node1.cluster.com:8082/v3/clusters/
```

```
682798077049224619/brokers/0"
          },
        "acls":
          {
            "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/acls"
          },
        "brokers":
          {
            "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/brokers"
          },
        "broker_configs":
          {
            "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/broker-configs"
          },
        "consumer_groups":
          {
            "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/consumer-groups"
          },
        "topics":
          {
            "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics"
          },
        "partition_reassignments":
          {
            "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/-/partitions/-/reassignment"
          }
      }
    ]
}
```

GET /v3/clusters/{string: cluster_id}

Retrieves metadata about a specific cluster.

Parameters

Parameters	Description
cluster_id (*string*)	Cluster's id.

Syntax

```
http://<host>:8082/v3/clusters/<cluster_id>
```

Request Example

```
curl -X GET -H "Content-Type: application/json" http://localhost:8082/v3/
clusters/682798077049224619
```

Response Example

```
{
  "kind":"KafkaCluster",
  "metadata":
```


last-updated: Apr 11, 2022

```
    {
      "self":"http://node1.cluster.com:8082/v3/clusters/682798077049224619",
      "resource_name":"crn:///kafka=682798077049224619"
    },
  "cluster_id":"682798077049224619",
  "controller":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/brokers/0"
    },
  "acls":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/acls"
    },
  "brokers":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/brokers"
    },
  "broker_configs":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/broker-configs"
    },
  "consumer_groups":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/consumer-groups"
    },
  "topics":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics"
    },
  "partition_reassignments":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/-/partitions/-/reassignment"
    }
}
```


GET /v3/clusters/{string: cluster_Id}/topics

Retrieves a list of topic names on the specific cluster.

Description

The behavior of the information retrieved depends on the configuration. See `streams.default.stream` in Configuration Parameters on page 3865.

Table

Parameters Defined	Response
streams.default.stream is defined	Returns a list of topic names and metadata in the default stream. Returns topic names and metadata that contains a stream path.
streams.default.stream is not defined	Returns {"error_code":80001,"message":"HPE Ezmeral Data Fabric Event Data Streams does not currently support this API. Set the streams.default.stream parameter to return topics for the default stream"}.

Table

Parameters	Description
cluster_id (*string*)	Cluster's id.

Syntax

```
http://<host>:8082/v3/clusters/<cluster_id>/topics
```

Request Example

```
$ curl -X GET -H "Content-Type: application/json" "http://localhost:8082/v3/
clusters/682798077049224619/topics"
```

Response Example

```
{
  "kind":"KafkaTopicList",
  "metadata":
    {
      "self":"http://node1.cluster.com:8082/v3/clusters/682798077049224619/
topics",
      "next":null
    },
  "data":
    [
        {
          "kind":"KafkaTopic",
          "metadata":
            {
              "self":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp",
              "resource_name":"crn:///kafka=682798077049224619/topic=str:tp"
            },
          "cluster_id":"682798077049224619",
          "topic_name":"/str:tp",
          "is_internal":false,
          "replication_factor":1,
          "partitions":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp/partitions"
            },
          "configs":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp/configs"
            },
          "partition_reassignments":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp/partitions/-/reassignment"
            }
        },
        {
          "kind":"KafkaTopic",
          "metadata":
            {
              "self":"http://node1.cluster.com:8082/v3/clusters/
```

```
682798077049224619/topics/str:tp-2",
              "resource_name":"crn:///kafka=682798077049224619/
topic=str:tp-2"
            },
          "cluster_id":"682798077049224619",
          "topic_name":"/str:tp-2",
          "is_internal":false,
          "replication_factor":1,
          "partitions":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions"
            },
          "configs":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/configs"
            },
          "partition_reassignments":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/-/reassignment"
            }
        }
    ]
}
```

GET /v3/clusters/{string: cluster_id}/topics/{string: topic_name}
Retrieves metadata about a specific topic within a cluster.

Description

The behavior of the information retrieved depends on the configuration. See `streams.default.stream` in Configuration Parameters on page 3865.

Table

Parameters Defined	Response
streams.default.stream is defined	Gets metadata about a specific MapR Event Store topic on this cluster. A fully qualified topic name can be passed or not. If the topic name is not fully qualified, the metadata is retrieved and appended to the default stream path. For example, topic1 is equivalent to default_stream:topic1.
streams.default.stream is not defined	Gets metadata about a specific MapR Event Store topic on this cluster. A fully qualified topic name is passed that contains the stream path.

Note: The full name for the MapR Event Store topic contains characters such as a forward slash (/) and a colon (:). Therefore, the topic should be encoded. For example, `/streaming_data/stream:topic-1` is equivalent to `%2Fstreaming_data%2Fstream%3Atopic-1`.

Table

Parameters	Description
cluster_id (*string*)	Cluster's id.
topic_name (*string*)	Name of the topic.

Syntax

```
http://<host>:8082/v3/clusters/<string: cluster_id>/topics/<string:
topic_name>
```

Request Example

```
$ curl -X GET -H "Content-Type: application/json" "http://localhost:8082/v3/
clusters/682798077049224619/topics/tp-2"
```

Response Example

```
{
  "kind":"KafkaTopic",
  "metadata":
    {
      "self":"http://node1.cluster.com:8082/v3/clusters/682798077049224619/
topics/str:tp-2",
      "resource_name":"crn:///kafka=682798077049224619/topic=str:tp-2"
    },
  "cluster_id":"682798077049224619",
  "topic_name":"/str:tp-2",
  "is_internal":false,
  "replication_factor":1,
  "partitions":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions"
    },
  "configs":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/configs"
    },
  "partition_reassignments":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/-/reassignment"
    }
}
```

GET /v3/clusters/{string: cluster_id}/topics/{string: topic_name}/partitions

Retrieves a list of partitions for the topic within a cluster.

Description

The behavior of the information retrieved depends on the configuration. See `streams.default.stream` in Configuration Parameters on page 3865.

Table

Parameters Defined	Response
streams.default.stream is defined	Gets metadata about specific MapR Event Store partitions within a topic on this cluster. The user could pass fully qualified topic name or not. If a fully qualified topic name is not used, metadata is retrieved and appended to the default stream path.

Table (Continued)

Parameters Defined	Response
streams.default.stream is not defined	Gets metadata about specific MapR Event Store partitions within topic on this cluster. The user could only pass fully qualified topic name that contains stream path.

Table

Parameters	Description
cluster_id (*string*)	Cluster's id.
topic_name (*string*)	Name of the topic.

Syntax

```
http://<host>:8082/v3/clusters/<cluster_id>/topics/<topic_name>/partitions
```

Request Example

```
$ curl -X GET -H "Content-Type: application/json" "http://localhost:8082/v3/
clusters/682798077049224619/topics/tp-2/partitions"
```

Response Example

```
{
  "kind":"KafkaPartitionList",
  "metadata":
    {
      "self":"http://node1.cluster.com:8082/v3/clusters/682798077049224619/
topics/tp-2/partitions",
      "next":null
    },
  "data":
    [
        {
          "kind":"KafkaPartition",
          "metadata":
            {
              "self":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/0",
              "resource_name":"crn:///kafka=682798077049224619/
topic=str:tp-2/partition=0"
            },
          "cluster_id":"682798077049224619",
          "topic_name":"/str:tp-2",
          "partition_id":0,
          "leader":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/0/replicas/0"
            },
          "replicas":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/0/replicas"
            },
          "reassignment":
            {
```

```
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/0/reassignment"
            }
        },
        {
          "kind":"KafkaPartition",
          "metadata":
            {
              "self":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/1",
              "resource_name":"crn:///kafka=682798077049224619/
topic=str:tp-2/partition=1"
            },
          "cluster_id":"682798077049224619",
          "topic_name":"/str:tp-2",
          "partition_id":1,
          "leader":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/1/replicas/0"
            },
          "replicas":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/1/replicas"
            },
          "reassignment":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/1/reassignment"
            }
        },
        {
          "kind":"KafkaPartition",
          "metadata":
            {
              "self":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/2",
              "resource_name":"crn:///kafka=682798077049224619/
topic=str:tp-2/partition=2"
            },
          "cluster_id":"682798077049224619",
          "topic_name":"/str:tp-2",
          "partition_id":2,
          "leader":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/2/replicas/0"
            },
          "replicas":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/2/replicas"
            },
          "reassignment":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/2/reassignment"
            }
        }
    ]
}
```

GET /v3/clusters/{string: cluster_id}/topics/{string: topic_name}/partitions/{string: partition_id}

Retrieves metadata about a specific partition within a topic and a cluster.

Description

The behavior of the information retrieved depends on the configuration. See `streams.default.stream` in Configuration Parameters on page 3865.

Table

Parameters Defined	Response
streams.default.stream is defined	Gets metadata about a specific partition within a MapR Event Store topic on this cluster. You can pass a fully qualified topic name or not. If a fully qualified topic name is not used, metadata is retrieved and appended to the default stream path.
streams.default.stream is not defined	Gets metadata about specific MapR Event Store partitions within a topic. The user could only pass fully qualified topic names that contains stream path.

Table

Parameters	Description
cluster_id (*string*)	Cluster's id.
topic_name (*string*)	Name of the topic.
partition_id (*int*)	ID of the partition to inspect.

Syntax

```
http://<host>:8082/v3/clusters/<cluster_id>/topics/<topic_name>/partitions/
<partition_id>
```

Request Example

```
$ curl -X GET -H "Content-Type: application/json" "http://localhost:8082/v3/
clusters/682798077049224619/topics/tp-2/partitions/1"
```

Response Example

```
{
  "kind":"KafkaPartition",
  "metadata":
    {
      "self":"http://node1.cluster.com:8082/v3/clusters/682798077049224619/
topics/str:tp-2/partitions/1",
      "resource_name":"crn:///kafka=682798077049224619/topic=str:tp-2/
partition=1"
    },
  "cluster_id":"682798077049224619",
  "topic_name":"/str:tp-2",
  "partition_id":1,
  "leader":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/1/replicas/0"
    },
  "replicas":
```


last-updated: Apr 11, 2022

```
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/1/replicas"
    },
  "reassignment":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/1/reassignment"
    }
}
```

GET /v3/clusters/{string: cluster_id}/topics/{string: topic_name}/partitions/{string: partition_id}/replicas
Retrieves a list of replicas within a partition, a topic, and a cluster.

Description

The behavior of the information retrieved depends on the configuration. See `streams.default.stream` in Configuration Parameters on page 3865.

Table

Parameters Defined	Response
streams.default.stream is defined	Gets metadata about specific replicas of the MapR Event Store partition within a topic. You can pass a fully qualified topic name or not. If a fully qualified topic name is not used, metadata is retrieved and appended to the default stream path.
streams.default.stream is not defined	Gets metadata about specific replicas of the MapR Event Store partition within a topic. The user could only pass fully qualified topic name that contains stream path.

Table

Parameters	Description
cluster_id (*string*)	Cluster's id.
topic_name (*string*)	Name of the topic.
partition_id (*int*)	ID of the partition to inspect.

Syntax

```
http://<host>:8082/v3/clusters/<cluster_id>/topics/<topic_name>/partitions/
<partition_id>/replicas
```

Request Example

```
$ curl -X GET -H "Content-Type: application/json" "http://localhost:8082/v3/
clusters/682798077049224619/topics/tp-2/partitions/1/replicas"
```

Response Example

```
{
  "kind":"KafkaReplicaList",
  "metadata":
    {
      "self":"http://node1.cluster.com:8082/v3/clusters/682798077049224619/
topics/tp-2/partitions/1/replicas",
      "next":null
```


last-updated: Apr 11, 2022

```
    },
  "data":
    [
        {
          "kind":"KafkaReplica",
          "metadata":
            {
              "self":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/str:tp-2/partitions/1/replicas/0",
              "resource_name":"crn:///kafka=682798077049224619/
topic=str:tp-2/partition=1/replica=0"
            },
          "cluster_id":"682798077049224619",
          "topic_name":"/str:tp-2",
          "partition_id":1,
          "broker_id":0,
          "is_leader":true,
          "is_in_sync":true,
          "broker":
            {
              "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/brokers/0"
            }
        }
    ]
}
```


GET /v3/clusters/{string: cluster_id}/topics/{string: topic_name}/partitions/{string: partition_id}/replicas/{string: broker_id}

Retrieves metadata about a specific replica within a partition, a topic, and a cluster.

Description

The behavior of the information retrieved depends on the configuration. See `streams.default.stream` in Configuration Parameters on page 3865.

Table

Parameters Defined	Response
streams.default.stream is defined	Gets metadata about specific broker of the MapR Event Store partition within a topic. You can pass a fully qualified topic name or not. If a fully qualified topic name is not used, metadata is retrieved and appended to the default stream path.
streams.default.stream is not defined	Gets metadata about specific broker of the MapR Event Store partition within a topic. The user could only pass fully qualified topic name that contains stream path.

Table

Parameters	Description
cluster_id (*string*)	Cluster's id.
topic_name (*string*)	Name of the topic.
partition_id (*int*)	ID of the partition to inspect.
broker_id (*int*)	ID of the broker to inspect.

last-updated: Apr 11, 2022

Syntax

```
http://<host>:8082/v3/clusters/<cluster_id>/topics/<topic_name>/partitions/
<partition_id>/replicas/<broker_id>
```

Request Example

```
$ curl -X GET -H "Content-Type: application/json" "http://localhost:8082/v3/
clusters/682798077049224619/topics/tp-2/partitions/1/replicas/0"
```

Response Example

```
{
  "kind":"KafkaReplica",
  "metadata":
    {
      "self":"http://node1.cluster.com:8082/v3/clusters/682798077049224619/
topics/str:tp-2/partitions/1/replicas/0",
      "resource_name":"crn:///kafka=682798077049224619/topic=str:tp-2/
partition=1/replica=0"
    },
  "cluster_id":"682798077049224619",
  "topic_name":"/str:tp-2",
  "partition_id":1,
  "broker_id":0,
  "is_leader":true,
  "is_in_sync":true,
  "broker":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/brokers/0"
    }
}
```

POST /v3/clusters/{string: cluster_Id}/topics

Creates a new topic on the specific cluster.

Description

The behavior of the information retrieved depends on the configuration. See `streams.default.stream` in Configuration Parameters on page 3865

Table

Parameters Defined	Response
streams.default.stream is defined	Creates a topic in the default stream and returns its metadata.
streams.default.stream is not defined	Returns {"error_code":80001,"message":"HPE Ezmeral Data Fabric Event Data Streams does not currently support this API. Set the streams.default.stream parameter to return topics for the default stream"}.

Table

Parameters	Description
cluster_id (*string*)	Cluster's id.

Syntax

```
http://<host>:8082/v3/clusters/<cluster_id>/topics
```

Request Example

```
$ curl -X POST -H "Content-Type: application/json" --data '{"topic_name":
"new-topic","partitions_count": 4}' "http://localhost:8082/v3/clusters/
682798077049224619/topics"
```

Request Response

```
{
  "kind":"KafkaTopic",
  "metadata":
    {
      "self":"http://node1.cluster.com:8082/v3/clusters/682798077049224619/
topics/new-topic",
      "resource_name":"crn:///kafka=682798077049224619/topic=new-topic"
    },
  "cluster_id":"682798077049224619",
  "topic_name":"new-topic",
  "is_internal":false,
  "replication_factor":0,
  "partitions":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/new-topic/partitions"
    },
  "configs":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/new-topic/configs"
    },
  "partition_reassignments":
    {
      "related":"http://node1.cluster.com:8082/v3/clusters/
682798077049224619/topics/new-topic/partitions/-/reassignment"
    }
}
```

DELETE /v3/clusters/{string: cluster_id}/topics/{string: topic_name}

Deletes a topic from the specific cluster.

Description

The behavior of the information retrieved depends on the configuration. See `streams.default.stream` in Configuration Parameters on page 3865.

Table

Parameters Defined	Response
streams.default.stream is defined	Deletes topic from the default stream.
streams.default.stream is not defined	Returns {"error_code":80001,"message":"HPE Ezmeral Data Fabric Event Data Streams does not currently support this API. Set the streams.default.stream parameter to return topics for the default stream"}.

Table

Parameters	Description
cluster_id (*string*)	Cluster's id.
topic_name (*string*)	Name of the topic.

Syntax

```
http://<host>:8082/v3/clusters/<string: cluster_id>/topics/<string:
topic_name>
```

Request Example

```
$ curl -X DELETE -H "Content-Type: application/json" "http://
localhost:8082/v3/clusters/68279807704922 4619/topics/new-topic"
```

Response Example

```
HTTP/1.1 204 No Content
```

Kafka Connect
Kafka Connect is a utility for streaming data between MapR Event Store For Apache Kafka and other storage systems.

Examples of other systems include:

- Relational databases
- Logs and metrics
- Hadoop and data warehouses
- NoSQL data stores

Kafka Connect makes it easy to integrate all your data via Kafka, making it available as realtime streams. For example, you can use Kafka Connect to:

- Stream changes from a relational database to make events available with low latency for stream processing applications.
- Import realtime logs and metrics into MapR Event Store For Apache Kafka and process them to detect anomalies.
- Implement a process of loading data into MapR Event Store For Apache Kafka from your primary data storage systems, performing filtering, transformations, and enrichment with a stream processing framework, and publish the data to the HPE Ezmeral Data Fabric File Store.

Note: Built-in security is not available.

Architecture of Kafka Connect
Kafka Connect for MapR Event Store For Apache Kafka has the following major models in its design: connector, worker, and data.

Connector Model

A connector is defined by specifying a Connector class and configuration options to control what data is copied and how to format it.

last-updated: Apr 11, 2022

- Each Connector instance is responsible for defining and updating a set of Tasks that actually copy the data.
- Kafka Connect manages the Tasks; the Connector is only responsible for generating the set of Tasks and indicating to the framework when they need to be updated.
- Source and Sink Connectors/Tasks are distinguished in the API to ensure the simplest possible API for both.

There are two types of tasks:

- **Source** - Source tasks ingest data from data storage systems and stream the data to MapR Event Store For Apache Kafka.
- **Sink** - Sink tasks stream data from MapR Event Store For Apache Kafka to other storage systems.

MapR Data Platform supports the following connectors:

- JDBC Source Connector

  The Kafka JDBC source connector is a type connector used to stream data from relational databases into MapR Event Store For Apache Kafka topics. JDBC Source Connector for MapR Event Store For Apache Kafka supports integration with Hive 2.1.

- JDBC Sink Connector

  The Kafka JDBC sink connector is a type connector used to stream data from MapR Event Store For Apache Kafka topics to relational databases that have a JDBC driver.

- HDFS Sink Connector

  The Kafka HDFS sink connector is a type connector used to stream data from MapR Event Store For Apache Kafka to MapR File System. By default, the resulting data is produced to MapR File System in Avro format. In addition, Parquet files can be written to MapR File System.

Worker Model

A Kafka Connect for MapR Event Store For Apache Kafka cluster consists of a set of Worker processes that are containers that execute Connectors and Tasks. A worker is a JVM process with a REST API that is able to execute streaming tasks.

- Workers automatically coordinate with each other to distribute work and provide scalability and fault tolerance.
- The Workers distribute work among any available processes, but are not responsible for management of the processes;
- Any process management strategy can be used for Workers. For example, cluster management tools like YARN or Mesos, configuration management tools like Chef or Puppet, or direct management of process lifecycles.

Data Model

Connectors copy streams of messages from a partitioned input stream to a partitioned output stream, where at least one of the input or output is *always* Kafka.

- Each of these streams is an ordered set messages where each message has an associated offset.
- The format and semantics of these offsets are defined by the Connector to support integration with a wide variety of systems; however, to achieve certain delivery semantics in the face of faults requires that offsets are unique within a stream and streams can seek to arbitrary offsets.

- Message contents are represented by Connectors in a serialization-agnostic format.
- Pluggable Converters are available for storing this data in a variety of serialization formats.
- Schemas are built-in, allowing important metadata about the format of messages to be propagated through complex data pipelines. However, schema-free data can also be use when a schema is simply unavailable.

Connectors, Tasks, and Workers
Describes how Kafka Connect for MapR Event Store For Apache Kafka works and how connectors, tasks, offsets, and workers are associated.

Connectors

Connectors (or a **connector instance**) are logical jobs that are responsible for managing the copying of data between MapR Event Store For Apache Kafka and another systems. Each connector instantiates a set of **tasks** that copies the data. By allowing the connector to break a single job into many tasks, support is built-in for parallelism and scalable data copying with very little configuration. **Connector plugins** are jars that add the classes that implement a connector.

Offsets

As connectors run, Kafka Connect tracks **offsets** for each one so that connectors can resume from their previous position in the event of failures or graceful restarts for maintenance. They track the current position in the stream of data being copied and because each connector may need to track many offsets for different **partitions** of the stream. For example, when loading data from a database, the offset might be a transaction ID that identifies a position in the database change log.

Users generally do not need to worry about the format of offsets, especially since they differ from connector to connector. However, Kafka Connect does require persistent storage for offset data to ensure it can recover from faults. This storage for offset data is configurable. See Standalone Worker Configuration Options on page 3910 and Distributed Worker Configuration Options on page 3910.

Workers

Connectors and tasks are logical units of work and must be scheduled to execute in a process. Kafka Connect calls these processes **workers.** With Kafka Connect for MapR streams, the worker processors run as a service. This service can be run in either standalone mode or distributed mode.

- In standalone mode, the cluster consists of a single worker that is supplied with tasks that are useful for testing and debugging purposes.
- In distributed mode, the cluster consisting from multiple workers with the same group.id, offset.storage.topic, and config.storage.topic. Connector tasks are submitted via the Kafka Connect REST API.

The following list the location of the standalone and distributed worker configuration files:

```
/opt/mapr/kafka/kafka-<version>/config/connect-standalone.properties
```

```
/opt/mapr/kafka/kafka-<version>/config/connect-distributed.properties
```

Note: Distributed mode is supported on MapR 5.2.1 and above

Note: Port 8083 is the default port.

Note: If you running multiple workers on the same node, the rest.port parameter must be different for each worker.

last-updated: Apr 11, 2022

Configuring in Standalone Mode
The section describes how to configure and execute workers in Kafka connect standalone mode.

Standalone mode is the simplest mode, where a single process is responsible for executing all connectors and tasks. Since it is a single process, it requires minimal configuration.

Configuring and running in standalone mode, involves configuring the standalone properties and connector parameters before executing the standalone shell command along with the properties files on the command line.

The following parameters must be provided in the **connect-standalone.properties** file.

- offset.storage.file.filename - Storage for connector offsets which are stored on the local filesystem in standalone mode. Using the same file leads to offset data being deleted or overwritten with different values.
- rest.port - Port the REST interface listens on for HTTP requests. If you run multiple standalone instances on the same host, this parameter must have different values for each instance.

Note: If you are running multiple standalone instances on the same host, these parameters must be different for each instances. Therefore, an additional properties file is created for the instance with different parameter values.

To run a worker in standalone mode:

1. Edit the **./config/connect-standalone.properties** file and add the name of the local file that will store the connector offsets.
2. Edit the **quickstart-sqlite.properties** file (JDBC connector configuration file).
3. Run the **./bin/connect-standalone.sh** command along with the properties files on the command line.

For example:

```
cd /opt/mapr/kafka/kafka-<version>
./bin/connect-standalone.sh
./config/connect-standalone.properties
/opt/mapr/kafka-connect-jdbc/kafka-connect-jdbc-<version>/etc/
kafka-connect-jdbc/quickstart-sqlite.properties
```

The first parameter is always a configuration file for the worker. This configuration gives you control over settings such as which cluster to use and the serialization format. See JDBC Connector on page 3915 for more information. All additional parameters should be connector configuration files. Each file contains a single connector configuration.

Configuring in Distributed Mode
This section describes how to configure and run workers in Kafka Connect distributed mode.

Distributed mode provides scalability and automatic fault tolerance for Kafka Connect. In distributed mode, multiple worker processes are started using the same group.id. These processes automatically coordinate to schedule execution of connectors and tasks across all available workers. If a worker is added, shuts down, or fails unexpectedly, the rest of the workers detect this and automatically coordinate to redistribute connectors and tasks across the updated set of available workers.

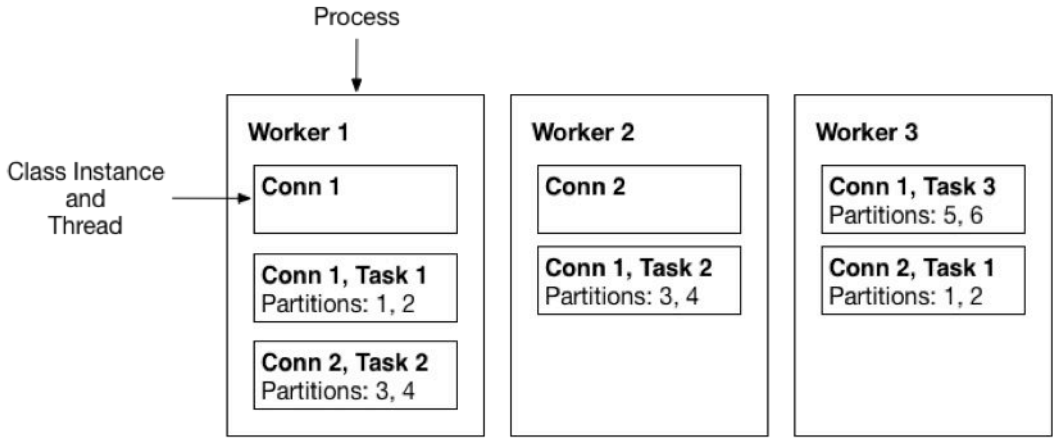
Note: Distributed mode is available as of EEP 2.0.1.

The following diagrams illustrates a three-node Kafka Connect distributed mode cluster where:

- Connectors are monitoring the source or sink system for changes that require reconfiguring tasks.
- Tasks are automatically balanced across the active workers by copying a subset of a connector's data.

last-updated: Apr 11, 2022

- The division of work between tasks is shown by the partitions that each task is assigned.



Interaction with a distributed-mode cluster is via the REST API (rather than on the command line). To create a connector, the workers are started and then REST request is made to create a connector. See REST API on page 3931.

Note: Kafka Connect workers do not have a special “leader” process that you have to interact with to use the REST API. All nodes can respond to REST requests, including creating, listing, modifying, and destroying connectors.

To run the worker in distributed mode:

1. In the `connect-distributed.properties` file, define the topics that will store the connector state, task configuration state, and connector offset state.

   In distributed mode, the workers need to be able to discover each other and have shared storage for connector configuration and offset data. In addition to the usual worker settings, ensure you have configured the following for the cluster:

   - **group.id** - ID that uniquely identifies the cluster these workers belong to. Ensure this is unique for all groups that work with a cluster.
   - **config.storage.topic** - Topic to store the connector and task configuration state in. Although this topic can be auto-created if your cluster has auto topic creation enabled, it is highly recommended that you create it before starting the cluster. This topic should **always** have a single partition and be highly replicated (3x or more).
   - **offset.storage.topic** - Topic to store the connector offset state in. To support large MapR Event Store For Apache Kafka clusters, this topic should have a large number of partitions (for example, 25 or 50 partitions and highly replicated (3x or more).
   - **rest.port** - Port where the REST interface listens for HTTP requests. If you run more than one worker per host (for example, if you are testing distributed mode locally during development), this setting must have different values for each instance.

2. Set the group.id value for all of the workers in the cluster.

   Note: All workers that belong to the same cluster must have the same group.id value.

last-updated: Apr 11, 2022

3. Start the Kafka Connect service in distributed mode:

```
maprcli node services -name kafka-connect -action start -nodes
<node_list>
```

For more information, see Managing Kafka Connect Services on page 3925.

Note: >Distributed mode does not have any additional command line parameters. If other instances are already running, new workers either start a new group or join an existing one, and then wait for work to do. For information on managing the connectors running in the cluster, see REST API on page 3931.

Connector Configuration
This section describes how and where connectors are configured.

Connector configurations are key-value mappings. For standalone mode, these parameters are defined in a properties file and passed to the Connect process on the command line. In distributed mode, they will be included in the JSON payload for the request that creates (or modifies) the connector. Most configurations are connector dependent, but there are a few settings common to all connectors:

- name - Unique name for the connector. Attempting to register again with the same name will fail.
- connector.class - The Java class for the connector
- tasks.max - The maximum number of tasks that should be created for this connector. The connector may create fewer tasks if it cannot achieve this level of parallelism.

Sink connectors also have one additional option to control their input:

- topics - A list of topics to use as input for this connector

For other options, consult the documentation for the JDBC and HDFS connectors. See JDBC Connector on page 3915 and HDFS Connector on page 3925.

Worker Configuration
This section describes how and where to configure workers.

Whether you're running standalone or distributed mode, Kafka Connect workers are configured by passing a properties file containing any required or overridden options as the first parameter to the worker process.

Common Worker Configuration Options
You can set common worker configuration options for standalone or distributed mode in the `connect-<standalone/distributed>.properties` file. The options control basic functionality, including which cluster to communicate with and data format.

Setting the Schema Registry URL for the Avro Converter

Set the Schema Registry URL for the converter through the following properties:

- `key.converter.schema.registry.url=<URL:PORT>`
- `value.converter.schema.registry.url=<URL:PORT>`
- For Avro, use `io.confluent.connect.avro.AvroConverter`.
-
-

If you do not set these properties, the Schema Registry URL is taken from ZooKeeper.

The following table describes the common worker configuration parameters:

last-updated: Apr 11, 2022

Parameter	Description
plugin.path	The comma-separated list of paths to directories that contain Kafka Connect plugins. • Type: string • Default: empty
key.converter	Converter class for key Connect data. This controls the format of the data that will be written to MapR Event Store For Apache Kafka for source connectors or read from MapR Streams for sink connectors. • Type: class • Default: empty
value.converter	Converter class for value Connect data. This controls the format of the data that will be written to MapR Event Store For Apache Kafka for source connectors or read from MapR Streams for sink connectors. • Type: class • Default: empty
internal.key.converter	Converter class for internal key Connect data that implements the Converter interface. Used for converting data like offsets and configs. • Type: class • Default:
internal.value.converter	Converter class for offset value Connect data that implements the Converter interface. Used for converting data like offsets and configs. • Type: class • Default:
offset.flush.interval.ms	Interval (milliseconds) at which to try committing offsets for tasks. • Type: long • Default: 60000
offset.flush.timeout.ms	Maximum number of milliseconds to wait for records to flush and partition offset data to be committed to offset storage before cancelling the process and restoring the offset data to be committed in a future attempt. • Type: long • Default: 5000
rest.advertised.host.name	If set, this is the hostname that will be given out to other workers to connect to. • Type: string
rest.advertised.port	If set, this is the port that will be given out to other workers to connect to. • Type: int

last-updated: Apr 11, 2022

Parameter	Description
rest.host.name	Hostname for the REST API. If this is set, it will only bind to this interface. • Type: string
rest.port	Port for the REST API to listen on. • Type: int • Default: 8083
task.shutdown.graceful.timeout.ms	Amount of time to wait (milliseconds) for tasks to shutdown gracefully. This is the total amount of time, not per task. All task have shutdown triggered, then they are waited on sequentially. • Type: long • Default: 5000
streams.consumer.streams.default.stream	If set, topic names can be used in the Sink task configuration without the stream name. The defined default stream is used.
streams.producer.producer.default.stream	If set, topic names can be used in the Source task configuration without the stream name. The defined default stream is used.

Standalone Worker Configuration Options
This section describes worker parameters that are specific to standalone configurations.

The offset.storage.file.filename and rest.port parameter are specific to the standalone worker configuration. These parameters are sent in the `connect-standalone.properties` file.

Table

Parameter	Description
rest.port	Port the REST interface listens on for HTTP requests. If you run multiple standalone instances on the same host, this setting must have different values for each instance. Type: int. Default: 8083
offset.storage.file.filename	The file to store connector offsets in. By storing offsets on disk, a standalone process can be stopped and started on a single node and resume where it previously left off. Type: string. Default: empty

Distributed Worker Configuration Options
This topic describes the worker parameters that are specific to distributed configurations.

In addition to the common worker configuration options, the following are available in distributed mode. These parameters are set in the `connect-distributed.properties` file.

Table

Parameter	Description	Type	Default
group.id	A unique string that identifies the Connect cluster group that the worker belongs to.	string	“”
config.storage.topic	The name of the MapR Event Store For Apache Kafka topic to store connector and task configuration data in. This *must* be the same for all workers with the same group.id. For example: `/path/to/stream:topic-prefix-`	string	“”

last-updated: Apr 11, 2022

Table (Continued)

Parameter	Description	Type	Default
status.storage.topic	The name of the MapR Event Store For Apache Kafka topic where connector and task configuration updates are stored. This *must* be the same for all workers with the same group.id.	string	“”
offset.storage.topic	The MapR Event Store For Apache Kafka topic to store offset data for connectors in. This *must* be the same for all workers with the same group.id. For example: `/path/to/stream:topic-prefix-`	string	“”
heartbeat.interval.ms	The expected time between heartbeats to the group coordinator when using Kafka’s group management facilities. Heartbeats are used to ensure that the worker’s session stays active and to facilitate rebalancing when new members join or leave the group. The value must be set lower than session.timeout.ms, but typically should be set no higher than 1/3 of that value. It can be adjusted even lower to control the expected time for normal rebalances.	int	3000
session.timeout.ms	The timeout used to detect failures when using Kafka’s group management facilities.	int	30000
connections.max.idle.ms	Close idle connections after the number of milliseconds specified by this config.	long	540000
receive.buffer.bytes	The size of the TCP receive buffer (SO_RCVBUF) to use when reading data.	int	32768
request.timeout.ms	The configuration controls the maximum amount of time the client will wait for the response of a request. If the response is not received before the timeout elapses the client will resend the request if necessary or fail the request if retries are exhausted.	int	40000
send.buffer.bytes	The size of the TCP send buffer (SO_SNDBUF) to use when sending data.	int	131072
worker.sync.timeout.ms	When the worker is out of sync with other workers and needs to resynchronize configurations, wait up to this amount of time before giving up, leaving the group, and waiting a backoff period before rejoining.	int	3000
worker.unsync.backoff.ms	When the worker is out of sync with other workers and fails to catch up within worker.sync.timeout.ms, leave the Connect cluster for this long before rejoining.	int	300000
client.id	An id string to pass to the server when making requests. The purpose of this is to be able to track the source of requests beyond just IP/port by allowing a logical application name to be included in server-side request logging.	string	“”
metadata.max.age.ms	The period of time in milliseconds after which we force a refresh of metadata even if we haven’t seen any partition leadership changes to proactively discover any new brokers or partitions.	long	300000

Table (Continued)

Parameter	Description	Type	Default
metric.reporters	A list of classes to use as metrics reporters. Implementing the MetricReporterinterface allows plugging in classes that will be notified of new metric creation. The JmxReporter is always included to register JMX statistics.	list	[]
metrics.num.samples	The number of samples maintained to compute metrics.	int	2
metrics.sample.window.ms	The number of samples maintained to compute metrics.	long	30000
reconnect.backoff.ms	The amount of time to wait before attempting to reconnect to a given host. This avoids repeatedly connecting to a host in a tight loop. This backoff applies to all requests sent by the consumer to the broker.	long	50
retry.backoff.ms	The amount of time to wait before attempting to retry a failed fetch request to a given topic partition. This avoids repeated fetching-and-failing in a tight loop.	long	100

Security Configuration Options
Describes Kafka Connect security parameters.

The following security parameters provide an authentication, encryption, and impersonation layer between the Kafka Connect REST API clients and the Kafka Connect REST Gateway.

These parameters are configurable in the `connect-distributed.properties` file.

```
/opt/mapr/kafka/kafka-&lt;version>/config/connect-distributed.properties
```

Note: Ensure that both an ssl_keystore and ssl_truststore file have been created.

Table

Parameter	Description	Type	Default
listeners	Comma-separated list of listeners that listen for API requests over either HTTP or HTTPS. If a listener uses HTTPS, the appropriate SSL configuration parameters need to be set as well. Each listener must include the protocol, hostname, and port. For example: http://localhost:8082	list	none
connect.enable.doAs	Specifies whether or not to enable impersonation for MapR Event Store For Apache Kafka topics. For this to take effect, PAM authentication must be enabled.	boolean	true
authentication.method	Specifies whether or not to enable PAM authentication. Set to NONE to disable.	string	BASIC
authentication.realm	Specifies realm for PAM authentication. Set to an empty string ("") to disable PAM. Set to jpamLogin to enable authentication	string	jpam
ssl.cipher.suites	A list of SSL cipher suites. This list is a comma-separated list. Leave blank to use Jetty's default.	list	none

last-updated: Apr 11, 2022

Table (Continued)

Parameter	Description	Type	Default
ssl.cipher.suites.exclude	A list of disabled SSL cipher suites. This is a comma-separated list.	list	• TLS_DHE.* • TLS_EDH.* • .*DES.* • .*MD5.* • .*RC4.*
ssl.disabled.protocols	The list of SSL protocols that will not be accepted by clients. This is a comma-separated list.	list	• SSLv3 • TLSv1.0
ssl.enabled.protocols	The list of SSL protocols that can be accepted from clients. The list is a comma-separated list. Leave blank to use Jetty's defaults.	list	empty
ssl.endpoint.identification.algorithm	The endpoint identification algorithm to validate the server hostname using the server certificate. IMPORTANT: Jetty requires that the key's CN, stored in the keystore, must match the FQDN if ssl_endpoint_identification_algorithm=https. Leave blank to use Jetty's default.	string	none
ssl.key.password	The password of the private key in the keystore file. If this parameter is not set, the property value is obtained from the ssl-client.xml file.	string	empty
ssl.keymanager.algorithm	The algorithm used by the key manager factory for SSL connections. Leave blank to use Jetty's default.	string	none
ssl.keystore.location	Location of the keystore file. If this parameter is not set, the property value is obtained from the ssl-client.xml file.	string	empty
ssl.keystore.password	The store password for the keystore file. If this parameter is not set, the property value is obtained from the ssl-client.xml file.	string	empty
ssl.keystore.type	The type of keystore file.	string	JKS
ssl.protocol	The SSL protocol used to generate the SslContextFactory.	string	TLSv1.2
ssl.provider	The SSL security provider name. Leave blank to use Jetty's default.	string	none
ssl.trustmanager.algorithm	The algorithm used by the trust manager factory for SSL connections. Leave blank to use Jetty's default.	string	none
ssl.truststore.location	Location of the trust store. Required only to authenticate HTTPS clients.	string	empty
ssl.truststore.password	The store password for the trust store file.	string	empty
ssl.truststore.type	The type of trust store file.	string	JKS

last-updated: Apr 11, 2022

SSL Security Configuration
Describes how to configure Kafka Connect security on a MapR Data Platform cluster.

Secure by Default

As of Core 6.0, the Installer performs the Kafka Connect configuration for new installations. This means that:

- If core is installed as *secure*, then Kafka Connect is also installed as *secure*.
- If core is installed as *unsecure*, then Kafka Connect is also installed as *unsecure*.

Important: In addition, every time the MapR configuration script is run with the `-R` option (`configure.sh -R`), the default settings for MapR core are re-established.

This means that if you manually configure Kafka Connect for *unsecure* on a *secure* MapR core, Kafka Connect will revert back to *secure* when `configure.sh -R` is run.

Manually Securing Kafka Connect Only

CAUTION: This configuration is ***not*** a typical configuration.

If you have an *unsecure* MapR Data Platform cluster, and you want to *secure* Kafka Connect, do the following:

1. Generate the server and client certificates.
2. Add any necessary property configurations to the `connect-distributed.properties` configuration file. For example:

```
listeners=http://0.0.0.0:8083
            ssl.keystore.location=<ssl-keystore-path>
            ssl.keystore.password=<ssl-keystore-password>
            ssl.key.password=<ssl-keystore-password>
```

3. Restart Kafka Connect.

```
maprcli node services -name kafka-connect -action restart -nodes <space
delimited list of nodes>
```

4. Run a curl command to ensure that HTTPS is enabled.

```
curl -X GET https://node1:8083/connectors --cacert <certificate-path>
```

Manually Unsecuring Kafka Connect

Warning: This scenario is ***NOT*** recommended or supported.

If you have a *secure* MapR Data Platform cluster, and you want to *unsecure* Kafka Connect, do the following:

1. In the `connect-distributed.properties` configuration file, change **https://** to **http://** for the listeners and remove the **ssl.*** properties. For example:

```
listeners=http://0.0.0.0:8083
```


last-updated: Apr 11, 2022

2. Restart Kafka Connect.

```
maprcli node services -name kafka-connect -action restart -nodes <space
delimited list of nodes>
```

User Impersonation
Describes how to disable, enable, and use impersonation with Kafka Connect.

User impersonation enables Kafka Connect jobs to be submitted as a particular user. Without impersonation, Kafka Connect submits jobs as the user that started the worker

On a MapR Data Platform cluster, the impersonated user is typically the `mapr` user or the user specified in the MAPR_USER environment variable. By default, impersonation and PAM authentication in Kafka Connect are enabled on all types of security.

Disabling User Impersonation

To disable user impersonation, you need to first disable the PAM authentication properties in the connect-distributed.properties file and then disable the connect.proxy.enable.doAs property.

1. Disable PAM authentication. Set the following properties in `opt/mapr/kafka/kafka-<version>/config/connect-distributed.properties:`
   - authentication.method=NONE
2. Once authentication is disabled, set the following property in `/opt/mapr/kafka/kafka-<version>/config/connect-distributed.properties:`
   - connect.proxy.enable.doAs=false

JDBC Connector
The topics describes the JDBC connector, drivers, and configuration parameters.

The JDBC connector allows you to import data from any relational database into MapR Event Store For Apache Kafka and export data from MapR Event Store For Apache Kafka to any relational database with a JDBC driver. By using JDBC, this connector can support a wide variety of databases without requiring custom code for each one.

The JDBC connector provides flexibility regarding which databases you can import data from and how that data is imported. JDBC connector implements the data copying functionality on the generic JDBC APIs, and relies on JDBC drivers to handle the database-specific implementation of those APIs.

The supported relational databases include:

- MySQL
- Oracle
- PostgreSQL
- SQLite
- SQL Server
- Hive is supported for the JDBC Source Connector

JDBC Driver
Kafka Connect for MapR Event Store For Apache Kafka provides a JDBC driver jar along with the connector configuration.

The JDBC driver (kafka-connect-jdbc) is set up by specifying the CLASSPATH variable. See Installing MapR Event Store For Apache Kafka Tools on page 202.

last-updated: Apr 11, 2022

The packaged connector is installed in the **share/java/kafka-connect-jdbc** directory, relative to the installation directory.

Alternatively, to add a new driver to the CLASSPATH,

1. Put the classpath of the connectors in the **kafka-connect-jdbc** directory:

```
/opt/mapr/kafka-connect-jdbc/kafka-connect-jdbc-<connector version>/
share/java/kafka-connect-jdbc/
```

2. Create a symlink into the **share/java/kafka-connect-jdbc/** directory.

JDBC Configuration Options
Use the following parameters to configure the Kafka Connect for MapR Event Store For Apache Kafka JDBC connector; they are modified in the `quickstart-sqlite.properties` file.

Configuration Modes

In *standalone* mode, JDBC connector configuration is specified in the **quickstart-sqlite.properties** file. Additional configurations such as the offset storage location and the port for the REST interface are specified in the **connect-standalone.properties** file. See Configuring in Standalone Mode on page 3906.

```
/opt/mapr/kafka-connect-jdbc/kafka-connect-jdbc-<version>/etc/
kafka-connect-jdbc/quickstart-sqlite.properties
/opt/mapr/kafka/kafka-<version>/config/connect-standalone.properties
```

In *distributed* mode, HDFS connector configuration is provided in the POST and PUT requests when creating or modifying the connector. See POST /connectors on page 3933 and PUT /connectors/(string:name)/config on page 3935 for more information about using the REST API. Additional configurations such as defining the topics that will store the connector state, task configuration state, and connector offset state are specified in the **connect-distributed.properties** file. See Configuring in Distributed Mode on page 3906 .

```
/opt/mapr/kafka/kafka-<version>/config/connect-distributed.properties
```

JDBC Source Configuration Options

Table

Parameters	Description
connection.url	JDBC connection URL for the database to load. • Type: string • Default: “”
connection.user	JDBC connection user. • Type: string • Default: NULL
connection.password	JDBC connection password. • Type: password • Default: NULL

Table (Continued)

Parameters	Description
connection.attempts	Maximum number of attempts to retrieve a valid JDBC connection. • Type: int • Default: 3
connection.backoff.ms	Backoff time in milliseconds between connection attempts. • Type: long • Default: 10000
table.whitelist	List of tables to include in copying. If specified, table.blacklist may not be set. • Type: list • Default: []
table.blacklist	List of tables to exclude from copying. If specified, table.whitelist may not be set. • Type: list • Default: []
numeric.precision.mapping	Whether or not to attempt mapping *numeric* values by precision to integral types. • Type: boolean • Default: false
schema.pattern	Schema pattern to fetch table metadata from the database. "" - Retrieves those without a schema. * - NULL (default) means that the schema name should not be used to narrow the search, all tables metadata would be fetched, regardless their schema.

last-updated: Apr 11, 2022

Table (Continued)

Parameters	Description
mode	The mode for updating a table each time it is polled. Options include: • bulk - perform a bulk load of the entire table each time it is polled. • incrementing - use a strictly incrementing column on each table to detect only new rows. Note that this will not detect modifications or deletions of existing rows. • timestamp - use a timestamp (or timestamp-like) column to detect new and modified rows. This assumes the column is updated with each write, and that values are monotonically incrementing, but not necessarily unique. • timestamp+incrementing - use two columns, a timestamp column that detects new and modified rows and a strictly incrementing column which provides a globally unique ID for updates so each row can be assigned a unique stream offset. • Type: string • Default: “” Valid Values: [, bulk, timestamp, incrementing, timestamp+incrementing] The name of the strictly incrementing column to use to detect new rows. Any empty value indicates the column should be autodetected by looking for an autoincrementing column. This column may not be nullable. • Type: string • Default: “” **Note:** If you are using Hive JDBC with incrementing or timestamp mode, you should set the `validate.non.null` property to false because there are no "not null" columns in Hive.
timestamp.column.name	The name of the timestamp column to use to detect new or modified rows. This column may not be nullable. • Type: string • Default: “”
validate.non.null	By default, the JDBC connector will validate that all incrementing and timestamp tables have NOT NULL set for the columns being used as their ID/timestamp. If the tables don’t, JDBC connector will fail to start. Setting this to false will disable these checks. • Type: boolean • Default: true **Note:** If this parameter is false, specify exactly all columns that need to be imported to MapR Event Store For Apache Kafka in the query parameter. For example instead of `"query" : "select * from table"`, use `"query" : "select col1, col2 from table"`

Table (Continued)

Parameters	Description
incrementing.column.name	The name of the strictly incrementing column to use to detect new rows. Any empty value indicates the column should be autodetected by looking for an auto-incrementing column. This column may not be nullable. • Type: string • Default: “”
query	If specified, the query to perform to select new or updated rows. Use this setting to join tables, select subsets of columns in a table, or filter data. If used, this connector will only copy data using this query – whole-table copying will be disabled. Different query modes may still be used for incremental updates, but in order to properly construct the incremental query, it must be possible to append a WHERE clause to this query (i.e. no WHERE clauses may be used). If you use a WHERE clause, it must handle incremental queries itself. • Type: string • Default: “”
poll.interval.ms	Frequency (milliseconds) to poll for new data in each table. • Type: int • Default: 5000
batch.max.rows	Maximum number of rows to include in a single batch when polling for new data. This setting can be used to limit the amount of data buffered internally in the connector. • Type: int • Default: 100
table.poll.interval.ms	Frequency (milliseconds) to poll for new or removed tables, which may result in updated task configurations to start polling for data in added tables or stop polling for data in removed tables. • Type: long • Default: 60000
topic.prefix	Prefix to prepend to table names to generate the name of the Kafka topic to publish data to, or in the case of a custom query, the full name of the topic to publish to. For example: `/path/to/stream:topic-prefix-`. • Type: string • Default: “”

last-updated: Apr 11, 2022

Table (Continued)

Parameters	Description
table.types	By default, the JDBC connector will only detect tables with type TABLE from the source Database. This config allows a command separated list of table types to extract. Options include: • TABLE • VIEW • SYSTEM TABLE • GLOBAL TEMPORARY • LOCAL TEMPORY • ALIAS • SYNONYM Typically, TABLE or VIEW are used. • Type: list • Default: TABLE
timestamp.delay.interval.ms	How long to wait after a row with certain timestamp appears before it is included in the result. You may choose to add some delay to allow transactions with earlier timestamp to complete. The first execution fetches all available records (for example, starting at timestamp 0) until the current time minus the delay. Every following execution retrieves data from the last time data was fetched until the current time minus the delay. • Type: long • Default: 0

JDBC Sink Configuration Options

Table

Parameters	Description
connection.url	JDBC connection URL. • Type: string • Default: “”
connection.user	JDBC connection user. • Type: string • Default: NULL
connection.password	JDBC connection password. • Type: password • Default: NULL

last-updated: Apr 11, 2022

Table (Continued)

Parameters	Description
insert.mode	The insertion mode to use. • INSERT - Use standard SQL INSERT statements. • UPSERT - Use the appropriate upsert semantics for the target database if it is supported by the connector. For example: INSERT or IGNORE • UPDATE - Use the appropriate update semantics for the target database if it is supported by the connector. For example: UPDATE • Type: string • Default: INSERT • Valid Values: insert, upsert, update
batch.size	Specifies how many records to attempt to batch together for insertion into the destination table, when possible. • Type: int • Default: 3000 • Valid Values: 0,...
table.name.format	A format string for the destination table name, which may contain `${topic}` as a placeholder for the originating topic name. For example, table_${topic} for the topic `orders` maps to the table name `table_orders`. • Type: string • Default: ${topic}
pk.mode	The primary key mode, also refer to pk.fields documentation for interplay. Supported modes are: • none - No keys utilized. • kafka - Kafka coordinates are used as the PK. • record_key - Field(s) from the record key are used, which may be a primitive or a struct. • record_value - Field(s) from the record value are used, which must be a struct. • Type: string • Default: none • Valid Values: none, kafka, record_key, record_value

last-updated: Apr 11, 2022

Table (Continued)

Parameters	Description
pk.fields	List of comma-separated primary key field names. The runtime interpretation of this config depends on the pk.mode: • none - Ignored as no fields are used as primary key in thes mode. • kafka - Must be a trio representing the Kafka coordinates. Defaults to __connect_topic,__connect_partition,__connect_offset if empty. • record_key - If empty, all fields from the key struct will be used, otherwise used to extract the desired fields - for primitive key only a single field name must be configured. • record_value - If empty, all fields from the value struct will be used, otherwise used to extract the desired fields. • Type: list • Default: ""
fields.whitelist	List of comma-separated record value field names. If empty, all fields from the record value are utilized, otherwise used to filter to the desired fields. Note: pk.fields is applied independently in the context of which field(s) form the primary key columns in the destination database, while this configuration is applicable for the other columns. • Type: list • Default: ""
auto.create	Whether to automatically create the destination table based on record schema if it is found to be missing by issuing CREATE. • Type: boolean • Default: false
auto.evolve	Whether to automatically add columns in the table schema when found to be missing relative to the record schema by issuing ALTER. • Type: boolean • Default: false
max.retries	The maximum number of times to retry on errors before failing the task. • Type: int • Default: 10 • Valid Values: 0,..
retry.backoff.ms	The time in milliseconds to wait following an error before a retry attempt is made. • Type: int • Default: 3000 • Valid Values: 0,..

Whitelists and Custom Query JDBC Examples
This section provides common usage scenarios using whitelists and custom queries.

Using Whitelists

Use a whitelist to limit changes to a subset of tables in a MySQL database, using id and modifiedcolumns that are standard on all whitelisted tables to detect rows that have been modified. This mode is the most robust because it can combine the unique, immutable row IDs with modification timestamps to guarantee modifications are not missed even if the process dies in the middle of an incremental update query.

The following is an example of a whitelist.

Note: Before running this example, you need to create the stream `/kafka-connect`

```
name=mysql-whitelist-timestamp-source
connector.class=io.confluent.connect.jdbc.JdbcSourceConnector tasks.max=10
connection.url=jdbc:mysql://mysql.example.com:3306/my_database?
user=alice&password=secret
table.whitelist=users,products,transactions
mode=timestamp+incrementing
timestamp.column.name=modified
incrementing.column.name=id
topic.prefix=/kafka-connect:mysql-
```

Using Custom Queries

Use a custom query instead of loading tables to join data from multiple tables. As long as the query does not include its own filtering, you can still use the built-in modes for incremental queries (in this case, using a timestamp column).

Note: This limits you to a single output per connector and because there is no table name, the topic “prefix” is actually the full topic name in this case.

The following is an example of a custom query.

Note: Before running this example, you need to create the stream `/kafka-connect`

```
name=mysql-whitelist-timestamp-source
        connector.class=io.confluent.connect.jdbc.JdbcSourceConnector
        tasks.max=10
        connection.url=jdbc:postgresql://postgres.example.com/test_db?
user=bob&password=secret&ssl=true
        query=SELECT users.id,
        users.name,
        transactions.timestamp,
        transactions.user_id,
        transactions.payment FROM users JOIN transactions ON (users.id =
transactions.user_id)
        mode=timestamp
        timestamp.column.name=timestamp
        topic.prefix=/kafka-connect:mysql-joined-data
```

Streaming Data JDBC Examples
This section provides common usage scenarios of streaming data between different databases to or from MapR Event Store For Apache Kafka.

Streaming Data from MapR Event Store For Apache Kafka to a MySQL Database

The following is example code for streaming data from MapR Event Store For Apache Kafka stream topics to a MySQL database.

```
POST /connectors HTTP/1.1
 Host: connect.example.com
 Content-Type: application/json
 Accept: application/json
{"name": "mysql-sink-connector",
"config": {
"connector.class":"io.confluent.connect.jdbc.JdbcSinkConnector",
"connection.url":"jdbc:mysql://hostname:3306/mysql_db?
user=<user>&password=<password>",
"auto.create":"true",
"topics":"/kafka-connect:topic1",
"tasks.max":"2",
"insert.mode":"insert"
}}
```

Streaming Data from a MySQL Database to MapR Event Store For Apache Kafka

The following is example code for streaming data from a MySQL database to MapR Event Store For Apache Kafka stream topics.

```
POST /connectors HTTP/1.1
 Host: connect.example.com
 Content-Type: application/json
 Accept: application/json
{"name": "mysql-source-connector",
"config":
{ "connector.class":"io.confluent.connect.jdbc.JdbcSourceConnector",
"connection.url":"jdbc:mysql://hostname:3306/newdb?
user=<user>&password=<password>"
"mode":"incrementing",
"incrementing.column.name":"id",
"topic.prefix":"/kafka-connect:mysql-",
"tasks.max":"1"
}}
```

Streaming Data from a Hive Database to MapR Event Store For Apache Kafka

The following is example code for streaming data from a Hive database to MapR Event Store For Apache Kafka stream topics.

```
POST /connectors HTTP/1.1
 Host: connect.example.com
 Content-Type: application/json
 Accept: application/json
{"name": "hive-source-connector",
"config": {
"connector.class":"io.confluent.connect.jdbc.JdbcSourceConnector",
"connection.url":"jdbc:hive2://hostname:10000/
database_name;user=<user>;password=<pa
ssword>",
"mode":"bulk",
"topic.prefix":"/kafka-connect:hive-",
"tasks.max":"1"
}}
```

Note: For a secure MapR Data Platform cluster, use next connection.url jdbc:hive2://hostname:10000/database_name;auth=maprsasl

.

Managing Kafka Connect Services
Lists the commands you use to start, stop, or restart Kafka Connect Services

Use the `maprcli node services` command with the `-action` parameter as follows:

```
maprcli node services -name kafka-connect -action start -nodes <node_list>
```

```
maprcli node services -name kafka-connect -action stop -nodes <node_list>
```

```
maprcli node services -name kafka-connect -action restart -nodes
<node_list>
```

For more information see node services on page 1730

HDFS Connector
These topics describe the Kafka Connect for MapR Event Store For Apache Kafka HDFS connector, driver, and configuration parameters.

The HDFS connector allows you to export data from MapR Event Store For Apache Kafka topics to MapR File System or HDFS files in a variety of formats. In addition, Hive integration is available, which can be use to make data immediately available for querying with HiveQL.

HDFS Configuration Options
Use the following parameters to configure the Kafka Connect for MapR Event Store For Apache Kafka HDFS connector.

Note: For the HDFS connector, both Avro and Parquet files can be written.

In *standalone* mode, specify the HDFS connector configuration in the **quickstart-hdfs.properties** file. You can also configure the offset storage location and the port for the REST interface, which are specified in the **connect-standalone.properties** file. See Configuring in Standalone Mode on page 3906.

```
/opt/mapr/kafka-connect-hdfs/kafka-connect-hdfs-<version>/etc/
kafka-connect-hdfs/quickstart-hdfs.properties
/opt/mapr/kafka/kafka-<version>/config/connect-standalone.properties
```

In *distributed* mode, HDFS connector configuration is provided in the POST and PUT requests when creating or modifying the connector. See POST /connectors on page 3933 and PUT /connectors/(string:name)/config on page 3935 for more information about using the REST API. Additional configurations such as defining the topics that will store the connector state, task configuration state, and connector offset state are specified in the **connect-distributed.properties** file. See Configuring in Distributed Mode on page 3906 .

```
/opt/mapr/kafka/kafka-<version>/config/connect-distributed.properties
```

Table

Parameter	Description
flush.size	Number of records written to the filesystem before invoking file commits. • Type: int • Default: ""

last-updated: Apr 11, 2022

Table (Continued)

Parameter	Description
hdfs.url	The filesystem connection URL. This configuration has the format of maprfs:// hostname:port and specifies the MapR filesystem to export data to. • Type: string • Default: ""
connect.hdfs.keytab	The path to the keytab file for the HDFS connector principal. This keytab file should only be readable by the connector user. • Type: string • Default: ""
connect.hdfs.principal	The principal used when the filesystem is using Kerberos for authentication. • Type: string • Default: ""
format.class	The format class used when writing data to the filesystem. • Type: string • Default: "io.confluent.connect.hdfs.avro.AvroFormat" **Note:** If you want to write to a Parquet set, use "io.confluent.connect.hdfs.parquet.ParquetFormat"
hadoop.conf.dir	The Hadoop configuration directory. • Type: string • Default: ""
hadoop.home	The Hadoop home directory. • Type: string • Default: ""
hdfs.authentication.kerberos	Specifies whether the filesystem uses Kerberos for authentication. • Type: boolean • Default: false
hdfs.namenode.principal	The Kerberos principal for CLDB. • Type: string • Default: ""
hive.conf.dir	The Hive configuration directory. • Type: string • Default: ""

last-updated: Apr 11, 2022

Table (Continued)

Parameter	Description
hive.database	The database used when the connector creates tables in Hive. • Type: string • Default: "default"
hive.home	The Hive home directory. • Type: string • Default: ""
hive.integration	Specifies whether Hive is integrated when running the connector. • Type: boolean • Default: false
hive.metastore.uris	The Hive metastore URIs. Can be an IP address or fully-qualified domain name and port of the metastore host. • Type: string • Default: ""
logs.dir	Top-level filesystem directory to store the write ahead logs. • Type: string • Default: "logs"
partitioner.class	The partitioner used when writing data to the filesystem. You can use DefaultPartitioner, which preserves the Kafka partitions; FieldPartitioner, which partitions the data to different directories according to the value of the partitioning field specified in partition.field.name; TimeBasedPartitioner, which partitions data according to the time ingested to the filesystem. • Type: string • Default: "io.confluent.connect.hdfs.partitioner.DefaultPartitioner"
rotate.interval.ms	The time interval (milliseconds) before invoking file commits. This configuration ensures that file commits are invoked every configured interval. This configuration is useful when data ingestion rate is low and the connector didn't write enough messages to commit files. The default value -1 means that this feature is disabled. • Type: long • Default: -1
schema.compatibility	The schema compatibility rule used when the connector is observing schema changes. The supported configurations are NONE, BACKWARD, FORWARD and FULL. • Type: string • Default: "NONE"

last-updated: Apr 11, 2022

Table (Continued)

Parameter	Description
topics	A list of topics to use as input for the HDFS connector. • Type: string • Default: ""
topics.dir	Top-level filesystem directory to store the data ingested from Kafka. • Type: string • Default: "topics"
locale	The locale used when partitioning with TimeBasedPartitioner. • Type: string • Default: ""
partition.duration.ms	The duration of a partition (milliseconds) used by TimeBasedPartitioner. The default value -1 means that TimeBasedPartitioner is not being used. • Type: long • Default: -1
partition.field.name	The name of the partitioning field when FieldPartitioner is used. • Type: string • Default: ""
path.format	This configuration is used to set the format of the data directories when partitioning with TimeBasedPartitioner. The format set in this configuration converts the Unix timestamp to proper directories strings. For example, if you setpath.format='year'=YYYY/'month'=MM/'day'=dd/'hour'=HH/, the data directories will have the format /year=2015/month=12/day=07/hour=15 • Type: string • Default: ""
shutdown.timeout.ms	Clean shutdown timeout. This makes sure that asynchronous Hive metastore updates are completed during connector shutdown. • Type: long • Default: 3000
timezone	The timezone to use when partitioning with TimeBasedPartitioner. • Type: string • Default: ""

Table (Continued)

Parameter	Description
filename.offset.zero.pad.width	Sets the width to the zero-pad offsets in the filesystem filenames. If the offsets are too short it provides fixed width filenames that can be ordered by simple lexicographic sorting. • Type: int • Default: 10
kerberos.ticket.renew.period.ms	The period in milliseconds to renew the Kerberos ticket. • Type: long • Default: 3600000 (milliseconds)
retry.backoff.ms	Used to notify Kafka Connect to retry delivering a message batch or performing recovery in case of transient exceptions. The retry backoff is in milliseconds. • Type: long • Default: 5000 (milliseconds)
schema.cache.size	The sized of the schema cache used in the Avro converter. • Type: int • Default: 1000
storage.class	The underlying storage layer. The default is MapR-FS. • Type: string • Default: "io.confluent.connect.hdfs.storage.HdfsStorage"

HDFS Examples
These examples provides sample code for streaming data to and from MapR File System.

Streaming Data from MapR Event Store For Apache Kafka to MapR File System

This example provides sample code for streaming data from MapR Event Store For Apache Kafka to MapR File System.

```
POST /connectors HTTP/1.1
        Host: connect.example.com
        Content-Type: application/json
        Accept: application/json

        {
        "name": "maprfs-sink-connector",
        "config": {
        "connector.class": "io.confluent.connect.hdfs.HdfsSinkConnector",
        "tasks.max": "1",
        "topics": "/kafka-connect:topic1",
        "hdfs.url": "maprfs:///",
        "flush.size": "5",
        "rotate.interval.ms": "1000"
        }
        }
```

Streaming Data from MapR Event Store For Apache Kafka to MapR File System in Parquet

This example provides sample code for streaming data from MapR Event Store For Apache Kafka to MapR File System in Parquet.

```
POST /connectors HTTP/1.1
        Host: connect.example.com
        Content-Type: application/json
        Accept: application/json

        {
        "name":"hdfs-connector-parquet",
        "config":{
        "connector.class":"io.confluent.connect.hdfs.HdfsSinkConnector",
        "tasks.max":"10",
        "topics":"/kafka-connect:topic2",
        "hdfs.url":"maprfs:///",
        "format.class":"io.confluent.connect.hdfs.parquet.ParquetFormat",
        "flush.size":"3"
        }
        }
```

Hive Integration

This topic describes how to integrate a Hive database with Kafka Connect for MapR Event Store For Apache Kafka.

Kafka Connect for MapR Event Store For Apache Kafka supports Hive integration. If a Hive database is enabled, an external Hive table is created and that can be queried via Hive shell.

Note: As of Kafka Connect 4.0.0 for MapR Event Store For Apache Kafka Hive 2.1 is supported.

The Hive table name is constructed using a topic name in the following manner:

- In the MapR Event Store For Apache Kafka topic, /stream_path:topic-name, the first forward slash (/) is removed, all other slashes are translated to underscores (_), and the colon (:) is translated to an underscore (_).
- All non-alphanumeric and non-underscore characters are removed from the string representing the Hive table name.

Renaming Topics for Hive usage

The following example shows a topic named /test-12:test1 is renamed for Hive usage.

```
$ hadoop fs -ls -R /topics
        drwxr-xr-x   - mapr mapr          1 2016-10-05 19:46 /topics/+tmp
        drwxr-xr-x   - mapr mapr          1 2016-10-05 19:46 /topics/+tmp/
test12_test1
        drwxr-xr-x   - mapr mapr          0 2016-10-05 19:50 /topics/+tmp/
test12_test1/partition=1
        drwxr-xr-x   - mapr mapr          1 2016-10-05 19:46 /topics/
test12_test1
        drwxr-xr-x   - mapr mapr          2 2016-10-05 19:50 /topics/
test12_test1/partition=1
        -rwxr-xr-x   3 mapr mapr        241 2016-10-05 19:47 /topics/
test12_test1/partition=1/test12_test1+1+0000000078+0000000080.avro
        -rwxr-xr-x   3 mapr mapr        241 2016-10-05 19:50 /topics/
test12_test1/partition=1/test12_test1+1+0000000081+0000000083.avro
```


last-updated: Apr 11, 2022

The following query and results shows the topic data in the Hive table.

```
> select * from test12_test1;
        OK
        16/10/05 20:06:59 INFO mapred.FileInputFormat: Total input paths to
process : 2
        18  data10  1
        18  data10  1
        18  data10  1
        18  data10  1
        18  data10  1
        18  data10  1
        Time taken: 0.128 seconds, Fetched: 6 row(s)
>
```

Streaming Data from MapR Event Store For Apache Kafka to the Hive database

This example provides sample code for streaming data from MapR Event Store For Apache Kafka to the Hive database.

```
POST /connectors HTTP/1.1
        Host: connect.example.com
        Content-Type: application/json
        Accept: application/json

        {
        "name":"hdfs-connector-hive",
          "config":{
          "hive.integration":"true",
          "hive.database":"db3",
            "hive.conf.dir":"/opt/mapr/hive/hive-1.2/conf",
          "hive.metastore.uris":"thrift://localhost:9083",
          "schema.compatibility":"BACKWARD",
          "connector.class":"io.confluent.connect.hdfs.HdfsSinkConnector",
          "tasks.max":"1",
          "topics":"/kafka-connect:topic3",
          "hdfs.url":"maprfs:///",
          "flush.size":"1"
          }
        }
```

REST API

The Kafka Connect REST API for MapR Event Store For Apache Kafka manages connectors.

In standalone mode, a connector request is submitted on the command line. This mode is useful for getting status information, adding and removing connectors without stopping the process, and testing and debugging.

In distributed mode, the REST API is the primary interface to the cluster. Requests can be made to any cluster member where the REST API automatically forwards requests.

Content Types

The REST API supports application/json as both the request and response entity content type. For example:

```
Accept: application/json
Content-Type: application/json
```

Status & Errors

The REST API returns standards-compliant HTTP statuses.

Note: By default, the Kafka Connect REST API for MapR Event Store For Apache Kafka service in run on port 8083.

Table

HTTP	URI	Description
GET	/connectors	Gets a list of active connectors.
POST	/connectors	Creates a new connector, returning the current connector information is successful.
GET	/connectors/(string:name)	Gets information about the connector.
GET	/connectors/(string:name)/config	Gets the configuration for the connector.
PUT	/connectors/(string:name)/config	Creates a new connector using the given configuration or updates the configuration for an existing connector.
GET	/connectors/(string:name)/tasks	Gets a list of tasks current running for the connector.
DELETE	/connectors/(string:name)/	Deletes a connector, halting all tasks and deleting its configuration.
GET	/connector-plugins	Lists the connector plugins available on this worker,
POST	/connectors/(string:name)/restart	Restarts a connector and its tasks.
GET	/connectors/(string:name)/tasks/(int:taskId)/status	Gets the status for a task.
POST	/connectors/(string:name)/tasks/(int:number of tasks)/restart	Restarts an individual task.
PUT	/connectors/(string:name)/pause	Pauses the connector and its tasks, which stops message processing until the connector is resumed.
PUT	/connectors/(string:name)/resume	Resumes a paused connector or do nothing if the connector is not paused.
GET	/connectors/(string:name)/status	Get current status of the connector, including whether it is running, failed or paused, which worker it is assigned to, error information if it has failed, and the state of all its tasks.

GET /connectors
Gets a list of active connectors.

Syntax

```
http://<host>:8083/connectors
```

Request Example

```
GET /connectors HTTP/1.1 Host: connect.example.com Accept: application/json
```

Response Example

The response JSON object is in the following form:

- **connectors** (*array*) – List of connector names.

```
HTTP/1.1 200 OK Content-Type: application/json  ["my-jdbc-source",
"my-hdfs-sink"]
```


last-updated: Apr 11, 2022

POST /connectors
Creates a new connector, returning the current connector information if successful.

Description

The POST request along with the parameters is used to create connectors in distributed mode.

The following table provides the parameters needed to create a new connector.

Table

Parameters	Description
name (*string*)	Name of the created connector
config (*map*)	Configuration parameters for the connector. See HDFS Connector on page 3925 and JDBC Connector on page 3915 for configuration options.
tasks (*array*)	List of active tasks generated by the connector.
tasks[i].connector (*string*)	Name of the connector that the task belongs to.
tasks[i].task (*int*)	Task ID within the connector.

Syntax

```
http://<host>:8083/connectors/?
name=<connector_name>&config=<config_parameters>
```

Request Example

```
POST /connectors HTTP/1.1 Host: connect.example.com Content-Type:
application/json Accept: application/json
{     "name": "hdfs-sink-connector",
        "config": {
          "connector.class":
"io.confluent.connect.hdfs.HdfsSinkConnector",
          "tasks.max": "1",
          "topics": "test-topic",
          "hdfs.url": "hdfs://fakehost:9000",
          "hadoop.conf.dir": "/opt/hadoop/conf",
          "hadoop.home": "/opt/hadoop",
          "flush.size": "100",
          "rotate.interval.ms": "1000"     }
 }
```

Response Example

The response JSON object is in the following form:

- **name** (*string*) – Name of the connector to create.
- **config** (*map*) – Configuration parameters for the connector. All values should be strings.
- **tasks** (*array*) – List of active tasks generated by the connector.

```
HTTP/1.1 201 Created Content-Type: application/json
{
      "name": "hdfs-sink-connector",
      "config":
```


last-updated: Apr 11, 2022

```
        {
          "connector.class": "io.confluent.connect.hdfs.HdfsSinkConnector",
          "tasks.max": "10",
          "topics": "test-topic",
          "hdfs.url": "hdfs://fakehost:9000",
          "hadoop.conf.dir": "/opt/hadoop/conf",
          "hadoop.home": "/opt/hadoop",
          "flush.size": "100",
          "rotate.interval.ms": "1000"
        },
      "tasks": [
          { "connector": "hdfs-sink-connector", "task": 1 },
          { "connector": "hdfs-sink-connector", "task": 2 },
          { "connector": "hdfs-sink-connector", "task": 3 }
       ]
}
```

GET /connector/(string:name)
Gets information about a specific connector.

Description

Table

Parameters	Description
name (*string*)	Name of the created connector.
config (*map*)	Configuration parameters for the connector.
tasks (*array*)	List of active tasks generated by the connector.
tasks[i].connector (*string*)	Name of the connector the task belongs to.
tasks[i].task (*int*)	Task ID within the connector.

Syntax

```
http://<host>:8083/connectors/<name>
```

Request Example

```
GET /connectors/hdfs-sink-connector HTTP/1.1 Host: connect.example.com
Accept: application/json
```

Response Example

```
HTTP/1.1 200 OK Content-Type: application/json
{     "name": "hdfs-sink-connector",
  "config": {
    "connector.class":
"io.confluent.connect.hdfs.HdfsSinkConnector",
    "tasks.max": "10",
    "topics": "test-topic",
    "hdfs.url": "hdfs://fakehost:9000",
    "hadoop.conf.dir": "/opt/hadoop/conf",
    "hadoop.home": "/opt/hadoop",
    "flush.size": "100",
    "rotate.interval.ms": "1000"     },
  "tasks": [
    { "connector": "hdfs-sink-connector", "task": 1 },
```

```
    { "connector": "hdfs-sink-connector", "task": 2 },
    { "connector": "hdfs-sink-connector", "task": 3 }     ]
 }
```

GET /connectors/(string:name)/config
Gets the configuration for the connector.

Description

Table

Parameters	Description
config (*map*)	Configuration parameters for the connector.

Syntax

```
http://<host>:8083/connectors/<string_name/config
```

Request Example

```
GET /connectors/hdfs-sink-connector/config HTTP/1.1 Host:
connect.example.com Accept: application/json
```

Response Example

```
HTTP/1.1 200 OK Content-Type: application/json
{     "connector.class": "io.confluent.connect.hdfs.HdfsSinkConnector",
      "tasks.max": "10",
      "topics": "test-topic",
      "hdfs.url": "hdfs://fakehost:9000",
      "hadoop.conf.dir": "/opt/hadoop/conf",
      "hadoop.home": "/opt/hadoop",
      "flush.size": "100",
      "rotate.interval.ms": "1000"
}
```

PUT /connectors/(string:name)/config
Creates a new connector using the given configuration or updates the configuration for an existing connector. Returns information about the connector after the change has been made.

Description

The PUT request along with the parameters is used to create connectors in distributed mode.

Table

Parameters	Description
name (*string*)	Name of the created connector.
config (*map*)	Configuration parameters for the connector. See HDFS Connector on page 3925 and JDBC Connector on page 3915 for configuration options.
tasks (*array*)	List of active tasks generated by the connector.
tasks[i].connector (*string*)	Name of the connector that the task belongs to.
tasks[i].task (*int*)	Task ID within the connector.

last-updated: Apr 11, 2022

Syntax

```
http://<host>:8083/connectors/<string_name>/config
```

Request Example

```
PUT /connectors/hdfs-sink-connector/config HTTP/1.1 Host:
connect.example.com Accept: application/json
{
    "connector.class": "io.confluent.connect.hdfs.HdfsSinkConnector",
    "tasks.max": "10",
    "topics": "test-topic",
    "hdfs.url": "hdfs://fakehost:9000",
    "hadoop.conf.dir": "/opt/hadoop/conf",
    "hadoop.home": "/opt/hadoop",
    "flush.size": "100",
    "rotate.interval.ms": "1000"
}
```

Response Example

The response JSON object is in the following form:

- config (map) – Configuration parameters for the connector. All values should be strings.

Note: In this example, the return status indicates that the connector was created. In the case of a configuration update, the status would be 200 OK.

```
HTTP/1.1 201 Created Content-Type: application/json
{
    "name": "hdfs-sink-connector",
    "config":
        {
            "connector.class":
"io.confluent.connect.hdfs.HdfsSinkConnector",
            "tasks.max": "10",
            "topics": "test-topic",
            "hdfs.url": "hdfs://fakehost:9000",
            "hadoop.conf.dir": "/opt/hadoop/conf",
            "hadoop.home": "/opt/hadoop",
            "flush.size": "100",
            "rotate.interval.ms": "1000"
            },
     "tasks": [
        { "connector": "hdfs-sink-connector", "task": 1 },
        { "connector": "hdfs-sink-connector", "task": 2 },
        { "connector": "hdfs-sink-connector", "task": 3 }     ]
}
```

GET /connectors/(string:name)/tasks
Gets a list of tasks currently running for the connector.

Description

Table

Parameters	Description
tasks (*array*)	List of active task configurations created by the connector.
tasks[i].id (*string*)	ID of the task.

Table (Continued)

Parameters	Description
tasks[i].id.connector (*string*)	Name of the connector that the task belongs to.
tasks[i].id.task (*int*)	Task ID within the connector.
tasks[i].config (*map*)	Configuration parameters for the task.

Syntax

```
http://<host>:8083/connectors/<connector_name>/tasks
```

Request Example

```
GET /connectors/hdfs-sink-connector/tasks HTTP/1.1 Host: connect.example.com
```

Response Example

```
HTTP/1.1 200 OK
[     {
          "task.class": "io.confluent.connect.hdfs.HdfsSinkTask",
          "topics": "test-topic",
          "hdfs.url": "hdfs://fakehost:9000",
          "hadoop.conf.dir": "/opt/hadoop/conf",
          "hadoop.home": "/opt/hadoop",
          "flush.size": "100",
          "rotate.interval.ms": "1000"     },
       {
          "task.class": "io.confluent.connect.hdfs.HdfsSinkTask",
          "topics": "test-topic",
          "hdfs.url": "hdfs://fakehost:9000",
          "hadoop.conf.dir": "/opt/hadoop/conf",
          "hadoop.home": "/opt/hadoop",
          "flush.size": "100",
          "rotate.interval.ms": "1000"     }
]
```

DELETE /connectors/(string:name)/
Deletes a connector by halting all tasks and deleting its configuration.

Syntax

```
http://<host>:8083/connectors/<connector_name>/
```

Request Example

```
DELETE /connectors/hdfs-sink-connector HTTP/1.1 Host: connect.example.com
```

Response Example

```
HTTP/1.1 204 No Content
```

GET /connector-plugins
Lists the connector plugins available on this worker.

Syntax

```
http://<host>:8083/connector-plugins>
```

Request Example

```
GET /connector-plugins/ HTTP/1.1
Host: connect.example.com
```

Response Example

```
HTTP/1.1 200 OK

[
    {
        "class": "io.confluent.connect.hdfs.HdfsSinkConnector"
    },
    {
        "class": "io.confluent.connect.jdbc.JdbcSourceConnector"
    }
]
```

POST /connectors/(string:name)/restart
Restarts a connector and its tasks.

Syntax

```
http://<host>:8083/connectors/<string_name>/restart
```

Request Example

```
POST /connectors/hdfs-sink-connector/restart HTTP/1.1
Host: connect.example.com
```

Response Example

Return 409 (Conflict) if rebalance is in process.

```
HTTP/1.1 200 OK
```

GET /connectors/(string:name)/tasks/(int:taskId)/status
Gets the status for a task.

Syntax

```
http://<host>:8083/connectors/<string_name>/tasks/<task ID>/status
```

Request Example

```
GET /connectors/hdfs-sink-connector/tasks/1/status HTTP/1.1
Host: connect.example.com
```

Response Example

```
HTTP/1.1 200 OK

{"state":"RUNNING","id":1,"worker_id":"192.168.86.101:8083"}
```

POST /connectors/(string:name)/tasks/(int:number of tasks)/restart
Restarts an individual task.

Syntax

```
http://<host>:8083/connectors/<string_name>/tasks/<number of tasks>/restart
```

Request Example

```
GET /connectors/hdfs-sink-connector/tasks/1/restart HTTP/1.1
Host: connect.example.com
```

Response Example

```
HTTP/1.1 200 OK
```

PUT /connectors/(string:name)/pause
Pauses the connector and its tasks, which stops message processing until the connector is resumed.

Syntax

```
http://<host>:8083/connectors/<string_name>/pause
```

Note: This call is asynchronous and the tasks will not transition to PAUSED state at the same time.

Request Example

```
PUT /connectors/hdfs-sink-connector/pause HTTP/1.1
Host: connect.example.com
```

Response Example

```
HTTP/1.1 202 Accepted
```

PUT /connectors/(string:name)/resume
Resumes a paused connector or do nothing if the connector is not paused.

Syntax

```
http://<host>:8083/connectors/<string_name>/resume
```

Note: This call is asynchronous and the tasks will not transition to RUNNING state at the same time.

last-updated: Apr 11, 2022

Request Example

```
PUT /connectors/hdfs-sink-connector/resume HTTP/1.1
Host: connect.example.com
```

Response Example

```
HTTP/1.1 202 Accepted
```

GET /connectors/(string:name)/status
Gets current status of the connector, including whether it is running, failed or paused, which worker it is assigned to, error information if it has failed, and the state of all its tasks.

Syntax

```
http://<host>:8083/connectors/<string_name>/status
```

Request Example

```
GET /connectors/hdfs-sink-connector/status HTTP/1.1
Host: connect.example.com
```

Response Example

The response JSON object includes the following:

- name (string) – Name of the connector
- connector (map) – Map containing connector status
- tasks[i] (map) – Map containing the task status

```
HTTP/1.1 200 OK

{
    "name": "hdfs-sink-connector",
    "connector": {
        "state": "RUNNING",
        "worker_id": "localhost:8083"
    },
    "tasks":
    [
        {
            "id": 0,
            "state": "RUNNING",
            "worker_id": "localhost:8083"
        },
        {
            "id": 1,
            "state": "FAILED",
            "worker_id": "localhost:8083",
            "trace":
"org.apache.kafka.common.errors.RecordTooLargeException\n"
        }
    ]
}
```

Saving Kafka Connect Configurations

Describes how Kafka Connect configurations are saved during an upgrade.

Starting in EEP 6.0.0, the configuration for a previously installed version of Kafka Connect is stored in a folder with a timestamp.

- Files are saved *and* overwritten by new configuration files:
  - when upgrading from 4.1.0 to 5.1.2.
  - when upgrading from 5.1.2 to 10.0.0.
- Files are saved only (*not* overwritten):
  - when upgrading from 5.1.2 to 5.1.2.

Example

The following example shows the list of configuration files that are saved after upgrading from EEP 6.3.1 to EEP 6.3.2:

```
ls /opt/mapr/kafka
kafka-1.1.1  kafka-1.1.1.202009040123  kafkaversion

ls /opt/mapr/kafka/kafka-1.1.1.202009040123/config/
connect-console-sink.properties    connect-distributed.properties
connect-file-source.properties  connect-standalone.properties
connect-console-source.properties  connect-file-sink.properties
connect-log4j.properties
```

Kafka Schema Registry

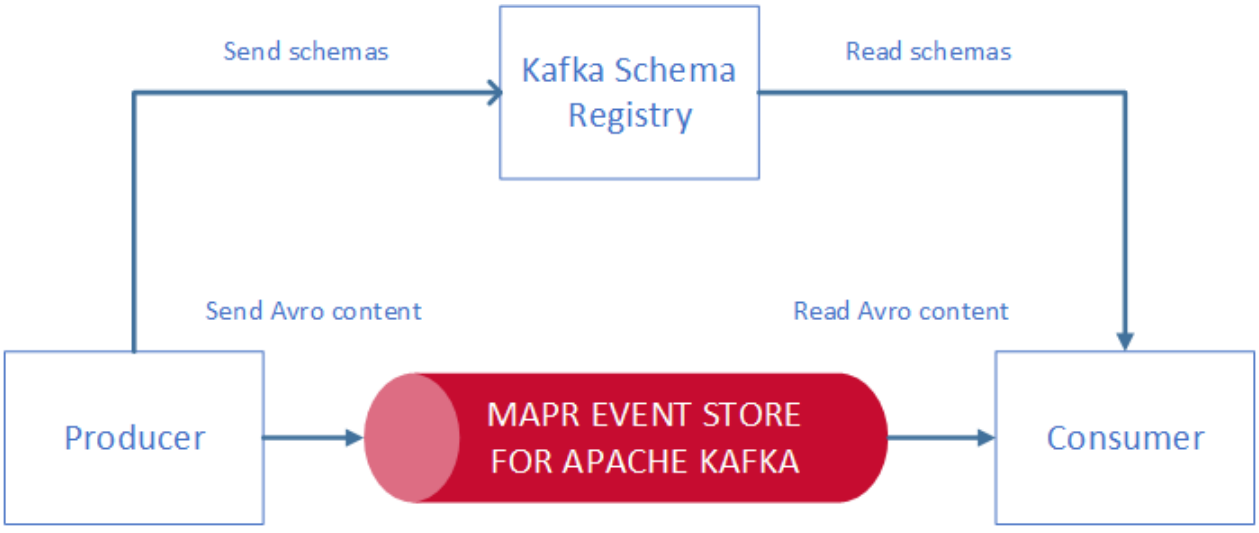
Kafka Schema Registry provides a RESTful interface for storing and retrieving schemas.

Note: This feature is presented as a *developer preview*. Developer previews are not tested for production environments, and should be used with caution.

Schema Registry can store and retrieve Avro schemas.

When you implement Kafka Schema Registry for your data schemas, they can self-evolve for compatibility with downstream consumers.

Kafka Schema Registry acts as a standalone serving layer for metadata, interacting with both the producer and consumer. It stores schemas for keys and values of records.



last-updated: Apr 11, 2022

Kafka Schema Registry enables you to perform the following tasks:

- Store a versioned history of all schemas.
- Provide multiple compatibility settings.
- Support schema evolution according to the configured compatibility settings and the expanded support for the schema format.
- Provide serializers that interface with Kafka clients and manage schema storage and retrieval for Kafka messages that are sent in one of the supported schema formats.
- Develop your own custom formats to use with this interface.

You can also perform these tasks:

- List schemas by subject and also list all versions of a subject (schema).
- Retrieve a schema by version or ID.
- Retrieve the latest version of a schema.
- Verify that a schema is compatible with a certain version.

Architecture

Kafka Schema Registry is designed to be distributed with a single master architecture. ZooKeeper coordinates the master election, based on the configuration. Kafka-coordinated master election is not currently supported.

MapR Event Store For Apache Kafka is designed to be the durable backend for schema registry, providing a write-ahead change log for the state of schema registry and the schemas it contains.

Interoperability

Kafka Schema Registry can interface with the following components:

- Kafka Client (producer, consumer APIs)
- KStreams
- KSQL
- Kafka Connect
- Kafka REST

Performance and Scalability Impact

You can improve performance by decreasing the size of the message payload. Without Kafka Schema Registry, the message payload contains the user data and the schema metadata. With the Kafka Schema Registry, the message payload contains the user data and only the schema ID that is unique for each schema.

For scalability, you can launch Kafka Schema Registry on several nodes.

For More Information

- Installing Kafka Schema Registry
- Confluent Schema Registry documentation

Building and Deploying Kafka Schema Registry
To build and deploy Kafka Schema Registry with Maven, you must first install development versions of Kafka `common` and `rest-utils` utilities.

You can run Kafka Schema Registry instances on several cluster nodes. One node is the primary node and the other ones are secondary nodes.

Kafka Schema Registry requires ZooKeeper and MapR Event Store For Apache Kafka.

The REST interface to schema registry includes a built-in Jetty server. The wrapper scripts `bin/schema-registry-start` and `bin/schema-registry-stop` are the recommended methods for starting and stopping the service.

In Apache Kafka, a schema is produced when:

- A message is produced and there is no equivalent schema in the schema registry
- A schema is created for key or value portion of the message

The associated schema subject is a “topic-key”:

- Each schema is associated with a version
- Every schema gets a globally unique ID

The consumer gets the messages’ schema using the schema ID:

```
$ curl -X GET http://localhost:8087/schemas/ids/1
       {"schema":"\"string\""}
```

You can also query a schema for a given topic using the associated schema subject. For example, for `topic1` (for either key or value), `schema`, `all`, `latest`, or `specific` schema versions can be queried using the following REST commands:

```
$ curl -X GET http://localhost:8087/subjects/topic1-value/versions
    [1]
$ curl -X GET http://localhost:8087/subjects/Kafka-value/versions/1
    {"subject":"Kafka-value","version":1,"id":1,"schema":"\"string\""}
$ curl -X GET http://localhost:8087/subjects/Kafka-value/versions/latest
    {"subject":"Kafka-value","version":1,"id":1,"schema":"\"string\""}
```

For a complete list of supported APIs, see Confluent Schema Registry API Reference.

Kafka Schema Registry Limitations
Describes the limitations related to the Kafka Schema Registry.

- Kafka REST support limitation

  Avro schema is not supported for Kafka REST.

- Security limitation

  MapR ticket based authentication, SSL, and other security mechanisms are not supported.

- MapR Installer

  You cannot use the MapR Installer to install the Kafka Schema Registry.

- Avro-console-producer and Avro-console-consumer

  You cannot use avro-console-producer and avro-console-consumer in the current release.

- SchemaParseException

  Loading data via JDBC Source connector from Hive throws an exception.

last-updated: Apr 11, 2022

- Replication Limitation

  Replication for Kafka Schema Registry is not supported.

- KSQL and Kafka Connect Limitation

  To work with Avro schemas, you need to manually modify the KSQL and Kafka Connect configuration files.

Kafka Schema Registry Use Cases
Describes typical use cases to register and query a schema and serialize and deserialize data.

Use Case 1: Registering and Querying a Schema for a Kafka Topic

While Kafka topics do not have a schema, having an external store that tracks this metadata for a given Kafka topic helps answer the following questions:

- What are the different events in any given Kafka topic?
- What can I put into a given Kafka topic?
- Do all Kafka events have a similar type of schema?
- How do I parse and use the data in a given Kafka topic?

Sample workflow code:

The following sample commands register and query a schema in a Kafka topic:

```
# Register a new version of a schema under the subject "Kafka-key"

$ curl -X POST -H "Content-Type: application/vnd.schemaregistry.v1+json" \
    --data '{"schema": "{\"type\": \"string\"}"}' \
    http://localhost:8087/subjects/Kafka-key/versions
    {"id":1}

# Register a new version of a schema under the subject "Kafka-value"

$ curl -X POST -H "Content-Type: application/vnd.schemaregistry.v1+json" \
    --data '{"schema": "{\"type\": \"string\"}"}' \
    http://localhost:8087/subjects/Kafka-value/versions
    {"id":1}

# List all subjects

$ curl -X GET http://localhost:8087/subjects
    ["Kafka-value","Kafka-key"]

# List all schema versions registered under the subject "Kafka-value"

$ curl -X GET http://localhost:8087/subjects/Kafka-value/versions
    [1]

# Fetch a schema by globally unique id 1

$ curl -X GET http://localhost:8087/schemas/ids/1
    {"schema":"\"string\""}

# Fetch version 1 of the schema registered under subject "Kafka-value"

$ curl -X GET http://localhost:8087/subjects/Kafka-value/versions/1
    {"subject":"Kafka-value","version":1,"id":1,"schema":"\"string\""}
```

```
# Fetch the most recently registered schema under subject "Kafka-value"

$ curl -X GET http://localhost:8087/subjects/Kafka-value/versions/latest
    {"subject":"Kafka-value","version":1,"id":1,"schema":"\"string\""}
```

Use Case 2: Serializing and Deserializing Data in a Kafka Topic

In addition to storing the schema metadata for a topic, Kafka Schema Registry also provides mechanisms for reading and writing data to a Kafka topic in supported formats.

You can plug the appropriate Serializer into the KafkaProducer to send messages to Kafka. A Serializer is available for each of the supported formats.

For Avro, use the `KafkaAvroSerializer`.

Currently, primitive types of `null`, `Boolean`, `Integer`, `Long`, `Float`, `Double`, `String`, `byte[]`, and the complex `IndexedRecord` type are supported.

Sending data of other types to KafkaAvroSerializer causes a SerializationException to occur. Typically, `IndexedRecord` is used for the value of the Kafka message. If used, the key of the Kafka message is often of one of the primitive types.

For example, when sending a message to a topic `t`, the schema for the key and the value is automatically registered in the Kafka Schema Registry under the subject `t-key` and `t-value`, respectively, if the compatibility test passes. The only exception is when the null type is never registered in the Kafka Schema Registry.

For consuming messages from a Kafka topic, the deserializer can be plugged analogically to the serializer.

Use Case 3: Supporting KSQL Streams or Tables in Supported formats

KSQL requires that you use the Kafka Schema Registry to create KSQL Streams or Tables in supported formats.

Schema Registry supports Avro format. Specify `VALUE_FORMAT='AVRO'` to work with topics that contain messages in Avro format.

Sample workflow code:

The following commands create and register a schema using an Avro console producer:

```
# Create a stream

$ maprcli stream create -path /sample-stream -produceperm p -consumeperm
p -topicperm p

# Use Avro console producer to create and register a schema

/sample-stream:pageviews-avro-topic

CREATE STREAM pageviews WITH (KAFKA_TOPIC='pageviews-avro-topic',
VALUE_FORMAT='AVRO');

CREATE TABLE users WITH (KAFKA_TOPIC='users-avro-topic',
VALUE_FORMAT='AVRO',
        KEY='userid');
```

Managing Kafka Schema Registry
Describes how to manage the internal stream for Kafka Schema Registry.

Schema Registry Internal Stream

By default, the `schema-registry-internal-stream` topic stores the schemas.
The `schema-registry-internal-stream` topic is located `in /apps/schema-registry/schema-registry-internal-stream`. The `kafkastore.stream` property in the `SR_CONF_DIR/schema-registry.properties` file sets the internal stream topic.
By default, the `kafkastore.stream` property is set to `/apps/schema-registry/schema-registry-internal-stream`.

The internal stream is automatically created if it does not already exist. If the internal stream already exists and has the same permissions as the default, the system returns a warning. If `kafkastore.stream` is set to a value other than the default, the system returns an error if the stream does not exist. You must explicitly set permissions on the stream you designate as the Schema Registry internal stream.

You can use Warden and `/opt/mapr/server/configure.sh` to manage and configure Kafka Schema Registry. Any time you change Schema Registry configurations, run `configure.sh` for changes to take effect.

Secure Cluster Default Permissions on the Internal Stream

By default, the internal stream is readable by all the users and writable only by the cluster administrator. For secure clusters, the default permissions are:

- `-produceperm u:$CLUSTER_ADMIN`
- `-consumeperm p`
- `-topicperm u:$CLUSTER_ADMIN`

The cluster admin is typically `mapr` or the value set for MAPR_USER.

Insecure Cluster Default Permissions on the Internal Stream

The stream has both read and write permissions for all users. The default permissions can be modified by means of the `maprcli stream` API. For insecure clusters, the default permissions are:

- `-produceperm p`
- `-consumeperm p`
- `-topicperm p`

Log Compaction for the Schema Registry Internal Stream

Log compaction is the process of purging messages previously published to a topic partition while retaining the latest version. Log compaction for the Schema Registry internal stream is enabled by default.

Important: Log compaction requires a gateway on the same cluster as the Schema Registry internal stream. Also, TTL should be disabled because it can interfere with log compaction.

Installing a Gateway

Run the command appropriate for your system:

- RedHat/CentOS

```
yum install mapr-gateway
```

- Ubuntu

```
apt-get install mapr-gateway
```

- SLES

```
zypper install mapr-gateway
```

Related concepts

Log Compaction on page 641
Log compaction purges previous, older messages that were published to a topic-partition and retains the latest version of the record.

Related information

Preparing Clusters for Log Compaction on page 1141
Describes how to prepare your environment so you can use log compaction.

Enabling High Availability for Kafka Schema Registry

You can enable high availability by installing multiple instances of schema registry on the same cluster.

One node in the cluster acts as the primary instance. Only the primary instance can publish writes to the underlying Kafka log. The primary node can also manage read requests.

All other secondary nodes manage only read requests. These nodes serve registration requests indirectly by forwarding them to the current primary node and returning any response supplied by the primary node.

1. Install schema registry on several nodes. For installation instructions, see Installing Kafka Schema Registry.
2. Configure high availability for the ensemble of ZooKeeper nodes. You can find the steps for this in the open source documentation, section Clustered (Multi-Server) Setup.
3. Modify the schema registry configuration file on each node to specify the `kafkastore.connection.url` property with the ZooKeeper nodes that form the HA ensemble.
4. Run the `configure.sh -R` command on each schema registry node.

Schema Registry Demos

Provides demonstrations for using Kafka Schema Registry to store and retrieve schemas.

The following topics demonstrate how to use Kafka Schema Registry to store and retrieve schemas in the supported formats:

Kafka Schema Registry Demo for Avro
Implements a Kafka Schema Registry demo example that stores and retrieves Avro schemas.

Maven Dependencies

Add the following repositories to the POM file to resolve Confluent and MapR dependencies:

```
<repositories>
   <repository>
       <id>confluent</id>
       <url>http://packages.confluent.io/maven/</url>
   </repository>
   <repository>
       <id>mapr-maven</id>
       <url>https://repository.mapr.com/maven/</url>
       <releases><enabled>true</enabled></releases>
       <snapshots><enabled>true</enabled></snapshots>
   </repository>
</repositories>
```

The following dependencies are needed for Avro and Kafka:

```
<dependency>
   <groupId>org.apache.avro</groupId>
   <artifactId>avro</artifactId>
   <version>1.9.2</version>
</dependency>
<dependency>
   <groupId>io.confluent</groupId>
   <artifactId>kafka-avro-serializer</artifactId>
   <version>5.1.2.0-mapr-700</version>
</dependency>
```

Creating a Java class that corresponds to the Avro schema

Add the following plugins to the `pom.xml` file:

- Plugin to build code:

```
<plugin>
  <groupId>org.apache.avro</groupId>
  <artifactId>avro-maven-plugin</artifactId>
  <version>1.8.2</version>
  <executions>
    <execution>
       <phase>generate-sources</phase>
       <goals>
           <goal>schema</goal>
           <goal>protocol</goal>
           <goal>idl-protocol</goal>
       </goals>
       <configuration>
          <sourceDirectory>${project.basedir}/src/main/resources/avro/</
sourceDirectory>
          <stringType>String</stringType>
          <createSetters>false</createSetters>
          <enableDecimalLogicalType>true</enableDecimalLogicalType>
          <fieldVisibility>private</fieldVisibility>
       </configuration>
     </execution>
   </executions>
</plugin>
```

- Plugin to force the discovery of the generated classes:

```
<plugin>
    <groupId>org.codehaus.mojo</groupId>
    <artifactId>build-helper-maven-plugin</artifactId>
    <version>3.0.0</version>
    <executions>
       <execution>
         <id>add-source</id>
         <phase>generate-sources</phase>
         <goals>
            <goal>add-source</goal>
         </goals>
         <configuration>
           <sources>
            <source>target/generated-sources/avro</source>
           </sources>
         </configuration>
       </execution>
     </executions>
</plugin>
```

- Create a file with filename extension `.avsc` in the `scr/main/resources` directory.

An example of an Avro schema is as follows:

```
{
  "namespace": "com.example",
  "type": "record",
  "name": "Employee",
  "doc" : "Represents an Employee at a company",
  "fields": [
      {"name": "firstName", "type": "string", "doc": "The persons given
name"},
```


last-updated: Apr 11, 2022

```
      {"name": "lastName", "type": "string"},
      {"name": "age",  "type": "int", "default": -1},
      {"name": "emails", "default":[], "type":{"type": "array", "items":
"string"}},
      {"name": "phoneNumber",  "type": "string"}
   ]
}
```

The `Employee.class` Java class is auto-generated in the `target/classes/com/example` directory after executing the following commands:

```
$ mvn clean
$ mvn package
```

You can use this class in your program after performing these steps.

Creating an Avro Producer

1. Import the following properties for the Kafka Producer:

```
import com.example.Employee;
import io.confluent.kafka.serializers.KafkaAvroSerializer;
import io.confluent.kafka.serializers.KafkaAvroSerializerConfig;
import org.apache.kafka.clients.producer.KafkaProducer;
import org.apache.kafka.clients.producer.ProducerConfig;
import org.apache.kafka.clients.producer.ProducerRecord;
import org.apache.kafka.common.serialization.IntegerSerializer;

import java.util.ArrayList;
import java.util.List;
import java.util.Properties;
```

2. Configure the following properties for Event Data Streams:

```
Properties properties = new Properties();
properties.setProperty(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG,
        IntegerSerializer.class.getName());

// Configure the KafkaAvroSerializer.
properties.setProperty(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG,
        KafkaAvroSerializer.class.getName());

//Schema registry location.
properties.setProperty(KafkaAvroSerializerConfig.SCHEMA_REGISTRY_URL_CONF
IG,
        "http://localhost:8087");

KafkaProducer<Integer, Employee> producer =
        new KafkaProducer<>(properties);
```

3. The following code sends n different objects of class `Employee.java` to the topic `avro_example` in the `/sample-stream` stream:

```
String topic = "/sample-stream:avro_example";
Employee employee;

for (int i = 0; i < n; i++) {

    List<String> emails = new ArrayList<>();
    for (int j = 0; j < i; j++) {
        emails.add("john" + j + ".doe" + i + "@mail.com");
    }

    employee = Employee.newBuilder()
            .setFirstName("John" + i)
            .setLastName("Doe")
            .setAge(i + 5)
            .setEmails(emails)
            .setPhoneNumber("+1-202-555-" + i + i + i + i)
            .build();

    ProducerRecord<Integer, Employee> record =
        new ProducerRecord(topic, i, employee);

    producer.send(record, (recordMetadata, e) -> {
        if (e == null) {
            System.out.println("Success!" );
            System.out.println(recordMetadata.toString());
        } else {
            e.printStackTrace();
        }
    });
}

producer.flush();
producer.close();
```

Creating an Avro Consumer

1. Import the following properties for the Kafka Consumer:

```
import com.example.Employee;
import io.confluent.kafka.serializers.KafkaAvroDeserializer;
import io.confluent.kafka.serializers.KafkaAvroDeserializerConfig;
import org.apache.kafka.clients.consumer.ConsumerConfig;
import org.apache.kafka.clients.consumer.ConsumerRecords;
import org.apache.kafka.clients.consumer.KafkaConsumer;
import org.apache.kafka.common.serialization.IntegerDeserializer;

import java.util.Collections;
import java.util.Properties;
```


last-updated: Apr 11, 2022

2. The properties to configure are similar to the Kafka producer, only Deserializers must be used instead of Serializers. Add one more property called `KafkaAvroDeserializerConfig.SPECIFIC_AVRO_READER_CONFIG`:

```
Properties properties = new Properties();
properties.put(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG,
        IntegerDeserializer.class.getName());

//Use Kafka Avro Deserializer.
properties.put(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG,
        KafkaAvroDeserializer.class.getName());

//Use Specific Record or else you get Avro GenericRecord.
properties.put(KafkaAvroDeserializerConfig.SPECIFIC_AVRO_READER_CONFIG,
"true");

//Schema registry location.
properties.put(KafkaAvroDeserializerConfig.SCHEMA_REGISTRY_URL_CONFIG,
        "http://localhost:8087");

KafkaConsumer<Integer, Employee> consumer =
        new KafkaConsumer<>(properties);
```

3. The following code reads objects of the `Employee.java` class from the `avro_example` topic in the `/sample-stream` stream:

```
String topic = "/sample-stream:avro_example";
consumer.subscribe(Collections.singletonList(topic));

try {
    while (true) {
        ConsumerRecords<Integer, Employee> records =
                consumer.poll(Duration.ofMillis(100));
        records.forEach(record -> {

            Employee employeeRecord = record.value();

            System.out.printf("%s %d %d %s \n", record.topic(),
                    record.partition(), record.offset(), employeeRecord);
        });
    }
} finally {
    consumer.close();
}
```

Structured Streaming in Spark
Starting in EEP 5.0.0, structured streaming is supported in Spark.

Related Links

Spark streaming is integrated with MapR Event Store For Apache Kafka for Apache Kafka.

- MapR Event Store For Apache Kafka Clients and Tools

Prerequisites for Using Structured Streaming in Spark
To deploy a structured streaming application in Spark, you must create a MapR Streams topic and install a Kafka client on all nodes in your cluster.

Creating a MapR Streams Topic

- Create a MapR Streams topic consisting of the stream path and topic name separated by a colon (:); for example, `/test_stream:topic1`.

Installing a Kafka Client

- Install a kafka-client on all nodes of your cluster or copy the `kafka-clients.jar` file from `/opt/mapr/lib/kafka-clients-<version>mapr<release>.jar` to `/opt/mapr/spark/spark-<version>/jars/`.

Using Structured Streaming to Create a Word Count Application

The example in this section creates a dataset representing a stream of input lines from Kafka and prints out a running word count of the input lines to the console.

Using Apache Kafka

Scala

```
val spark = SparkSession
       .builder
       .appName("StructuredKafkaWordCo
unt")
       .getOrCreate()

import spark.implicits._
//Create a DataSet representing the
stream of input lines from Kafka
val lines = spark
       .readStream
       .format("kafka")
       .option("kafka.bootstrap.server
s", bootstrapServers)
       .option(subscribeType, topics)
       .load()
       .selectExpr("CAST(value AS
STRING)")
       .as[String]
//Generate a running word count
val wordCounts =
lines.flatMap(_.split("
")).groupBy("value").count()
//Run the query that prints the
running counts to the console
val query = wordCounts.writeStream
       .outputMode("complete")
       .format("console")
       .option("checkpointLocation",
checkpointLocation)
       .start()

query.awaitTermination()
```

Java

```
SparkSession spark = SparkSession
             .builder()
             .appName("JavaStructured
KafkaWordCount")
             .getOrCreate();
//Create a DataSet representing the
stream of input lines from Kafka
Dataset<String> lines = spark
             .readStream()
             .format("kafka")
```


last-updated: Apr 11, 2022

```
              .option("kafka.bootstrap
.servers", bootstrapServers)
              .option(subscribeType,
topics)
              .load()
              .selectExpr("CAST(value
AS STRING)")
              .as(Encoders.STRING());
//Generate a running word count
Dataset<Row> wordCounts =
lines.flatMap(
(FlatMapFunction<String, String>)
x -> Arrays.asList(x.split("
")).iterator(),
Encoders.STRING()).groupBy("value").co
unt();

//Run the query that prints the
running counts to the console
StreamingQuery query =
wordCounts.writeStream()
              .outputMode("complete")
              .format("console")
              .start();

query.awaitTermination();
```

Python

```
spark = SparkSession\
          .builder\
          .appName("StructuredKafkaWor
dCount")\
          .getOrCreate()

#Create a DataSet representing the
stream of input lines from Kafka
lines = spark\
          .readStream\
          .format("kafka")\
          .option("kafka.bootstrap.ser
vers", bootstrapServers)\
          .option(subscribeType,
topics)\
          .load()\
          .selectExpr("CAST(value AS
STRING)")

#Split the lines into words
words = lines.select(
#explode turns each item in an array
into a separate row
explode(
        split(lines.value, ' ')
       ).alias('word')
     )

#Generate a running word count
wordCounts =
words.groupBy('word').count()

#Run the query that prints the
running counts to the console
```

```
query = wordCounts\
           .writeStream\
           .outputMode('complete')\
           .format('console')\
           .start()

query.awaitTermination()
```

Using MapR Event Store for Apache Kafka

For MapR Event Store, the topic name consists of the stream name and topic, and the bootstrap servers are not used. For example:

```
var topic: String = "/user/mapr/stream:reviews"
val df1 = spark.readStream.format("kafka").option("kafka.bootstrap.servers",
   "maprdemo:9092").option("subscribe", topic).option("group.id",
   "testgroup").option("startingOffsets",
"earliest").option("failOnDataLoss",
          false).option("maxOffsetsPerTrigger", 1000).load()
```

Writing a Structured Spark Stream to MapR Database JSON Table

The example in this section writes a structured stream in Spark to MapR Database JSON table.

To write a structured Spark stream to MapR Database JSON table, use `MapRDBSourceConfig.Format` for Java and Scala and `com.mapr.db.spark.streaming` for Python to format the `tablePath`, `idFieldPath`, `createTable`, `bulkMode`, and `sampleSize` parameters.

Scala

```
import
com.mapr.db.spark.streaming.MapRDBSour
ceConfig
import org.apache.spark.sql.streaming.
{DataStreamReader, DataStreamWriter}
import org.apache.spark.sql.
{DataFrame, Row, SparkSession}

def dataStreamWriter(spark:
SparkSession, df: DataFrame):
DataStreamWriter[Row] = {
import spark.implicits._

df.select($"value" as "_id")
  .writeStream
  .format(MapRDBSourceConfig.Format)
  .option(MapRDBSourceConfig.TablePath
Option, "/table/path")
  .option(MapRDBSourceConfig.IdFieldPa
thOption, "value")
  .option(MapRDBSourceConfig.CreateTab
leOption, true)
  .option(MapRDBSourceConfig.BulkModeO
ption, true)
  .option(MapRDBSourceConfig.SampleSiz
eOption, 1000)
  .outputMode("append")
}
```

Java

```
import
com.mapr.db.spark.streaming.MapRDBSour
ceConfig;
```

```
import org.apache.spark.sql.Dataset;
import org.apache.spark.sql.Row;
import
org.apache.spark.sql.SparkSession;
import
org.apache.spark.sql.streaming.DataStr
eamReader;
import
org.apache.spark.sql.streaming.DataStr
eamWriter;
import
org.apache.spark.sql.streaming.Streami
ngQueryException;

DataStreamWriter<Row>
dataStreamWriter(Dataset<Row> df) {
   return df.selectExpr("CAST(value
AS STRING) as _id")
            .writeStream()
            .format(MapRDBSourceConfig
.Format())
            .option(MapRDBSourceConfig
.TablePathOption(), "/table/path")
            .option(MapRDBSourceConfig
.IdFieldPathOption(), "value")
            .option(MapRDBSourceConfig
.CreateTableOption(), true)
            .option(MapRDBSourceConfig
.BulkModeOption(), true)
            .option(MapRDBSourceConfig
.SampleSizeOption(), 1000)
            .outputMode("append");
 }
```

Python

```
from pyspark.sql import *

def data_stream_writer_func(df,
checkpoint_dir, table_path):
  return df.selectExpr("CAST(value AS
STRING) as _id") \
           .writeStream \
           .format("com.mapr.db.spark.
streaming") \
           .option("checkpointLocation
", checkpoint_dir) \
           .option("tablePath",
table_path) \
           .option("idFieldPath",
"value") \
           .option("createTable",
True) \
           .option("bulkMode", True) \
           .option("sampleSize", 1000)
```

Writing a Spark Stream Word Count Application to MapR Database

The example in this section writes a Spark stream word count application to MapR Database.

Scala

```
val spark = SparkSession
      .builder
```

```
      .appName("StructuredKafkaWordCou
nt")
      .getOrCreate()

import spark.implicits._
//Create a DataSet representing the
stream of input lines from Kafka
val lines = spark
      .readStream
      .format("kafka")
      .option("kafka.bootstrap.servers
", bootstrapServers)
      .option(subscribeType, topics)
      .load()
      .selectExpr("CAST(value AS
STRING)")
      .as[String]

//Generate a running word count
val wordCounts =
lines.flatMap(_.split("
")).groupBy("value").count()

//Run the query that saves the result
to MapR-DB
val query = wordCounts.writeStream
      .format(MapRDBSourceConfig.Forma
t)
      .option(MapRDBSourceConfig.Table
PathOption, resultTable)
      .option(MapRDBSourceConfig.Creat
eTableOption, true)
      .option(MapRDBSourceConfig.IdFie
ldPathOption, "value")
      .outputMode("complete")
      .start()

query.awaitTermination()
```

Java

```
SparkSession spark = SparkSession
           .builder()
           .appName("JavaStructuredKaf
kaWordCount")
           .getOrCreate();

//Create a DataSet representing the
stream of input lines from Kafka
Dataset<String> lines = spark
            .readStream()
            .format("kafka")
            .option("kafka.bootstrap.s
ervers", bootstrapServers)
            .option(subscribeType,
topics)
            .load()
            .selectExpr("CAST(value
AS STRING)")
            .as(Encoders.STRING());

//Generate a running word count
Dataset<Row> wordCounts =
```


last-updated: Apr 11, 2022

```
lines.flatMap(
(FlatMapFunction<String, String>)
x -> Arrays.asList(x.split("
")).iterator(),
Encoders.STRING()).groupBy("value").co
unt();

//Run the query that saves the result
to MapR-DB
StreamingQuery query =
wordCounts.writeStream()
            .format(MapRDBSourceConfig
.Format())
            .option(MapRDBSourceConfig
.TablePathOption(), resultTable)
            .option(MapRDBSourceConfig
.CreateTableOption(), true)
            .option(MapRDBSourceConfig
.IdFieldPathOption(), "value")
            .outputMode("complete");
            .start();

query.awaitTermination();
```

Python

```
spark = SparkSession\
           .builder\
           .appName("StructuredKafkaWo
rdCount")\
           .getOrCreate()

#Create a DataSet representing the
stream of input lines from Kafka
lines = spark\
          .readStream\
          .format("kafka")\
          .option("kafka.bootstrap.ser
vers", bootstrapServers)\
          .option(subscribeType,
topics)\
          .load()\
          .selectExpr("CAST(value AS
STRING)")

#Split the lines into words
words = lines.select(
#Explode turns each item in an array
into a separate row
explode(
       split(lines.value, ' ')
       ).alias('word')
)

#Generate a running word count
wordCounts =
words.groupBy('word').count()

#Run the query that saves the result
to MapR-DB
query = wordCounts\
              .writeStream\
              .format("com.mapr.db.spa
```

```
rk.streaming") \
                .option("tablePath",
table_path) \
                .option("createTable",
True) \
                .option("idFieldPath",
"value") \
                .outputMode('complete')\
                .start()

query.awaitTermination()
```

S3 Gateway

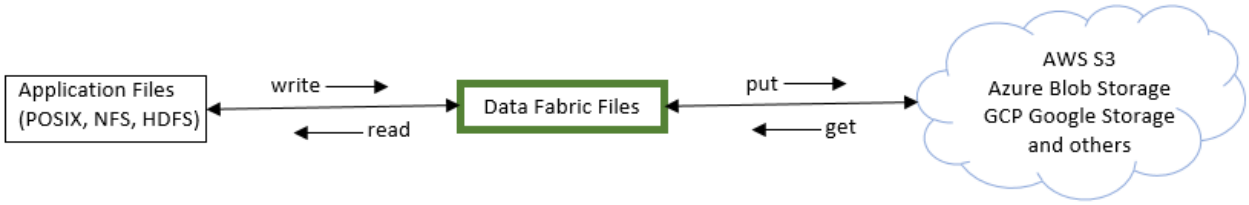
The S3 gateway is a service that provides an S3-compatible interface to expose data in MapR Data Platform as objects. The S3 gateway manages all inbound S3 API requests to put data into and get data out of cloud storage.

Notice: The S3 gateway is included in EEP 6.0.0 - EEP 8.0.0 repositories.

The S3 gateway can expose data generated through multiple data protocols, such as NFS, POSIX, S3, and HDFS as objects.

The MapR Data Platform filesystem stores an object as a file. The file can be of any data type, but must have a unique name as part of the S3 API call. Files can be accessed by the S3 API requests and by MapR Data Platform file interfaces. Data objects are grouped into a logical container called a bucket. A bucket correlates with a folder in MapR Data Platform.

The following image shows the flow of data to and from MapR Data Platform:



You can use the S3 API to `create`, `list`, or `delete` a bucket. You can also use it to `get`, `put`, `list`, or `delete` a data object within a bucket. The first time the S3 API accesses a file, it generates metadata. Metadata is required to fulfill the API requests.

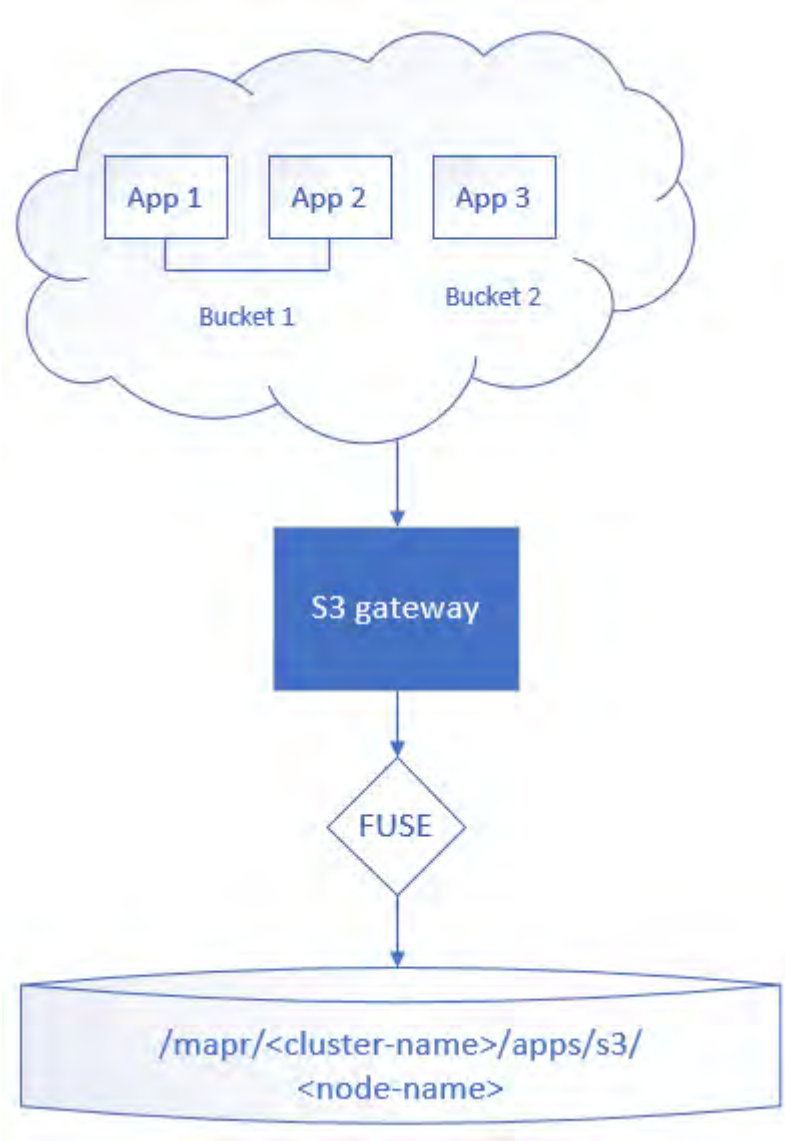
The S3 gateway also supports object notification through MapR Event Store For Apache Kafka. See Using the MapR Event Store For Apache Kafka for S3 Bucket Event Notifications on page 3965.

S3 Deployment Mode

The S3 gateway only supports the *Amazon S3 standalone deployment mode* because each instance of the S3 gateway can only interact with one bucket or set of buckets at a time.

When you use the S3 API in standalone mode, each S3 gateway instance must have its own back-end directory in the MapR Data Platform filesystem. You can either map a volume mount point to the directory or use the directory path itself. An S3 instance exclusively uses the allocated directory or volume in the filesystem to serve an exclusive set of buckets.

The following deployment scenario shows one S3 gateway per cluster that supports multiple applications and multiple buckets with bucket sharing:



This scenario is useful if you want an application to access multiple buckets without knowing about bucket locations beforehand. The single S3 gateway instance serves all requests without having to partition any buckets.

If you need to migrate buckets to another S3 instance, you can move or copy the buckets to another directory or volume. See AWS CLI. If a bucket does not exist, an application can create a bucket through any S3 gateway; however, the bucket created will only be served through the one gateway.

Authorization to Access Data

By default, the S3 gateway provides a two-tier authorization model that starts with an S3 bucket policies check at the S3 REST API level, followed by a file permissions check on the MapR Data Platform.

When an S3 gateway instance receives a request from a tenant to access a bucket or object, it first checks for bucket policies that reference that particular tenant. If the tenant does not have access via the bucket policy, the request fails and no other checks are performed.

If the tenant has access via the bucket policy, the filesystem performs the next check using the mapped UID and GID credentials for the tenant.

last-updated: Apr 11, 2022

Configuring S3 Gateway on page 3961 describes how to modify the type of authorization, configure tenants and credentials, and secure data. To see a configuration example, see Multi-Tiered Authorization Example.

Related Links

- For information about installation and configuration, see Installing S3 Gateway on page 204
- For a complete list of supported APIs, see Amazon S3 documentation
- For release-specific information, see S3 Gateway Release Notes on page 6260
- For information about object tiering (archiving files in the cloud), see Data Tiering on page 469.

Configuring S3 Gateway
Describes how to configure the S3 gateway.

Notice: The S3 gateway is included in EEP 6.0.0 - EEP 8.0.0 repositories.

To configure the S3 Gateway, add S3 gateway tenants (users) and credentials to the `tenants.json` file and change the deployment mode in the `minio.json` file. After you update and save the files, restart the `objectstore service.`

Important: If a bucket does not have custom policies applied and `mixed` deployment mode is set, you must remove the policy file for the bucket and restart the `objectstore service` to avoid permissions issues any time you:

- change the owner of the bucket
- add new tenants that are also bucket owners

The bucket policy file is located in `FS_PATH/.minio.sys/buckets/BUCKET_NAME/policy.json.` To restart the objectstore service, see Restarting the S3 Gateway Service on page 3964.

Distributed Mode

S3 gateway 2.1.0 and later supports distributed mode. S3 gateway 2.0.x does not support distributed mode.

Note the following points related to S3 gateway 2.1.0 and later in distributed mode:

- In distributed mode, instances of S3 gateway share user and policy information only, as well as locks between nodes.
- Distributed mode does not include proxy or load balancing in front of instances. You must configure and install proxy and load balancing manually.
- Distributed mode only works for S3 mode.
- Three or more nodes is recommended for S3 gateway in distributed mode. If only one node is active, the node will be in read-only mode.

To enable distributed mode for S3 gateway 2.1.0 and later:

1. Configure S3 gateway 2.1.0 or later on three or more nodes.
2. Disable caching in Fuse clients.

last-updated: Apr 11, 2022

3. On each S3 gateway node, update the `/opt/mapr/objectstore-client/objectstore-client-<version>/conf/minio.json` file with the `accessKey`, `secretKey`, and `ldap`. The value for each of these properties must be identical across all the `minio.json` files.

4. On all S3 gateway 2.1.0 or later nodes, verify that the `distributedHosts` property contains the list of nodes with mount paths, for example:

   ```
   "distributedHosts": "http(s)://HOST1:PORT1/MOUNT_PATH1 http(s)://
   HOST2:PORT2/MOUNT_PATH2"
   ```

   Note that the mount path on all nodes should target the same folder in the filesystem.

5. If the cluster is secure with self-signed certificates, copy the public certificates for each host to the `/opt/mapr/objectstore-client/objectstore-client-<version>/conf/certs/CAs` directory. To ignore validation of the certificates, set `"insecureSkipVerify": true`.

About Credentials

In the S3 world, credentials represent the application and not the identity of the end user. The application layer is responsible for end-user verification. The S3 administrator must assign S3 credentials for the application or set of applications and optionally, map those S3 credentials to a data-fabric identity.

As defined in the Amazon S3 documentation, the S3 REST API uses a “key” and “secret” (in a REST-like manner) as credentials to authenticate to the underlying object storage and authorize access to data.

S3 gateway supports a multi-tenant scenario in which the S3 administrator can configure one or more credentials with the appropriate data-fabric credential mapping. The S3 administrator can assign credentials to a user and, optionally, map them to a data-fabric identity.

For an overview of tenants and multi-tenancy, see Multitenancy on File System on page 488.

Add Tenants and Credentials in the tenants.json File

The `tenants.json` file describes the tenants configuration. This file consists of a JSON object with two keys: `credentials` and `tenants`. The `credentials` key relates more to S3 gateway authorization, whereas the `tenants` key relates to the users in the system.

The `credentials` key contains an array of objects with the following fields:

- `accessKey` - S3 format access key
- `secretKey` - S3 format secret key
- `Tenant` - Internal tenant name, used to link an access key to an operating system user

The `tenants` key contains an array of objects with the following fields:

- `uid` - Operating system user ID for file impersonation
- `gid` - Operating system group ID for file impersonation
- `name` - Internal tenant name, used to link an access key to an operating systems user

You add or update tenants (object storage users) and credentials in the `/opt/mapr/objectstore-client/objectstore-client-<version>/conf/tenants.json` file. The `/opt/mapr/objectstore-client/objectstore-client-<version>/conf` directory also contains a `tenants-sample.json` file that you can use for reference.

Example of a `tenants.json` configuration:

```
{
    "tenants": [
        {
            "name": "tenant1",
            "uid": 5001,
            "gid": 5001
        }
    ],

    "credentials": [
        {
            "accessKey": "accessKey1",
            "secretKey": "secretKey1",
            "tenant": "tenant1"
        }
    ]
}
```

Set the Deployment Mode (Authorization Type) in the minio.json File

The deployment mode sets the type of authorization being used. The default deployment mode is `mixed`. You can edit the `deploymentMode` parameter in the `/opt/mapr/objectstore-client/objectstore-client-<version>/conf/minio.json` file to change the deployment mode setting.

The S3 gateway supports the following deployment modes:

Option	Deployment Mode	Description
1	fs_only	Enforced by MapR File System file permissions only. The S3 bucket policy is disabled. Access is granted based on the bucket owner UID and GID, not read/write/execute permissions. • Configuration maps the application key or secret to a MapR ID. • S3 policy is not used and the policy check is skipped. • MapR file security validates inbound mapped UID and GID to authorize read or write file permissions.
2	mixed	The default setting. Uses a mix of the MapR File System and S3 bucket policy security enforcement. • Configuration maps the application key or secret to a MapR ID. • S3 bucket policy and MapR file security are enforced. Specifically, the two authorization models. • The access key is ignored and the filesystem permissions determine access.

last-updated: Apr 11, 2022

Option	Deployment Mode	Description
3	s3_only	Enforced by s3 bucket policy only. • No configuration to map the application key or secret to a MapR ID. • All the files in the filesystem are owned by the user that runs the S3 gateway processes (typically the `mapr` user). • Access is controlled by the secret access key and the key ID.

Secure the Data

To prevent unauthorized access to data, secure the data based on the configured deployment mode:

- For `mixed` mode, set folder permissions and upload the S3 policies.
- For `fs_only` mode, set folder permissions to allow authorized users access to folders.
- For `s3_only` mode, upload the S3 policies.

Set the Path to Mount the Filesystem

If you are using `fs_only` or `mixed` mode, set the path to mount the filesystem.

- **Server node** - In the `minio.json` file, set the `fsPath` parameter to the filesystem mount path. By default, the path is set to `/mapr/<clustername>/apps/s3/<nodename>`. If you want to share existing folders with users, set the `fsPath` parameter to point to the directory with the folders that you want to share. All the folders in the directory (to which `fsPath` points) are accessible as buckets.
- **Edge node** - If you completed the Edge Node Installation steps, you may have already set the path to mount the filesystem when you ran the objectstore `configure.sh` script with the `--path` parameter. No action is required.

For more information about deployment modes, see S3 Gateway on page 3959.

Restarting the S3 Gateway Service

Start, restart, or stop the S3 gateway by using the command appropriate for the node:

Server Node

```
/opt/mapr/bin/maprcli node services -name objectstore -nodes
<node_name> -action [ start |
          restart | stop ]
```

Edge Node

```
sudo /opt/mapr/objectstore-client/objectstore-client-<version>/bin/
objectstore.sh [ start |
        stop ]
```

Multiple-Tiered Authorization Example

The example shows a multi-tiered configuration where a bucket policy and MapR ACEs are both used for authorization.

last-updated: Apr 11, 2022

 **Notice:** The S3 gateway is included in EEP 6.0.0 - EEP 8.0.0 repositories.

Tenant A creates bucket B1. The system applies on the bucket, the default bucket policy of `read/write/execute` permission to only the owner and no access to other tenants. These permissions are UNIX permissions: `0700`.

Tenant A calls the REST API for bucket policy resulting in the following policy on bucket B1:

```
"Version": "2012-10-17",
"Id": "ExamplePolicy01",
"Statement": [
   {
     "Sid": "ExampleStatement01",
     "Effect": "Allow",
     "Principal": {
           "AWS": "B"
       },
      "Action": [
        "s3:GetObject",
      ],
      "Resource": [
        "arn:aws:s3:::B1/*",
      ]
    }
  ]
}
```

This policy is processed by the S3 gateway and allows access to tenant B. The policy indicates that the *ACE* is set up for the objects, as `-readdir u:B | u:A` and is applied to the bucket directory, while the *ACE* expression `-readfile u:B | u:A` is applied to all existing files. In mixed mode, tenant B has access.

Using the MapR Event Store For Apache Kafka for S3 Bucket Event Notifications
You can use the MapR Event Store For Apache Kafka to receive event notifications for buckets. For example, when an object is created in a bucket. Install the event store, create a stream and topic, and then enable notifications in `/opt/mapr/objectstore-client/objectstore-client-<version>/conf/config.json`. You can then configure S3 event notifications through the S3 REST API, and use the AWS CLI to manage the notifications.

 **Notice:** The S3 gateway is included in EEP 6.0.0 - EEP 8.0.0 repositories.

To enable bucket-notification functionality, complete the following steps:

1. Install the `mapr-kafka*` package:

   ```
   zypper|yum|apt-get install mapr-kafka
   ```

2. Create a stream and topic:

   ```
   maprcli stream create -path /path/to/stream
   maprcli stream topic create -path /path/to/stream -topic nameOfTopic
   ```

3. Set up `/opt/mapr/objectstore-client/objectstore-client-<version>/conf/config.json` in block notifications:

```
"notify": {
  "kafka": {
    "notifyBlockName": {
      "enable": true,
      "brokers": [""],
      "topic": "/path/to/stream:nameOfTopic"
      }
   }
}
```

4. Restart the S3 gateway.

Install and configure the AWS CLI to manage notifications from the command line. See Install and Configure the AWS CLI for installation and configuration details.

The following examples demonstrate some useful commands:

- To read a topic:

```
./kafka/kafka-<version>/bin/kafka-console-consumer.sh --topic /path/to/
stream:nameOfTopic --bootstrap-server some:9092 --from-beginning
```

- To add notifications to a bucket:

```
aws s3api --endpoint-url http://node1.cluster.com:9000
put-bucket-notification-configuration  --notification-configuration
file://fileWithNotificationRules.json --bucket bucketName
```

- To get notifications for a bucket:

```
aws s3api --endpoint-url https://node1.cluster.com:9000
get-bucket-notification-configuration  --bucket bucketName
```

Configure S3 Event Notifications

You can configure event notifications in the S3 REST API if you want to receive notifications about specific events that happen in a bucket. To enable notifications, add a notification configuration that identifies the events you want published to a topic. For additional details and instruction, see Configuring Amazon S3 Event Notifications.

The following examples demonstrate how to configure event notifications:

Create an object event for all names

```
{
  "QueueConfigurations": [
  {
   "Id": "1",
   "QueueArn":
"arn:minio:sqs::notifyBlockName:kafka"
,
   "Events": [
       "s3:ObjectCreated:*"
      ],
   "Filter": {
       "Key": {
       "FilterRules": [
```

```
      {
        "Name": "prefix",
        "Value": "*"
       }
     ]
    }
   }
  }
 ]
}
```

Remove object events that have names with the `mybuck` prefix

```
{
  "QueueConfigurations": [
    {
     "Id": "1",
     "QueueArn":
"arn:minio:sqs::notifyBlockName:kafka"
,
     "Events": [
          "s3:ObjectRemoved:*"
        ],
       "Filter": {
            "Key": {
            "FilterRules": [
              {
                "Name": "prefix",
                "Value": "mybuck"
              }
            ]
          }
        }
      }
    ]
 }
```

AWS CLI

You can install and use the AWS Command Line Interface (CLI) with the S3 gateway.

Installing the AWS CLI

Notice: The S3 gateway is included in EEP 6.0.0 - EEP 8.0.0 repositories.

To install the AWS CLI on Ubuntu, RedHat/CentOS, or SLES, run:

```
pip install awscli
```

To configure the AWS CLI, use the AWS `configure` command with the values shown in the following table:

Value	Description
AWS Access Key ID	Insert the `accessKey` from the `tenants.json` file
AWS Secret Access Key	Insert the `secretKey` from the `tenants.json` file
Default region name	Specify `'us-west-1'`
Default output format	Specify `json`

last-updated: Apr 11, 2022

Using the AWS CLI

The following table provides examples of commands that can help you manage buckets and files.

Note: The `public.crt` file is the public certificate for the node that is generated by MapR Data Platform.

```
# Create a bucket and upload a file
aws s3 mb s3://mybucket/ --endpoint-url https://
<ip-address>:9000 --ca-bundle /absolute/path/to/public.crt

echo "HelloS3World" > /tmp/helloS3world.txt
aws s3 cp /tmp/helloS3world.txt s3://mybucket/
helloS3world.txt --endpoint-url https://<ip-address>:9000 --ca-bundle /
absolute/path/to/public.crt

# Move an object from one bucket to another
aws s3 mv s3://firstbucket/helloS3world.txt s3://mybucket/
helloS3world.txt --endpoint-url https://<ip-address>:9000 --ca-bundle /
absolute/path/to/public.crt

# Delete an object from a bucket
aws s3 rm s3://mybucket/helloS3world.txt --endpoint-url https://
<ip-address>:9000 --ca-bundle /absolute/path/to/public.crt

# Delete the bucket
aws s3 rb s3://mybucket/ --endpoint-url https://
<ip-address>:9000 --ca-bundle /absolute/path/to/public.crt

# List all files in the bucket
aws s3 ls s3://mybucket/ --endpoint-url https://
<ip-address>:9000 --ca-bundle /absolute/path/to/public.crt

# Show all files on the filesystem
ls -al /mapr/demo.mapr.com/apps/s3/$(hostname)/mybucket/
```

Note: The public SSL certificate is generated when installing the S3 gateway at `/opt/mapr/objectstore-client/objectstore-client-1.0.0/conf/certs/public.crt`.

For a non-secure cluster, use `http` instead of `https` and do not use `--ca-bundle /absolute/path/to/public.crt`. For example:

```
# List all files in the bucket for secure cluster
aws s3 ls s3://mybucket/ --endpoint-url https://
<ip-address>:9000 --ca-bundle /absolute/path/to/public.crt

# List all files in the bucket for non-secure cluster
aws s3 ls s3://mybucket/ --endpoint-url http://<ip-address>:9000
```

Related concepts

Understanding the Key Store and Trust Store Files on page 1408
Provides a comprehensive listing of the key store and trust store files.

Related reference

configure.sh on page 2053
Describes the syntax and parameters of the `configure.sh` script that you run for a number of tasks including setting up MapR client nodes, and configuring services for a node.

Logs for the S3 Gateway

Logs for the S3 Gateway are available after the first run. You can access logs in `/opt/mapr/objectstore-client/objectstore-client-<version>/logs/`.

 **Notice:** The S3 gateway is included in EEP 6.0.0 - EEP 8.0.0 repositories.

The `minio.log` is rotated daily. Logs from previous days are archived to `minio.log-<date>.gz`.

To change the level of detail (verbosity) displayed in `minio.log`, you can edit the `logLevel` field in the `minio.json` file:

```
/opt/mapr/objectstore-client/objectstore-client-<version>/conf/minio.json
```

Specify a number (1 through 6) to set the log verbosity level:

```
1 - Panic (less detail)
2 - Fatal
3 - Error
4 - Warning
5 - Info
6 - Debug (more detail)
```

Troubleshooting S3 Gateway
This section shows some common error messages generated by the S3 gateway.

 **Notice:** The S3 gateway is included in EEP 6.0.0 - EEP 8.0.0 repositories.

ERROR: Bucket does not exist

Make sure you are performing an operation on a bucket that is not empty.

ERROR: s3a://bucketname: No such file or directory

Add a `/` at the end of your command, after the bucket name:

```
hadoop fs -ls s3a://bucketname/
```

S3 Gateway Limitations
The S3 Gateway has certain limitations and restrictions involving protocol support, object naming, and specific object and bucket APIs that are not supported.

 **Notice:** The S3 gateway is included in EEP 6.0.0 - EEP 8.0.0 repositories.

Multiple protocol support limitation	You can write data via S3 API and read data via NFS/ POSIX. But, you CANNOT write to NFS and read via the S3 API as NFS has no ability to populate the necessary S3 metadata required by S3 gateway.
S3 gateway instance limitation	You can use only one instance of S3 gateway on any given bucket.
Object name character restrictions	You cannot use object names that contain characters `^*|" are NOT supported in Windows and other file systems which do not support them.
S3 object APIs not supported	• ObjectACL (Use bucket policies instead) • ObjectTorrent • ObjectVersions

S3 Bucket APIs not supported	• BucketACL (Use bucket policies instead) • BucketCORS (CORS enabled by default on all buckets for all HTTP verbs) • BucketLifecycle (Not required for Minio erasure coded backend) • BucketReplication (Use mc mirror instead) • BucketVersions, BucketVersioning (Use s3git) • BucketWebsite (Use caddy or nginx) • BucketAnalytics, BucketMetrics, BucketLogging (Use bucket notification APIs) • BucketRequestPayment • BucketTagging
File deletion limitation	If you delete a file using S3 gateway, it will delete the metadata file but not the metadata directory of the file.

Myriad

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Myriad enables the co-existence of Apache Hadoop and Apache Mesos on the same physical infrastructure. By running Hadoop YARN as a Mesos framework, YARN applications and Mesos frameworks can run side-by-side while dynamically sharing cluster resources. Resource allocation and management of Mesos services is supported by MapR Warden. The `configure.sh` script provides additional options for configuring Myriad.

With Apache Myriad, you can:

- Run your operational applications (including those running in Docker) side-by-side with your analytic applications.
- Achieve Hadoop multi-tenancy by provisioning logical Hadoop clusters for each user or group.

Key Features

- YARN running as a Mesos Framework, with resource manager and node managers running inside Mesos containers.
- YARN clusters running on Mesos that can allocate resources in one of the following ways:
- Static - Administrators can use an API or a GUI to add or remove node managers or auxiliary services like the JobHistoryServer.
- Fine-grained - Administrators can provision thin node managers that are dynamically resized based on application demand.
- High Availability (HA) and graceful restart of YARN daemons.
- Ability to launch multiple YARN clusters on the same set of nodes.

- Support for YARN FairScheduler and all functionality such as hierarchical queues with weights. Ability to deploy YARN Resource Manager using Marathon. This feature leverages Marathon's dynamic scheduling, process supervision, and integration with service discovery (Mesos-DNS).
- Ability to run MapReduce version 2 and associated libraries such as Hive and Pig.

Configure Myriad

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

This section provides the steps for configuring Myriad.

1. Modify the **myriad-config-default.yml** file.

   As a minimum, the following Myriad configuration parameters must be set:

   - mesosMaster
   - zkServers
   - YARN_HOME

   Note: The value of mesosMaster parameters may be similar to `<mesosMasterHost>:5050` or `zk://<zkHost>:5181/<path_to_mesos_master>`

last-updated: Apr 11, 2022

2. Configure YARN to use Myriad.
   Edit the **$YARN_HOME/etc.hadoop/yarn-site.xml** file and add the following properties:

```
<property>
    <name>yarn.resourcemanager.hostname</name>
    <value>RESOURCEMANAGER_HOSTNAME</value>
</property>

<property>
  <name>yarn.resourcemanager.recovery.enabled</name>
  <value>true</value>
</property>

<property>
  <name>yarn.resourcemanager.scheduler.class</name>
<value>org.apache.myriad.scheduler.yarn.MyriadFairScheduler</value>
</property>

<property>
  <name>yarn.nodemanager.resource.cpu-vcores</name>
  <value>${nodemanager.resource.cpu-vcores}</value>
</property>

<property>
  <name>yarn.nodemanager.resource.memory-mb</name>
  <value>${nodemanager.resource.memory-mb}</value>
</property>

<property>
  <name>yarn.nodemanager.address</name>
  <value>${myriad.yarn.nodemanager.address}</value>
</property>

<property>
  <name>yarn.nodemanager.webapp.address</name>
  <value>${myriad.yarn.nodemanager.webapp.address}</value>
</property>

<property>
  <name>yarn.nodemanager.webapp.https.address</name>
  <value>${myriad.yarn.nodemanager.webapp.address}</value>
</property>

<property>
  <name>yarn.nodemanager.localizer.address</name>
  <value>${myriad.yarn.nodemanager.localizer.address}</value>
</property>

<property>
  <name>mapreduce.shuffle.port</name>
  <value>${myriad.mapreduce.shuffle.port}</value>
</property>

<property>
  <name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle,mapr_direct_shuffle,myriad_executor</value>
</property>

<property>
  <name>yarn.nodemanager.aux-services.myriad_executor.class</name>
  <value>org.apache.myriad.executor.MyriadExecutorAuxService</value>
</property>
```


last-updated: Apr 11, 2022

```
<property>
  <name>yarn.resourcemanager.store.class</name>
<value>org.apache.hadoop.yarn.server.resourcemanager.recovery.MyriadFileS
ystemRMStateStore</value>
</property>

<property>
  <name>yarn.scheduler.minimum-allocation-mb</name>
  <value>0</value>
 </property>

<property>
  <name>yarn.scheduler.minimum-allocation-vcores</name>
  <value>0</value>
</property>

<property>
  <name>yarn.scheduler.minimum-allocation-disks</name>
  <value>0</value>
</property>
```

See Configuring Services on page 3973 for information about configuring JobHistoryServer and other services. See Use Myriad for information about using Myriad.

Configuring Mesos-DNS

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

After installing Mesos-DNS on its master node, configure Mesos-DNS by updating values in the `config.json` file on the node where Mesos-DNS is installed and update the `/etc/resolv.conf` file on every node in the cluster.

Note: The Mesos-DNS package can be installed on any node that is designated as its master. For more information, see https://github.com/mesosphere/mesos-dns/tree/master/docs.

Important: The first nameserver specified in the `/etc/resolv.conf` file must be the IP address where Mesos-DNS is installed. This is to prevent issues associated with slow interconnectivity.

In the following example, 10.10.100.16 is the Mesos-DNS IP address:

```
vi /etc/resolv.conf

nameserver 10.10.100.16
nameserver 10.10.1.10
nameserver 12.250.1.3
```

Configuring Services

Services, such as JobHistoryServer, can be launched as a task from the Myriad Framework.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

To configure services, define the service in the Myriad configuration file, `myriad-config-default.yml`. Once defined, the Myriad Service REST API can be used to launch an instance as a task for the service.

The services and service properties are as follows:

```
<!-- Define services as a task -->
services:
```


last-updated: Apr 11, 2022

```
    serviceName:        # Name of the service
    jvmMaxMemoryMB:     # Memory needed for service
    cpus:               # CPU needed for Service
    ports:              # Map of ports: port property and value
                        # (Default: 0 which indicates that port will be
assigned randomly)
                        # Uses the syntax, <name>: <value>
    envSettings:        # Any environment settings
    taskName:           # Again service name
    maxInstances:       # If defined maximum number of instances this
service can have per myriad framework
    command:            # Command to be executed
    serviceOptsName:    # Name of the env. variable that may need to be set
for the service that will include env. settings
```

To configure additional services:

1. Edit the `myriad-config-default.yml` file.
2. For the `services` property, specify a service and provide the appropriate parameters.

   For example, to define a TimeLineServer task:

   ```
   services:
       jobhistory:
           jvmMaxMemoryMB: 64
           cpus: 0.5
           ports:
              myriad.mapreduce.jobhistory.admin.address: 10033
              myriad.mapreduce.jobhistory.address: 10020
              myriad.mapreduce.jobhistory.webapp.address: 19888
           envSettings:
           taskName: jobhistory
           serviceOptsName: HADOOP_JOB_HISTORYSERVER_OPTS
           command: $YARN_HOME/bin/mapred historyserver
           maxInstances: 1
       timelineserver:
           jvmMaxMemoryMB: 1024
           cpus: 1
           ports:
               myriad.mapreduce.timeline.address: 10200
               myriad.mapreduce.timeline.webapp.address: 8188
           envSettings:
           taskName: timelineserver
           command: $YARN_HOME/bin/yarn timelineserver
           maxInstances: 1
   ```

3. Copy or update the `myriad-config-default.yml` file on every node in the cluster.
4. Restart the initial Resource Manager from Marathon.

Configuring Multiple YARN Clusters

This topic describes how to set up multiple Myriad frameworks and multiple YARN clusters in the Mesos environment.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Multiple YARN clusters are created in the Mesos environment when multiple Myriad frameworks are each running a Resource Manager. The Resource Manager spawns Node Managers and Job History Service processes which form a cluster. To create multiple YARN clusters, a Resource Manager is run under the Myriad framework with the cluster name specified. The cluster name differentiates between

staging, system, and local volume MapReduce directories. When configuring for multiple YARN clusters, run `configure.sh` on the nodes where you want the 1st Myriad framework and YARN cluster, then run `configure.sh` with different parameters on the nodes where you want the 2nd Myriad framework and YARN cluster, and so on.

1. Run `configure.sh` on the slave nodes for the YARN cluster. Specify a Resource Manager hostname, Job History name, Myriad framework, and cluster prefix name.

   Note: It is recommended that you use the same value for the `-RM, -HS, -MF, and -MCL` parameters.

   For example, if you set the parameter value to **framework1** for each of the parameters, the `configure.sh` parameters could be similar to the following:

   ```
   /opt/mapr/server/configure.sh \
   -C 10.10.100.16 \
   -Z 10.10.100.16 -u mapr -g mapr \
   -N myCluster \
   -F /root/disk.list \
   -RM framework1.marathon.mesos \
   -HS jobhistory.framework1.mesos \
   -MF framework1 \
   -MCL framework1
   ```

2. Run `configure.sh` on the slave nodes for each new YARN cluster. Specify new values for Resource Manager hostname, Job History name, Myriad framework, and cluster prefix name.

   For example, on the subsequent YARN cluster, if you set the parameter value to **framework2** for each of the parameters, the `configure.sh` parameters could be similar to the following:

   ```
   /opt/mapr/server/configure.sh \
   -C 10.10.100.16 \
   -Z 10.10.100.16 -u mapr -g mapr \
   -N myCluster \
   -F /root/disk.list \
   -RM framework2.marathon.mesos \
   -HS jobhistory.framework2.mesos \
   -MF framework2 \
   -MCL framework2
   ```

For example, the following creates three Myriad frameworks and the associated YARN cluster name prefix (framework1, framework2, and framework3). In the following examples, -N is set to **myCluster**, the primary IP address is 10.10.100.16, and the values for the -RM, -HS, -MF, and -MCL parameters change for each YARN cluster.

Create the 1st cluster for framework1:

```
// On slave nodes 1 - 3, run:
/opt/mapr/server/configure.sh \
-C 10.10.100.16 \
-Z 10.10.100.16 -u mapr -g mapr \
-N myCluster \
-F /root/disk.list \
-RM framework1.marathon.mesos \
-HS jobhistory.framework1.mesos \
-MF framework1 \
-MCL framework1
```


last-updated: Apr 11, 2022

Create the 2nd cluster for framework2:

```
// On slave nodes 4 - 6, run:
/opt/mapr/server/configure.sh \
-C 10.10.100.16 \
-Z 10.10.100.16 -u mapr -g mapr \
-N myCluster \
-F /root/disk.list \
-RM framework2.marathon.mesos \
-HS jobhistory.framework2.mesos \
-MF framework2 \
-MCL framework2
```

Create the 3rd cluster for framework3:

```
// On slave nodes 7 - 9, run:
/opt/mapr/server/configure.sh \
-C 10.10.100.16 \
-Z 10.10.100.16 -u mapr -g mapr \
-N myCluster \
-F /root/disk.list \
-RM framework3.marathon.mesos \
-HS jobhistory.framework3.mesos \
-MF framework3 \
-MCL framework3
```

Configuring Multiple YARN Clusters (Binary)

This topic describes how to set up multiple Myriad frameworks and multiple YARN clusters in the Mesos environment for binary distribution.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

1. Set up a Myriad framework.

   Note: For each Myriad framework, setup different cluster name and resource manager hostname in yarn-site.xml file. See Using Multiple YARN Clusters on page 3984 for more information.

   a) Edit the `myriad-config-default.yml` file in the /opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/ and update the `nodeManagerUri` and `YARN_HOME` parameters:

   ```
   nodeManagerUri: maprfs://<full path to Hadoop tar file>
       // For example: maprfs:///dist/hadoop-2.7.0.framework1.tar.gz
   YARN_HOME: hadoop-2.7.0
       //  (or the relative path to the hadoop directory that will be
   created in the tar file)
   ```

2. Create a tar file for the Myriad framework.

   ```
   cd /opt/mapr/hadoop
   tar -czvf hadoop-2.7.0.framework1.tar.gz hadoop-2.7.0
   ```

3. Copy the Myriad framework tar file to the location specified by the `nodeManagerUri` parameter in the `myriad-config-default.yml` file.

4. Repeat steps 1 - 3 for each Myriad framework.

Using a Different Scheduler
Myriad uses MyriadFairScheduler by default. However, MyriadCapacityScheduler and MyriadFifoScheduler can also be used to run jobs.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The Myriad Scheduler property is set in the `yarn-site.xml` file:

```
<property>
  <name>yarn.resourcemanager.scheduler.class</name>
  <value>org.apache.myriad.scheduler.yarn.MyriadFairScheduler</value>
  <description>You can configure other schedulers from following list:
     org.apache.myriad.scheduler.yarn.MyriadCapacityScheduler,
     org.apache.myriad.scheduler.yarn.MyriadFifoScheduler</description>
</property>
```

Important: Fine-grained scaling only works with a MyriadFairScheduler configuration.

To use either MyriadCapacityScheduler or MyriadFifoScheduler:

1. Edit the `/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop/yarn-site.xml` file.
2. For the `yarn.resourcemanager.scheduler.class` property, specify either `org.apache.myriad.scheduler.yarn.MyriadCapacityScheduler` or `org.apache.myriad.scheduler.yarn.MyriadFifoScheduler`
3. Modify the YARN scheduler minimum allocation properties.
   The configure.sh sets these properties to zero (0) by default. When using a different scheduler, the properties must be set (as a minimum) to the following values:

   ```
   yarn.scheduler.minimum-allocation-mb = 1024
   yarn.scheduler.minimum-allocation-vcores = 1
   yarn.scheduler.minimum-allocation-disks = 0
   ```

4. Copy or update the `yarn-site.xml` file on every node in the cluster.
5. Restart the initial Resource Manager from Marathon.

Use Myriad

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The Myriad environment is established by running an initial application for the Resource Manager. The initial application ID is established when you configure Myriad with `/opt/mapr/services/configure.sh` with the Resource Manager property (-RM) and the value: <RM app-name>>.marathon.mesos. For example, if the value of Resource Manager property is rm.marathon.mesos:

rm	The ID given to the Resource Manager when launched using Marathon. Mesos-DNS constructs the Resource Manager hostname using the ID.
marathon	The Mesos framework.

last-updated: Apr 11, 2022

If you are using Marathon, launch Marathon, and run an initial Resource Manager application. See Starting Resource Manager for more information.

Note: The Resource Manager can not be managed by the `maprcli nodes services` command.

Starting Resource Manager
A Resource Manager for Myriad must be launched before the Myriad UI becomes available.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The initial Resource Manager is started from Marathon. In addition, only one (1) Resource Manager can be launched on a cluster.

Note: Starting the Resource Manager from the command line results in an error, see Troubleshooting Myriad for a workaround.

To start the Resource Manager:

1. Launch Marathon with http://hostname:8080.
2. Click on **New App**.
3. Create a new application for the resource manager and specify:
   - ID - Default: rm
   - CPU - 0.2 minimum
   - Memory - 2048 minimum
   - Instances - 1 only
   - Command - `env && yarn resourcemanager`

   For example:

Table

Parameter	Example
ID	rm
CPU	0.2
Memory	2048
Instances	1
Command	`env && yarn resourcemanager`

Important: Only one instance of Myriad Resource Manager can be running. Scaling up instances (specifying more than one instance) may result in unpredictable behavior. Creating a new Myriad application from Marathon results in a new framework.

4. Access the Myriad UI with <hostname>:8192

   If you have multiple nodes in your cluster, Myriad may be started on any one of the nodes. To obtain the Myriad host, access the Mesos UI (`http://<hostname>:5050`) and check the **Active Tasks > Hosts** panel.

   If the Mesos-DNS entry is added on your local machine, the host can be accessed with the following address: `http://rm.marathon.mesos:8192`.

last-updated: Apr 11, 2022

Launching Myriad Services

The Myriad services are launched automatically and accessed from their designated ports.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Mesos primary, Mesos secondary, and Mesos Marathon services are launched automatically after configuration (running `configure.sh`). Myriad is launched once you run an initial Resource Manager, normally from the Marathon UI. In a browser, Mesos, Marathon, and Myriad are accessible (launched) from the following ports:

Table

Application	Port
Mesos Master UI	http://<hostname>:5050
Marathon UI	http://<hostname>:8080
Myriad UI	http://<hostname>:8192 **Note:** The Myriad UI is available once an initial Resource Manager is launched. **Note:** If you have multiple nodes in your cluster, Myriad may be started on any one of the nodes. To obtain the Myriad host, access the Mesos UI and check the **Active Tasks > Hosts** panel.
Control System	http://<hostname>:8443

Note: If your environment has both Mesos Marathon and Spark installed on the same node, a conflict occurs because the default port for both is 8080. To resolve this conflict, change the port for one of the applications.

Monitoring Myriad

Myriad is monitored using several application user interfaces.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following user interfaces are used to monitor application processes, tasks, configurations, and flexup or flexdown resources:

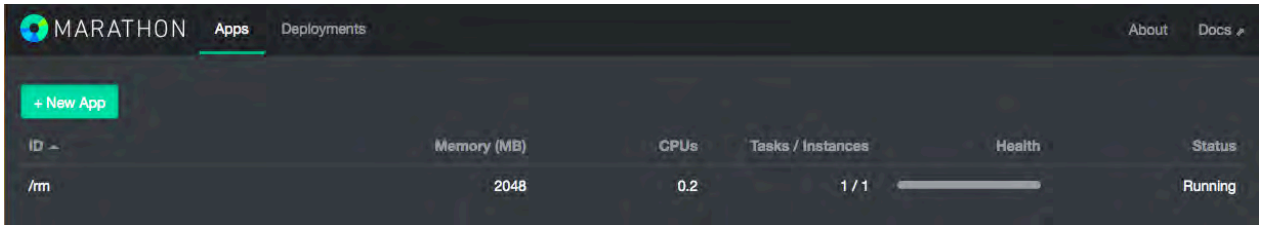
- Marathon UI
- Mesos UI
- Myriad UI
- Control System

Marathon UI

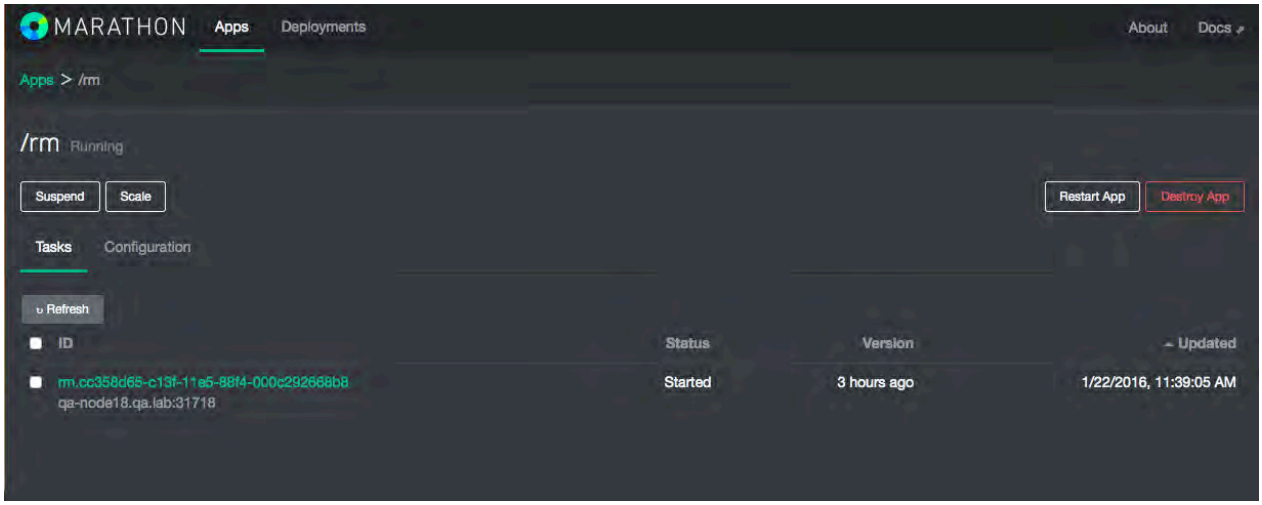
The Marathon UI is used to manage application processes. The **Apps** panel provides created, suspended, scaled, and refreshed as well as restarted and destroyed. To view task details, click on the application link.

The following screenshot shows the Marathon home page:

last-updated: Apr 11, 2022



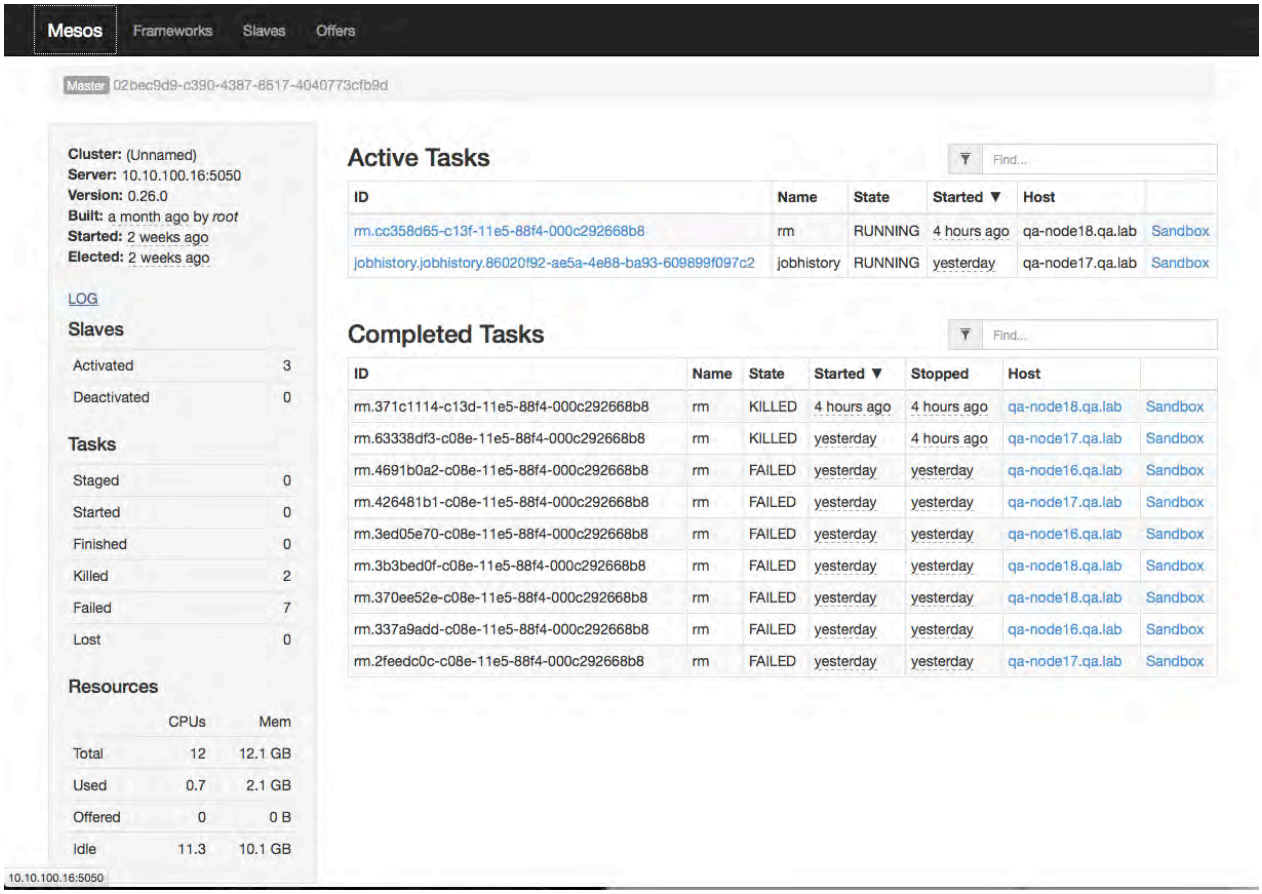
The following screenshot shows the Marathon application details page:



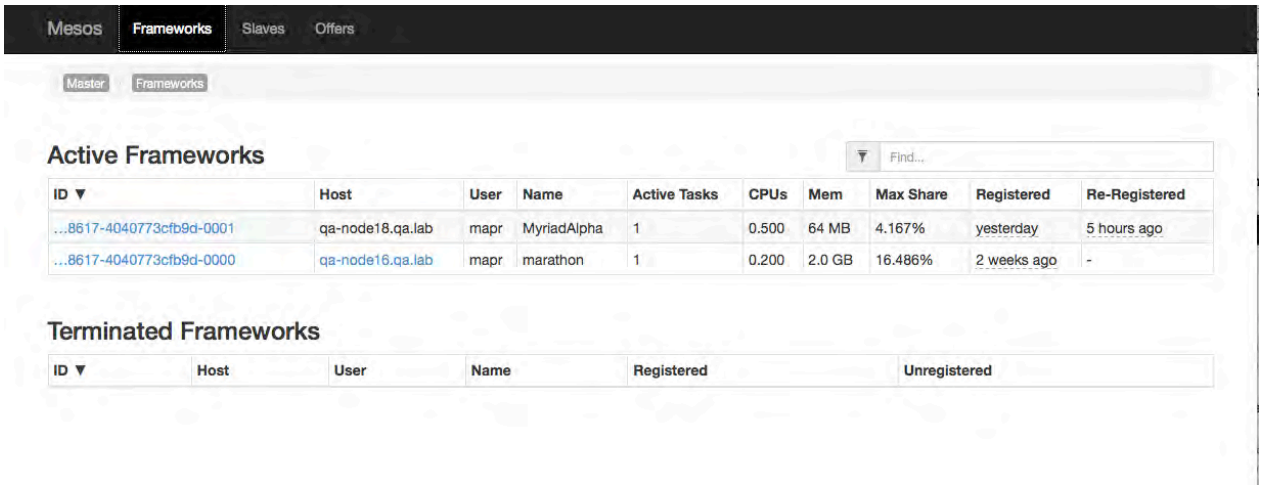
Mesos UI

The Mesos UI is used to monitor Mesos active and completed tasks, provide cluster information, provide active and terminated frameworks, Mesos secondary information, and Offers.

The following screenshot shows the Mesos home page:



The following screenshot shows the Mesos Frameworks page:



Myriad UI

The Myriad UI is used to flexup or flexdown Node Manager instances, that is, to allocate more or less resources using a predefined configuration as well as to monitor your tasks and display Myriad configuration.

- The main Myriad UI page shows the underlying REST API.

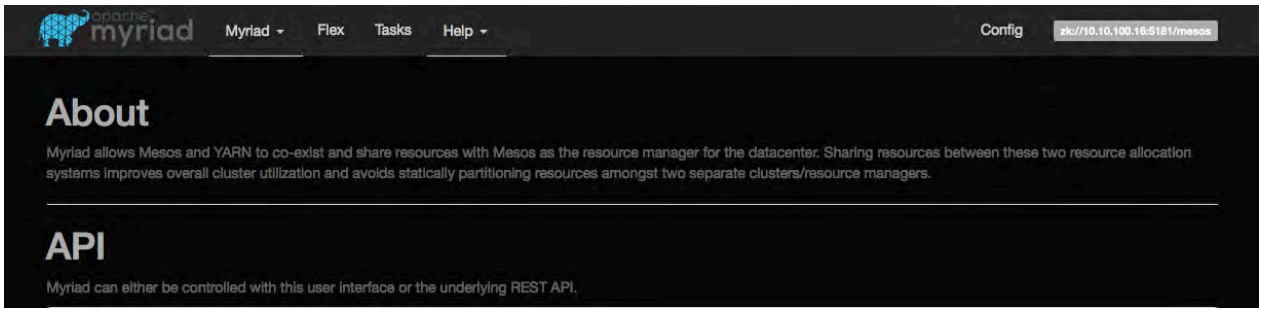
- The **Flex** button allows you to flex up or flex down instances by specifying the profile and number of instances to flex.

  Note: At this time, only one (1) instance is allowed.

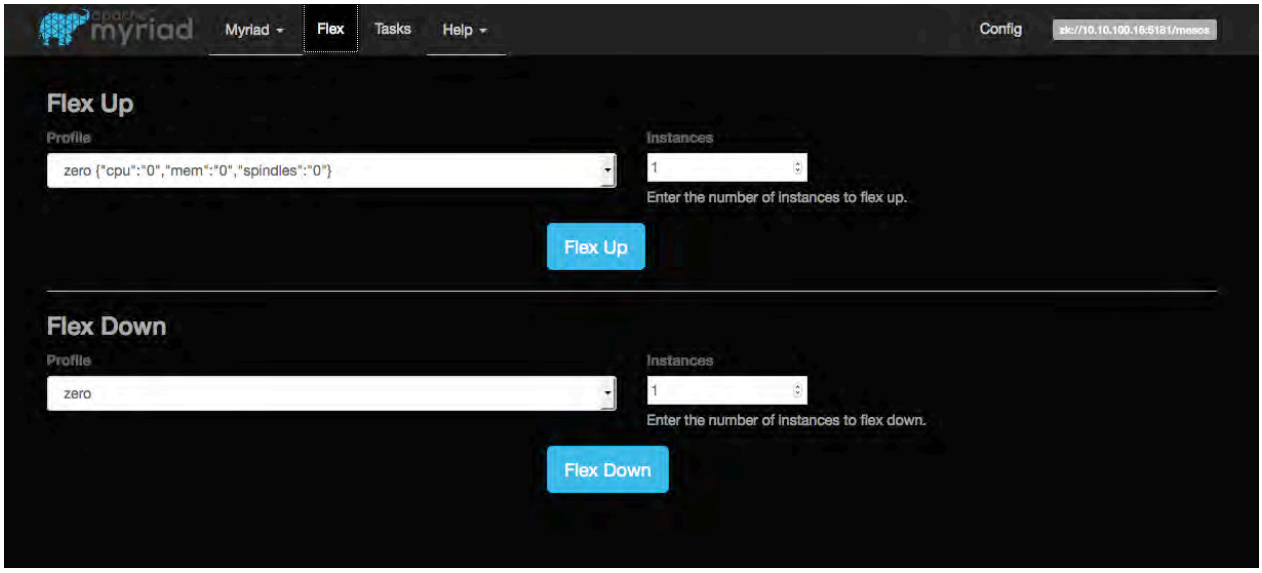
- The **Task** button shows the current active, killable, pending, and staging tasks.
- The **Config** button shows the defined profiles and the parameters in the Myriad configuration file (`myriad-config-default.yml`).

Flexing up and flexing down changes are reflected in the Mesos UI under the Myriad Framework name.

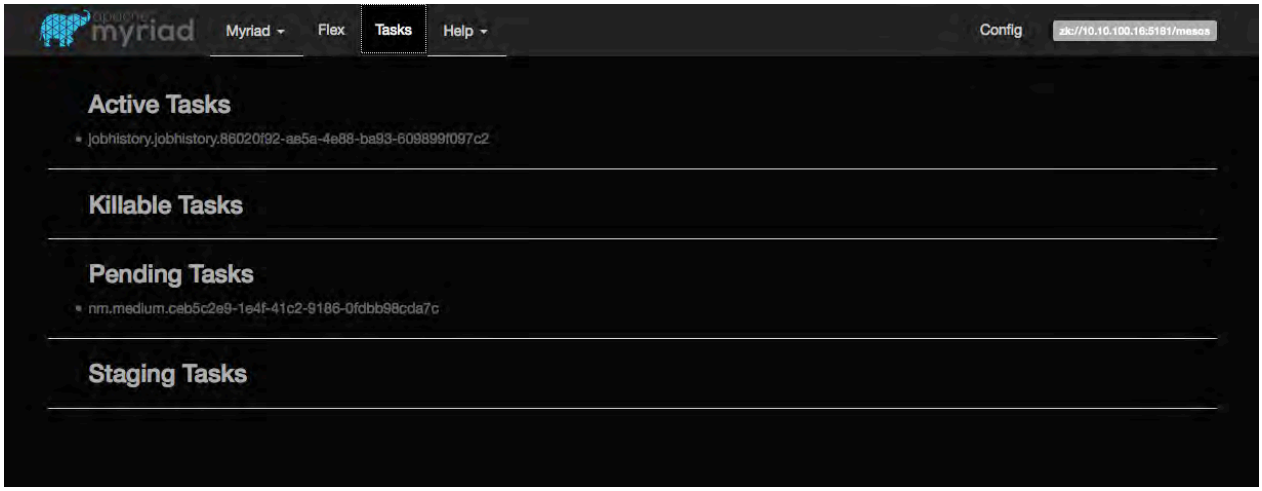
The following screenshot shows the Myriad home page:



The following screenshot shows the Myriad Flex page:



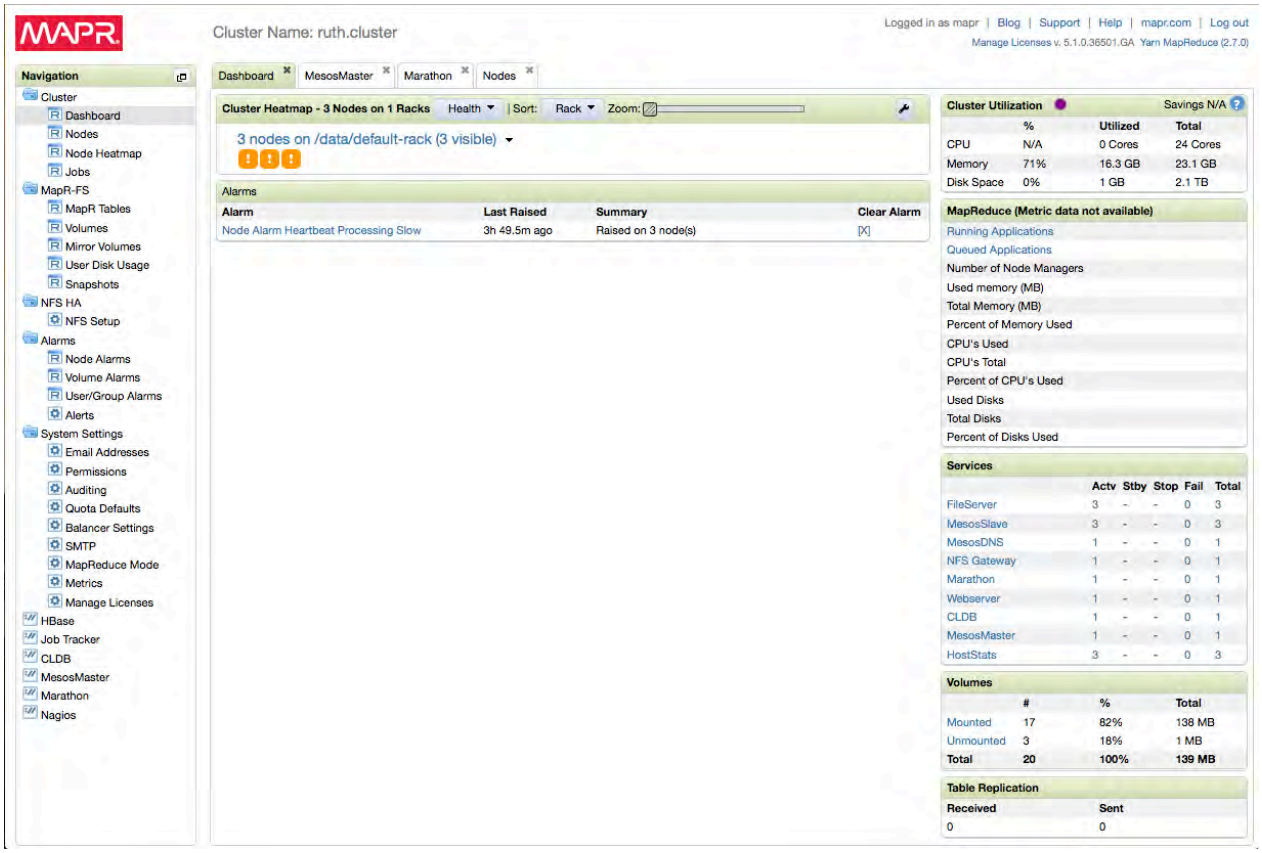
The following screenshot shows the Myriad Tasks page:



The Control System

The Control System is used as a central access point to navigate to Mesos Primary instance, Mesos Secondary instance, Mesos-DNS and Marathon information about the configured and running services in your cluster.

The following screenshot shows the Control System interface when Myriad is installed and configured:



The following Control System panel shows the Myriad services:

last-updated: Apr 11, 2022

Services					
	Actv	Stby	Stop	Fail	Total
FileServer	3	-	-	0	3
MesosSlave	3	-	-	0	3
MesosDNS	1	-	-	0	1
NFS Gateway	1	-	-	0	1
Marathon	1	-	-	0	1
Webserver	1	-	-	0	1
CLDB	1	-	-	0	1
MesosMaster	1	-	-	0	1
HostStats	3	-	-	0	3

Using Multiple YARN Clusters

This topic describes how to create multiple YARN clusters and to run Hadoop jobs in a multiple YARN cluster environment.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Multiple YARN clusters are created in the Mesos environment when multiple Myriad frameworks are each running a Resource Manager. Where there are multiple YARN clusters, clients must specify a cluster name prefix to identify which cluster the Hadoop job should be run on.

Creating Multiple YARN Clusters

The Resource Manager spawns Node Managers and Job History Service processes which form a cluster. To create multiple YARN clusters, a Resource Manager is run under the Myriad framework along with the cluster name prefix. The cluster name prefix differentiates between staging, system, and local volume MapReduce directories.

Note: The cluster name prefix setting is a value of the `cluster.name.prefix` property in the **yarn-site.xml** file. Also, you must set it an an `envSettings` variable for each service on a cluster. For example, the following show the `cluster.name.prefix` setting for jobHistoryServer:

```
services:
     jobhistory:
     jvmMaxMemoryMB: 64
     cpus: 0.5
     ports:
       myriad.mapreduce.jobhistory.admin.address: 10033
       myriad.mapreduce.jobhistory.address: 10020
       myriad.mapreduce.jobhistory.webapp.address: 19888
     envSettings: -Dcluster.name.prefix=/framework1
     taskName: jobhistory
     serviceOptsName: HADOOP_JOB_HISTORYSERVER_OPTS
     command: $YARN_HOME/bin/mapred historyserver
     maxInstances: 1
```

Cluster name prefix and Resource Manager parameters:

```
-Dyarn.resourcemanager.hostname=<RM appName>.marathon.mesos
-Dcluster.name.prefix=/<clusterNamePrefix>
```


last-updated: Apr 11, 2022

For example, the `cluster.name.prefix` and `yarn.resourcemanager.hostname` properties are both **framework1** in the **yarn-site.xml** file:

```
env && export
YARN_RESOURCEMANAGER_OPTS="-Dyarn.resourcemanager.hostname=framework1.marath
on.mesos
-Dcluster.name.prefix=/framework1"
&&/opt/mapr/hadoop/hadoop-2.7.0/bin/yarn resourcemanager
```

Running Jobs from the Client-side

When running Hadoop jobs from clients within a multiple YARN cluster environment, additional parameters are required. In this scenario, one client could submit jobs to a specific YARN cluster and a second client could submit jobs to a completely different YARN cluster. The following parameters are specified when multiple YARN clusters are implemented.

Table

Parameter	Parameter Values	Description
cluster.name.prefix	/<clusterNamePrefix>	Prefix for each Myriad cluster name. This is the top-level directory where all the staging and shuffle data is written for a specific Myriad framework. Specified by the `-MCL` parameter when running `configure.sh`. See Configure Myriad for more information.
yarn.resourcemanager.hostname	<RM appName>.marathon.mesos	Mesos hostname for Resource Manager. Specified by the `-RM` parameter when running `configure.sh`. See Configure Myriad for more information.
yarn.resourcemanager.dir	/var/mapr/cluster/yarn/<clusterNamePrefix>/rm	Directory for Resource Manager specific data. The directory is based on the cluster name prefix specified by the `-MCL` parameter when running `configure.sh`. See Configure Myriad for more information. .
mapr.mapred.localvolume.root.dir.name	/<clusterNamePrefix>/nodeManager	Directory for local volume MapReduce specific data. The directory is based on the cluster name prefix.

Command-line Parameters

```
-Dcluster.name.prefix=/<clusterNamePrefix>
-Dyarn.resourcemanager.hostname=<RM appName>.marathon.mesos
-Dyarn.resourcemanager.dir=/var/mapr/cluster/yarn/<clusterNamePrefix>/rm
-Dmapr.mapred.localvolume.root.dir.name=/<clusterNamePrefix>/nodeManager
```

For example, where the `cluster.name.prefix` and `yarn.resourcemanager.hostname` properties are both **framework1**:

```
hadoop jar
/opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/mapreduce/
```


last-updated: Apr 11, 2022

```
hadoop-mapreduce-examples-2.7.0-mapr-1506-SNAPSHOT.jar wordcount
-Dcluster.name.prefix=/framework1
-Dyarn.resourcemanager.hostname=framework1.marathon.mesos
-Dyarn.resourcemanager.dir=/var/mapr/cluster/yarn/framework1/rm
-Dmapr.mapred.localvolume.root.dir.name=/framework1/nodeManager
/test.log test.out
```

yarn-site.xml Properties

Alternatively, rather than specifying these parameters dynamically, add them to the client-side `yarn-site.xml` file.

For example, where the `cluster.name.prefix` and `yarn.resourcemanager.hostname` properties are both **framework1**:

```
// Property example for cluster.name.prefix
  <property>
    <name>cluster.name.prefix</name>
    <value>/framework1</value>
  </property>

// Property example for yarn.resourcemanager.hostname
  <property>
    <name>yarn.resourcemanager.hostname</name>
    <value>framework1.marathon.mesos</value>
  </property>

// Property example for yarn.resourcemanager.dir
  <property>
    <name>yarn.resourcemanager.dir</name>
    <value>/var/mapr/cluster/yarn/framework1/rm</value>
  </property>

// Property example for mapr.mapred.localvolume.root.dir.name
  <property>
    <name>mapr.mapred.localvolume.root.dir.name</name>
    <value>/framework1/nodeManager</value>
  </property>
```

Modifying Myriad Profiles

Myriad profiles can be changed and expanded depending on your requirements.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following are the default Myriad profiles:

```
profiles:
    zero:  # NMs launched with this profile dynamically obtain CPU/Memory
from Mesos
        cpu: 0
        mem: 0
        spindles: 0
    small:
        cpu: 2
        mem: 2048
        spindles: 1
    medium:
        cpu: 4
        mem: 4096
        spindles: 2
```


last-updated: Apr 11, 2022

```
    large:
        cpu: 10
        mem: 12288
        spindles: 4
```

To modify Myriad profiles:

1. Edit the `myriad-config-default.yml` file.
2. For the `profiles` property, you can modify existing profiles and create new profiles
   For example, to add a "super" profile:

   ```
      huge:
         cpu: 15
         mem: 12288
         spindles: 5
   ```

3. Copy or update the `myriad-configure-default.yml` file on every node in the cluster.
4. Restart the initial Resource Manager from Marathon.

Myriad REST API

Securing the Myriad REST API

To secure the Myriad REST API, set the `isSecure` property to **true** in the **myriad-config-default.yml** file. In this case, Myriad uses the same security type as the value of the `hadoop.http.authentication.type` parameter in the **core-site.xml** file

Note: On a secure cluster, the default value of this property is `org.apache.hadoop.security.authentication.server.MultiMechsAuthenticationHandler`. Authentication, such as Kerberos or plain (using PAM), is determined automatically.

Scaling the Cluster

To scale a cluster up or down, use the Cluster API. The Cluster API provide flexup and flex down capability the changes the size of one or more instances in a cluster. The instance size is a profile parameter that is a predefined value of zero, small, medium, and large. These predefined values are specified in the Myriad configuration file (myriad-config-default.yml).

Specify Service Instances

To specify the number of instances for a service, use the Service API. Services are configured in the Myriad configuration file (myriad-config-default.yml).

Retrieving Configuration and State

To retrieve the Myriad configuration and the Myriad Scheduler state, use the Configuration API and State API. See Myriad open source REST API for more information.

The Myriad REST API provides the following functionality:

Table

API	HTTP Method	URI	Description
Cluster	PUT	/api/cluster/flexup	Expands the cluster size. Parameters: profile, instances, constraints
Cluster	PUT	/api/cluster/flexdown	Shrinks the cluster size. Parameters: profile, instances, constraints
Service	PUT	/api/cluster/flexupservice	Increases the number of instances for a service. Parameters: instances, serviceName
Service	PUT	/api/cluster/flexdownservice	Shrinks the number of instances for a service. Parameters: instances, serviceName
Configuration	GET	/api/config	Retrieves the Myriad configuration.
State	GET	api/state	Retrieves a snapshot of the Myriad Scheduler state.
Framework	GET	/api/framework/shutdown/frameworkd	Destroys the Myriad framework. **Note:** With this release, GET is the method used to destroy the Myriad framework.

Troubleshoot Myriad
Explains how to troubleshoot Myriad.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Resource Manager not Starting

The Resource Manager cannot be started from the command line using the following command:

```
$ yarn resourcemanager
```

The following error occurs:

```
java.lang.UnsatisfiedLinkError: no mesos in java.library.path
```

Workaround:

Manually pass the MESOS_NATIVE_LIBRARY environment variable when starting the Resource Manager. For example:

```
env && export MESOS_NATIVE_LIBRARY=/usr/local/lib/libmesos.so && yarn
          resourcemanager
```

Note: This is not an issue when starting Resource Manager from the Marathon UI.

Marathon Fails

The Mesos-3602 fix in Mesos version 0.26 causes an issue when using the maprfs path for the URI. When starting Marathon, an extra forward slash (/) is appended to the URI path. The application fails with a message similar to the following:

```
hadoop fs -copyToLocal '/maprfs:///dist/hadoop-2.7.0.myriad1.tar.gz'
'/opt/mapr/slaves/67d1f64c-449b-4609-82f3-5da309f3c5c5-S9/
frameworks/67d1f64c-449b-4609-82f3-5da309f3c5c5-0000/
executors/myriad1.63bbb98c-c072-11e5-b686-0cc47a587d20/runs/
427fe309-82c5-4f8b-9fa3-6dd39a4a5ef4/hadoop-2.7.0.myriad1.tar.gz
-copyToLocal: java.net.URISyntaxException: Expected scheme-specific part at
index 7: maprfs:
```

Workaround:

Replace `maprfs:///` with `hdfs:///` when specifying the URI on Marathon application.

Note: This issue is fixed by MESOS-4304, which is available with in the Mesos 0.27 release.

Oozie

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Apache Oozie™ is a workflow scheduler system to manage Apache Hadoop jobs. Using Oozie, you can set up *workflows* that execute MapReduce applications and *coordinators* that manage workflows.

Oozie Workflow jobs are Directed Acyclical Graphs (DAGs) of actions. Oozie Coordinator jobs are recurrent Oozie Workflow jobs triggered by time (frequency) and data availability.

Oozie is integrated with the rest of the Hadoop stack and supports several types of Hadoop jobs out-of-the-box, such as Java map-reduce, Streaming map-reduce, Pig, Hive, Sqoop, and Distcp, as well as system-specific jobs, such as Java programs and shell scripts.

This section contains documentation for Oozie on the MapR Data Platform and provides all relevant details needed to use Oozie with MapR. The documentation in this section does not duplicate the documentation on the Apache Oziesite.

Configure Oozie

This topic describes how to configure Oozie.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

To set up Oozie using a non-default administrative user (non-`mapr`), set the `yarn.resourcemanager.principal` property to the non-`mapr` user in the `yarn-site.xml` file. For more information, see yarn-site.xml.

In Oozie 4.3.0, when you modify the `yarn-site.xml` file on Oozie server nodes, you must rebuild the Oozie war file to include the latest changes using one of the following two methods:

last-updated: Apr 11, 2022

1. First method:
   a. Rebuild the Oozie war files:

```
/opt/mapr/oozie/oozie-<version>/bin/oozie-setup.sh hadoop
<version> /opt/mapr/hadoop/hadoop<version> -secure
```

   b. Restart Oozie:

```
maprcli node services -name oozie -action restart -nodes <space
delimited list of nodes>
```

2. Second method:
   - Or, for EEP 4.x and Oozie 5.1 (EEP 6.x), you can run the following command:

```
/opt/mapr/server/configure.sh -R
```

Configuring Oozie on a Secure Cluster

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The default configuration for Oozie on a secure MapR cluster uses MapR tickets to authenticate between the Oozie client and server. The Oozie server uses MapR tickets to authenticate the connection between the YARN components and the JobClient embedded in the Oozie server. Starting in the EEP 4.0 release, for secure clusters, Oozie is also configured to use SSL encryption. This default configuration is in place once Oozie is installed and the security features for your cluster are enabled. See Enabling and Disabling Security. No further configuration is required. See User Impersonation for Oozie to enable user impersonation for Oozie.

The sections below provide instructions to manually configure Oozie features on a secure MapR cluster:

Oozie Server secured by default to use SSL

Starting in EEP 6.0.0, for Oozie 4.3.0 and above, Oozie server uses SSL by default for secured clusters.

1. By default, on a secure cluster, Oozie reads the `ssl-client.xml` file and configures SSL using this file.
2. If you want to use a custom SSL configuration, enable custom secure on your cluster.

Configuring the Oozie Server to use SSL (Oozie 5.1.0.0 and above)

Configure the settings necessary for enabling SSL or TLS support in the `oozie-site.xml` file.

1. Set the `oozie.https.enabled` property to `true`:

```
<property>
  <name>oozie.https.enabled</name>
  <value>true</value>
  <description>Controls whether SSL encryption is enabled.</description>
</property>

<property>
  <name>oozie.https.truststore.file</name>
  <value>/opt/mapr/conf/ssl_truststore</value>
  <description>Path to a TrustStore file.</description>
</property>

<property>
  <name>oozie.https.keystore.file</name>
  <value>/opt/mapr/conf/ssl_keystore</value>
  <description>Path to a KeyStore file.</description>
</property>

<property>
  <name>oozie.https.keystore.pass</name>
  <value><password></value>
  <description>Password to the KeyStore.</description>
</property>
```

   Note: To configure the Oozie for `http`, set the `oozie.https.enabled` property to `false`.

2. Connect to the Oozie Web UI using SSL (HTTPS):

```
https://oozie.server.hostname:11443/oozie
```

You can also encrypt the Oozie keystore password. For more information, refer to the open source documentation.

Configuring Oozie Clients to Use SSL (Oozie 4.3.0)

To configure the Oozie clients, follow this step:

- Specify the path to the keystore and password at `/opt/mapr/oozie/oozie-<version>/conf/oozie-client-env.sh`:

```
export
OOZIE_CLIENT_OPTS="${OOZIE_CLIENT_OPTS} -Djavax.net.ssl.trustStore=/opt/
mapr/conf/ssl_truststore"
```

Using Kerberos to Securely Authenticate Between the Oozie Client and Server

Oozie can use Kerberos to secure authentication between the Oozie client and server. The Oozie server uses the Kerberos principal and keytab information specified in the Java Authentication and Authorization (JAAS) configuration file at `/opt/mapr/conf/mapr.login.conf`. Generate a Kerberos principal of the form `http/ <fqdn>@<realm>` and store the keytab in the cluster's keytab file. The default keytab file location is `/opt/mapr/conf/mapr.keytab`.

To use Kerberos authentication on a specific invocation of Oozie without modifying your client, use the `-auth KERBEROS` option when you start Oozie, as in the following example:

```
$ bin/oozie admin -status -auth KERBEROS
```


last-updated: Apr 11, 2022

Defining a Custom Principal and Keytab File

You can use custom Kerberos principals and keytab files if you wish. To specify the locations of these custom Kerberos principals and keytab files, make the following modifications to the `oozie-site.xml` file:

- Explicitly change the authentication type to Kerberos.

```
<property>
    <name>oozie.authentication.type</name>
    <value>kerberos</value>
    <description>
    Defines authentication used for Oozie HTTP endpoint.
    Supported values are: simple | kerberos |
#AUTHENTICATION_HANDLER_CLASSNAME#
    </description>
</property>
```

- Modify the following entries to use your custom principals and keytab. The principal takes the form `HTTP/<hostname>`, where *hostname* is the URL used by the client to connect to the server.

```
<property>
    <name>oozie.service.HadoopAccessorService.keytab.file</name>
    <value>/opt/mapr/conf/mapr.keytab</value>
    <description>
      Location of the Oozie user keytab file.
    </description>
</property>

<property>
    <name>local.realm</name>
    <value>{local.realm}</value>
    <description>
      Kerberos Realm used by Oozie and Hadoop. Using 'local.realm' aligns
with Hadoop configuration
    </description>
</property>

<property>
    <name>oozie.service.HadoopAccessorService.kerberos.principal</name>
    <value>mapr/<hostname>@${local.realm}</value>
    <description>
       Kerberos principal for Oozie service.
    </description>
</property>

<property>
    <name>oozie.authentication.kerberos.principal</name>
    <value>HTTP/<hostname>@${local.realm}</value>
    <description>
      Indicates the Kerberos principal to be used for the HTTP endpoint.
The principal MUST start with 'HTTP/' per the Kerberos HTTP SPNEGO
specification.
    </description>
</property>
```

- Optional: If you plan to run Oozie actions that require talking to external services, add the `oozie.credentials.credentialclasses` to `oozie-site.xml`. For more details, see the Oozie documentation.

```
<property>
    <name>oozie.credentials.credentialclasses</name>
    <value>
         hcat=org.apache.oozie.action.hadoop.HCatCredentials,
         hbase=org.apache.oozie.action.hadoop.HbaseCredentials,
         hive2=org.apache.oozie.action.hadoop.Hive2Credentials
    </value>
</property>
```

Note: No specific configuration is required for configuring Oozie to use MapR-SASL.

Disabling Cached Tokens

After a client authenticates to Oozie, the authentication token received by the client is cached in the user's home directory in the `.oozie-auth-token` file. As long as the cached token remains valid, future authentication requests from the same client use that token and succeed, even if the client's Kerberos or MapR credentials have expired or have been revoked. You can disable use of the cache file by using the `oozie` command-line interface with the `-Doozie.auth.token.cache false` option.

Configuring Security Headers for Web Servers for Oozie

This section describes how to configure response headers for REST API servers used in the Oozie web UI.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

About the Headers File

The XML file with security headers is located at:

```
/opt/mapr/oozie/oozie-<version>/conf/security-headers.xml
```

The `security-headers.xml` file contains the following headers:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE properties SYSTEM
"http://java.sun.com/dtd/properties.dtd">
<properties>
    <comment>Security headers that is used to minimize the possibility of
cross-site scripting and other attacks</comment>
    <entry key="X-XSS-Protection">1; mode=block</entry>
    <entry key="X-Content-Type-Options">nosniff</entry>
    <entry
key="Strict-Transport-Security">max-age=31536000;includeSubDomains</entry>
    <entry key="Content-Security-Policy">default-src https:</entry>
</properties>
```

This table describes each header:

Header	Description	Default Value
X-XSS-Protection	Stops pages from loading when reflected cross-site scripting (XSS) is detected. Supported by IE, Chrome, and Safari.	`1: mode=block`

last-updated: Apr 11, 2022

Header	Description	Default Value
X-Content-Type-Options	Indicates that the MIME types advertised in the `Content-Type` headers should not be changed and should be followed.	`nosniff`
Strict-Transport-Security	Tells all browsers that the website should only be accessed using HTTPS instead of using HTTP.	`max-age=31536000;includeSubDomains`
Content-Security-Policy	Allows web-site administrators to control resources the user agent is allowed to load for a given page. This helps guard against cross-site scripting attacks (XSS).	`default-src https:`

Configuring Security Headers for Oozie

To enable security headers for Oozie, add the following to the `oozie-site.xml` file, and replace `<version>` with your Oozie version:

```
<property>
<name>oozie.server.response.headers</name>
<value>/opt/mapr/oozie/oozie-<version>/conf/security-headers.xml</value>
</property>
```

Configuring Custom Headers

To configure custom headers for web servers, edit the `headers.xml` file, and add `Custom-header` as follows:

```
<entry key="Custom-header">custom-value</entry>
```

Security Headers Auto-Configuration

If you install Oozie on a secure cluster (MapR-SASL or Kerberos) and run the following command after Oozie installation, Oozie automatically configures itself to enable security headers, and no additional action is needed:

```
/opt/mapr/server/configure.sh -R
```

Configure High Availability for Oozie

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

To configure HA for Oozie on your cluster, the cluster must meet the following prerequisites:

- Your cluster must have access to a database with support for multiple concurrent connections. To prevent this database from becoming a single point of failure, the database must support HA. HA for the Oozie service works regardless of the database's HA status. See MySQL Data Store for Oozie.
- To prevent the ZooKeeper service from becoming a single point of failure, your cluster must have at least 3 ZooKeeper nodes. You can still configure HA for Oozie on clusters with a single ZK node.
- Multiple nodes on the cluster must have Oozie installed. For installation instructions, see the Installing MapR and MapR Ecosystem Components on page 128.

  Note: For greater consistency of behavior on your cluster, verify that all of the Oozie servers have the same configuration.

- A load balancer, virtual IP, or round-robin DNS set up, such as HAProxy. To prevent the load balancer from becoming a single point of failure, the load balancer must support HA.

1. Verify that the Oozie servers are all configured to connect to the same database. Do not start Oozie.
2. On each Oozie node, edit the `oozie-site.xml` file to add the following section, which changes the results in Oozie using the Zookeeper's version of the services, overriding the default implementations:

```
<property>
    <name>oozie.services.ext</name>
    <value>
        org.apache.oozie.service.ZKLocksService,
        org.apache.oozie.service.ZKXLogStreamingService,
        org.apache.oozie.service.ZKJobsConcurrencyService,
        org.apache.oozie.service.ZKUUIDService
    </value>
</property>
```

3. On each Oozie node, edit the `oozie-site.xml` file to include a comma-separated list of the host names and ports for the ZooKeeper servers. For example:

```
<property>
     <name>oozie.zookeeper.connection.string</name>
     <value>zk1:5181,zk2:5181,zk3:5181</value>
</property>
```

4. On each Oozie node, edit the `oozie-site.xml` file to include the Oozie server host name:

```
<property>
     <name>oozie.http.hostname</name>
     <value><FQDN></value>
</property>
```

5. On each Oozie node, edit the `oozie-site.xml` file to specify the namespace. Each Oozie server that communicates to other Oozie servers must use the same namespace:

```
<property>
     <name>oozie.zookeeper.namespace</name>
     <value>oozie</value>
</property>
```

6. On each Oozie node, change the value of the `OOZIE_BASE_URL` property in the `oozie-site.xml` file to point to the load balancer or virtual IP.

```
<property>
    <name>oozie.base.url</name>
    <value>http://my.loadbalancer.hostname:<oozie_port_number>/oozie</
value>
</property>
```

   The `<oozie_port_number>` depends on whether your cluster is secure.

last-updated: Apr 11, 2022

7. On all nodes, update the services line in the `warden.oozie.conf` file (stored at `/opt/mapr/conf/conf.d`; or for a fresh installation, at `/opt/mapr/oozie/oozie-<version>/conf/warden.oozie.conf`) from:

   ```
   services=oozie:1:cldb
   ```

   to

   ```
   services=oozie:all:cldb
   ```

8. Run the `configure.sh -R` command.

9. Run Oozie share lib update command to make sure that all Oozie services use the latest and the same version:

   ```
   {OOZIE_HOME}/bin/oozie admin -oozie="http(s)://
   my.loadbalancer.hostname:<oozie_port_number>" -sharelibupdate
   ```

Configure a MySQL Data Store for Oozie

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Follow these steps to configure Oozie to use a MySQL database as the Oozie data store (instead of the default Apache Derby database). The MySQL database can also be used to support high availability (HA).

1. Using the `mysql` command-line tool, create a MySQL database, user, and password for the Oozie user. For example:

   ```
   mysql> create database oozie;
   mysql> grant all privileges on oozie.* to '<oozie-user>'@'%' identified
   by '<oozie-passwd>';
   ```

2. Stop the Oozie server:

   ```
   maprcli node services -name oozie -action stop -nodes <node-list>
   ```

3. Set the following `JPAService` properties in the `oozie-site.xml` file: **Note**: In the JDBC URL property, use the correct hostname (where MySQL is running).

   ```
   <property>
       <name>oozie.service.JPAService.jdbc.driver</name>
       <value>com.mysql.jdbc.Driver</value>
   </property>
   <property>
       <name>oozie.service.JPAService.jdbc.url</name>
       <value>jdbc:mysql://localhost:3306/oozie</value>
       </property>
   <property>
       <name>oozie.service.JPAService.jdbc.username</name>
       <value>oozie</value>
   </property>
   <property>
       <name>oozie.service.JPAService.jdbc.password</name>
       <value>oozie</value>
   </property>
   ```

4. For earlier EEP 4.1.0 versions, copy the MySQL JDBC driver `mysql-connector-java-<version>-bin.jar` file to the following directory.

   ```
   /opt/mapr/oozie/oozie-<oozieversion>/libext
   ```

   For EEP 4.1.0 and later, the JDBC driver file `mysql-connector-java-<version>-bin.jar` is automatically copied from the `/opt/mapr/lib/` directory to the `/opt/mapr/oozie/oozie-<oozieversion>/libext` directory.

   See the repository to confirm default MySQL JDBC driver in `/opt/mapr/oozie/oozie-<oozieversion>/libext` directory is compatible with your MySQL or MariaDB version. Download and update the default JDBC driver file with the driver file that is compatible with your MySQL or Maria DB version.

5. Start the Oozie server:

   ```
   maprcli node services -name oozie -action start -nodes <node-list>
   ```

6. Check that MySQL is now in use by looking at the contents of the `oozie.log` file. For example:

   ```
   cat /opt/mapr/oozie/oozie-<version>/logs/oozie.log |grep mysql
   2015-07-24 06:10:07,023  INFO JPAService:541 - SERVER[local.novalocal]
   USER[-]
   GROUP[-] TOKEN[-] APP[-] JOB[-] ACTION[-] JPA configuration:
   DriverClassName=com.mysql.jdbc.Driver,Url=jdbc:mysql://localhost:3306/
   oozie,Username=oozie,...
   ```

7. To encrypt the Oozie database user password, see Encrypt the Oozie Database User Password on page 3997.

Configuring an Oracle Schema
You must create schemas for Oracle databases manually.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Create an Oozie SQL-with-schema file to reveal the schema:

```
bin/ooziedb.sh create -sqlfile oozie.sql -run
```

For more information, see the Apache Oozie documentation.

Encrypt the Oozie Database User Password

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Follow these steps to encrypt the password when Oozie uses a MySQL database as the Oozie data store (instead of the default Apache Derby database).

1. Configure Oozie to use a MySQL database as described in Configure a MySQL Data Store for Oozie on page 3996.

2. [OPTIONAL] Export the Hadoop credential store password as a system variable:

   ```
   $ export HADOOP_CREDSTORE_PASSWORD=password
   ```

3. Add `oozie.service.jpaservice.jdbc.password` to the `jceks` keystore:

```
$ hadoop credential create
oozie.service.jpaservice.jdbc.password -provider jceks://path/to/
oozie.jceks
Enter the password:
Enter the password again:
oozie.service.jpaservice.jdbc.password has been successfully created.
org.apache.hadoop.security.alias.JavaKeyStoreProvider has been updated.
```

4. Verify that the MySQL password was added:

```
Keystore type: JCEKS
Keystore provider: SunJCE

Your keystore contains 1 entry

Alias name: oozie.service.jpaservice.jdbc.password
Creation date: Apr 11, 2018
Entry type: SecretKeyEntry
```

5. Once the `jceks` file is created, add the `hadoop.security.credential.provider.path` property to the `oozie-site.xml` file with the path to the `jceks` file. The `jceks` path location can be `maprfs` or a local file (`local-fs`).

```
<property>
   <name>hadoop.security.credential.provider.path</name>
   <value>jceks://path/to/oozie.jceks</value>
</property>
```

6. Update the `password` property to use `*****` instead of a word-readable password:

```
<property>
  <name>oozie.service.JPAService.jdbc.password</name>
  <value>*****</value>
</property>
```

Encrypt the Oozie Keystore Password

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Starting from Oozie 5.1.0.0, follow these steps to encrypt the keystore password when Oozie is configured to use SSL.

Note: Oozie 5.1.0.0 is configured to use SSL by default on secure clusters.

1. [OPTIONAL] Export the Hadoop credential store password as a system variable:

```
$ export HADOOP_CREDSTORE_PASSWORD=password
```

2. Add `oozie.https.keystore.pass` to the `jceks` keystore:

```
$ hadoop credential create oozie.https.keystore.pass -provider jceks://
path/to/oozie.jceks
Enter the password:
Enter the password again:
oozie.https.keystore.pass has been successfully created.
org.apache.hadoop.security.alias.JavaKeyStoreProvider has been updated.
```

3. Once the `jceks` file is created, add the `hadoop.security.credential.provider.path` property to the `oozie-site.xml` file along with the path to the jceks file. The jceks path location can be `maprfs` or a local file (`local-fs`).

```
<property>
   <name>hadoop.security.credential.provider.path</name>
   <value>jceks://path/to/oozie.jceks</value>
</property>
```

4. Update the `password` property to use `*****` instead of a word-readable password:

```
<property>
  <name>oozie.https.keystore.pass</name>
  <value>*****</value>
</property>
```

Note: You can use the same `jceks` file for storing both database and keystore passwords.

User Impersonation for Oozie

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Oozie supports *impersonation*, which enables Oozie to run jobs as a user other than the user that started the Oozie server. You can set up proxy user functionality if you want Oozie to impersonate a user from a set of hosts, or to impersonate a user that belongs to a set of groups. When you configure proxy user functionality, the proxy user can perform “doAs” operations. Add configuration properties to the oozie-site.xml and core-site.xml files to configure proxy user functionality.

Add the following configuration properties to the oozie-site.xml file:

- `oozie.service.ProxyUserService.proxyuser.#USER#.hosts`
- `oozie.service.ProxyUserService.proxyuser.#USER#.groups`

Replace #USER# with the username of the proxy that can perform “doAs” operations. For the host property, you can add a list of host names as the value. For the group property, you can add a list of groups as the value. Alternatively, you can add a wildcard character (*) as the value for host and group properties. To add multiple users, copy the property and replace #USER# with the proxy user name.

Host Example

```
<property>
        <name>oozie.service.ProxyUserService.proxyuser.mapr.hosts</name>
        <value>*</value>
</property>
```


last-updated: Apr 11, 2022

Group Example

```
<property>
        <name>oozie.service.ProxyUserService.proxyuser.mapr.groups</name>
        <value>*</value>
</property>
```

Add the following configuration properties to the core-site.xml:

- `hadoop.proxyuser.#USER#.hosts`
- `hadoop.proxyuser.#USER#.groups`

Replace #USER# with the username of the proxy.

When you add the host property, the proxy user can only connect from a host to impersonate a user. When you add the group property, the proxy user can impersonate any member of any group.

Host Example

```
<property>
    <name>hadoop.proxyuser.mapr.hosts</name>
    <value>*</value>
</property>
```

Group Example

```
<property>
    <name>hadoop.proxyuser.mapr.groups</name>
    <value>*</value>
</property>
```

Updating the Oozie Shared Libraries

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You must update the Oozie shared libraries when:

- the `oozie.service.WorkflowAppService.system.libpath` property in the `oozie-site.xml` file is changed from the default value (`/oozie/share/lib`).
- a core patch is installed.

By default, the Oozie shared libraries are uploaded to the `maprfs:///oozie/share/lib` directory.

To update the Oozie shared libraries, complete the following steps:

1. Run the following command to copy the new Oozie shared libraries to the MapR filesystem:

   ```
   {OOZIE_HOME}/bin/oozie-setup.sh sharelib create -fs
   maprfs:/// -locallib /opt/mapr/oozie/oozie-<version>/share
   ```

2. Run the following command to update the Oozie classpath with the new shared libraries:

   ```
   {OOZIE_HOME}/bin/oozie admin -sharelibupdate -oozie https://<FQDN>:11443/
   oozie
   ```


last-updated: Apr 11, 2022

Use Oozie

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following sections provide information about using Oozie:

Manage Oozie Services and Interface

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You can manage Oozie services and the Oozie interface through the command line or Oozie Web UI.

The following sections provide information about managing Oozie:

Starting, Stopping, and Restarting Oozie Services

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The Warden daemon starts the Oozie server automatically at installation time. You can start and stop Oozie from the command line or from the Control System. Using the `maprcli node services` command enables you to start Oozie on multiple nodes at one time.

Starting, Stopping, and Restarting Oozie using maprcli

Complete the following steps to start | stop | restart Oozie from the command line:

1. Make a list of nodes on which Oozie is configured.
2. Issue the `maprcli node services` command with either `start`, `restart`, or `stop`, and specify the nodes on which Oozie is configured, separated by spaces.

   Example:

   ```
   maprcli node services -name oozie -action start|stop|restart -nodes
   <nodes list>
   ```

Enabling the Oozie Web UI

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The Oozie web UI can display your job status, logs, and other related information. You must enable the Oozie Web UI after you install Oozie. However, depending on how you enable the Web UI, you may also need to perform manual steps to start the Web UI after you run `configure.sh` on the cluster.

1. Download the Ext JS 2.2 library as a zip archive (`ext-2.2.zip`). You can download this library from the Apache Oozie Quick Start guide. Search for "ExtJS 2.2" on the page to find the link.

   Note: The Ext JS 2.2 library is required only for the Oozie Web UI. A rewrite of the Oozie web UI has been proposed in the Oozie user community.

2. Copy the Ext JS 2.2 library into the `libext` directory:

   ```
   cp ext-2.2.zip /opt/mapr/oozie/oozie-<version>/libext/
   ```

3. **Applicable only for Oozie 4.X versions:** Perform one of the following options based on your preference:
   - If you want the Oozie Web UI to start whenever Oozie starts or restarts, perform the following steps:

last-updated: Apr 11, 2022

a. If Oozie is running, shut it down:

```
maprcli node services -name oozie -action stop -nodes <space
delimited list of nodes>
```

b. Run the oozie-setup.sh script, and specify the hadoop version.

```
/opt/mapr/oozie/oozie-<version>/bin/oozie-setup.sh -hadoop
<version> /opt/mapr/hadoop/hadoop-<version>
```

Note: If you are enabling the Oozie Web UI on a secure cluster and SSL must be configured, run the command with the `-secure` option:

```
/opt/mapr/oozie/oozie-<version>/bin/oozie-setup.sh -hadoop
<version> /opt/mapr/hadoop/hadoop-<version> -secure
```

- If you want to manually start the Web UI when Oozie starts or restarts, perform the following steps:

a. If Oozie is running, shut it down:

```
maprcli node services -name oozie -action stop -nodes <space
delimited list of nodes>
```

b. Run the `oozie-setup.sh` script and specify the path to the `extjs` file.

```
cd /opt/mapr/oozie/oozie-<version>
bin/oozie-setup.sh prepare-war -extjs ~/ext-2.2.zip
```

4. Restart Oozie.

```
maprcli node services -name oozie -action restart -nodes <space
delimited list of nodes>
```

Checking the Status of Oozie

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Once Oozie is installed, you can check the status using the command line or the Oozie web console. The `<oozie_port_number>` noted in the URLs depend on whether your cluster is secure.

Check Status from Command Line

Use the `oozie admin` command:

```
/opt/mapr/oozie/oozie-<version>/bin/oozie admin -oozie http(s)://
<oozie_node>:<oozie_port_number>/oozie -status
```

The following output indicates normal operation:

```
System mode: NORMAL
```

Check Status from the Oozie Web Console

Point your browser to `http://<oozie_node>:<oozie_port_number>/oozie`

Note:

- For non-secure clusters:

```
Oozie url: http://<oozie_node>:11000/oozie
```

- For secure clusters:

```
Oozie url: https://<oozie_node>:11443/oozie
```

Setup and Run Oozie Examples

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

To get familiar with Oozie, set up and try the examples.

1. Complete the following steps to set up the examples and copy them to the cluster:
   a) Extract the oozie examples archive `oozie-examples.tar.gz`:

```
cd /opt/mapr/oozie/oozie-<version>
tar xvfz ./oozie-examples.tar.gz -C /opt/mapr/oozie/oozie-<version>/
```

   b) Copy the examples to MapR filesystem. Run the following command as a user that has permissions to write to the specified MapR filesystem directory:

```
hadoop fs -put examples maprfs:///user/<user_name>/examples
```

2. Complete the following steps to run the examples:
   a) Choose an example and run it with the `oozie job` command. You can use the following commands to run the following examples. The `<oozie_port_number>` depends on whether your cluster is secure.
      - MapReduce

```
/opt/mapr/oozie/oozie-<version>/bin/oozie job -oozie="http://
localhost:<oozie_port_number>/oozie" -config /opt/mapr/oozie/
oozie-<version>/examples/apps/map-reduce/job.properties -run
```

      - Spark

```
/opt/mapr/oozie/oozie-<version>/bin/oozie job -oozie="http://
localhost:<oozie_port_number>/oozie" -config /opt/mapr/oozie/
oozie-<version>/examples/apps/spark/job.properties -run
```

      Note: To run the packaged Hive examples, make `/tmp` on MapR filesystem world-writable. Set `/tmp` to 777. Example:

```
hadoop fs -chmod -R 777 /tmp
```

      If /tmp does not exist, create `/tmp` and then set it to 777.

last-updated: Apr 11, 2022

Check the Status of a Job

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You can check the status of a job from the command line or the Oozie Web Console.

From the command line, issue the following (substituting the job ID for the `<job id>` placeholder):

```
# /opt/mapr/oozie/oozie-<version>/bin/oozie job -info <job id>
```

To check status from the Oozie Web Console, point your browser to `http://<Oozie_node>:<oozie_port_number>/oozie` and click **All Jobs**. The `<oozie_port_number>` depends on whether your cluster is secure.

Run Hive Jobs with Oozie

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You run Hive jobs with Oozie by configuring a Hive workflow.

1. **Configure a Hive workflow.** You can configure Oozie to perform a workflow by connecting to Hive Metastore or Hiveserver2.

   Configure a Hive Workflow with Connection to Hive Metastore

   a. To use a metastore server for the Hive job, add the following parameter to the hive-site.xml file:

   ```
   <property>
     <name>hive.metastore.uris</name>
     <value>thrift://<IP address>:<port></value>
     <description>IP address (or fully-qualified domain name) and port
   of the metastore host</description>
   </property>
   ```

   b. Copy the edited `hive-site.xml` file to the same location as your `workflow.xml` file.

   c. If you are using the Hive-on-Tez engine, and you have changed the default `tez-site.xml` configuration, perform one of the following steps:

   - Copy the tez-site.xml file to the same location as your workflow.xml file:

     1. Remove the forbidden (forbidden for Oozie) property from `tez-site.xml`:

     ```
     <property>
       <name>fs.defaultFS</name>
       <value>maprfs:///</value>
     </property>
     ```

     2. Make sure that you update the value for `tez.lib.uris` property after removing the `fs.defaultFS` property. For example:

     ```
     tez.lib.uris=maprfs:///apps/tez/tez-<version>,maprfs:///apps/tez/
     tez-<version>/lib
     ```

     3. Specify the `tez-site.xml` file in the `job-xml` parameter of your workflow.

   - Update `<OOZIE_HOME>/conf/action-conf/hive.xml` with the new `tez-site.xml` properties.

last-updated: Apr 11, 2022

d. Edit the `workflow.xml` file to include the following:

1. Specify the `hive-site.xml` in the `job-xml` parameter.
2. Specify the name of the script (for example, `script.q`) that contains the hive query in the script parameter.
3. Optionally, add properties used by the Oozie launcher job. Add the prefix `oozie.launcher` to the property names.

```
<workflow-app xmlns="uri:oozie:workflow:0.2" name="hive-wf">
    <start to="hive-node"/>
    <action name="hive-node">
        <hive xmlns="uri:oozie:hive-action:0.2">
            <job-tracker>${jobTracker}</job-tracker>
            <name-node>${nameNode}</name-node>
            <prepare>
                <delete path="${nameNode}/user/${wf:user()}/$
{examplesRoot}/output-data/hive"/>
                <mkdir path="${nameNode}/user/${wf:user()}/$
{examplesRoot}/output-data"/>
            </prepare>
            <job-xml>hive-site.xml</job-xml>
            <!- Add this property if you copied tez-site.xml to the
same location as your workflow.xml file -->
            <job-xml>tez-site.xml</job-xml>
            <configuration>
                <property>
                    <name>mapred.job.queue.name</name>
                    <value>${queueName}</value>
                </property>
            </configuration>
            <script>script.q</script>
            <param>INPUT=/user/${wf:user()}/${examplesRoot}/
input-data/table</param>
            <param>OUTPUT=/user/${wf:user()}/${examplesRoot}/
output-data/hive</param>
        </hive>
        <ok to="end"/>
        <error to="fail"/>
    </action>

    <kill name="fail">
        <message>Hive failed, error message[$
{wf:errorMessage(wf:lastErrorNode())}]</message>
    </kill>
    <end name="end"/>
</workflow-app>
```

Configure a Hive Workflow with Connection to HiveServer2

a. Copy the edited `hive-site.xml` file to the same location as your `workflow.xml` file.

b. On a Kerberos secure cluster for Oozie 4.3.0, perform the following steps:

1. Copy the `hive-site.xml` file to the `${OOZIE_HOME}/conf/action-conf/` directory.
2. Rebuild the Oozie war file:

```
/opt/mapr/oozie/oozie-<version>/bin/oozie-setup.sh
-hadoop <version> /opt/mapr/hadoop/hadoop-<version>
```


last-updated: Apr 11, 2022

c. Edit the `workflow.xml` file to include the following:

1. Specify the JDBC URL used by Beeline for connections to Hiveserver2 in the `jdbc-url` element. See Connecting to HiveServer2 for details.

2. Specify the name of the script (for example, `script.q`) that contains the hive query in the script element.

```
<?xml version="1.0" encoding="UTF-8"?>
<workflow-app xmlns="uri:oozie:workflow:0.5" name="hive2-wf">
    <start to="hive2-node"/>
    <action name="hive2-node">
        <hive2 xmlns="uri:oozie:hive2-action:0.1">
            <job-tracker>${jobTracker}</job-tracker>
            <name-node>${nameNode}</name-node>
            <prepare>
                <delete path="${nameNode}/user/${wf:user()}/
output-data/hive2"/>
                <mkdir path="${nameNode}/user/${wf:user()}/
output-data"/>
            </prepare>
            <configuration>
                <property>
                    <name>mapred.job.queue.name</name>
                    <value>${queueName}</value>
                </property>
            </configuration>
            <jdbc-url>jdbc:hive2://localhost:10000/default</
jdbc-url>
            <script>script.q</script>
            <param>INPUT=/user/${wf:user()}/input-data/table</
param>
            <param>OUTPUT=/user/${wf:user()}/output-data/hive2</
param>
        </hive2>
        <ok to="end"/>
        <error to="fail"/>
    </action>
    <kill name="fail">
        <message>Hive2 (Beeline) action failed, error message[$
{wf:errorMessage(wf:lastErrorNode())}]</message>
    </kill>
    <end name="end"/>
</workflow-app>
```

Run Spark Jobs with Oozie

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Complete the following steps to configure Oozie to run Spark jobs:

Configure a Spark action:

1. For running a Spark action through Oozie, you should be able to connect to Hive on a secure cluster. Make sure the `hive-site.xml` file that is used by Oozie has the following property set:

```
<property>
  <name>hive.metastore.sasl.enabled</name>
  <value>true</value>
</property>
```

2. To add Spark configuration files (`spark-defaults.conf`, `hive-site.xml`, etc) to a Spark action, copy the files to the `{OOZIE_HOME}/share/lib/spark/` directory.

3. If needed, update the Oozie shared libraries as described in Updating the Oozie Shared Libraries on page 4000.

4. Run pySpark using Spark Action:

   a. Run pySpark using Spark Action by specifying pyspark and py4j zip files to the sharelib:

   ```
   cp /{SPARK_HOME}/python/lib/ pyspark*.zip {OOZIE_HOME}/share/lib/
   spark/
   cp /{SPARK_HOME}/python/lib/py4j*src.zip {OOZIE_HOME}/share/lib/spark/
   ```

   b. Update the Oozie shared libraries as described in Updating the Oozie Shared Libraries on page 4000.

5. When you configure a Spark action in the `workflow.xml`, specify the `master` and `mode` elements of the Spark job:

   - For Spark standalone mode, specify the Spark Master URL in the `master` element. For example, if your SparkMaster URL is `spark://ubuntu2:7077`, you would replace the `<master> [SPARK MASTER URL]</master>` in the example below with `<master> spark://ubuntu2:7077</master>`.

   - For Spark on YARN mode, specify `yarn-client` or `yarn-cluster` in the `master` element. For example, for `yarn-cluster` mode, you would replace `<master> [SPARK MASTER URL]</master>` with `<master>yarn</master>` and `<mode>[SPARK MODE]</mode>` with `<mode>cluster</mode>`.

     Here is an example of a Spark action within a `workflow.xml` file:

```
<workflow-app xmlns='uri:oozie:workflow:0.5' name='SparkFileCopy'>
  <start to='spark-node' />
  <action name='spark-node'>
     <spark xmlns="uri:oozie:spark-action:0.1">
        <job-tracker>${jobTracker}</job-tracker>
        <name-node>${nameNode}</name-node>
        <master>[SPARK MASTER URL]</master>
        <mode>[SPARK MODE]</mode>
        <name>Spark-FileCopy</name>
        <class>org.apache.oozie.example.SparkFileCopy</class>
        <jar>${nameNode}/user/${wf:user()}/${examplesRoot}/apps/
spark/lib/oozie-examples.jar</jar>
        <arg>${nameNode}/user/${wf:user()}/${examplesRoot}/input-data/
text/data.txt</arg>
        <arg>${nameNode}/user/${wf:user()}/${examplesRoot}/output</arg>
      </spark>
      <ok to="end" />
      <error to="fail" />
   </action>
   <kill name="fail">
      <message>Workflow failed, error
               message[${wf:errorMessage(wf:lastErrorNode())}]
      </message>
   </kill>
  <end name='end' />
</workflow-app>
```


last-updated: Apr 11, 2022

Oozie 5.1.0 API Changes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Oozie 5.1.0 includes the following changes in the Web Services API. The complete documentation is available at http://oozie.apache.org/docs/5.1.0/WebServicesAPI.html

API	Description
Get missing dependencies	This endpoint is to show all missing dependencies of coordinator action(s).
Purge Command	Oozie admin purge command cleans up the Oozie Workflow, Coordinator, or Bundle records based on the specified parameters. The unit for parameters is `days`. The purge command deletes the workflow records (`wf=30`) older than 30 days, coordinator records (`coord=7`) older than 7 days, and bundle records (`bundle=7`) older than 7 days. The limit (`limit=10`) defines the number of records to be fetched at a given time. Turn on or off (`oldCoordAction true or false`) enables coordinator action record purging for long running coordinators. If any of these parameters are not specified, it is taken from the `oozie-default` or `oozie-site` configuration file.

Oozie 4.3.0 API Changes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Oozie 4.3.0 includes the following changes in the Web Services API. Full documentation is available at http://oozie.apache.org/docs/4.3.0/WebServicesAPI.html

API	Description
Validate a local file	This endpoint is to validate a workflow, coordinator, bundle XML file.
Validate a file in MapR File System	You can validate a workflow, coordinator, bundle XML file in MapR File System. The XML file must already exist in MapR File System.

Oozie Known Issues

This topic describes known issues related to Oozie.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Spark Oozie Share Library Update Required if Core Patch Applied

If you installed a Core patch for EEP version 6.3.0, 6.3.1, or 7.0.0, the Spark Oozie Share Lib must be updated with the `maprbuildversion` JAR either manually or through an Oozie upgrade.

- When you upgrade to Oozie in EEP 6.3.2, 7.0.1, or later and run `/opt/mapr/server/configure.sh -R`, the `maprbuildversion` JAR is automatically copied from `$MAPR_HOME/lib` to `$OOZIE_HOME/share/lib/spark`.
- Alternatively, you can manually copy the `maprbuildversion` JAR from `$MAPR_HOME/lib` to `$OOZIE_HOME/share/lib/spark` and then run `/opt/mapr/server/configure.sh -R`.

last-updated: Apr 11, 2022

Oozie Share Library Update Required if Core Patch Applied

After a core patch installation, update the Oozie shared libraries under the maprfs directory, as described in Updating the Oozie Shared Libraries on page 4000.

Pig

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Apache Pig™ is a platform for analyzing large data sets. Apache Pig™ is a high-level language for expressing data analysis programs coupled with an infrastructure to evaluate these programs. The infrastructure layer consists of a compiler that produces sequences of Map-Reduce programs. The language layer consists of a textual language called *Pig Latin*.

The following sections provide documentation about installing, upgrading, configuring and using Pig. You can also refer to the documentation available on the Apache Pig website.

Configure Pig

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Pig runs in MapReduce mode by default. If you want to run Pig in local mode (`pig -x local`), add the following properties to the `/opt/mapr/pig/pig-<version>/conf/pig.properties file:`

```
fs.file.impl=org.apache.hadoop.fs.LocalFileSystem
io.file.buffer.size=4096
```

The `io.file.buffer.size` property defines how much data is buffered during read and write operations. This value defaults to `-1`, which causes Pig to fail if the local filesystem is used.

Use Pig

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Once Pig is installed, the executable is located in `/opt/mapr/pig/pig-<version>/bin/pig`.

The following topics provide information about using Pig. You can also refer to the documentation available on the Apache Pig website.

Get Started with Pig

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

In this tutorial, we will use Pig to run a MapReduce application that counts the words in the `/in/constitution.txt` file located in the `mapr` user's directory on the cluster, and store the results in the file `wordcount.txt`.

1. Download the ZIP file that contains constitution.txt and then extract the constitution.txt file.
2. Load the file onto the cluster and place it in the directory `/user/mapr/in`.
3. In the terminal, type the command `pig` to start the Pig shell.
4. At the `grunt>` prompt, type the following lines (press ENTER after each): After you type the last line, Pig starts a MapReduce application to count the words in the file `constitution.txt`.

```
A = LOAD '/user/mapr/in' USING TextLoader() AS (words:chararray);
```

```
B = FOREACH A GENERATE FLATTEN(TOKENIZE(*));
```

```
C = GROUP B BY $0;
```

```
D = FOREACH C GENERATE group, COUNT(B);
```

```
STORE D INTO '/user/mapr/wordcount';
```

5. When the MapReduce application is complete, type `quit` to exit the Pig shell and take a look at the contents of the directory `/myvolume/wordcount` to see the results.

Use ORC Storage with Pig

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

This section presents simple examples for using Hive ORC storage. Use the `grunt` shell in Pig to execute the commands. For more details, see the Pig documentation.

ORC is typically used to read (load) and write (store) data as follows:

```
<VAR_NAME> = load '/path/to/orc/formatted/file' using OrcStorage();
store <VAR_NAME> into '/path/to/output/orc/file' using OrcStorage('');
```

Example: Create an ORC file in the file system by storing the data in a Hive table and uploading it to Pig

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You can create an ORC format file in the file system by using Hive to load a text file into a table with ORC storage. Then, you can upload the resulting ORC format file to Pig.

1. Create a sample test data file:

```
cd /home/mapr
nano test_pig.data
chown mapr:mapr test_pig.data
```


last-updated: Apr 11, 2022

2. Add data to the file.

```
John,Smith
Brian,May
Rodger,Taylor
John,Deacon
Max,Plank
Freddie,Mercury
Albert,Einstein
Fedor,Dostoevsky
Lev,Tolstoy
Niccolo,Paganini
```

Note: Do not include any extra lines at the end of the file.

3. Upload the test data to a Hive table:

```
sudo -u mapr hive
hive> create table test_pig(first_name string, last_name string) ROW
FORMAT DELIMITED FIELDS TERMINATED BY ',';
hive> load data local inpath '/home/mapr/test_pig.data' overwrite into
table test_pig;
```

4. Create a Hive table with ORC storage:

```
hive> create table test_pig_orc(first_name string, last_name string)
stored as orc tblproperties ("orc.compress"="NONE");
hive> insert overwrite table test_pig_orc select * from test_pig;
hive> select * from test_pig_orc;
```

5. Check that the ORC file was created:

```
hadoop fs -ls /user/hive/warehouse/test_pig_orc
```

6. Upload the ORC file to Pig:

```
sudo -u mapr pig
grunt> B = load '/user/hive/warehouse/test_pig_orc/000000_0'
using OrcStorage();
grunt> dump B;
```

Example: ***Upload a text file to the file system and use Pig to save it as an ORC file***

1. Upload the file to the file system:

```
cd /home/mapr
sudo -u mapr hadoop fs -put ./test_pig.data /test_pig.data
sudo -u mapr hadoop fs -mkdir /output
```

2. Start Pig and save the text file in ORC format:

```
sudo -u mapr pig
grunt> A = LOAD '/test_pig.data' using PigStorage(',') AS
(first_name:chararray, last_name:chararray);
grunt> store A into '/output/A' using OrcStorage('');
```


last-updated: Apr 11, 2022

3. Verify that the ORC file was created:

```
hadoop fs -ls /output/A/
```

Integrate Pig

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

This following topics provide information about how to integrate Pig with other ecosystem components. You can also refer to the documentation available on the Apache Pig website.

Integrate Pig and HBase
This document shows an example of a Pig and HBase integration. The goal of integration is to upload data from the MapR File System to Pig and then move the data to an HBase table.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Configuring Pig and HBase

No additional configuration is needed to integrate HBase and Pig.

Pig and HBase Integration Example

1. Create sample data, and upload the data to the MapR File System:
   a. Create a sample data file:

```
vim input.csv
```

   b. Add data to the file:

```
1,aaa,bbb
2,ccc,ddd
3,rrr,fff
4,ttt,yyy
```

   c. Upload the data to the MapR File System:

```
hadoop fs -put input.csv /user/mapr/input.csv
```

2. Create a sample table in HBase:
   a. Start the HBase shell:

```
hbase shell
```

   b. Create a table:

```
hbase(main):012:0> create 'sample_names', 'info'
```

3. Load the data to Pig, and store the data in HBase:

a. Start the Pig shell:

```
pig
```

b. Load the data to Pig:

```
raw_data = LOAD '/user/mapr/input.csv' USING PigStorage(',') AS
(listing_id: chararray, fname: chararray, lname: chararray);
```

c. Store the data in HBase:

```
STORE raw_data INTO 'sample_names' USING
org.apache.pig.backend.hadoop.hbase.HBaseStorage ('info:fname
info:lname');
```

4. Verify the data in HBase:

a. Start the HBase shell:

```
hbase shell
```

b. Query the data:

```
hbase(main):017:0* scan 'sample_names'
```

The result is:

```
ROW
COLUMN+CELL


 1    column=info:fname, timestamp=1574946889082, value=aaa
 1    column=info:lname, timestamp=1574946889082, value=bbb
 2    column=info:fname, timestamp=1574946889091, value=ccc
 2    column=info:lname, timestamp=1574946889091, value=ddd
 3    column=info:fname, timestamp=1574946889091, value=rrr
 3    column=info:lname, timestamp=1574946889091, value=fff
 4    column=info:fname, timestamp=1574946889091, value=ttt
 4    column=info:lname, timestamp=1574946889091, value=yyy
```

Integrate Pig and MapR Database

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

To configure Pig to work with MapR Database tables, perform the following steps:

1. On the client node where Pig is installed, add the following string to `/opt/mapr/conf/env.sh`:

```
export PIG_CLASSPATH=$PIG_CLASSPATH:/location-to-hbase-jar
```

2. If the client node where Pig is installed also has the `mapr-hbase` package installed, add the location of the `hbase-<version>.jar` file to the `PIG_CLASSPATH` variable from the previous step:

```
export PIG_CLASSPATH="$PIG_CLASSPATH:/opt/mapr/hbase/hbase-<version>/
hbase-<version>.jar"
```

3. If the client node where Pig is installed does not have the `mapr-hbase` package installed, copy the HBase JAR from a node that does have HBase installed to a location on the Pig client node. Add the HBase JAR's location to the definition from previous steps:

```
export PIG_CLASSPATH=$PIG_CLASSPATH:/opt/mapr/lib/hbase-<version>.jar
```

4. Add the HBase JAR to the Hadoop classpath:

```
export HADOOP_CLASSPATH="/opt/mapr/hbase/hbase-<version>/
hbase-<version>-mapr.jar:$HADOOP_CLASSPATH"
```

5. Launch a Pig job and verify that Pig can access HBase tables by using the HBase table name directly. Do not use the `hbase://` prefix.

Pig 0.16.0 API

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

This section contains the following topics:

New API in Pig 0.16.0

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Pig 0.16.0 includes the following new classes and interfaces.

New Classes

Class	Description

org.apache.pig.piggybank.evaluation.string.REPLACE_MULTI	REPLACE_MULTI implements eval function to replace all occurrences of search keys with replacement values. Replacement values are specified in Map. For example:
	`input_data = LOAD 'input_data' as (name);`
	`-- name = 'Hello World!' replaced_name = FOREACH input_data \`
	`GENERATE REPLACE_MULTI ( name, [ ''#'_', '!'#'', 'e'#'a', 'o'#'oo' ] );`
	`-- replaced_name = Halloo_Woorld`
	The first argument is the source string on which `REPLACE_MULTI` operation is performed. The second argument is a map having search key with replacement value pairs.

Class	Description
org.apache.pig.piggybank.storage.apachelog.LogFormatLoader	This is a pig loader that can load Apache HTTPD access logs written in (almost) any Apache HTTPD LogFormat. *Basic usage:* Feed the loader your (custom) logformat specification and it will show the fields that can be extracted from this logformat.
org.apache.pig.CounterBasedErrorHandler	Handles errors thrown by the `StoreFuncInterface.putNext()`.
org.apache.pig.backend.hadoop.HKerberos	Support for logging in using a Kerberos keytab file. Kerberos is an authentication system that uses tickets with limited validity time. Running a Pig script on a Kerberos secured Hadoop cluster limits the running time to at most the remaining validity time of the Kerberos tickets. When doing really complex analytics, this may become a problem as the job may need to run for a longer time than these ticket times allow. A Kerberos keytab file is a Kerberos specific form of the password of a user. It is possible to enable a Hadoop job to request new tickets when they expire by creating a keytab file and making it part of the job that is running in the cluster. This will extend the maximum job duration beyond the maximum renew time of the Kerberos tickets.
org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.PigWritableComparators	Byte only raw comparators for faster comparison for non-orderby jobs. This does not reuse `JobControlCompiler.Pig<DataType>WritableComparator`, which extends `PigWritableComparator`. The `PigNullablePartitionWritable.compare` is not that efficient in cases where tuple is iterated for null checking instead of taking advantage of `TupleRawComparator.hasComparedTupleNull()`. This also skips multi-query index checking.
org.apache.pig.backend.hadoop.executionengine.physicalLayer.relationalOperators.StoreFuncDecorator	This class is used to decorate the `StoreFunc#putNext(Tuple)`. It handles errors by calling `OutputErrorHandler#handle(String, long, Throwable)` if the `StoreFunc` implements `ErrorHandling`.
org.apache.pig.backend.hadoop.executionengine.tez.runtime.PigInputFormatTez	Extends `org.apache.hadoop.mapreduce.InputFormat` and implements Pig and Tez specific functions.
org.apache.pig.backend.hadoop.executionengine.tez.util.TezUDFContextSeparator	Extends a visitor for the TezOperPlan class and serializes all (`LoadFunc`, `StoreFunc`, `UserFunc`).
org.apache.pig.impl.io.compress.BZip2CodecWithExtensionBZ	For historical reasons, Pig supports `.bz` and `.bz2` for `bzip2` extension. This class returns the additional `bzip2` file extension, `.bz`, as a string.
org.apache.pig.impl.util.UDFContextSeparator	`TezUDFContextSeparator` extends `PhyPlanVisitor`, which is the visitor class for the Physical Plan. To use this, create the visitor with the plan to be visited. Call the `visit()` method to traverse the plan in a depth first fashion. This class also visits the nested plans inside the operators. Extend this class to modify the nature of each visit and to maintain any relevant state information between the visits to two different operators.
org.apache.pig.parser.RegisterResolver	Resolves a JAR with a scripting language or namespace.

last-updated: Apr 11, 2022

Class	Description
org.apache.pig.tools.DownloadResolver	Makes a list of URIs of the downloaded JARs.

New Interfaces

Interface	Description
org.apache.pig.ErrorHandler	The interface that handles errors thrown by `StoreFuncInterface.putNext(Tuple)`.
org.apache.pig.ErrorHandling	The interface to enable handling of errors during `StoreFunc#putNext(Tuple)`.

Deprecated API in Pig 0.16.0

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following interfaces, classes, fields, and methods have been deprecated in Pig 0.16.0.

Deprecated Interfaces

Instead of this deprecated interface...	Use...
org.apache.pig.PigToStream	PigStreamingBase
org.apache.pig.StreamToPig	PigStreamingBase

Deprecated Classes

Instead of this deprecated class...	Use...
org.apache.pig.piggybank.evaluation.math.ABS	ABS
org.apache.pig.piggybank.evaluation.math.ACOS	ACOS
org.apache.pig.builtin.ARITY	SIZE
org.apache.pig.piggybank.evaluation.math.ASIN	ASIN
org.apache.pig.piggybank.evaluation.math.ATAN	ATAN
org.apache.pig.piggybank.evaluation.math.Base	Base
org.apache.pig.piggybank.evaluation.math.CBRT	CBRT
org.apache.pig.piggybank.evaluation.math.CEIL	CEIL
org.apache.pig.ComparisonFunc	N/A
org.apache.pig.piggybank.evaluation.stats.COR	COR
org.apache.pig.piggybank.evaluation.math.COS	COS
org.apache.pig.piggybank.evaluation.math.COSH	COSH
org.apache.pig.piggybank.evaluation.stats.COV	COV
org.apache.pig.data.DefaultTupleFactory	TupleFactory
org.apache.pig.piggybank.evaluation.math.DoubleAbs	DoubleAbs
org.apache.pig.piggybank.evaluation.math.DoubleBase	DoubleBase
org.apache.pig.piggybank.evaluation.math.DoubleRound	DoubleRound

last-updated: Apr 11, 2022

Instead of this deprecated class...	Use...
org.apache.pig.piggybank.evaluation.math.EXP	EXP
org.apache.pig.piggybank.evaluation.math.FloatAbs	FloatAbs
org.apache.pig.piggybank.evaluation.math.FloatRound	FloatRound
org.apache.pig.piggybank.evaluation.math.FLOOR	FLOOR
org.apache.pig.piggybank.evaluation.string.INDEXOF	INDEXOF
org.apache.pig.piggybank.evaluation.math.IntAbs	IntAbs
org.apache.pig.piggybank.storage.JsonMetadata	N/A
org.apache.pig.piggybank.evaluation.string.LASTINDEXOF	LAST_INDEX_OF
org.apache.pig.piggybank.evaluation.string.LcFirst	LCFIRST
org.apache.pig.piggybank.evaluation.math.LOG	LOG
org.apache.pig.piggybank.evaluation.math.LOG10	LOG10
org.apache.pig.piggybank.evaluation.math.LongAbs	LongAbs
org.apache.pig.piggybank.evaluation.string.LOWER	LOWER
org.apache.pig.piggybank.storage.PigStorageSchema	PigStorage with `-schema` option
org.apache.pig.piggybank.evaluation.string.RegexExtract	REGEX_EXTRACT
org.apache.pig.piggybank.evaluation.string.RegexExtractAll	REGEX_EXTRACT_ALL
org.apache.pig.piggybank.evaluation.string.REPLACE	REPLACE
org.apache.pig.piggybank.evaluation.math.ROUND	ROUND
org.apache.pig.piggybank.evaluation.math.SIN	SIN
org.apache.pig.piggybank.evaluation.math.SINH	SINH
org.apache.pig.piggybank.evaluation.string.Split	STRSPLIT
org.apache.pig.piggybank.evaluation.math.SQRT	SQRT
org.apache.pig.piggybank.evaluation.string.SUBSTRING	SUBSTRING
org.apache.pig.piggybank.evaluation.math.TAN	TAN
org.apache.pig.piggybank.evaluation.math.TANH	TANH
org.apache.pig.piggybank.evaluation.util.ToBag	TOBAG
org.apache.pig.piggybank.evaluation.util.Top	TOP
org.apache.pig.piggybank.evaluation.util.ToTuple	TOTUPLE
org.apache.pig.piggybank.evaluation.string.Trim	TRIM
org.apache.pig.piggybank.evaluation.string.UcFirst	UCFIRST
org.apache.pig.piggybank.evaluation.string.UPPER	UPPER

last-updated: Apr 11, 2022

Instead of this deprecated class...	Use...
org.apache.pig.impl.util.WrappedIOException	**Note:** This class was introduced to overcome the limitation that before Java 1.6, IOException did not have a constructor which took a Throwable argument. Since Pig code is now compiled with Java 1.6 and EvalFunc and LoadFunc user implementations should also use Java 1.6, they can use IOException instead. From Java 1.6, IOException has constructors which take a Throwable argument.

Deprecated Fields

Instead of this deprecated field...	Use...
org.apache.pig.tools.pigstats.PigStatsUtil.FS_COUNTER_GROUP	MRPigStatsUtil.FS_COUNTER_GROUP
org.apache.pig.PigConfiguration.INSERT_ENABLED **Note:** Will be removed in Pig 0.16	PigConfiguration.PIG_SCRIPT_INFO_ENABLED.
org.apache.pig.PigConfiguration.MAX_SCRIPT_SIZE **Note:** Will be removed in Pig 0.16.	PigConfiguration.PIG_SCRIPT_MAX_SIZE
org.apache.pig.PigConfiguration.OPT_FETCH **Note:** Will be removed in Pig 0.16.	PigConfiguration.PIG_OPT_FETCH
org.apache.pig.PigConfiguration.PARTAGG_MINREDUCTION **Note:** Will be removed in Pig 0.16.	PigConfiguration.PIG_EXEC_MAP_PARTAGG_MINREDUCTION
org.apache.pig.PigConfiguration.PROP_CACHEDBAG_MEMUSAGE **Note:** Will be removed in Pig 0.16.	PigConfiguration.PIG_CACHEDBAG_MEMUSAGE
org.apache.pig.PigConfiguration.PROP_EXEC_MAP_PARTAGG **Note:** Will be removed in Pig 0.16.	PigConfiguration.PIG_EXEC_MAP_PARTAGG
org.apache.pig.PigConfiguration.PROP_NO_COMBINER **Note:** Will be removed in Pig 0.16.	#PROP_NO_COMBINER1
org.apache.pig.PigConfiguration.SCHEMA_TUPLE_SHOULD_ALLOW_FORCE	N/A
org.apache.pig.PigConfiguration.SCHEMA_TUPLE_SHOULD_USE_IN_FOREACH	N/A
org.apache.pig.PigConfiguration.SCHEMA_TUPLE_SHOULD_USE_IN_FRJOIN	N/A
org.apache.pig.PigConfiguration.SCHEMA_TUPLE_SHOULD_USE_IN_MERGEJOIN	N/A

Instead of this deprecated field...	Use...
org.apache.pig.PigConfiguration.SCHEMA_TUPLE_SHOULD_USE_IN_UDF	N/A
org.apache.pig.impl.PigContext.scriptFiles	N/A
org.apache.pig.PigConfiguration.SHOULD_USE_SCHEMA_TUPLE	N/A
org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.PigInputFormat.sJob	UDFContext in the following way to get the job's Configuration: `UdfContext.getUdfContext().getJobConf()`
org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.PigGenericMapReduce.sJobConf	UDFContext in the following way to get the job's Configuration: `UdfContext.getUdfContext().getJobConf()`
org.apache.pig.tools.pigstats.PigStatsUtil.TASK_COUNTER_GROUP	MRPigStatsUtil.TASK_COUNTER_GROUP

Deprecated Methods

Instead of this deprecated method...	Use...
org.apache.pig.PigStreamingBase.deserialize(byte[])	N/A
org.apache.pig.impl.io.FileLocalizer.fileExists(String, DataStorage)	FileLocalizer.fileExists(String, PigContext)
org.apache.pig.impl.io.FileLocalizer.fullPath(String, DataStorage)	FileLocalizer.fullPath(String, PigContext)
org.apache.pig.tools.pigstats.JobStats.getAvgMapTime()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getAvgMapTime()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getAvgMapTime()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getAvgMapTime()	N/A
org.apache.pig.tools.pigstats.JobStats.getAvgREduceTime()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getAvgREduceTime()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getAvgREduceTime()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getAvgREduceTime()	N/A
org.apache.pig.impl.PigContext.getConf()	PigContext.getProperties()
org.apache.pig.tools.pigstats.PigStatusReporter.getCounter(Enum)	PigStatusReporter.incrCounter(java.lang.Enum<?>, long) This method returns MR counter which is not compatible with Tez mode. Use `incrCounter()` that is compatible with both MR and Tez mode.
org.apache.pig.tools.pigstats.PigStatusReporter.getCounter(String, String)	PigStatusReporter.incrCounter(java.lang.Enum<?>, long) This method returns MR counter which is not compatible with Tez mode. Use incrCounter() that is compatible with both MR and Tez mode.
org.apache.pig.tools.pigstats.PigStatsUtil.getEmptyPigStats()	N/A

last-updated: Apr 11, 2022

Instead of this deprecated method...	Use...
org.apache.pig.tools.pigstats.JobStats.getHadoopCounters()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getHadoopCounters()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getHadoopCounters()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getHadoopCounters()	N/A
org.apache.pig.tools.pigstats.PigStats.getJobClient()	N/A
org.apache.pig.tools.pigstats.mapreduce.SimplePigStats.getJobClient()	N/A
org.apache.pig.backend.hadoop.executionengine.HExecutionEngine.getJobConf()	N/A
org.apache.pig.tools.pigstats.JobStats.getMapInputRecords()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getMapInputRecords()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getMapInputRecords()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getMapInputRecords()	N/A
org.apache.pig.tools.pigstats.JobStats.getMapOutputRecords()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getMapOutputRecords()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getMapOutputRecords()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getMapOutputRecords()	N/A
org.apache.pig.tools.pigstats.JobStats.getMaxMapTime()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getMaxMapTime()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getMaxMapTime()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getMaxMapTime()	N/A
org.apache.pig.tools.pigstats.JobStats.getMaxReduceTime()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getMaxReduceTime()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getMaxReduceTime()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getMaxReduceTime()	N/A
org.apache.pig.tools.pigstats.JobStats.getMinMapTime()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getMinMapTime()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getMinMapTime()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getMinMapTime()	N/A

last-updated: Apr 11, 2022

Instead of this deprecated method...	Use...
org.apache.pig.tools.pigstats.JobStats.getMinReduceTime()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getMinReduceTime()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getMinReduceTime()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getMinReduceTime()	N/A
org.apache.pig.tools.pigstats.JobStats.getMultiInputCounters()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getMultiInputCounters()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getMultiInputCounters()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getMultiInputCounters()	N/A
org.apache.pig.tools.pigstats.JobStats.getMultiStoreCounters()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getMultiStoreCounters()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getMultiStoreCounters()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getMultiStoreCounters()	N/A
org.apache.pig.tools.pigstats.JobStats.getNumberMaps()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getNumberMaps()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getNumberMaps()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getNumberMaps()	N/A
org.apache.pig.tools.pigstats.JobStats.getNumberReduces()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getNumberReduces()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getNumberReduces()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getNumberReduces()	N/A
org.apache.pig.tools.pigstats.JobStats.getProactiveSpillCountObjects()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getProactiveSpillCountObjects()
org.apache.pig.tools.pigstats.JobStats.getProactiveSpillCountRecs()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getProactiveSpillCountRecs()
org.apache.pig.tools.pigstats.JobStats.getReduceInputRecords()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getReduceInputRecords()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getReduceInputRecords()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getReduceInputRecords()	N/A

last-updated: Apr 11, 2022

Instead of this deprecated method...	Use...
org.apache.pig.tools.pigstats.JobStats.getReduceOutputRecords()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getReduceOutputRecords()
org.apache.pig.tools.pigstats.tez.TezVertexStats.getReduceOutputRecords()	N/A
org.apache.pig.tools.pigstats.tez.TezDAGStats.getReduceOutputRecords()	N/A
org.apache.pig.tools.pigstats.JobStats.getSMMSpillCount()	If you are using mapreduce, please cast JobStats to org.apache.pig.tools.pigstats.mapreduce.MRJobStats, then use MRJobStats.getSMMSpillCount()
org.apache.pig.EvalFunc.isAsynchronous()	N/A
org.apache.pig.impl.io.FileLocalizer.isDirectory(String, DataStorage)	FileLocalizer.isDirectory(String, PigContext)
org.apache.pig.impl.io.FileLocalizer.isFile(String, DataStorage)	FileLocalizer.isFile(String, PigContext)
org.apache.pig.impl.logicalLayer.schema.Schema.isTwoLevelAccessRequired() **Note:** twoLevelAccess is no longer needed.	N/A
org.apache.pig.impl.io.FileLocalizer.open(String, ExecType, DataStorage)	FileLocalizer.open(String, PigContext)
org.apache.pig.data.Tuple.reference(Tuple)	N/A
org.apache.pig.data.BagFactory.registerBag(DataBag)	As of Pig 0.11, bags register with the *SpillableMemoryManager* themselves. Register a bag with the *SpillableMemoryManager*. If the bags created by an implementation of BagFactory are managed by the *SpillableMemoryManager* then this method should be called each time a new bag is created.
org.apache.pig.PigStreamingBase.serialize(Tuple)	N/A
org.apache.pig.ResourceStatistics.setmBytes(Long)	ResourceStatistics.setSizeInBytes(Long)
org.apache.pig.impl.logicalLayer.schema.Schema.setTwoLevelAccessRequired(boolean) twoLevelAccess is no longer needed.	N/A
org.apache.pig.tools.pigstats.ScriptState.start(String, PigContext)	ScriptState.start(ScriptState)

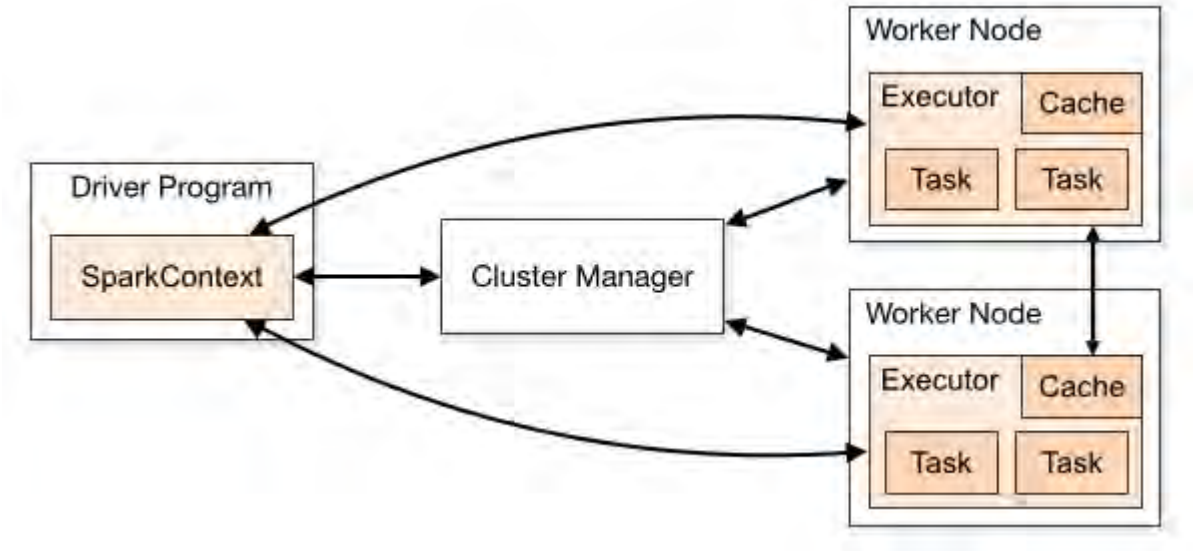
Sentry

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The Sentry documentation has moved. Sentry documentation can now be found at Sentry on page 3817.

Apache Spark

Apache Spark is an open-source processing engine that you can use to process Hadoop data. The following diagram shows the components involved in running Spark jobs. See Spark Cluster Mode Overview for additional component details.



MapR Data Platform supports the following types of cluster managers:

- Spark's standalone cluster manager
- YARN

The configuration and operational steps for Spark differ based on the Spark mode you choose to install. The steps to integrate Spark with other components are the same when using either Standalone of YARN cluster mode, except where otherwise noted.

This section provides documentation about configuring and using Spark with MapR Data Platform, but it does not duplicate the Apache Spark documentation.

You can also refer to additional documentation available on the Apache Spark Product Page.

Getting Started with Spark Interactive Shell
After you have a basic understanding of Apache Spark and have it installed and running on your cluster, you can use it to load datasets, apply schemas, and query data from the Spark interactive shell.

Reading Data from MapR File System

1. Copy sample data into MapR File System:

   - For this example, the dataset constitutes a CSV file of a list of auctions.

   - Download the file from GitHub: https://github.com/mapr-demos/getting-started-spark-on-mapr/tree/master/data.

   - Copy the file into your cluster, in the `/apps/` directory, using the `cp/scp` or `hadoop put` command:

   ```
   scp ./data/auctiondata.csv mapr@[mapr-cluster-node]:/mapr/[cluster-name]/
   apps/
   or
   $ hadoop fs -put ./data/auctiondata.csv /apps
   ```

   - This dataset is from eBay online auctions. The dataset contains the following fields:

   ```
   auctionid - Unique identifier of an auction.
   bid - Proxy bid placed by a bidder.
   bidtime - Time (in days) that the bid was placed from the start of the
   auction.
   bidder - eBay username of the bidder.
   ```


last-updated: Apr 11, 2022

```
bidderrate - eBay feedback rating of the bidder.
openbid - Opening bid set by the seller.
price - Closing price that the item sold for (equivalent to the second
highest bid + an increment).
item - Type of item.
```

The table below shows the fields with some sample data:

auctionid	bid	bidtime	bidder	bidderrate	openbid	price	item	daystolive
8213034705	95	2.927373	jake7870	0	95	117.5	xbox	3

2. Start the Spark interactive shell:

   - `$SPARK_HOME` represents the home of your Spark installation in MapR, for example: `/opt/mapr/spark/spark-2.2.1/`.

   ```
   $ $SPARK_HOME/bin/spark-shell --master local[2]
   ```

3. Once the Spark shell is ready, load the dataset:

   ```
   scala> val auctionData = spark.read.textFile("/apps/auctiondata.csv")
   ```

4. Display the first entry:

   ```
   scala> auctionData.first()
   ```

5. Count the number of entries:

   ```
   scala> auctionData.count()
   ```

6. Use other Spark actions:

   ```
   // Displays first 20 lines
   scala> auctionData.show()

   // Displays first 3 lines - change value to see more/less
   scala> auctionData.take(3)
   ```

7. Transform the dataset into a new one that contains only `xbox` lines, and count them:

   ```
   scala> val auctionWithXbox = auctionData.filter(line =>
   line.contains("xbox"))
   scala> auctionWithXbox.count()
   ```

   - This could also be done in a single line by chaining transformations and actions:

   ```
   scala> auctionData.filter(line => line.contains("xbox")).count()
   ```

8. Use Spark Dataframes:

```
scala> val auctionDataFrame =
spark.read.format("csv").option("inferSchema",
true).load("/apps/
auctiondata.csv").toDF("auctionid","bid","bidtime","bidder","bidderrate",
"openbid","price","item","daystolive")
```

9. Use a `filter` transformation on the Dataframe:

```
scala> auctionDataFrame.filter($"price" < 30).show()
```

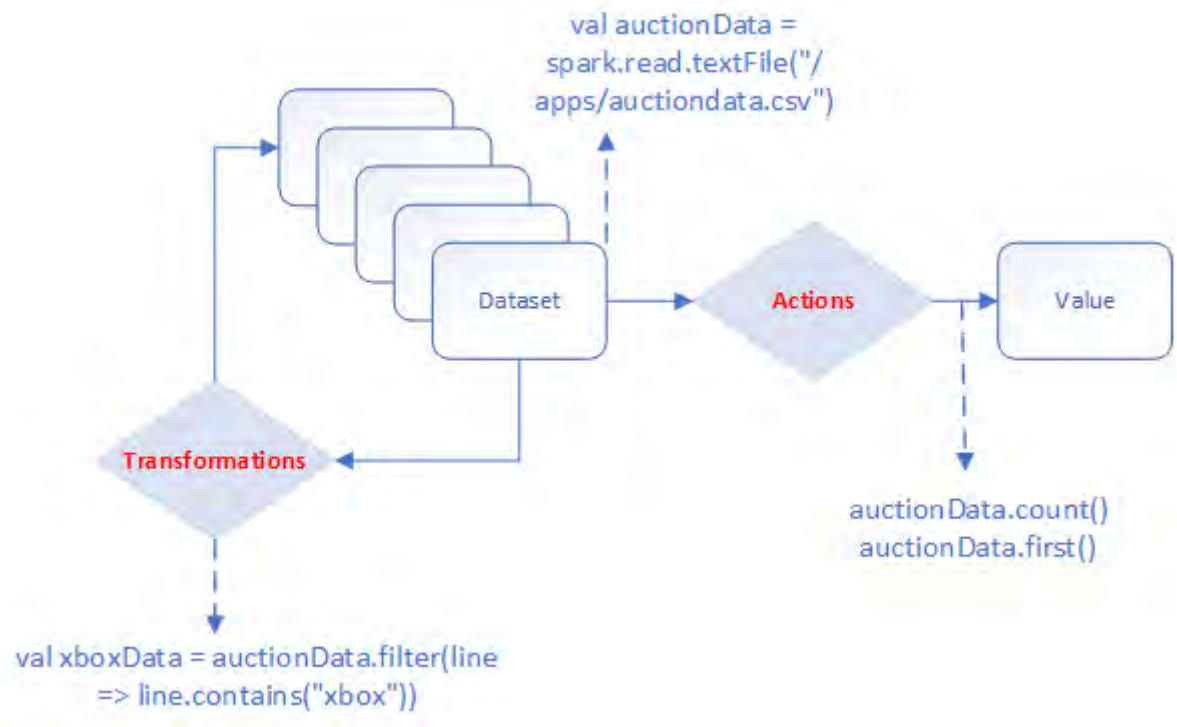


Figure 30: Schematic representation of performing transformations and actions on a dataset

Writing Data from MapR File System

Using the same dataset, save all `xbox` items as a file in MapR File System:

- You can use the `filter($"item" === "xbox")` filter and `write.json` or other options to save the result of the action to MapR File System.

```
scala> auctionDataFrame.filter($"item" === "xbox").write.json("/apps/
results/json/xbox")
```

This command creates the `/apps/results/json/xbox directory` in which you will see the JSON file(s) created. You can use the same command to create Parquet or any other file format:

```
scala> auctionDataFrame.filter($"item" === "xbox").write.parquet("/apps/
results/parquet/xbox")
```


last-updated: Apr 11, 2022

Writing Data to MapR Database JSON

The first step when you are working with MapR Database JSON is to define a document `_id` that uniquely identifies the document.

Add a new `_id` field in the csv file and generate UUIDs to add to this field.

To load the Dataframe into the MapR-DB JSON:

```
dataframe.saveToMapRDB("tableName", createTable = true, bulkInsert = false,
idFieldPath = "_id")
```

The following commands will create a table and insert the data into: `/apps/auction_json_table`.

```
scala> import spark.implicits._
scala> import java.util.UUID
scala> import org.apache.spark.sql.SparkSession
scala> import org.apache.spark.sql.types._
scala> import org.apache.spark.sql.SaveMode
scala> import com.mapr.db.spark.sql._  // import the MapR-DB OJAI Connector
scala> val generateUUID = udf(() => UUID.randomUUID().toString) // create
UDF to generate UUID
scala> // showing that you can create your own schema
  val customSchema =
  StructType(
   Array(
     StructField("actionid",StringType,true),
     StructField("bid",DoubleType,true),
     StructField("bidtime",DoubleType,true),
     StructField("bidder",StringType,true),
     StructField("bidderrate",IntegerType,true),
     StructField("openbid",DoubleType,true),
     StructField("price",DoubleType,true),
     StructField("item",StringType,true),
     StructField("daystolive",IntegerType,true)
     )
   )
```

You can now query the table using the MapR Database shell. Open a terminal on your cluster and run the following command:

```
$ mapr dbshell
maprdb mapr:> find /apps/auction_json_table --limit 10
```

Reading Data from MapR Database JSON

Now that you have the data in MapR Database JSON, you can create and query a Spark Dataframe using the following commands:

```
scala> import com.mapr.db.spark.sql._
scala> import org.apache.spark.sql.SparkSession
scala> val dataFromMapR = spark.loadFromMapRDB("/apps/auction_json_table")
scala> dataFromMapR.printSchema
scala> dataFromMapR.count

scala> dataFromMapR.filter($"price" < 30).show()   // use a filter
```

Related Links

- Spark configure.sh on page 4038

- MapR Sandbox for Hadoop on page 103
- MapR Data Science Refinery on page 3032
- HPE Ezmeral Spark blog

Apache Spark Feature Support

MapR Data Platform supports most Apache Spark features. However, there are some exceptions.

Spark SQL and Apache Derby Support on Spark	If you are using Spark SQL with Derby database without Hive or Hive Metastore installation, you will see the following exception: `java.lang.RuntimeException: Unable to instantiate org.apache.hadoop.hive.ql.metadata.SessionHiveMetaStoreClient` Add the `hive-service-2.3.*.jar` and `log4j2` jars to `/opt/mapr/spark/spark-3.x.x/jars` location to use Spark SQL with Derby Database without Hive or Hive Metastore installation. The `log4j2` jars are located at `/opt/mapr/lib/log4j2/log4j-*.jar` location. Spark 3.1.2 and Spark 3.2.0 does not support `log4j1.2` logging on HPE Ezmeral Data Fabric.
Symlink Support on Spark 2.4.4	For full Symlink support on Spark 2.4.4, request the patch. See Applying a Patch on page 437.
Spark Thrift JDBC/ODBC Server Support	Running the Spark Thrift JDBC/ODBC Server on a secure cluster is supported only on Spark 2.1.0 or later. You can run the Spark Thrift JDBC/ODBC Server to enable connections to Hive 1.2.1 using Beeline; however, you can connect only to Hive versions supported by your Spark version.
Spark SQL and Hive Support for Spark 2.1.0	Spark 2.1.0 is able to connect to Hive 2.1 Metastore; however, only features of Hive 1.2 are supported.
Spark SQL and Hive Support for Spark 2.0.1	Spark SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation. The following Hive functions are not supported in Spark SQL: • Tables with buckets • UNION type • Unique join • Column statistics collecting • Output formats: File format (for CLI), Hadoop Archive • Block-level bitmap indexes and virtual columns • Automatic determination of the number of reducers for JOIN and GROUP BY

last-updated: Apr 11, 2022

- Metadata-only query
- Skew data flag
- STREAMTABLE hint in JOIN
- Merging of multiple small files for query results

Spark SQL and Hive Support for Spark 1.6.1

Spark SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation. The following Spark SQL operations support the following Hive table formats:

	Hive 1.2 Table Format				
Spark SQL Operations	AVRO	ORC	Parquet	RC	default
create	Yes	Yes	Yes	Yes	Yes
drop	Yes	Yes	Yes	Yes	Yes
insert into	Yes	Yes	Yes	Yes	Yes
insert overwrite	Yes	Yes	Yes	Yes	Yes
select	Yes	Yes	Yes	Yes	Yes
load data	Yes	Yes	Yes	Yes	Yes

Spark Standalone

This section includes topics about configuring and using Spark in Standalone mode.

To integrate Spark with other ecosystem components, see Integrating Spark.

For additional documentation, see the Apache Spark website.

Installing Spark Standalone

This topic describes how to use package managers to download and install Spark Standalone from the EEP repository.

To set up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

Spark is distributed as four separate packages:

Package	Description
`mapr-spark`	Install this package on any node where you want to install Spark. This package is dependent on the `mapr-client` package.
`mapr-spark-master`	Install this package on Spark master nodes. Spark master nodes must be able to communicate with Spark worker nodes over SSH without using passwords. This package is dependent on the `mapr-spark` and the `mapr-core` packages.
`mapr-spark-historyserver`	Install this optional package on Spark History Server nodes. This package is dependent on the `mapr-spark` and `mapr-core` packages.
`mapr-spark-thriftserver`	Install this optional package on Spark Thrift Server nodes. This package is available starting in the EEP 4.0 release. It is dependent on the `mapr-spark` and `mapr-core` packages.

last-updated: Apr 11, 2022

Run the following commands as `root` or using `sudo`.

1. Create the `/apps/spark` directory on the cluster filesystem, and set the correct permissions on the directory.

```
hadoop fs -mkdir /apps/spark
hadoop fs -chmod 777 /apps/spark
```

   Note: Beginning with EEP 6.2.0, the `configure.sh` script creates the `/apps/spark` directory automatically.

2. Install Spark using the appropriate commands for your operating system:

On CentOS / Red Hat	`yum install mapr-spark mapr-spark-master mapr-spark-historyserver mapr-spark-thriftserver`
On Ubuntu	`apt-get install mapr-spark mapr-spark-master mapr-spark-historyserver mapr-spark-thriftserver`
On SLES	`zypper install mapr-spark mapr-spark-master mapr-spark-historyserver mapr-spark-thriftserver`

   Note: The `mapr-spark-historyserver`, `mapr-spark-master`, and `mapr-spark-thriftserver` packages are optional.

   Spark is installed into the `/opt/mapr/spark` directory.

3. For Spark 2.x:

   Copy the `/opt/mapr/spark/spark-<version>/conf/slaves.template` into `/opt/mapr/spark/spark-<version>/conf/slaves`, and add the hostnames of the Spark worker nodes. Put one worker node hostname on each line.

   For example:

```
localhost
worker-node-1
worker-node-2
```

4. Set up passwordless ssh for the `mapr` user such that the Spark master node has access to all secondary nodes defined in the `conf/slaves` file for Spark 2.x .

5. As the `mapr` user, start the worker nodes by running the following command in the master node. Since the Master daemon is managed by the Warden daemon, do not use the `start-all.sh` or `stop-all.sh` command.

   For Spark 2.x:

```
/opt/mapr/spark/spark-<version>/sbin/start-slaves.sh
```


last-updated: Apr 11, 2022

6. If you want to integrate Spark with MapR Event Store For Apache Kafka, install the Streams Client on each Spark node:
   - On Ubuntu:
     ```
     apt-get install mapr-kafka
     ```
   - On RedHat/CentOS:
     ```
     yum install mapr-kafka
     ```
7. If you want to use a Streaming Producer, add the `spark-streaming-kafka-producer_2.11.jar` from the MapR Data Platform Maven repository to the Spark classpath (`/opt/mapr/spark/spark-<versions>/jars/`).
8. After installing Spark Standalone but before running your Spark jobs, follow the steps outlined at Configuring Spark Standalone on page 4030.

Configuring Spark Standalone

Starting in EEP 4.0, after following the steps outlined in the sub-topics in this section, you must run configure.sh -R as the final step in the configuration process.

Configure High Availability for SparkMaster
You configure high availability for the Spark Primary instance so that the instance does not become the single point of failure.

By using ZooKeeper to provide leader election and some state storage, you can launch multiple primary nodes in your cluster that are connected to the same ZooKeeper instance. Zookeeper elects one primary node to be the “leader,” and the others remain in standby mode. If the leader goes down, Zookeeper elects another primary node, recovers the old primary node's state, and resumes scheduling.

1. Set `SPARK_DAEMON_JAVA_OPTS` in `spark-env.sh` with the appropriate ZooKeeper information for the cluster.
   ```
   export SPARK_DAEMON_JAVA_OPTS="-Dspark.deploy.recoveryMode=ZOOKEEPER
   -Dspark.deploy.zookeeper.url=<zookeeper1:5181,zookeeper2:5181,...>
   -Djava.security.auth.login.config=/opt/mapr/conf/
   mapr.login.conf -Dzookeeper.sasl.client=false
   ```
2. Restart the Spark Primary instance and Spark History Server services:
   - For Spark 2.0.1 and later:
     ```
     maprcli node services -nodes <node-ip> -name spark-master -action
     restart
     ```
   - For Spark 1.6.1:
     ```
     maprcli node services -nodes <node-ip> -name spark-master -action
     restart
     maprcli node services -nodes <node-ip> -name
     spark-historyserver -action restart
     ```

3. On the primary node, restart the Spark Secondary instances as the `mapr` user.

   For Spark 2.x:

   ```
   /opt/mapr/spark/spark-<version>/sbin/stop-slaves.sh
   /opt/mapr/spark/spark-<version>/sbin/start-slaves.sh
   ```

Configure Scratch Directory for Spark Standalone

By default, Spark uses the `/tmp` directory as scratch space. Map output files and RDDs are stored in the scratch directory. To use a different directory, or a comma-separated list of multiple directories, set `SPARK_LOCAL_DIRS` to the path to the new directory by adding the following line to the `$SPARK_HOME/conf/spark-env.sh` file:

```
export SPARK_LOCAL_DIRS=$SPARK_HOME/<path to scratch directory>
```

Make this change before starting the Spark services.

Community Edition (Without NFS Support)

Reserve space on your local disk to use as the scratch directory for Spark.

Enterprise Edition and Enterprise Database Edition (With NFS Support)

Create a local volume on each node with the `maprcli volume create` command, or from the Control System. Mount that local volume with NFS to a directory. Set that directory as the scratch directory for Spark.

Note: Due to https://issues.apache.org/jira/browse/SPARK-6313, make sure to set `spark.files.useFetchCache=false` in your `spark-defaults.conf` file.

Using Spark Standalone

For a simple test of your Spark installation, run the following command:

- On Spark 2.0.1 or later:

  ```
  /opt/mapr/spark/spark-<version>/bin/run-example --master spark://<Spark
  Master node hostname>:7077 SparkPi 10
  ```

- On Spark 1.6.1:

  ```
  MASTER=spark://<Spark Master node hostname>:7077 /opt/mapr/spark/
  spark-<version>/bin/run-example org.apache.spark.examples.SparkPi 10
  ```

For more information about running Spark applications, see the Apache Spark Documentation.

Run the Spark Shell in Standalone Mode

- To run the Spark shell, use the following command:
  - On Spark 2.0.1 and later:

    ```
    /opt/mapr/spark/spark-<version>/bin/spark-shell --master spark://<Spark
    Master node hostname>:7077
    ```

- On Spark 1.6.1:

```
MASTER=spark://<Spark Master node hostname>:7077 /opt/mapr/spark/
spark-<version>/bin/spark-shell
```

Security with Spark Standalone

Starting in the EEP 4.0 release, for secure clusters, you no longer need to manually configure your cluster to enable Spark security features. Using the MapR installer for new installations or running `configure.sh -R` for manual installs and upgrades automatically enables security features on secure clusters. See Spark configure.sh on page 4038 for details, including instructions on how to avoid enabling security features on secure clusters.

When running Spark applications on a secure cluster, you must pass the `-Dmapr_sec_enabled` flag to Spark. For secure clusters, this flag is set in `spark-env.sh`. For situations where your Spark application does not invoke this script, e.g., a Spark web service, you must manually pass the flag.

Spark on YARN

This section contains topics about installing, configuring and using Spark on YARN.

Important: Spark 2.0.1 (and later) YARN mode is supported only on clusters in MRv2 (YARN) mode. It is not supported on clusters in MRv1 (classic) mode.

To integrate Spark with other ecosystem components, see Integrating Spark on page 4113

For additional documentation, see the Apache Spark documentation.

Installing Spark on YARN

This topic describes how to use package managers to download and install Spark on YARN from the EEP repository.

To set up the EEP repository, see Step 10: Install Ecosystem Components Manually on page 176.

Spark is distributed as three separate packages:

Package	Description
`mapr-spark`	Install this package on each node where you want to install Spark. This package is dependent on the `mapr-client` package.
`mapr-spark-historyserver`	Install this optional package on Spark History Server nodes. This package is dependent on the `mapr-spark` and `mapr-core` packages.
`mapr-spark-thriftserver`	Install this optional package on Spark Thrift Server nodes. This package is available starting in the EEP 4.0 release. It is dependent on the `mapr-spark` and `mapr-core` packages.

To install Spark on YARN (Hadoop 2), execute the following commands as `root` or using `sudo`:

1. Verify that JDK 1.7 or later is installed on the node where you want to install Spark.
2. Create the `/apps/spark` directory on the cluster filesystem, and set the correct permissions on the directory:

```
hadoop fs -mkdir /apps/spark
hadoop fs -chmod 777 /apps/spark
```

Note: Beginning with EEP 6.2.0, the `configure.sh` script creates the `/apps/spark` directory automatically when using the Installer. However, you must manually create this directory when performing a manual installation.

last-updated: Apr 11, 2022

3. Install the packages:

On Ubuntu	`apt-get install mapr-spark mapr-spark-historyserver mapr-spark-thriftserver`
On CentOS / Red Hat	`yum install mapr-spark mapr-spark-historyserver mapr-spark-thriftserver`
On SLES	`zypper install mapr-spark mapr-spark-historyserver mapr-spark-thriftserver`

   Note: The `mapr-spark-historyserver` and `mapr-spark-thriftserver` packages are optional.

4. If you want to integrate Spark with MapR Event Store For Apache Kafka, install the Streams Client on each Spark node:
   - **On Ubuntu:**

     ```
     apt-get install mapr-kafka
     ```

   - **On CentOS / Red Hat**:

     ```
     yum install mapr-kafka
     ```

5. If you want to use a Streaming Producer, add the `spark-streaming-kafka-producer_2.11.jar` from the MapR Maven repository to the Spark classpath (`/opt/mapr/spark/spark-<versions>/jars/`).

   For repository-specific information, see Maven Artifacts for MapR on page 4155

6. After installing Spark on YARN but before running your Spark jobs, follow the steps outlined at Configuring Spark on YARN on page 4033.

Configuring Spark on YARN

Starting in EEP 4.0, after following the steps outlined in the sub-topics in this section, you must run configure.sh -R as the final step in the configuration process.

Configure data-fabric Client Node to Run Spark Applications

When Spark runs on YARN, data-fabric client nodes require the `hadoop-yarn-server-web-proxy` JAR file to run Spark applications. On Windows, the client node also requires an update to the SPARK_DIST_CLASSPATH. A data-fabric client node (a node with the mapr-client package, but without mapr-core packages) is also known as an edge node.

The mapr-client package does not include the JAR file required to run Spark applications. Therefore, you must copy the `/opt/mapr/hadoop/hadoop-2.x.x/share/hadoop/yarn/hadoop-yarn-server-web-proxy-<version>.jar` from a data-fabric cluster node to the same location on the data-fabric client node from which you want to run the Spark application.

Configure Spark JAR Location

last-updated: Apr 11, 2022

By default, Spark on YARN uses Spark JAR files that are installed locally. The Spark JAR files can also be added to a world-readable location on MapR File System. When you add the JAR files to a world-readable location, YARN can cache them on nodes to avoid distributing them each time an application runs. Complete the following steps to add the Spark JAR files to a world-readable locaton on MapR File System:

1. Create a zip archive containing all the JARs from the `SPARK_HOME/jars` directory. For example:

   ```
   cd /opt/mapr/spark/spark-<version>/jars/
   zip /opt/mapr/spark/spark-<version>/spark-jars.zip ./*
   ```

2. Copy the zip file from the local filesystem to a world-readable location on MapR File System. You can upload it to the home of the current user:

   ```
   hadoop fs -put /opt/mapr/spark/spark-<version>/spark-jars.zip
   ```

   For example:

   ```
   hadoop fs -put /opt/mapr/spark/spark-3.2.0/spark-jars.zip /user/mapr/
   ```

3. Set the `spark.yarn.archive` property in the `spark-defaults.conf` file located in `/opt/mapr/spark/spark-<version>/conf/spark-defaults.conf` to point to the world-readable location where you added the zip file. Apply this setting on the node where you will be submitting your Spark jobs.

   ```
   spark.yarn.archive maprfs:///<path to zip>
   ```

   For example:

   ```
   spark.yarn.archive maprfs:///user/mapr/spark-jars.zip
   ```

Configure Spark with the NodeManager Local Directory Set to MapR File System

This procedure configures Spark to use the mounted NFS directory instead of the `/tmp` directory on the local filesystem. Note that spill to disk should be configured to spill to the MapR File System node local storage only if local disks are unavailable or space is limited on those disks.

1. Install the `mapr-loopbacknfs` and `nfs-utils` packages if they are not already installed. For reference, see Installing the mapr-loopbacknfs Package on page 400 and Setting Up MapR NFS.

2. Start the mapr-loopbacknfs service by following the steps at Managing the mapr-loopbacknfs Service on page 1236.

3. To configure Spark Shuffle on NFS, complete these steps **on all nodes**:
   a) Create a local volume for Spark Shuffle:

   ```
   sudo -u mapr maprcli volume
   create -name mapr.$(hostname -f).local.spark -path /var/mapr/local/$
   (hostname -f)/spark -replication 1 -localvolumehost $(hostname -f)
   ```

b) Point the NodeManager local directory to the Spark Shuffle volume mounted through NFS by setting the following property in the `/opt/mapr/hadoop/hadoop-<version>/etc/hadoop/yarn-site.xml` file on the NodeManager nodes:

```
<property>
    <name>yarn.nodemanager.local-dirs</name>
    <value>/mapr/my.cluster.com/var/mapr/local/${mapr.host}/spark</
value>
</property>
```

c) (Optional) Configure how many times the NodeManager can attempt to delete application-related directories from a volume when Spark is configured to use the mounted NFS directory instead of the /tmp directory on the local filesystem. Increasing the value (default is 2) of this property can prevent application cache data from accumulating in the volume. This functionality is available by default starting in EEP 7.1.0. For previous EEP versions, request the patch. See Applying a Patch.

```
<property>
    <name>yarn.nodemanager.max-retry-file-delete</name>
    <value>2</value>
</property>
```

d) Restart the NodeManager service and the Resource Manager service on the main node to pick up the `yarn-site.xml` changes:

```
maprcli node services -name nodemanager -action restart -nodes <node
1> <node 2> <node 3>
maprcli node services -name resourcemanager -action restart -nodes
<node 1> <node 2> <node 3>
```

Using Spark on YARN

This section includes information about using Spark on YARN in a data-fabric cluster.

For a simple test of your Spark installation, run the following command as the mapr user:

- On Spark 2.0.1 or later:

```
/opt/mapr/spark/spark-<version>/bin/run-example --master
yarn --deploy-mode client SparkPi 10
```

- On Spark 1.6.1:

```
MASTER=yarn-client /opt/mapr/spark/spark-<version>/bin/run-example
org.apache.spark.examples.SparkPi 10
```

Note: These commands will fail if it is run as the root user.

For more information about running Spark applications, see the Apache Spark documentation.

Deployment Modes

Spark is preconfigured for YARN and does not require any additional configuration to run.

Two deployment modes can be used to launch Spark applications on YARN:

- In `cluster` mode, jobs are managed by the YARN cluster. The Spark driver runs inside an Application Master (AM) process that is managed by YARN. This means that the client can go away after initiating the application.

last-updated: Apr 11, 2022

- In `client` mode, the Spark driver runs in the client process, and the Application Master is used only to request resources from YARN.

MapR recommends using `cluster` deployment mode instead of `client` mode. If the Spark client that runs the job exits after submitting the job, there is no impact on job completion.

Note: In `cluster` deployment mode, the local directories used by the Spark executors and the Spark driver are the local directories that are configured for YARN (`yarn.nodemanager.local-dirs`).

Note: `SPARK_LOCAL_DIRS` is ignored when you run Spark on YARN.

Run Spark from the Spark Shell

In `yarn-client` mode, complete the following steps to run Spark from the Spark shell:

1. Navigate to the Spark-on-YARN installation directory, and insert your Spark version into the command.

```
cd /opt/mapr/spark/spark-<version>/
```

2. Issue the following command to run Spark from the Spark shell:
   - On Spark 2.0.1 and later:

```
./bin/spark-shell --master yarn --deploy-mode client
```

   - On Spark 1.6.1:

```
MASTER=yarn-client ./bin/spark-shell
```

Note: You must use `yarn-client` mode to run Spark from the Spark shell. The `yarn-cluster` mode is not supported.

Security with Spark on YARN

Starting in the EEP 4.0 release, for secure clusters, you no longer need to manually configure your cluster to enable Spark security features. Using the MapR installer for new installations or running `configure.sh -R` for manual installs and upgrades automatically enables security features on secure clusters. See Spark configure.sh on page 4038 for details, including instructions on how to avoid enabling security features on secure clusters.

When running Spark applications on a secure cluster, you must pass the `-Dmapr_sec_enabled` flag to Spark. For secure clusters, this flag is set in `spark-env.sh`. For situations where your Spark application does not invoke this script, e.g., a Spark web service, you must manually pass the flag.

Configure Authentication for Spark on YARN
When authentication is enabled, authentication keys are randomly generated for each job.

Note: Starting in EEP 4.0, for secure clusters, you can skip this step. For new installs done through the 6.0 MapR Installer, the installer enables this configuration. For manual installs and upgrades, running configure.sh -R, as the final step in the configuration process, enables these settings.

last-updated: Apr 11, 2022

Complete the following step to manually configure authentication on a non-secure cluster or in versions earlier than EEP 4.0:

1. Configure the following property in the `spark-defaults.conf` file on each Spark node:

   ```
   spark.authenticate true
   ```

   The `spark-defaults.conf` file is in the following location: `/opt/mapr/spark/spark-<version>/conf/`

2. If you have configured an external shuffle service, you must also add the following property to `yarn-site.xml` on page 2216 on each Spark node:

   ```
   <property>
         <name>spark.authenticate</name>
         <value>true</value>
   </property>
   ```

Configure SSL Encryption for Spark on YARN

Starting in EEP 6.0.0, you can remove `spark.ssl.keyStorePassword`, `spark.ssl.trustStorePassword`, and `spark.ssl.keyPassword` from the `spark-defaults.conf` file for additional security. These passwords are stored in the `/opt/mapr/conf/ssl-client.xml` file and Spark can access passwords from this file itself.

Note: If passwords are present in both `/opt/mapr/conf/ssl-client.xml` and `/opt/mapr/spark/spark-2.3.1/conf/spark-defaults.conf` files, then the password from the `spark-defaults.conf` file is used.

Complete the following step to manually configure encryption for the Spark HTTP file and broadcast servers:

In the `spark-defaults.conf` file on each spark node, configure the following properties. Starting in EEP 6.0.0, the configured algorithms mentioned in the following code are no longer available for your web service to pick up. You must remove the `spark.ssl.enabledAlgorithms TLS_RSA_WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_256_CBC_SHA` line to let parties negotiate the matching ciphers.

- For Spark 2.0.1 and later:

  ```
  spark.ssl.fs.enabled true
  spark.ssl.keyPassword <ssl-keystore-password>
  spark.ssl.keyStore /opt/mapr/conf/ssl_keystore
  spark.ssl.keyStorePassword <ssl-keystore-password>
  spark.ssl.trustStore /opt/mapr/conf/ssl_truststore
  spark.ssl.trustStorePassword <ssl-keystore-password>
  spark.ssl.protocol TLSv1.2
  spark.ssl.enabledAlgorithms
  TLS_RSA_WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_256_CBC_SHA
  ```

Note: Starting in EEP 4.0, for secure clusters, you can skip this step. For new installs done through the 6.0 MapR Installer, the installer enables this configuration. For manual installs and upgrades, running configure.sh -R, as the final step in the configuration process, enables these settings.

last-updated: Apr 11, 2022

- For Spark 1.6.1:

```
spark.ssl.akka.enabled true
spark.ssl.fs.enabled true
spark.ssl.keyPassword <ssl-keystore-password>
spark.ssl.keyStore /opt/mapr/conf/ssl_keystore
spark.ssl.keyStorePassword <ssl-keystore-password>
spark.ssl.trustStore /opt/mapr/conf/ssl_truststore
spark.ssl.trustStorePassword <ssl-keystore-password>
spark.ssl.protocol TLSv1.2
spark.ssl.enabledAlgorithms
TLS_RSA_WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_256_CBC_SHA
```

Spark UI SSL is not needed when running Spark on YARN because encryption is provided by the YARN protocol. **For versions prior to EEP 4.1.0**, to enable users logged in with a normal user account (not mapr or root) to run spark jobs on the cluster, disable Spark SSL for Spark-on-YARN jobs. To disable Spark SSL, add `spark.ssl.ui.enabled false` to the `spark-defaults.conf` file on each spark node. The `spark-defaults.conf` file is in the following location: `/opt/mapr/spark/spark-<version>/conf/`. Make sure SSL is enabled for the Spark history server.

When you manually configure encryption for Spark, set the same protocol and algorithms for each node. Otherwise, the connection between those components might fail.

Configure SASL Encryption for Spark on YARN

Note: Starting in EEP 4.0, for secure clusters, you can skip the steps outlined in this section. For new installs done through the 6.0 MapR Installer, the installer enables this configuration. For manual installs and upgrades, running configure.sh -R, as the final step in the configuration process, enables these settings.

Complete the following steps to manually enable SASL encryption on a non-secure cluster or in versions earlier than EEP 4.0:

1. Verify that authentication is enabled, or configure authentication. SASL encryption uses the same authentication keys.

   For more information, see Configure Authentication for Spark on YARN on page 4036

2. Configure the following property in the `spark-defaults.conf` file on each spark node.

   ```
   spark.authenticate.enableSaslEncryption true
   ```

   The `spark-defaults.conf` file is in the following location: `/opt/mapr/spark/spark-<version>/conf/`

Spark configure.sh
Starting in the EEP 4.0 release, run `configure.sh -R` to complete your Spark configuration when manually installing Spark or upgrading to a new version.

The command is the following:

```
/opt/mapr/server/configure.sh -R
```

Note: You do not need to run this script for new installs, if you are using the MapR installer in EEP 4.0 or later.

In the case of Spark Standalone on page 4028 and Spark on YARN on page 4032, this is the last step in the configuration process.

All security configuration properties are specified within the following comment block in the SPARK_HOME/conf/spark-defaults.conf file:

```
#SECURITY BLOCK
...
#END OF THE SECURITY CONFIGURATION BLOCK
```

Important: Do not remove these comments from the file, as well as any other comments within the block inserted by configure.sh. The script uses these comments to locate security properties.

Note: To set ports to special values, use the spark.driver.port and spark.blockManager.port properties.

Starting in EEP 6.0.0, Spark services such as the History Server, Thrift Server, or Master are restarted by configure.sh only for changes to the following Spark configuration files: spark-defaults.conf, spark-env.sh, hive-site.xml, or log4j.properties. If these files are unchanged, configure.sh does not restart any of the Spark services.

An update to Spark causes the conf directory from the previous the Spark version to be saved to the spark-<old-version>.<old-timestamp> directory. If your Spark version did not change during the update, then configurations from the spark-<old-version>.<old-timestamp> directory is automatically copied to the spark-<version> directory by the configure.sh script.

If you use .customSecure, at the first run, the configure.sh script copies the hive-site.xml file from Hive. For subsequent times, the hive-site.xml file is not copied from Hive and you would need to manually modify the $SPARK_HOME/conf/hive-site.xml file.

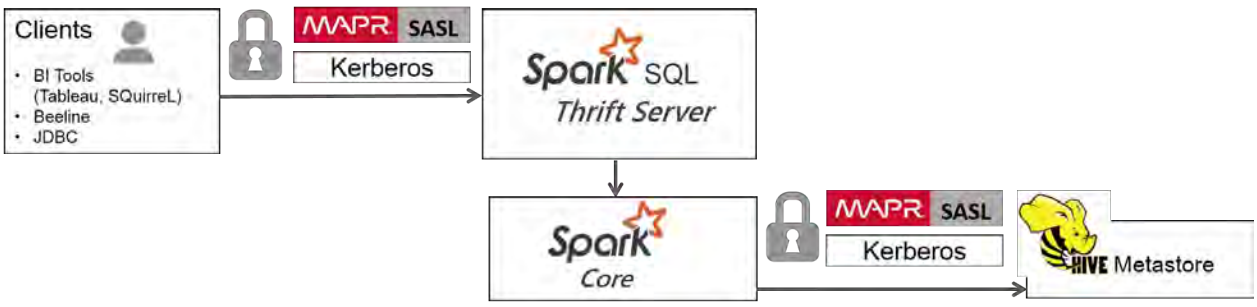
Spark SQL Thrift Server
Spark SQL Thrift (Spark Thrift) was developed from Apache Hive HiveServer2 and operates like HiveSever2 Thrift server.

Spark Thrift is supported on secure clusters. You can run the Spark Thrift server and connect to Hive versions supported by Spark 2.1.0 and later with Business Intelligence (BI) tools or the Beeline command-line tool.

Starting in the EEP 4.0 release, the Spark Thrift server is available as a separate package. To install this package, see Installing Spark Standalone on page 216 or Installing Spark on YARN on page 218, depending on the type of cluster manager you are installing.

In EEP 3.0, MapR introduces additional security mechanisms for Spark with the Spark Thrift server. MapR-SASL and Kerberos are supported:

- For JDBC connections into Spark Thrift server
- Between Spark and Hive metastore



To enable these security mechanisms for the Spark Thrift server, starting in the EEP 4.0 release, for secure clusters, running configure.sh -R configures MapR-SASL security. The script modifies or creates a SPARK_HOME/conf/hive-site.xml file as follows:

last-updated: Apr 11, 2022

- If Hive is installed in your cluster, the script copies `HIVE_HOME/conf/hive-site.xml` to `SPARK_HOME/conf` and modifies the file.
- If Hive is not installed and you are using MapR-SASL security, the script creates a new `SPARK_HOME/conf/hive-site.xml` file.
- Each time the script runs, if there is a pre-existing `SPARK_HOME/conf/hive-site.xml` file, the script saves a copy of the file in `SPARK_HOME/conf/hive-site.xml.old` before modifying it.

You can configure security manually by following the steps outlined in sub-topics listed on this page.

To launch the Spark Thrift server, perform the procedures required to configure Apache Spark on page 4022 to use Hive on page 3405.

Important:

- Starting in the EEP 4.0 release, if you start and stop the Spark Thrift server using Warden, the connection port number is 2304. If you start and stop by running the `/opt/mapr/spark/<spark-version/sbin/{start,stop}-thriftserver.sh` scripts, the port number remains 10000.
- Starting in the EEP 5.0.4 and EEP 6.3.0 releases, if you start and stop the Spark Thrift server by running the `/opt/mapr/spark/<spark-version/sbin/{start,stop}-thriftserver.sh` scripts, the port number remains 2304.

Default Behavior

The default behavior of the Spark Thrift server is as follows:

1. After installation, the Spark Thrift server is started in the local master mode.
2. If the Spark master package is installed, then Spark Thrift server is started in the standalone master mode.
3. If the `spark.master` property is set in the `spark-defaults.conf` file, then Spark Thrift server uses the master set by this property.

Known Limitations

- MapR-SASL support is implemented for Spark 2.1.0 and later versions of Spark. For Spark version information, see Component Versions for Released EEPs on page 5586.
- The ODBC drivers do not support MAPR-SASL.
- Username and password authentication through PAM is not supported in EEP 3.0.
- Spark Thrift server supports only features and commands in Hive 1.2.
- Although Spark 2.1.0 can connect to Hive 2.1 Metastore, only Hive 1.2 features and commands are supported by Spark 2.1.0.

Related Links

For information related to Spark Thrift server, see:

MapR	Apache

• Hive Release Notes:
  • Hive 1.2.1-1703 Release Notes on page 5999
  • Hive 2.1-1703 Release Notes on page 5992
• Hive and Tez
• Integrate Spark SQL with Hive
• Hive
• Authentication for HiveServer2
• Spark Feature Support
	• Apache Spark 2.1.0 Security
	• Apache Thrift
	• Setting Up HiveServer2

Spark Thrift Server Clients

With Spark Thrift server, you can use JDBC and ODBC connection interfaces that enable a variety of external tools to access Spark and run SQL queries.

- The ODBC interface is used by BI tools (often produced by MapR partners such as Tableau or Microstrategy).
- The JDBC interface is used by clients such as SQuirrel SQL or the Beeline simple SQL shell.

MapR Hive JDBC clients that connect to HiveServer2 can also connect to Spark Thrift server without additional configuration. For details about clients, see HiveServer2 Clients and Connecting to HiveServer2.

Important: Starting in the EEP 4.0 release, if you start and stop the Spark Thrift server using Warden, the connection port number is 2304. If you start and stop by running the `/opt/mapr/spark/<spark-version/sbin/{start,stop}-thriftserver.sh` scripts, the port number is 10000. Beginning with EEP 6.3.0, the connection port number is 2304 for both start/stop methods (using Warden and using `thriftserver.sh` scripts).

MapR-SASL JDBC Connection String Format

If you start and stop the Spark Thrift server through Warden, starting in EEP 4.0, then the JDBC connection string format for MapR-SASL environments is:

```
jdbc:hive2://<hostname>:2304/default;auth=maprsasl;ssl=true
```

Otherwise, the port you use depends on the EEP version:

EEP 4.0 through 6.2.x	`jdbc:hive2://<hostname>:10000/default;auth=maprsasl;ssl=true`
EEP 6.3.0 and later	`jdbc:hive2://<hostname>:2304/default;auth=maprsasl;ssl=true`

Kerberos JDBC Connection String Format

If you start and stop the Spark Thrift server through Warden, starting in EEP 4.0, then the JDBC connection string format for clusters secured with Kerberos is:

```
jdbc:hive2://<hostname>:2304/default;principal=mapr/<FQDN@REALM>;ssl=true
```

Otherwise, the port you use depends on the EEP version:

last-updated: Apr 11, 2022

EEP 4.0 through 6.2.x	`jdbc:hive2://<hostname>:10000/default;principal=mapr/<FQDN@REALM>;ssl=true`
EEP 6.3.0 and later	`jdbc:hive2://<hostname>:2304/default;principal=mapr/<FQDN@REALM>;ssl=true`

Starting the Thrift Server on a Custom Port

To start the Spark Thrift Server on a custom port, use the `hive.server2.thrift.port` option. For example, you can specify the following in the `/opt/mapr/spark/spark-2.4.4/conf/hive-site.xml` file:

```
<property>
<name>hive.server2.thrift.port</name>
<value>34512</value>
</property>
```

For more information, see the Apache HiveServer2 documentation.

Using Authentication with Spark Thrift Server

Spark Thrift server supports both MapR-SASL and Kerberos authentication. The authentication method that you configure for the Spark Thrift server determines how the connection is secured. Clients might require additional configuration and specific connection strings based on the authentication type.

To enable authentication, see:

- Configuring Spark Thrift Server with MapR-SASL on page 4042
- Configuring Spark Thrift Server with Kerberos on page 4043

To configure PAM for Spark Thriftserver, run `configure.sh` on the secure cluster.

For information about Hive integration, see:

- Integrate Spark SQL with Hive
- Setting Up HiveServer2
- Hive
- Spark Feature Support

Configuring Spark Thrift Server with MapR-SASL
Describes how to enable and start the Spark Thrift server on all nodes.

You can configure Spark Thrift server to use MapR-SASL for its communications with various components on a secure MapR cluster. Minimal configuration is required.

Note: Starting in EEP 4.0, for secure clusters, you can skip the steps outlined in this section. For new installs done through the 6.0 MapR Installer, the installer enables this configuration. For manual installs and upgrades, running configure.sh -R enables these settings.

To manually enable MapR-SASL authentication on a non-secure cluster or in versions earlier than EEP 4.0:

1. Verify that the `hive.server2.authentication` property in `hive-site.xml` is set to the value, `MAPRSASL`.

```
<property>
     <name>hive.server2.authentication</name>
     <value>MAPRSASL</value>
</property>
```

2. Restart Spark Thrift server to apply this change. `sbin` is in your Spark directory at `/opt/mapr/spark/spark-<spark_version>/`.

   Important: The MapR administrative user (generally, the account named `mapr`) should start the Spark Thrift server. Then, process identifier (PID) files will be owned by this user, and impersonation support (where applicable) will function correctly.

```
./sbin/stop-thriftserver.sh
./sbin/start-thriftserver.sh
```

Bringing up the Spark Thrift server on every node

When you start and stop Warden after enabling Spark or after running configure.sh on page 2053 or after installing a patch, Spark starts only on one (1) node and not on all nodes. This happens because by default, the Warden configuration file for Spark has the value `1` instead of `all`. For example:

```
# grep services /opt/mapr/conf/conf.d/warden.spark-thriftserver.conf
services=spark-thriftserver:1:cldb
```

To fix this issue permanently:

1. Modify `/opt/mapr/spark/spark-2.4.0/warden/warden.spark-thriftserver.conf` and change `1` to `all`:

```
# grep services /opt/mapr/spark/spark-2.4.0/warden/
warden.spark-thriftserver.conf
            services=spark-thriftserver:all:cldb
```

2. Run `/opt/mapr/server/configure.sh -R`.

The change is then propogated to the `/opt/mapr/conf/conf.d/warden.spark-thriftserver.conf` file.

Configuring Spark Thrift Server with Kerberos

You can configure Spark Thrift server to use Kerberos for its communications with various components on a secure MapR cluster if necessary.

Note: MapR clusters do not provide Kerberos infrastructure. The information in this section assume a Linux-based Kerberos environment, and the specific commands for your environment may vary. Consult with your Kerberos administrator for assistance.

To enable Kerberos authentication:

1. Create a Kerberos identity and keytab. You can use the following commands in a Linux-based Kerberos environment to set up the identity and update the keytab file.
   - The `hive.keytab` file must be owned and readable only by the `mapr` user.

last-updated: Apr 11, 2022

- FQDN@REALM is case-sensitive.

```
# kadmin
          : addprinc -randkey mapr/<FQDN@REALM>
          : ktadd -k /opt/mapr/conf/hive.keytab mapr/<FQDN@REALM>
```

2. Configure the following properties in `hive-site.xml` on each node where HiveServer2 is installed:

Property	Value
hive.server2.authentication	KERBEROS
hive.server2.authentication.kerberos.principal	`mapr/FQDN@REALM` (where `mapr/FQDN@REALM` is the principal that you want to use for the Spark Thrift server)
hive.server2.authentication.kerberos.keytab	`/opt/mapr/conf/mapr.keytab` (where `/opt/mapr/conf/mapr.keytab` is path to the keytab that must be used)

```
<property>
     <name>hive.server2.authentication</name>
     <value>KERBEROS</value>
     <description>authenticationtype</description>
</property>
<property>
      <name>hive.server2.authentication.kerberos.principal</name>
      <value>mapr/FQDN@REALM</value>
      <description>Spark Thrift server principal. If _HOST is used as
the FQDN portion,
      it will be replaced with the actual hostname of the running
instance.
      </description>
</property>
<property>
     <name>hive.server2.authentication.kerberos.keytab</name>
     <value>/opt/mapr/conf/mapr.keytab</value>
     <description>Keytab file for Spark Thrift server principal</
description>
</property>
```

3. Reconfigure the following options in `env.sh` (`/opt/mapr/conf/env.sh`) on each node where HiveServer2 is installed:

Note: These configurations are listed in the portion of the file that begins with `if [ "$MAPR_SECURITY_STATUS" = "true" ];`. However, you should make the changes in the `/opt/mapr/conf/env_override.sh` file. For more information, see About env_override.sh on page 2290.

Existing Configuration	Required Configuration
`MAPR_HIVE_SERVER_LOGIN_OPTS="-Dhadoop.login=maprsasl_keytab"`	`MAPR_HIVE_SERVER_LOGIN_OPTS="-Dhadoop.login=hybrid"`
`MAPR_HIVE_LOGIN_OPTS="-Dhadoop.login=maprsasl"`	`MAPR_HIVE_LOGIN_OPTS="-Dhadoop.login=hybrid"`

last-updated: Apr 11, 2022

4. Restart Spark Thrift server to apply this change. `sbin` is in your Spark directory at `/opt/mapr/spark/spark-<spark_version>/`.

   Important: The MapR administrative user (generally, the account named `mapr`) should start Spark Thrift server. Then, process identifier (PID) files will be owned by this user, and impersonation support (where applicable) will function correctly.

```
./sbin/stop-thriftserver.sh
./sbin/start-thriftserver.sh
```

Related Links

For information about working with HiveServer, see:

- Setting Up HiveServer2
- Hive

Configuring Spark Thrift Server Encryption

Spark Thrift server encryption is supported when authentication is enabled. You can configure encryption with MapR SASL or with SSL/TLS.

Configuring Encryption with MapR SASL or Kerberos

Starting in EEP 4.0, for secure clusters, you can skip the steps outlined in this section. For new installs done using MapR Installer, the Installer enables this configuration. For manual installs and upgrades, running configure.sh -R enables these settings.

To manually configure encryption with MapR-SASL or Kerberos authentication on a non-secure cluster or in versions earlier than EEP 4.0, complete the following steps:

1. Set the `hive.server2.thrift.sasl.qop` property in `hive-site.xml` to the value `auth-conf`. The SASL Quality of Protection (QOP), or *sasl.qop*, setting and the authentication with confidentiality (`auth-conf`) value support authentication:

```
<property>
    <name>hive.server2.thrift.sasl.qop</name>
    <value>auth-conf</value>
</property>
```

2. Restart Spark Thrift server to apply the change:

   Important: The MapR administrative user (generally, the account named `mapr`) should start Spark Thrift server. Then, process identifier (PID) files are owned by this user, and impersonation support (where applicable) functions correctly.

```
./sbin/stop-thriftserver.sh
./sbin/start-thriftserver.sh
```

Configuring Encryption with SSL/TLS

To enable encryption with SSL/TLS:

last-updated: Apr 11, 2022

1. Add the following properties to the `/opt/mapr/spark/spark-<version>/conf/spark-defaults.conf` file:

```
spark.ssl.enabled true
spark.ssl.fs.enabled true
spark.ssl.trustStore /opt/mapr/conf/ssl_truststore
spark.ssl.keyStore /opt/mapr/conf/ssl_keystore
spark.ssl.protocol TLSv1.2
spark.ssl.keyStorePassword      mapr123
spark.ssl.trustStorePassword    mapr123
```

   After the properties are added, event logs will indicate that the job is encrypted.

2. To connect using Beeline with encryption, add the following properties to the `/opt/mapr/spark/spark-<version>/conf/hive-site.xml` file:

```
<property>
  <name>hive.server2.use.SSL</name>
  <value>true</value>
  <description>enable/disable SSL </description>
</property>

<property>
  <name>hive.server2.keystore.path</name>
  <value>/opt/mapr/conf/ssl_keystore</value>
  <description>path to keystore file</description>
</property>

<property>
  <name>hive.server2.keystore.password</name>
  <value>mapr123</value>
  <description>keystore password</description>
</property>
```

3. To start the Spark Thriftserver, use the following command:

```
/opt/mapr/spark/spark-<version>/sbin/start-thriftserver.sh --hiveconf
hive.server2.thrift.port=2304 --master yarn --deploy-mode client
```

   The following example shows a connection string using Beeline (PAM+SSL):

```
./bin/beeline
Beeline version 1.2.0-mapr-1808-spark by Apache Hive
beeline>  !connect jdbc:hive2://node1.cluster.com:2304/
default;ssl=true;user=mapr;password=mapr;sslTrustStorePassword=mapr123;ss
lTrustStore=/opt/mapr/conf/ssl_truststore
Connecting to jdbc:hive2://node1.cluster.com:2304/
default;ssl=true;user=mapr;password=mapr;sslTrustStorePassword=mapr123;ss
lTrustStore=/opt/mapr/conf/ssl_truststore
Connected to: Spark SQL (version 2.1.0-mapr-mep-3.x-1808)
Driver: Hive JDBC (version 1.2.0-mapr-1808-spark)
Transaction isolation: TRANSACTION_REPEATABLE_READ
1: jdbc:hive2://node1.cluster.com:2304/defaul>
```

Enabling High Availability for Spark Thrift Server

With MEPs 5.0.4 or 6.3.0 and later, you can enable high availability for the Spark Thrift Server. Unlike a HiveServer2 high-availability configuration, Spark has no concept of active-active instances. However, after configuration, you can can use Beeline to connect to the Spark Thrift Server on each node.

To enable high availability, use the following steps:

1. Install Spark Thrift Server on all the cluster nodes where it is needed:

On Ubuntu	`apt-get install mapr-spark-thriftserver`
On Red Hat / CentOS	`yum install mapr-spark-thriftserver`
On SLES	`zypper install mapr-spark-thriftserver`

last-updated: Apr 11, 2022

2. Add the following properties to the `/opt/mapr/spark/spark-<spark_version>/conf/hive-site.xml` file on all the nodes where the Spark Thrift Server is installed

```
<property>
<name>hive.zookeeper.quorum</name>
<value><zk_host1_>,<zk_host_2>,…,<zk_host_n></value>
</property>

<property>
<name>hive.zookeeper.client.port</name>
<value><zk_port></value>
</property>

<property>
<name>hive.server2.support.dynamic.service.discovery</name>
<value>true</value>
</property>

<property>
<name>hive.server2.zookeeper.namespace</name>
<value><zk_namespace></value>
</property>
```

For example:

```
<property>
<name>hive.zookeeper.quorum</name>
<value>node1.cluster.com,node2.cluster.com,node3.cluster.com</value>
</property>

<property>
<name>hive.zookeeper.client.port</name>
<value>5181</value>
</property>

<property>
<name>hive.server2.support.dynamic.service.discovery</name>
<value>true</value>
</property>

<property>
<name>hive.server2.zookeeper.namespace</name>
<value>ts2-ts2</value>
</property>
```

Note: The values that you provide for the `hive.server2.zookeeper.namespace` property should be different for the `hive-site.xml` in the Spark and Hive directories.

3. Restart the Spark Thrift Server to apply the changes following the script in the `.sbin` directory at `/opt/mapr/spark/spark-<spark_version>/` or by running a `maprcli` command on all configured nodes:

```
./sbin/stop-thriftserver.sh
./sbin/start-thriftserver.sh
```

or

```
maprcli node services -nodes <host_1>,<host_2>,<host_n> -name
spark-thriftserver -action restart
```

4. Launch the Zookeeper command line interface, and check the Spark Thriftserver znode by running the following commands:

```
/opt/mapr/zookeeper/zookeeper-<version>/bin/zkCli.sh -server <ip:port of
zookeeper instance>
ls /<hive.server2.zookeeper.namespace>
```

For example:

```
/opt/mapr/zookeeper/zookeeper-3.4.11/bin/zkCli.sh -server
node1.cluster.com:5181
ls /ts2-ts2
[serverUri=node1.cluster.com:2304;version=;sequence=0000000000]
```

5. Using Beeline, you can connect to the Spark Thrift Server by using the following string:

```
beeline> !connect jdbc:hive2://<hostname -f>:5181/
default;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=<hive.server2.z
ookeeper.namespace>;
```

For example:

```
./bin/beeline
Warning: Unable to determine $DRILL_HOME
Beeline version 1.2.0-mapr-spark-MEP-6.0.0-1912 by Apache Hive
beeline> !connect jdbc:hive2://node1.cluster.com:5181/
default;ssl=true;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=ts2-ts
2;auth=maprsasl;
Connecting to jdbc:hive2://node1.cluster.com:5181/
default;ssl=true;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=ts2-ts
2;auth=maprsasl;
20/03/29 21:38:19 WARN MaprSaslClient: SASL Server qopProperty:
auth-confis different from Client: auth-conf,auth-int,auth.Using Server
one
Connected to: Spark SQL (version 2.4.4.0-mapr-630)
Driver: Hive JDBC (version 1.2.0-mapr-spark-MEP-6.0.0-1912)
Transaction isolation: TRANSACTION_REPEATABLE_READ
1: jdbc:hive2://node1.cluster.com:5181/defaul> show databases;
+-----------------+
| databaseName |
+-----------------+
| default             |
+-----------------+
1 row selected (0.11 seconds)
```


last-updated: Apr 11, 2022

Note: High availability for the Spark Thrift Server can be used in conjunction with HiveServer2 high availability. For more information about HiveServer2 high availability, see Enabling High Availability for Hive on page 3537.

Spark History Server SSL
Describes how to enable SSL for Spark History Server.

Note: For secure clusters, Spark History Server UI authentication is enabled by default.
If passwords are present in both `/opt/mapr/conf/ssl-client.xml` and `/opt/mapr/spark/spark-<spark_version>/conf/spark-defaults.conf` files, the password from the `spark-defaults.conf` file is used

Starting in EEP 4.0, for secure clusters, you can skip this step. For new installs done through the 6.0 MapR Installer, the installer enables this configuration. For manual installs and upgrades, running configure.sh -R enables these settings.

To configure SSL manually in a non-secure cluster or in versions earlier than EEP 4.0, add the following properties to the `spark-default.conf` file:

```
#HistoryServer https configure
spark.yarn.historyServer.address <Spark History Server node
hostname>:18480
spark.ssl.protocol tls
spark.ssl.historyServer.enabled true
spark.ssl.trustStore $MAPR_HOME/conf/ssl_truststore
spark.ssl.keyStore $MAPR_HOME/conf/ssl_keystore
spark.ssl.trustStorePassword <ssl-keystore-password>
spark.ssl.keyStorePassword <ssl-keystore-password>
```

Note: If a cluster is secure and you use unsecured ports with HTTP, you will be automatically redirected to HTTPS with secure ports. For example, if on the secure cluster you go to `http://node1:18080`, you will be redirected to `https://node1:18480`.

MapR Database Connectors for Apache Spark
This section describes the MapR Database connectors that you can use with Apache Spark.

Apache Spark is a software framework that is used to process data in memory in a distributed manner. Spark is replacing MapReduce in many use cases. The MapR Database Connectors for Spark enable users to write applications that access MapR Database JSON and Binary tables.

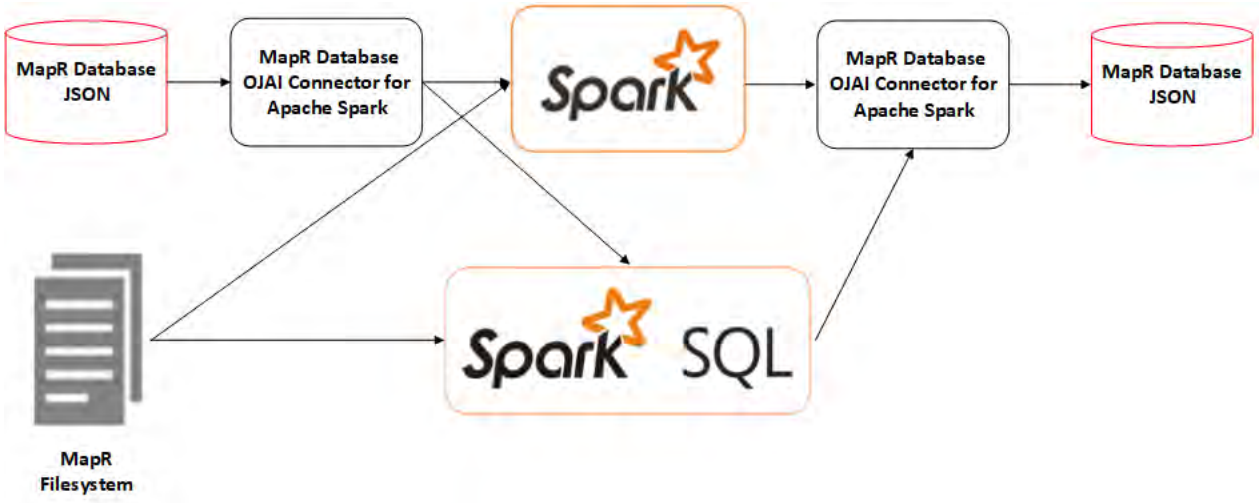
Understanding the MapR Database OJAI Connector for Spark
Using the MapR Database OJAI connector for Spark enables you build real-time and batch pipelines between your data and MapR Database JSON. Before getting started, it is important that you understand Spark terminology and workflow, system requirements and support, and OJAI connector and API features.

The MapR Database OJAI connector includes a set of APIs that enable you to write applications that consume MapR Database JSON tables and use them in Spark. The MapR Database OJAI Connector for Apache Spark is a companion to the MapR Database Binary Connector for Apache Spark on page 4101, which provides the equivalent functionality for MapR Database Binary tables.

MapR Database OJAI Connector with Spark Workflow

You can use the MapR Database OJAI Connector to extract data from MapR Database or MapR File System and transform that data using either Spark or Spark SQL, and then load it into MapR Database

JSON:



MapR Database OJAI Connector for Apache Spark Features

Principal features of the MapR Database OJAI Connector for Apache Spark include the following:

- Support for Scala and, beginning with EEP 4.1, Java and Python APIs

  This matrix shows the programming languages and features supported:

	Scala	Java	Python
RDD	Yes	Yes	No
DataFrame	Yes	Yes	Yes
Dataset	Yes	Yes	No
DStream	Yes	No	No

- APIs that enable you to load data from a MapR Database JSON table to an Apache Spark RDD, DataFrame, or Dataset
- Projection and filter pushdown for better performance
- Custom partitioner for RDDs that enables you to partition data for better performance
- APIs that save an Apache Spark RDD, DataFrame, or DStream to a MapR Database JSON table using either normal or bulk insert
- Support for Scala and Java bean classes
- Support for data locality
- Support for secondary indexes starting from EEP 7.0.0 and EEP 6.3.1.

The following features are not supported:

- MapR Database Binary tables

  Only MapR Database JSON tables are supported; access to MapR Database binary tables is provided through the MapR Database Binary Connector.

- Secondary indexes are not suported for previous EEP 7.0.0 and EEP 6.3.1 versions.

last-updated: Apr 11, 2022

Supported Product Versions and System Requirements

To use the MapR Database OJAI Connector for Apache Spark, you must have the following minimum software versions:

- MapR: 5.2.1 or later
- EEP 3.0 or later
- Spark 2.1.0 or later
- Scala 2.11 or later
- Java 8 or later

Support for DataFrames and Datasets is available starting in the EEP 4.0 release.

OJAI API

The MapR Database OJAI Connector for Apache Spark uses the OJAI API internally to access MapR Database JSON tables.

Related information

Spark Programming Guide
Spark SQL, DataFrames and Datasets Guide
Spark Streaming Programming Guide

Configuring the MapR Database OJAI Connector for Apache Spark
Before using the MapR Database OJAI Connector for Apache Spark, you must edit the `pom.xml` file for your project.

Add the Spark core dependency into the `pom.xml` file:

Note: If all dependent JAR files are already present on the node, consider setting the `scope` parameter to `provided`. For example:

```
<scope>provided</scope>
```

Setting the scope this way reduces the size of the JAR file.

```
<dependency>
  <groupId>org.apache.spark</groupId>
  <artifactId>spark-core_<scala_version></artifactId>
  <version><spark_artifact_version></version>
</dependency>
```

Add the Spark Maven dependency to the `pom.xml` file:

```
<dependency>
  <groupId>com.mapr.db</groupId>
  <artifactId>maprdb-spark</artifactId>
  <version><spark_artifact_version></version>
</dependency>
```

For example, see the dependencies for Spark 2.4.4.0 (EEP 6.3.0 release):

```
<dependency>
  <groupId>org.apache.spark</groupId>
  <artifactId>spark-core_2.11</artifactId>
  <version>2.4.4.0-mapr-630</version>
```

```
</dependency>
<dependency>
  <groupId>com.mapr.db</groupId>
  <artifactId>maprdb-spark</artifactId>
  <version>2.4.4.0-mapr-630</version>
</dependency>
```

To enable Maven to download dependencies, add the following repository information to the `pom.xml` file:

```
<repository>
  <id>mapr-releases</id>
  <url>https://repository.mapr.com/maven/</url>
  <snapshots>
    <enabled>false</enabled>
  </snapshots>
  <releases>
    <enabled>true</enabled>
  </releases>
</repository>
```

Related concepts

Maven Artifacts for MapR on page 4155
Maven artifacts can be used for dependency management when developing applications based on the MapR platform.

Loading Data from MapR Database Using the MapR Database OJAI Connector for Apache Spark
The MapR Database OJAI Connector for Apache Spark supports loading data as an Apache Spark RDD. Starting in the EEP 4.0 release, the connector introduces support for Apache Spark DataFrames and Datasets. DataFrames and Datasets perform better than RDDs. Whether you load your MapR Database data as a DataFrame or Dataset depends on the APIs you prefer to use. It is also possible to convert an RDD to a DataFrame.

Loading Data from MapR Database as an Apache Spark RDD

You can use the following API to load JSON-format data from a MapR Database table into an Apache Spark RDD of a JSON document:

Scala

For loading as an RDD, apply the following method on a `SparkContext` object:

```
def loadFromMapRDB[T](table: String):
RDD[T]
```

Java

For loading as an RDD, apply the following method on a `MapRDBJavaSparkContext` object:

```
mapRDBSparkContext.loadFromMapRDB(tabl
eName: String, clazz: Class)
```

Note: The only required parameter to the methods is tableName. All the others are optional.

The following example creates a `userprofilesRDD` by calling `loadFromMapRDB` from `SparkContext` (Scala) or `MapRDBSparkContext` (Java) and supplying the table (`"/tmp/user_profiles"`):

Scala

```
import com.mapr.db.spark._

val userprofilesRDD
```


last-updated: Apr 11, 2022

```
= sc.loadFromMapRDB("/tmp/
user_profiles")
```

Java

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;
import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;

MapRDBJavaSparkContext
mapRDBSparkContext = new
MapRDBJavaSparkContext(sc);
JavaRDD userprofilesRDD =
mapRDBSparkContext.loadFromMapRDB("/tm
p/user_profiles")
```

The following example creates a `usersInfo` RDD by calling `loadFromMapRDB` from `SparkContext` (Scala) or `MapRDBSparkContext` (Java) and supplying the table (`"/tmp/UserInfo"`):

Scala

```
import com.mapr.db.spark._

val usersInfo =
sc.loadFromMapRDB("/tmp/UserInfo")
```

Java

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;

MapRDBJavaRDD<OJAIDocument> usersInfo
=
mapRDBSparkContext.loadFromMapRDB("/tm
p/UserInfo")
```

In the previous example, the `usersInfo` data contains the following information:

- Address (map type)
- Date of birth (date type)
- First name (string type)
- Interests (string type)
- Last name (string type)

The following prints the fields and shows the output for a sample user:

Scala

```
usersInfo.foreach(println(_))
```

Java

```
usersInfo.foreach(System.out::println)
;
```

```
{
  "address":
     {"Pin":95035,"city":"milpitas","street":"350 holger way"},
  "dob":"1947-11-29",
  "first_name":"David",
  "interests":["football","books","movies"],
  "last_name":"Jones"
}
```

The following example shows a join operation performed on two different JSON documents using `address.city` as the join key:

Scala

```
import com.mapr.db.spark._

val maprd1 = sc.loadFromMapRDB("/tmp/
user_profiles")

val maprd2 = sc.loadFromMapRDB("/tmp/
user_income")

val collection = maprd1.map(a =>
(a.`address.city`[String],a))
.cogroup(maprd2.map(a=>(a.`address.cit
y`[String],a)))
.map(a =>
(a._1,a._2._1.size,a._2._2.size)).coll
ect
```

Java

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;
import scala.Tuple2;
import scala.Tuple3;
import java.util.Collection;

MapRDBJavaRDD<OJAIDocument> maprd1 =
mapRDBSparkContext.loadFromMapRDB("/tm
p/user_profiles");
MapRDBJavaRDD<OJAIDocument> maprd2 =
mapRDBSparkContext.loadFromMapRDB("/tm
p/user_income");

List collection =
maprd1.mapToPair(a -> new
Tuple2<>(a.getString("address.city"),
a))
.cogroup(maprd2.mapToPair(a -> new
Tuple2<>(a.getString("address.city"),
a)))
.map(a -> new Tuple3<>(a._1,
((Collection<?>)a._2._1).size(),
((Collection<?>)a._2._2).size()))
.collect();
```

The resulting RDD, `collection`, contains the count of the users in the `user_profiles` and `user_income` MapR Database tables.

The following example adds a new field into all the JSON documents:

Scala

```
import com.mapr.db.spark._

val maprd = sc.loadFromMapRDB("/tmp/
user_profiles")
val documents = maprd.map(a =>
{ a.`address.country` = "USA";
a}).collect
documents.saveToMapRDB("/tmp/
cleaned_user_profiles")
```

Java

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;

MapRDBJavaRDD<OJAIDocument> maprd =
mapRDBSparkContext.loadFromMapRDB("/tm
p/user_profiles");
List<OJAIDocument> documents =
maprd.map(a ->
{a.set("address.country", "USA");
return a;})

    .collect();
mapRDBSparkContext.saveToMapRDB(docume
nts, "/tmp/cleaned_user_profiles");
```

Improving Performance by Using Projection Pushdown and Filter Pushdown

To improve performance, you can supply a WHERE clause and projection fields to the `loadFromMapRDB` API. In the following example, a condition is supplied to the `loadFromMapRDB` function and only certain fields are specified in the SELECT clause:

Scala

```
import com.mapr.db.spark._

val userprofilesRDD =
sc.loadFromMapRDB("/tmp/
user_profiles")
                        .where([condit
ion])
                        .select("addre
ss",

"first_name",
                                "_id",

"last_name")
```

Java

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;
import org.ojai.store.QueryCondition;
```


last-updated: Apr 11, 2022

```
MapRDBJavaSparkContext
mapRDBSparkContext = new
MapRDBJavaSparkContext(spark.sparkCont
ext());
MapRDBJavaRDD userprofilesRDD =
mapRDBSparkContext.loadFromMapRDB("/tm
p/user_profiles")

            .where([condition])

            .select("address",

                    "first_name",

                    "_id",


"last_name");
```

The data is loaded based on the condition. The condition is pushed down to the server, and the server returns data based on the filtering. Only the fields specified in the SELECT clause are projected.

In the following example, the WHERE clause is used as a filter condition:

Scala

```
import com.mapr.db.spark._

val userprofilesRDD =
sc.loadFromMapRDB("/tmp/
user_profiles")
                      .where(field("sa
lary") >= 100)
```

Java

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;
import org.ojai.store.QueryCondition;

MapRDBJavaSparkContext
mapRDBSparkContext = new
MapRDBJavaSparkContext(spark.sparkCont
ext());
MapRDBJavaRDD userprofilesRDD =
mapRDBSparkContext.loadFromMapRDB("/tm
p/user_profiles")
.where(MapRDB.newCondition().is("salar
y",
QueryCondition.Op.GREATER_OR_EQUAL,
100));
```

The `userprofilesRDD` includes only those documents with a salary field greater than 100.

By specifying an `_id` field, you can find and retrieve a row for a given key:

Scala

```
import com.mapr.db.spark._

val userprofilesRDD =
sc.loadFromMapRDB("/tmp/
user_profiles")
```

```
                                .where(field("_i
d") === "k2")
```

Java

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;
import org.ojai.store.QueryCondition;

MapRDBJavaSparkContext
mapRDBSparkContext = new
MapRDBJavaSparkContext(spark.sparkCont
ext());
MapRDBJavaRDD userprofilesRDD =
mapRDBSparkContext.loadFromMapRDB("/tm
p/user_profiles")
                  .where(MapRDB.newCon
dition().is("_id",
QueryCondition.Op.EQUAL, "k2"));
```

WHERE Clause Semantics

The `loadFromMapRDB` API supports a WHERE clause to push down the filter to the JSON document API, ensuring that only relevant documents are propagated to the RDD.

You can use two options to provide the filter condition:

- Scala domain-specific language (DSL)
- `QueryCondition` (from OJAI API)

Following is an example of using `loadFromMapRDB` and supplying a condition by using Scala DSL:

Scala

```
Condition isDoe = field("last_name")
=== "Doe"
val userprofilesRDD
= sc.loadFromMapRDB("/tmp/
user_profiles").where(isDoe)
```

For more information about using Scala DSL, see Scala DSL for Specifying Filter Conditions on page 4060.

Following is an example of passing the condition using the `QueryCondition` API:

Scala

```
val maprd  =
sc.loadFromMapRDB(tableName)
            .where(MapRDB.newConditio
n()
            .is("_id",
QueryCondition.Op.EQUAL, "k2")
            .build())
```

Java

```
MapRDBJavaRDD rdd =
mapRDBJavaSparkContext.loadFromMapRDB(
tableName)
            .where(MapRDB.newConditio
n().is("_id",
```

```
QueryCondition.Op.EQUAL,
"k2").build());
```

For more information about `QueryCondition`, see Querying with Conditions on page 2642.

Note: For additional information, see Java Examples in the source code.

Creating an Apache Spark RDD of a Class

When loading data as an Apache Spark RDD, if you have a custom class in your application, you can present the data as objects of your class.

Scala

You must define the custom class using Jackson semantics for Scala modules. The following example defines a custom `User` class:

```
case class User (@JsonProperty("_id")
id:String,

@JsonProperty("first_name")
firstName:String,

@JsonProperty("last_name") lastName:
String,
            @JsonProperty("dob") dob:
ODate,

@JsonProperty("interests") interests:
List[String])
```

In the following example, by supplying `User` as a type parameter to the function while loading the MapR Database table, you can create an RDD of the `User` class:

```
val userprofilesRDD =
sc.loadFromMapRDB[User]("/tmp/
user_profiles")
                        .where("conditio
n")
```

When specifying a bean class, the SELECT clause is unnecessary and is ignored.

Java

You must define a custom bean class as follows:

```
public static class Person implements
Serializable {
            private String _id;
            private String firstName;
            private String lastName;
            private Date dob;
            private Seq<String>
interests;
            public String get_id()
{ return _id; }
            public void set_id(String
_id) { this._id = _id; }
            public String
getFirstName() { return firstName; }
            public void
setFirstName(String firstName)
```


last-updated: Apr 11, 2022

```
{ this.firstName = firstName; }
            public String
getLastName() { return lastName; }
            public void
setLastName(String lastName)
{ this.lastName = lastName; }
            public Date getDob()
{ return dob; }
            public void setDob(Date
dob) { this.dob = dob; }
            public Seq<String>
getInterests() { return interests; }
            public void
setInterests(Seq<String> interests)
{ this.interests = interests; }
           }
```

In the following example, by supplying the `User` bean class as a type parameter while loading the MapR Database table, you can create a `MapRDBJavaRDD` of the `User` class:

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;

MapRDBJavaSparkContext
mapRDBSparkContext = new
MapRDBJavaSparkContext(spark.sparkCont
ext());
MapRDBJavaRDD userprofilesRDD =
mapRDBSparkContext.loadFromMapRDB("/tm
p/user_profiles",

User.class).where(<condition>);
```

Scala DSL for Specifying Filter Conditions

When loading data from MapR Database as an Apache Spark RDD, you can use Scala DSL to specify filter conditions. This section shows examples of these filter conditions.

In the following examples, a class named `field` is introduced to represent a field in a condition. The field condition takes an argument as a String. The following table shows conditions written using Scala DSL:

Condition	Example
equality	`val idOnlyPredicate = field("_id") === "k2"`
greatherThan	`val simplePredicateWithComparisonOperator = field("a.c.d") > 10`
notexists	`val simpleNotExistsPredicate = field("a.c.e") notexists`
IN	`val inPredicate = field("a.c.d") in Seq(ODate.parse("2011-05-21"), ODate.parse("2013-02-22"))`
typeof	`val simpleTypeOfPredicate = field("a.c.d") typeof "INT"`

Condition	Example
complex condition with and	`val inPredicateWithMapAndArray = (field("a.c.d") in Seq(5,10)) and (field("a.c.e") notin Seq("aaa","bbb"))`
another complex condition	`val compositePredicateWithAndOnly = ((field("a.b") notexists ) and (field("p.q") typeof "DATE")) and (field("a.c.d") > 20L)`
between	`val predicateWithBetweenOp = field("a.c.d") between (ODate.parse("2015-01-15"), ODate.parse("2015-05-15"))`
predicate with equality check on Sequence of elements (representing array)	`val eqPredicateWithList = field("a.b") === Seq(12345L, "xyz")`
predicate with equality check on a map	`val eqWithMapPredicate = field("a") === Map("k" -> "kite", "m" -> "map")`

The MapR Database OJAI Connector for Apache Spark supports these predicates:

- `>`
- `>=`
- `<`
- `<=`
- `===`
- `!=`
- `between`
- `exists`
- `notin`
- `in`
- `notexists`
- `typeof`
- `nottypeof`
- `like`
- `notlike`

last-updated: Apr 11, 2022

- `matches`
- `notmatches`
- `sizeOf`

Here are examples for these operators:

- `field("a") > 10`
- `field("a") >= 10`
- `field("a") < 10`
- `field("a") <= 10`
- `field("a") === 10`
- `field("a") === Seq("aa", 10)`
- `field("a") === Map("aa" -> 10)`
- `field("a") != 10`
- `field("a") != Seq("aa", 10)`
- `field("a") != Map("aa" -> 10)`
- `field("a) between (10,20)`
- `field("a") exists`
- `field("a") notin Seq(10,20)`
- `field("a") in Seq(10, 20)`
- `field("a") notexists`
- `field("a") typeof "INT"`
- `field("a") nottypeof "INT"`
- `field("a") like "%s"`
- `field("a") notlike "%s"`
- `field("a") matches "*s"`
- `field("a") notmatches "*s"`

For `typeof`, these are the right-hand side values:

- `"INT"`
- `"INTEGER"`
- `"LONG"`
- `"BOOLEAN"`

- "STRING"
- "SHORT"
- "BYTE"
- "NULL"
- "FLOAT"
- "DOUBLE"
- "DECIMAL"
- "DATE"
- "TIME"
- "TIMESTAMP"
- "INTERVAL"
- "BINARY"
- "MAP"
- "ARRAY"

The sizeOf operator can have the following operations:

- sizeOf(field("a")) === 10
- sizeOf(field("a")) < 10
- sizeOf(field("a")) > 10
- sizeOf(field("a")) >= 10
- sizeOf(field("a")) <= 10
- sizeOf(field("a")) != 10

Java DSL for Specifying Filter Conditions

When loading data from MapR Database as an Apache Spark RDD, you can use Java DSL to specify filter conditions. This section shows examples of these filter conditions.

Condition	Example
equality	`QueryCondition equality = MapRDB.newCondition().is("_id", QueryCondition.Op.EQUAL, "k2").build();`
greatherThan	`QueryCondition greatherThan = QueryCondition simpleWithComparisonOperator = MapRDB.newCondition().is("a.b.c", QueryCondition.Op.GREATER, 10).build();`

last-updated: Apr 11, 2022

Condition	Example
notexists	`QueryCondition notexists = MapRDB.newCondition().notExists("a.c.e").build();`

IN	`List<ODate> odateList = new ArrayList<>();`
	`odateList.add(ODate.parse("2011-05-21"));`
	`odateList.add(ODate.parse("2013-02-22"));`
	`QueryCondition in = MapRDB.newCondition().in("a", odateList).build();`

typeof	`QueryCondition typeOf = MapRDB.newCondition().typeOf("a.c.d", Value.Type.INT).build();`

complex condition with and	`QueryCondition complexConditionWithAnd = MapRDB.newCondition()`
	`.and()`
	`.condition(MapRDB.newCondition().in("a", Arrays.asList(5, 10)))`
	`.condition(MapRDB.newCondition().notIn("b", Arrays.asList("aaa", "bbb")))`
	`.close().build();`

another complex condition	`QueryCondition anotherComplexCondition = MapRDB.newCondition()`
	`.and()`
	`.condition(MapRDB.newCondition().notExists("a.b"))`
	`.condition(MapRDB.newCondition().typeOf("p.q", Value.Type.DATE))`
	`.condition(MapRDB.newCondition().is("a.c.d", QueryCondition.Op.GREATER, 20L))`
	`.close().build();`

The MapR Database OJAI Connector for Apache Spark supports these predicates:

- `is (LESS, LESS_OR_EQUAL, EQUAL, NOT_EQUAL, GREATER_OR_EQUAL, GREATER)`
- `equals`
- `and`
- `exists`
- `in`
- `like`
- `matches`
- `notEquals`
- `notExists`
- `notIn`
- `notLike`
- `notMatches`

- notTypeOf
- or
- sizeOf
- typeOf

Here are examples for these operators:

- MapRDB.newCondition().is("a", QueryCondition.Op.GREATER, 10);
- MapRDB.newCondition().is("a",QueryCondition.Op.GREATER_OR_EQUAL, 10);
- MapRDB.newCondition().is("a", QueryCondition.Op.LESS, 10);
- MapRDB.newCondition().is("a", QueryCondition.Op.LESS_OR_EQUAL, 10);
- MapRDB.newCondition().is("a", QueryCondition.Op.EQUAL, 10);
- MapRDB.newCondition().is("a", QueryCondition.Op.NOT_EQUAL, 10);
- MapRDB.newCondition().exists("a");
- MapRDB.newCondition().notIn("a", Arrays.asList(10, 20));
- MapRDB.newCondition().in("a", Arrays.asList(10, 20));
- MapRDB.newCondition().notExists("a");
- MapRDB.newCondition().typeOf("a", Value.Type.INT);
- MapRDB.newCondition().notTypeOf("a", Value.Type.INT);
- MapRDB.newCondition().like("a", "%s");
- MapRDB.newCondition().notLike("a", "%s");
- MapRDB.newCondition().matches("a", "*s");
- MapRDB.newCondition().notMatches("a", "*s");

For `typeof`, these are the right-hand side values:

- "INT"
- "INTEGER"
- "LONG"
- "BOOLEAN"
- "STRING"
- "SHORT"
- "BYTE"
- "NULL"

- "FLOAT"
- "DOUBLE"
- "DECIMAL"
- "DATE"
- "TIME"
- "TIMESTAMP"
- "INTERVAL"
- "BINARY"
- "MAP"
- "ARRAY"

The `sizeOf` operator can have the following operations:

- `MapRDB.newCondition().sizeOf("a", QueryCondition.Op.EQUAL, 10);`
- `MapRDB.newCondition().sizeOf("a", QueryCondition.Op.LESS, 10);`
- `MapRDB.newCondition().sizeOf("a", QueryCondition.Op.GREATER, 10);`
- `MapRDB.newCondition().sizeOf("a", QueryCondition.Op.LESS_OR_EQUAL, 10);`
- `MapRDB.newCondition().sizeOf("a", QueryCondition.Op.GREATER_OR_EQUAL, 10);`
- `MapRDB.newCondition().sizeOf("a", QueryCondition.Op.NOT_EQUAL, 10);`

Using the Custom Partitioner with the MapR Database OJAI Connector for Apache Spark

In any distributed computing system, partitioning data is crucial to achieve the best performance. Apache Spark provides a mechanism to register a custom partitioner for partitioning the pipeline. The MapR Database OJAI Connector for Apache Spark includes a custom partitioner you can use to optimally partition data in an RDD.

The MapR Database OJAI Connector for Apache Spark's custom partitioner takes the following classes as keys:

- `String`
- `ByteBuffer` (as serializable `ByteBuffer`)

You can register this custom partitioner with either the `partitionBy` function or the `repartitionAndSortWithinPartitions` function.

The connector supports two versions of the custom partitioner. One version takes a MapR Database JSON table as an input. The partition information of the table is used to partition the data, so the `saveToMapRDB` call can use a `bulkInsert` to store the data. The `bulkInsert` option requires that you have the data already sorted on the `_id` key.

The other version of the custom partitioner takes an array of splits as an input.

Specifying tablename for the Partitioner

If you already have a table that has been created and partitioned based on a set of keys, you can can specify that the RDD be partitioned in the same way (using the same set of keys). In the following example, `/srctable` is provided as a reference partitioner for `/dsttable`:

Scala

```
sc.loadFromMapRDB("/srctable")
            .keyBy(doc =>
doc._id[String])
            .repartitionAndSortWithinP
artitions(MapRDBSpark.newPartitioner[S
tring]("/dsttable"))
            .saveToMapRDB("/
dsttable", createTable = false,
bulkInsert = true)
```

Specifying a String Seq as an Array of Splits

In the following example, the first line creates an array of splits as `id1`, `id2` ... `id9`. The rest of the example splits the RDD based on the array of splits:

Scala

```
val dstSplits: Array[String] = (1 to
9 by 3).map("id" + _).toArray
val partitionRDD =
sc.loadFromMapRDB("/srctable")
            .keyBy(doc =>
doc._id[String])
            .repartitionAndSortWithinP
artitions(MapRDBSpark.newPartitioner[S
tring](dstSplits))
            .saveToMapRDB("/
dsttable", createTable = true,
bulkInsert = true)
```

Specifying a ByteBuffer Seq as an Array of Splits

Suppose you have an array of byte buffers to use as the array of splits for the partitioner. You must convert the byte buffers to serializable byte buffers first:

Scala

```
// Converting bytebuffer to
serializable bytebuffer
val dstSplits =
arrayOfByteBuffer.map(x =>
MapRDBSpark.serializableBinaryValue(x)
)
sc.loadFromMapRDB("/srctable")
  //KeyBy serializable bytebuffer
  .keyBy(doc =>
doc.getBinarySerializable(binaryField)
)
  .repartitionAndSortWithinPartitions(
MapRDBSpark.newPartitioner(dstSplits))
  .saveToMapRDB("/dsttable",
createTable = true, bulkInsert = true)
```

Specifying tablename for the Partitioner with ByteBuffer as Id Fields

Suppose you have a table with keys that are binary or `ByteBuffer`, and you have an RDD with some rows and some values. You can repartition the RDD based on the partitions of the table. The following example reads the document from `/srctable`, but you could provide any table. In the second line, the example specifies a `keyBy` call on an ID that is binary serializable. In the last line, `/dsttable` is the RDD that has a key of serializable `ByteBuffers`:

Scala

```
sc.loadFromMapRDB("/srctable")
  .keyBy(doc =>
doc.getIdBinarySerializable())
  .repartitionAndSortWithinPartitions(
MapRDBSpark.newPartitioner[ByteBuffer]
("/dsttable"))
```

Note: You must provide the key type of the `PairedRDD` on which the partitioning is specified. If the IDs are serializable bytebuffers, specify `ByteBuffer`. Otherwise, specify `String`.

After the data is partitioned with the custom partitioner, all the downstream transformations should be non-partition-changing transformations. Here is the code for passing on partitioner for an RDD:

Scala

```
user_profiles.repartitionAndSortWithin
Partitions
   (MapRDBSpark.newPartitioner[String]
(<table-name>))
```

Or you can use the `partitionBy` function on the RDD:

Scala

```
user_profiles.partitionBy(MapRDBSpark.
newPartitioner[String](<table-name>))
```

The key of the data for this partitioner should be of the same type as that of the key of the table name. This partitioner yields a single partition if the table supplied to it is not pre-split. The number of partitions is calculated based on the table's existing tablet information.

For a table created with the `bulkInsert` option set to `true`, one of the following applies:

- If the table is pre-split, then the resulting partitions can be > 1.
- If the table is no-split, then the resulting partitions will be 1 if no partition information is available from the RDD lineage.

Loading Data from MapR Database as an Apache Spark DataFrame

To load data from a MapR Database JSON table into an Apache Spark DataFrame, invoke the following API:

Scala

For loading as a DataFrame, apply the following method on a `SparkSession` object:

```
def loadFromMapRDB[T](tableName:
String,
                schema: StructType):
DataFrame

import com.mapr.db.spark.sql._
```

```
val df
= sparkSession.loadFromMapRDB[T]
("/tmp/user_profiles"): DataFrame
```

Java

For loading as a DataFrame (Datasets of Row), apply the following method on a `MapRDBJavaSession` object:

```
def loadFromMapRDB(tableName: String,
schema: StructType, sampleSize:
Double):DataFrame

import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;
import
org.apache.spark.sql.SparkSession;

MapRDBJavaSession maprSession = new
MapRDBJavaSession(spark);
maprSession.loadFromMapRDB("/tmp/
user_profiles");
```

Note: Java supports only DataSets of Row (Dataset<Row>).

Python

For loading as a DataFrame, apply the following method on a `SparkSession` object:

```
loadFromMapRDB(table_name, schema,
sample_size)

from pyspark.sql import SparkSession

df = spark.loadFromMapRDB("/tmp/
user_profiles")
```

Note: PySpark supports only DataFrames (Dataset<Row>).

Note: The only required parameter to the methods is tableName. All the others are optional.

This creates a DataFrame object corresponding to the MapR Database table specified by the `tableName` parameter.

Both DataFrames and MapR Database tables work with structured data. DataFrames need a fixed schema, whereas MapR Database allows for a flexible schema. When loading data into a DataFrame, you can map your data to a schema by specifying the schema parameter in the `loadFromMapRDB` call. You can also provide an application class as the type `[T]` parameter in the call. These two approaches are the preferred methods for loading data into DataFrames.

For data exploration use cases, you might not know the schema of your MapR Database table. For those situations, the MapR Database OJAI connector for Apache Spark can infer the schema by sampling data from the table.

Whenever possible, the MapR Database OJAI Connector for Apache Spark pushes projections and filters for better performance. This allows MapR Database to project and filter data before returning it to your client application.

The following subtopics describe these techniques.

last-updated: Apr 11, 2022

Optimizing MapR Database Lookups in Spark Jobs

The `lookupFromMapRDB()` API utilizes the primary and secondary indexes on a MapR Database table to optimize table lookups and outputs the results to an Apache Spark DataFrame.

Important: The `lookupFromMapRDB()` API functionality requires a patch. The patch works with EEP 6.2.0 (Core 6.1.0, Spark 2.4.0.0) and EEP 6.3.0 (Core 6.1.0, Spark 2.4.4.0). To install patches, see Applying a Patch

The `loadfromMapRDB()` API in MapR Database Connectors for Apache Spark is optimized to load massive amounts of data from MapR Database tables with high throughput. In cases where a Spark job needs to lookup a small number of documents based on the equality (or short range) condition on a primary or secondary key, the `lookupFromMapRDB()` API should be used.

Invoke the `lookupFromMapRDB()` API when the filter conditions in short range and equality queries reference primary and secondary keys. If the filter condition references any non-primary keys (fields other than the `_id` field), a secondary index must exist on the secondary keys. Indexes on the filtering keys is essential to achieving reasonable performance of lookup queries in MapR Database tables.

The `lookupFromMapRDB()` API uses the secondary keys in indexes to lookup values in the primary table. For example, if a query contains the filter conditions `mydate = '2012-03-26'` and `myid = '120026015'`, a secondary index (of type composite) created on the `mydate` and `myid` fields must exist for the query to quickly output results.

Examples on the following tabs demonstrate how to invoke the `lookupFromMapRDB()` API to perform a lookup in a MapR Database table and output the results to an Apache Spark DataFrame:

Scala

```
import com.mapr.db.spark.sql._
import spark.implicits._
val df = spark.lookupFromMapRDB("/
tbl")
df.filter("mydate" === "2012-03-26"
&& $"myid" === 120026015).show
```

Java

```
SparkSession sparkSession =
SparkSession.builder().getOrCreate();
MapRDBJavaSession mapRDBJavaSession =
new MapRDBJavaSession(sparkSession);
Dataset<Row> df2 =
mapRDBJavaSession.lookupFromMapRDB("/
tbl");
df2.filter("mydate = '2012-03-26' and
myid = '120026015'").show();
```

Python

```
from pyspark.sql import SparkSession
df = spark.lookupFromMapRDB("/tbl")
df.filter("mydate = '2012-03-26' and
myid = '120026015'").show()
```

Loading Data into a DataFrame Using an Explicit Schema

If you know the schema of your data, you can specify an explicit schema when loading a DataFrame.

The following example loads data into a user profile table using an explicit schema:

Scala

```
import
org.apache.spark.sql.SparkSession
import com.mapr.db.spark.sql._
```


last-updated: Apr 11, 2022

```
val addressSchema =
StructType(StructField("Pin",
IntegerType) ::

StructField("city", StringType) ::

StructField("street", StringType) ::
Nil)

val personSchema =
StructType(StructField("_id",
StringType) ::

StructField("first_name",
StringType) ::

StructField("last_name",
StringType) ::

StructField("address",
addressSchema) ::

StructField("interests",
ArrayType(StringType)) :: Nil)

val df
= sparkSession.loadFromMapRDB("/tmp/
user_profiles", personSchema)
```

Java

```
import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;
import
org.apache.spark.sql.SparkSession;

StructField[] addressSchema = {
                                new
StructField("Pin", IntegerType, true,
Metadata.empty()),
                                new
StructField("city",StringType, true,
Metadata.empty()),
                                new
StructField("street", StringType,
true, Metadata.empty())
                               };
StructField[] schemaFields = {
   new StructField("_id", StringType,
true, Metadata.empty()),
   new StructField("first_name",
StringType, true, Metadata.empty()),
   new StructField("address", new
StructType(addressSchema), true,
Metadata.empty()),
   new StructField("interests", new
ArrayType(StringType, true), true,
Metadata.empty())
                              };
StructType personSchema = new
StructType(schemaFields);
MapRDBJavaSession maprSession = new
```

```
MapRDBJavaSession(sparkSession);
Dataset<Row> df =
maprSession.loadFromMapRDB("/tmp/
user_profiles", personSchema);
```

Python

```
from pyspark.sql import SparkSession

addressSchema = [StructField("Pin",
IntegerType(), True),
                 StructField("city",
StringType(), True),

StructField("street", StringType(),
True)]
schemaFields = [StructField("_id",
StringType(), True),

StructField("first_name",
StringType(), True),

StructField("last_name",
StringType(), True),

StructField("address",
StructType(addressSchema), True),

StructField("interests",
ArrayType(StringType()), True)]
personSchema =
StructType(schemaFields)

df
= spark_session.loadFromMapRDB("/tmp/
user_profiles", personSchema)
```

To create the DataFrame object named `df`, pass the schema as a parameter to the load call. Invoke the `loadFromMapRDB` method on a `SparkSession` object.

The resulting schema of the object is the following:

```
df.printSchema()
 ---------------------------------
 root
 |-- _id: String (nullable = true)
 |-- first_name: String (nullable = true)
 |-- last_name: String (nullable = true)
 |-- address: Struct (nullable = true)
 |    |-- Pin: integer (nullable = true)
 |    |-- city: string (nullable = true)
 |    |-- street: string (nullable = true)
 |-- interests: array (nullable = true)
 |    |-- element: string (containsNull = true)
```

When specifying `StructField` in a schema, optionally specify whether the field is nullable. In the example above, all fields are nullable.

Depending on the nullability of the field in the schema and the existence of fields in the MapR Database table, the load returns an `InvalidSchema` exception in the following cases:

- The schema contains a non-nullable field and the load attempts to put a NULL value into the field.

- The schema contains a non-nullable field and the field does not exist in the MapR Database table.
- The MapR Database table has fields that do not exist in the specified schema.

Loading Data into a DataFrame Using a Type Parameter

If the structure of your data maps to a class in your application, you can specify a type parameter when loading into a DataFrame.

Specify the application class as the type parameter in the load call. The load infers the schema from the class.

The following example creates a DataFrame with a `Person` schema by passing the `Person` class as the type parameter in the load call:

Scala

```
import
org.apache.spark.sql.SparkSession
import com.mapr.db.spark.sql._

case class Address(Pin: Integer,
street: String, city: String)

          case class Person(_id:
String,
          First_name: String,
          last_name: String,
          Address: Address,
          Interests: Seq[String])

val df
= sparkSession.loadFromMapRDB[Person]
("/tmp/user_profiles")
```

Java

```
import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;
import
org.apache.spark.sql.SparkSession;

public static class Address
implements Serializable {
     private Integer pin;
     private String street;
     private String city;

     public Integer getPin() { return
pin; }
     public void setPin(Integer pin)
{ this.pin = pin; }
     public String getStreet()
{ return street; }
     public void setStreet(String
street) { this.street = street; }
     public String getCity() { return
city; }
     public void setCity(String city)
{ this.city = city; }
}
public static class Person implements
Serializable {
     private String _id;
     private String firstName;
```


last-updated: Apr 11, 2022

```
     private String lastName;
     private Date dob;
     private Seq<String> interests;

     public String get_id() { return
_id; }
     public void set_id(String _id)
{ this._id = _id; }
     public String getFirstName()
{ return firstName; }
     public void setFirstName(String
firstName) { this.firstName =
firstName; }
     public String getLastName()
{ return lastName; }
     public void setLastName(String
lastName) { this.lastName =
lastName; }
     public Date getDob() { return
dob; }
     public void setDob(Date dob)
{ this.dob = dob; }
     public Seq<String>
getInterests() { return interests; }
     public void
setInterests(Seq<String> interests)
{ this.interests = interests; }
}
MapRDBJavaSession maprSession = new
MapRDBJavaSession(sparkSession);
Dataset<Row> df =
maprSession.loadFromMapRDB(tableName,
Person.class);
```

You must invoke the `loadFromMapRDB` method on a `SparkSession` or `MapRDBJavaSession` object.

All fields in an application bean class are nullable by default. The only circumstance in which the load returns an `InvalidSchema` exception is if the MapR Database table contains fields not included in the bean class.

The resulting schema of the object is as follows:

```
df.printSchema()
 ---------------------------------
 root
 |-- _id: String (nullable = true)
 |-- first_name: String (nullable = true)
 |-- last_name: String (nullable = true)
 |-- address: Struct (nullable = true)
 |    |-- Pin: integer (nullable = true)
 |    |-- street: string (nullable = true)
 |    |-- city: string (nullable = true)
 |-- interests: array (nullable = true)
 |    |-- element: string (containsNull = true)
```

Loading Data into a DataFrame Using Schema Inference

If you do not know the schema of the data, you can use schema inference to load data into a DataFrame. This section describes how to use schema inference and restrictions that apply

When you do not specify a schema or a type when loading data, schema inference triggers automatically. The MapR Database OJAI Connector for Apache Spark internally samples documents from the MapR Database JSON table and determines a schema based on that data sample. By default, the sample size is

1000 documents. Alternatively, you can specify a sample size parameter. The parameter is optional in the `loadFromMapRDB` call and is named `sampleSize`. The following example specifies using a sample size of 100 documents:

Scala

```
import
org.apache.spark.sql.SparkSession
import com.mapr.db.spark.sql._

val df =
sparkSession.loadFromMapRDB(tableName,
 sampleSize : 100)
```

Java

```
import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;
import
org.apache.spark.sql.SparkSession;

MapRDBJavaSession maprSession = new
MapRDBJavaSession(spark);
Dataset<Row> df =
maprSession.loadFromMapRDB(tableName,
100);
```

Python

```
from pyspark.sql import SparkSession

df = spark.loadFromMapRDB(table_name,
100)
```

Important: Because schema inference relies on data sampling, it is non-deterministic. It is not well suited for production use where you need predictable results. Inferring schema results in reading sample rows from the table, hence execution time varies with number of rows in the source table.

Sampling Using Reader Functions

An alternative to sampling data using the `loadFromMapRDB` call is to use reader functions.

To use the DataFrame reader function (for Scala only), call the following methods:

```
val df = sparkSession.read.maprdb(tableName)
```

To use the reader function with basic Spark, call the `read` function on a `SQLContext` object as follows:

Scala

```
import org.apache.spark.sql.SQLContext

val df =
sqlContext.read.format("com.mapr.db.sp
ark.sql")
         .option("tableName",
<table-name>)
         .option("sampleSize",
100).load()
```

Java

```
import
org.apache.spark.sql.SQLContext;
```


last-updated: Apr 11, 2022

```
Dataset<Row> df = sqlContext.read()
                    .format("com.mapr.db
.spark.sql")
                    .option("tableName",
 <table-name>).load();
```

Python

```
from pyspark.sql import SQLContext

df = sql_context.read\
     .format("com.mapr.db.spark.sql.De
faultSource")\
     .option("tableName",
<table-name>).load()
```

Type Conflict Resolution When Sampling

When sampling data during schema inference, you might encounter conflicting value types within a field. The connector uses the following rules to resolve type conflicts:

- If the two conflicting types are each one of the following, the resolved type is the wider of the two types:
  - `ByteType`
  - `ShortType`
  - `IntegerType`
  - `LongType`
  - `FloatType`
  - `DoubleType`

  The type list above is arranged in increasing order of width. For example, if one document contains a field of type `ByteType` and the other contains a field of type `FloatType`, the resultant type is `FloatType`.
- If one of the types is `DecimalType`, then the resultant type is `DecimalType`, if and only if `DecimalType` is the wider of the two types.
- If the two types are `StructType`, each with different fields, then the resultant type is a new `StructType` that contains all the fields in each `StructType`.
- If the two types are `ArrayType`, each with different element types, then the resultant type is a new `ArrayType` where the type of the elements in the array is resolved using the aforementioned rules.
- If none of the above rules can be used for resolving type conflicts, then during data conversion, the load reports a `ConflictType` exception.

Suppose `Name` contains String values in some rows and a map with `first_name` and `last_name` as nested fields in other rows. During schema inference, the conflict resolution logic encounters two different types for the same field, `StringType` and `MapType`. It will note the conflict and return a `ConflictType` exception later when converting the data during the load.

By default, conflict exceptions occur during data conversion. To change this so that the exception is returned during the conflict resolution stage, set the `FailOnConflict` option to true :

Scala

```
val df =
spark.read.maprdb(<tableName>,
Map("sampleSize" -> 100,
"FailOnConflict" -> true))
```

Invalid Schemas

When using schema inference, missing and extra fields are resolved in the following ways:

- If a field in the inferred schema is missing in the MapR Database JSON document, the field is set to null.
- If there are fields in a MapR Database JSON document that are not in the inferred schema, the load returns an `InvalidSchema` exception.

Type Mapping Between MapR Database JSON and DataFrames

This table maps data types between MapR Database JSON OJAI and Apache Spark DataFrame.

Note: Not all DataFrame data types are supported by MapR Database, for a list of supported data types, see JSON Documents on page 508.

OJAI Data Type	DataFrame Data Type
Boolean	BooleanType
String	StringType
Byte	ByteType
Short	ShortType
Int	IntegerType
Long	LongType
Float	FloatType
Double	DoubleType
Decimal	DecimalType
Date	DateType
Time	TimestampType
TimeStamp	TimeStampType
Interval	CalendarIntervalType
Binary	BinaryType
Map	StructType
Array	ArrayType

Note: The OJAI `Time` data type is converted to a Spark `TimestampType` with the date set to the epoch date. Spark SQL does not support a `TIME` type.

Loading Data from MapR Database as an Apache Spark Dataset

You can use one of three ways to load data from MapR Database into an Apache Spark Dataset:

- Load the data into a Dataset.
- Load the data into a DataFrame, and then convert it to a Dataset.

last-updated: Apr 11, 2022

- Load the data into a Dataset using a custom encoder.

Load into a Dataset

Scala

For loading as a Dataset, apply the following method on a `SparkSession` object:

```
def loadFromMapRDB[T](table: String,
schema : StructType).as [T]: Dataset

import com.mapr.db.spark.sql._

val ds
= sparkSession.loadFromMapRDB[T]
("/tmp/user_profiles").as [T]: Dataset
```

Java

For loading as a Dataset, apply the following method on a `MapRDBJavaSession` object:

```
def loadFromMapRDB[T <:
java.lang.Object](tableName: String,
schema: StructType, sampleSize:
Double, clazz: Class[T]): Dataset[T]

import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;

MapRDBJavaSession maprSession = new
MapRDBJavaSession(sparkSession);

Dataset<Row> ds =
maprSession.loadFromMapRDB("/tmp/
user_profiles");
```

Note: The only required parameter to the methods is tableName. All the others are optional.

Load into DataFrame and Convert to Dataset

To load the data as a DataFrame, see Loading Data from MapR Database as an Apache Spark DataFrame on page 4068. To convert the DataFrame to a Dataset, use the `as[<type>]` method. The `<type>` can be any of the basic types in Scala.

The following code example creates a `Dataset[Person]` using the `as[<type>]` method:

Scala

```
import
org.apache.spark.sql.SparkSession
import com.mapr.db.spark.sql._

case class Address(Pin: Integer,
street: String, city: String)

case class Person (_id:String,
              first_name:String,
              last_name: String, dob:
java.sql.Date,
              Interests: Seq[String,
address: Address)
```

```
val ds
= sparkSession.loadFromMapRDB[Person]
("/tmp/user_profiles").as[Person]
```

Java

```
import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;

public static class Address
implements Serializable {
     private Integer pin;
     private String street;
     private String city;

     public Integer getPin() { return
pin; }
     public void setPin(Integer pin)
{ this.pin = pin; }
     public String getStreet()
{ return street; }
     public void setStreet(String
street) { this.street = street; }
     public String getCity() { return
city; }
     public void setCity(String city)
{ this.city = city; }
}

public static class Person implements
Serializable {
     private String _id;
     private String firstName;
     private String lastName;
     private Date dob;
     private Seq<String> interests;
               public String get_id()
{ return _id; }
               public void
set_id(String _id) { this._id = _id; }
               public String
getFirstName() { return firstName; }
               public void
setFirstName(String firstName)
{ this.firstName = firstName; }
               public String
getLastName() { return lastName; }
               public void
setLastName(String lastName)
{ this.lastName = lastName; }
               public Date getDob()
{ return dob; }
               public void setDob(Date
dob) { this.dob = dob; }
               public Seq<String>
getInterests() { return interests; }
               public void
setInterests(Seq<String> interests)
{ this.interests = interests; }
               }
Dataset<Person> ds =
```


last-updated: Apr 11, 2022

```
maprSession.loadFromMapRDB(tableName,
Person.class);
```

Load into Dataset Using Custom Encoder

You can create a custom encoder for Java bean classes by calling the `Encoders.bean` method. `Encoders.bean` only support Java classes. To create a Dataset of the Scala class, the previous code can be used. The following example shows how to load into a Dataset by creating a custom encoder for a Java class named `beanClass`:

Scala

```
import
org.apache.spark.sql.SparkSession

import com.mapr.db.spark.sql._

val ds =
sparkSession.loadFromMapRDB("/tmp/
user_profiles")
         .as(Encoders.bean(beanClass))
```

Java

```
import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;

maprSession.loadFromMapRDB("/tmp/
user_profiles").as(Encoders.bean(beanC
lass);
```

Filter Pushdown

After you have loaded data into a Dataset, you can apply filter pushdowns. The following example filters on `first_name`:

Scala

```
ds.filter($"first_name" === "David")
```

Java

```
ds.filter(col("first_name").equalTo("D
avid")).show();
```

See Projection and Filter Pushdown with Apache Spark DataFrames and Datasets on page 4080 for other examples.

Projection and Filter Pushdown with Apache Spark DataFrames and Datasets

Projection and filter pushdown improve query performance. When you apply the `select` and `filter` methods on DataFrames and Datasets, the MapR Database OJAI Connector for Apache Spark pushes these elements to MapR Database where possible.

Projection Pushdown

Projection pushdown minimizes data transfer between MapR Database and the Apache Spark engine by omitting unnecessary fields from table scans. It is especially beneficial when a table contains many columns.

When you invoke the following `select` method on a DataFrame, the connector pushes the projection:

Scala

```
import
org.apache.spark.sql.SparkSession
import com.mapr.db.spark.sql._

val df
= sparkSession.loadFromMapRDB("/tmp/
user_profiles")
df.select("_id", "first_name",
"last_name")
```

Java

```
import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;

MapRDBJavaSession maprSession = new
MapRDBJavaSession(sparkSession);
Dataset<Row> df =
maprSession.loadFromMapRDB("/tmp/
user_profiles");
df.select("_id", "first_name",
"last_name");
```

Python

```
from pyspark.sql import SparkSession

df
= spark_session.loadFromMapRDB("/tmp/
user_profiles")
df.select("_id", "first_name",
"last_name")
```

The equivalent example using Datasets is as follows:

Scala

```
import
org.apache.spark.sql.SparkSession
import com.mapr.db.spark.sql._

val ds
= sparkSession.loadFromMapRDB[Person]
("/tmp/user_profiles").as[Person]
ds.select("_id", "first_name",
"last_name")
```

Java

```
import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;

MapRDBJavaSession maprSession = new
MapRDBJavaSession(sparkSession);
Dataset<Row> ds =
maprSession.loadFromMapRDB("/tmp/
user_profiles", Person.class);
ds.select("_id", "first_name",
"last_name");
```


last-updated: Apr 11, 2022

Filter Pushdown

Filter pushdown improves performance by reducing the amount of data passed between MapR Database and the Apache Spark engine when filtering data.

Consider the following example:

Scala

```
import
org.apache.spark.sql.SparkSession
import com.mapr.db.spark.sql._

val df
= sparkSession.loadFromMapRDB("/tmp/
user_profiles")
df.filter("first_name = 'Bill'")
```

Java

```
import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;

MapRDBJavaSession maprSession = new
MapRDBJavaSession(spark);
Dataset<Row> df =
maprSession.loadFromMapRDB("/tmp/
user_profiles");
df.filter("first_name = 'Bill'")
```

Python

```
from pyspark.sql import SparkSession

df
= spark_session.loadFromMapRDB("/tmp/
user_profiles")
df.filter("first_name = 'Bill'")
```

The MapR Database OJAI Connector for Apache Spark pushes the filter `firstName = 'Bill'` down to MapR Database.

The equivalent example using Datasets is as follows:

Scala

```
import
org.apache.spark.sql.SparkSession
import com.mapr.db.spark.sql._

val ds
= sparkSession.loadFromMapRDB[Person]
("/tmp/user_profiles").as[Person]
ds.filter($"first_name" === "Bill")
```

Java

```
import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;

Dataset ds =
maprSession.loadFromMapRDB("/tmp/
user_profiles").as(Encoders.bean(Perso
n.getClass()));
```

```
ds.filter(col("first_name").equalTo("B
ill"));
```

The following DataFrame filters those rows in which first_name is either `"David"` or `"Peter"`:

Scala

```
df.filter($"first_name" === "David"
|| $"first_name" === "Peter")
```

Java

```
df.filter(col("first_name").equalTo("D
avid").or(col("first_name").equalTo("P
eter")))
```

Python

```
df.filter((col("first_name") ==
"David") | (col("first_name") ==
"Peter"))
```

The following DataFrame retrieves only the rows in which the first_name is `"David"` and the last_name is `"Jones"`:

Scala

```
df.filter($"first_name" === "David"
&& $"last_name" === "Jones")
```

Java

```
df.filter(col("first_name").equalTo("D
avid").and(col("last_name").equalTo("J
ones")))
```

Python

```
df.filter((col("first_name") ==
"David") & (col("last_name") ==
"Jones"))
```

The following uses a `not` condition to return rows where the first_name is not `"David"` and the last_name is not `"Peter"`:

Scala

```
df.filter(not($"first_name" ===
"David || $"last_name" === "Peter"))
```

Java

```
df.filter(not(col("first_name").equalT
o("David").or(col("last_name").equalT
o("Peter"))))
```

Python

```
df.filter(~((col("first_name") ==
"David") | (col("last_name") ==
"Peter")))
```

The MapR Database OJAI Connector pushes down all of the filters shown in the earlier examples. It can push down the following types of filters, provided that the field is not an `Array` or `Map`:

- Equal To (=)

last-updated: Apr 11, 2022

- Not Equal To (`!=`)
- Less Than (<)
- Less Than or Equal To (<=)
- Greater Than (>)
- Greater Than or Equal To (>=)
- In Predicate (`IN`)
- Like predicate (`LIKE`)
- `AND`, `OR`
- `NOT`

Restrictions

Pushdowns with DataFrames and Datasets are not supported in the following scenarios:

- Filters on complex types, including arrays, maps, and structs

  For example, a filter on a field in a map, as shown in the following example, is not pushed down:

  Scala
  ```
  df.filter($"address.city" === "Milpitas")
  ```

  Java
  ```
  df.filter(col("address.city").equalTo("Milpitas"));
  ```

  Python
  ```
  df.filter(col("address.city") == "Milpitas")
  ```

- Filters with functions `sizeof`, `typeof`, and `matches`

  Spark SQL does not support these functions.

- Projections on complex types, including arrays, maps, and structs

  For example, if you select an element of an array, as shown in the following example, it is not pushed down:

  Scala
  ```
  ds.select($"hobbies" (0))
  ```

  Java
  ```
  df.select(col("hobbies").getItem(0));
  ```

  Python
  ```
  df.select(col("hobbies").getItem(0))
  ```

These limitations do not apply to pushdowns on RDDs. An alternative is to apply the pushdown using an RDD, and then convert the RDD to a DataFrame.

Note: MapR Database 6.0 introduces support for Secondary Indexes on page 544, but the MapR Database OJAI Connector for Spark does not currently leverage them.

Converting an Apache Spark RDD to an Apache Spark DataFrame

When APIs are only available on an Apache Spark RDD but not an Apache Spark DataFrame, you can operate on the RDD and then convert it to a DataFrame.

You can convert an RDD to a DataFrame in one of two ways:

- Use the helper function, `toDF`.
- Convert the RDD to a DataFrame using the `createDataFrame` call on a `SparkSession` object.

Using the toDF Helper Function

The `toDF` method is available through `MapRDBTableScanRDD`. The following example loads an RDD that filters on `first_name` equal to `"Peter"` and projects the `_id` and `first_name` fields, and then converts the RDD to a DataFrame:

Scala

```
import com.mapr.db.spark.sql._

val df =
sc.loadFromMapRDB(<table-name>)
           .where(field("first_name")
=== "Peter")
           .select("_id",
"first_name").toDF()
```

Using SparkSession.createDataFrame

With this approach, you can convert an `RDD[Row]` to a DataFrame by calling `createDataFrame` on a `SparkSession` object. The API for the call is as follows:

Scala

```
def createDataFrame(RDD, schema:
StructType)
```

You might need to first convert an `RDD[OJAIDocument]` to an `RDD[Row]`. The following example shows how to do this:

Scala

```
val df = sparkSession.createDataFrame(
         rdd.map(doc
=>MapRDBSpark.docToRow(doc, schema)),
schema)
```

`rdd` is of type `RDD[OJAIDocument]`. The `docToRow` call converts `rdd` to an `RDD[Row]` that is then passed to `createDataFrame`.

Working with Complex JSON Document Types

The MapR Database OJAI Connector for Apache Spark provides APIs to process JSON documents loaded from MapR Database.

Suppose you want to calculate the number of users located in each city:

Scala

```
import com.mapr.db.spark.sql._

val customerprofilesRDD =
```


last-updated: Apr 11, 2022

```
sc.loadFromMapRDB("/tmp/
user_profiles")
val numberOfCustaccCities =
customerprofilesRDD.map(a =>
(a.`address.city`[String],a))
                                .groupByKe
y()
                                .map(a =>
(a._1, a._2.size))
```

Java

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;
import scala.Tuple2;
import java.util.Collection;

MapRDBJavaRDD<OJAIDocument>
customerprofilesRDD =
mapRDBSparkContext.loadFromMapRDB("/tm
p/user_profiles");
JavaRDD numberOfCustaccCities =
customerprofilesRDD.mapToPair
   (a -> new
Tuple2<>(a.getString("address.city"),
a)).groupByKey()
   .map(a -> new Tuple2<>(a._1,
((Collection<?>)a._2).size()));
```

If you have not provided an explicit cast, then the object is returned as `AnyRef`. To access methods specific to a class, such as `String` or `Integer`, you can cast it to a specific type later in the process.

Now suppose you want to collect all the addresses (`address` is of type `Map`) of all customers:

Scala

```
import com.mapr.db.spark.sql._

val customerprofilesRDD
= sc.loadFromMapRDB("/tmp/
user_profiles")
val customersAddress =
customerprofilesRDD.map(a =>
a.address).collect
```

Java

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;

MapRDBJavaRDD<OJAIDocument>
customerprofilesRDD =
mapRDBSparkContext.loadFromMapRDB("/tm
p/user_profiles");
List<String> customersAddress =
customerprofilesRDD.map(a ->
a.getString("address")).collect();
```

`customersAddress` contains all of the addresses, but is returned as an `AnyRef` object.

The MapR Database OJAI Connector for Apache Spark introduces three new classes to wrap complex JSON types:

Class	Type
`DBMapValue`	`Map[String, AnyRef]`
`DBArrayValue`	`Array[AnyRef]`
`DBBinaryValue`	`ByteBuffer`

These classes are not exposed; however, you can access the underlying elements of `DBArrayValue` and `DBMapValue` by using the same functions as in `Seq` and `Map`. `DBArrayValue` works like a sequence, while `DBMapValue` works like a map.

`DBBinaryValue` is a class wrapper around `ByteBuffer`. `ByteBuffer` is not serializable, so you will get serialization errors if you use the `ByteBuffer` in Spark code. You must ensure that byte buffers are converted to `DBBinaryValue` or serialized byte buffers. The MapR Database OJAI Connector for Apache Spark provides an API to convert `ByteBuffers` to serializable byte buffers.

Accessing Values in a Map

`DBMapValue` is a type of `Map[String, AnyRef]`. Any functions that you can use to access values in the `Map`, you can also use to access values in `DBMapValue`. In the following example, `customeraddress` contains the address of the customers who reside in San Jose. `customeraddress` is an `Array[DBMapValue]`:

Scala

```
val customerAddress = maprd.map(a =>
a.address[Map[String, AnyRef]])
            .filter(a => a!= null &&

a.get("city").contains("San Jose"))
            .collect
```

This example can also be written in Scala using a functional approach as follows:

Scala

```
val customerAddress = maprd.map(a
=> (a.address[Map[String, AnyRef]],
a).join(my_documents)
            .filter(a =>
Option(a).map(a =>

a.get("city").contains("San
Jose")).getOrElse(false)))
            .collect
```

Note: You can push the condition specified in the filter condition to the MapR Database table scan by using the `where` clause.

Accessing the Array JSON Object

This example uses a sequence to access the Array JSON object:

Scala

```
val custInterests = maprd.map(a =>
a.interests[Seq[AnyRef]])
                    .filter(a => a!=
null && a(0) == "sports")
                    .collect
```


last-updated: Apr 11, 2022

ByteBuffer Serialization

The MapR Database OJAI Connector for Apache Spark provides the following API to enable serialization of the `ByteBuffer`:

Scala

```
MapRDBSpark.serializableBinaryValue(byteBuffer)
```

The following example shows an array of byte buffers or binary values that are converted to serialized byte buffers by using `MapRDBSpark.serializableBinaryValue`:

Scala

```
val dstSplits =
arrayOfByteBuffer.map(x =>
MapRDBSpark.serializableBinaryValue(x)
)
```

Saving Data to a MapR Database JSON Table

The MapR Database OJAI Connector for Apache Spark provides an API to save an Apache Spark RDD to a MapR Database JSON table. Starting in the EEP 4.0 release, the connector introduces support for saving Apache Spark DataFrames and DStreams to MapR Database JSON tables.

Saving an Apache Spark RDD to a MapR Database JSON Table

Saving an RDD[OJAIDocument] to MapR Database

The MapR Database OJAI Connector for Apache Spark provides the following API to save an `RDD[OJAIDocument]` to a MapR Database table:

Scala

For saving an RDD, apply the following method on the RDD:

```
def saveToMapRDB(tablename: String,
createTable: Boolean =
false, bulkInsert: Boolean =
false, idFieldPath: String =
DocumentConstants.ID_KEY) : Unit
```

Java

For saving an RDD, apply one of the following methods on a `MapRDBJavaSparkContext` object:

```
def saveToMapRDB[D](javaRDD:
JavaRDD[D], tableName: String,
createTable: Boolean, bulkInsert:
Boolean, idField: String): Unit

def saveRowRDDToMapRDB(javaRDD:
JavaRDD[Row], tableName: String,
createTable: Boolean, bulkInsert:
Boolean, idField: String): Unit

def saveToMapRDB[K, V <: AnyRef]
(javaPairRDD: JavaPairRDD[K, V],
keyClazz: Class[K], valueClazz:
Class[V], tableName: String,
createTable: Boolean, bulkInsert:
Boolean): Unit
```

Note: The only required parameter to the methods is tableName. All the others are optional.

last-updated: Apr 11, 2022

In the following example, `address` and `first_name` data is loaded from the `"/tmp/user_profiles"` table, stored as an RDD (`userprofilesRDD`), and then saved to the `"/tmp/user_firstname_and_address"` table:

Scala

```
import com.mapr.db.spark._

val userprofilesRDD =
sc.loadFromMapRDB("/tmp/
user_profiles")
                      .where("condition
")
                      .select("address"
, "first_name")

userprofilesRDD.saveToMapRDB("/tmp/
user_firstname_and_address")
```

Java

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;
import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;

MapRDBJavaSparkContext
mapRDBSparkContext = new
MapRDBJavaSparkContext(sc);
JavaRDD userprofilesRDD =
mapRDBSparkContext.loadFromMapRDB("/tm
p/user_profiles")
              .where("condition")
              .select("address",
"first_name");
mapRDBSparkContext.saveToMapRDB(userpr
ofilesRDD, "/tmp/
user_firstname_and_address", true,
false, "_id");
```

The MapR Database OJAI Connector for Apache Spark also provides the following API to insert an `RDD[OJAIDocument]` to a MapR Database table:

Note: The `insertToMapRDB` API is available starting in the EEP 4.1.0 release.

Scala

```
import com.mapr.db.spark._

val userprofilesRDD =
sc.loadFromMapRDB("/tmp/
user_profiles")
userprofilesRDD.insertToMapRDB(tablena
me, createTable = true, bulkInsert =
false, idFieldPath = "_id")
```

Java

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;
import
org.apache.spark.sql.SparkSession;
```


last-updated: Apr 11, 2022

```
MapRDBJavaSparkContext
mapRDBSparkContext = new
MapRDBJavaSparkContext(spark.sparkCont
ext());
MapRDBJavaRDD<OJAIDocument>
userprofilesRDD =
mapRDBSparkContext.loadFromMapRDB("/tm
p/user_profiles");
mapRDBSparkContext.insertRowRDDToMapRD
B(userprofilesRDD, tablename);
```

Note: The `insertToMapRDB` API throws an exception if a row with the same ID already exists.

This API supports the following parameters:

Scala

Parameter	Default	Description
`tableName`	Not applicable	The name of the MapR Database table in which you are saving the document.
`createTable`	`false`	Creates the table before saving the documents. Note that if the table already exists and `createTable` is set to true, the API throws an exception.
`idFieldPath`	`_id`	Specifies the key to be used for the document.
`bulkInsert`	`false`	Loads a group of rows simultaneously. `bulkInsert` is similar to a bulk load in MapReduce.

Java

Parameter	Default	Description
`RDD (JavaRDD or JavaPairRDD)`	Not applicable	Specifies the RDD which you are saving to the MapR Database table.
`tableName`	`Not applicable`	Specifies the name of the MapR Database table in which you are saving the document.
`createTable`	`false`	Creates the table before saving the document. Note that if the table already exists and `createTable` is set to true, the API throws an exception.
`idFieldPath`	`_id`	Specifies the key to be used for the document.
`bulkInsert`	`false`	Loads a group of rows simultaneously. `bulkInsert` is similar to a bulk load in MapReduce.

Parameter	Default	Description
`keyClazz (Only for JavaPairRDD)`	Not applicable	Specifies the class type which is the key in the `JavaPairRDD` which you are saving into the MapR Database table.
`valueClazz(Only for JavaPairRDD)`	Not applicable	Specifies the class type which is the value in the `JavaPairRDD` which you are saving into the MapR Database table.

In Java, `saveToMapRDB` method works with `JavaRDD` and `JavaPairRDD`. For saving `JavaRDD[Row]`, use the `saveRowRDDToMapRDB` method.

The following example specifies a key by using the `idFieldPath` parameter and the `bulkInsert` value to save the RDD:

Scala

```
import com.mapr.db.spark._

userprofilesRDD.saveToMapRDB("/tmp/
user_firstname_and_address",

idFieldPath = "user_id",

bulkInsert = false)
```

Java

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;

MapRDBJavaSparkContext
mapRDBSparkContext = new
MapRDBJavaSparkContext(spark.sparkCont
ext());
mapRDBSparkContext.saveToMapRDB(userpr
ofilesRDD, "/tmp/
user_firstname_and_address", false,
false, "user_id");
```

The following example saves an RDD of `Person` objects into the newly created `/tmp/Userinfo` table:

Scala

```
import com.mapr.db.spark._

val sparkConf =
new SparkConf().setAppName("json
app").setMaster("local[*]")
val sc = new SparkContext(sparkConf)
val people =
sc.parallelize(getUsers())
people.saveToMapRDB("/tmp/UserInfo",
createTable= true)
```

Java

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;

SparkConf sparkConf = new
```

```
SparkConf().setAppName("json
app").setMaster("local[*]");
SparkContext sc = new
SparkContext(sparkConf);
JavaRDD rdd =
sc.parallelize(getUsers());
mapRDBSparkContext.saveToMapRDB(rdd,
"/tmp/UserInfo", true);
```

The following example shows the `getUsers` function that allocates the `Person` objects:

Scala

```
def getUsers(): Array[Person] = {
    val users: Array[Person] =

    Array(
       Person("DavUSCalif", "David",
"Jones",
          ODate.parse("1947-11-29"),
          Seq("football", "books",
"movies"),
          Map("city" -> "milpitas",
"street" -> "350 holger way",
"Pin" -> 95035)),

       Person("PetUSUtah", "Peter",
"pan",
          ODate.parse("1974-1-29"),
          Seq("boxing", "music",
"movies"),
          Map("city" -> "salt lake",
"street" -> "351 lake way", "Pin" ->
89898)),

       Person("JamUSAriz", "James",
"junior",
          ODate.parse("1968-10-2"),
          Seq("tennis", "painting",
"music"),
          Map("city" -> "phoenix",
"street" -> "358 pond way", "Pin" ->
67765)),

       Person("JimUSCalif", "Jimmy",
"gill",
          ODate.parse("1976-1-9"),
          Seq("cricket",
"sketching"),
          Map("city" -> "san jose",
"street" -> "305 city way", "Pin" ->
95652)),

       Person("IndUSCalif",
"Indiana", "Jones",
          ODate.parse("1987-5-4"),
          Seq("squash", "comics",
"movies"),
          Map("city" -> "sunnyvale",
"street" -> "35 town way", "Pin" ->
95985)))
```

```
    users
  }
```

Saving a JSON Document to MapR Database

To save a JSON document using the MapR Database OJAI Connector for Apache Spark, you must first convert the JSON document into an OJAI document and then save the RDD, as shown in the following example:

Scala

```
import com.mapr.db.spark._

val documents = sc.parallelize((1 to
10)
            .map(i => s"""{"_id":
"$i", "test": "$i"}"""))
val maprd = documents.map(a =>
MapRDBSpark.newDocument(a))
maprd.saveToMapRDB("/tmp/testData")
```

Java

```
import
com.mapr.db.spark.api.java.MapRDBJavaS
parkContext;
import
org.apache.spark.api.java.JavaSparkCon
text;

JavaRDD<String> documents =
JavaSparkContext.fromSparkContext(sc)
            .parallelize(Arrays.asLis
t(1, 2, 3, 4, 5, 6, 7, 8, 9, 10))
            .map(i -> { return
"{\"id\": \"" + i + "\", \"test\":
\"" + i + "\"}"; });
JavaRDD<OJAIDocument> maprd =
documents.map(MapRDBSpark::newDocument
);
mapRDBSparkContext.saveToMapRDB(maprd,
 "/tmp/testData");
```

An `_id` field is required to save JSON data into a table, so an `_id` field must be present. If you need only to convert the JSON data to an OJAI document (without saving to MapR Database), the `_id` field is not required. If the MapR Database table already contains a record with the same `_id` value, MapR Database replaces the record. Otherwise, it inserts a new record.

Just as you can load a JSON document into a Scala bean class (see Creating an RDD of a Class), you can save the RDD of Scala class objects in a MapR Database JSON table. `saveToMapRDB` can save any bean object as a JSON document by converting it to a JSON document.

Table Splits and saveToMapRDB

If the `createTable` parameter is set to true, `saveToMapRDB` can use the partition information from the RDD's lineage to create the splits for a new table:

Scala

```
sc.loadFromMapRDB("/tmp/
user_profiles").saveToMapRDB("/
userProfiles",
```

```
                createTable = true)
```

Suppose `/tmp/user_profiles` has a table with five splits. `saveToMapRDB` uses this information to create the `/userProfiles` table with the same number and range of splits. You can also supply this information by using `MapRDBSpark.newPartitioner`:

Scala

```
sc.loadFromMapRDB("/tmp/
user_profiles").keyBy(doc =>
doc.get("_id"))
  .repartitionAndSortWithinPartitions(
MapRDBSpark.newPartitioner[String]
      ("/profiles"))
  .saveToMapRDB("/userProfiles",
createTable = true)
```

For more information about partitioning, see Using the Custom Partitioner with the MapR Database OJAI Connector for Apache Spark on page 4066.

Saving an Apache Spark DataFrame to a MapR Database JSON Table

To save an Apache Spark DataFrame to a MapR Database, invoke the `saveToMapRDB` method on the `DataFrame` object (Scala). This returns a `DataFrameWriter` object, from which you can invoke the `saveToMapRDB` method. For Java and Python, invoke the `saveToMapRDB` method on the `MapRDBJavaSession` object or `SparkSession` object, respectively.

If a row with the same ID already exists, the `savetoMapRDB` method updates or overwrites that row. If you want an exception to be thrown in this case, you can use the `insertToMapRDB` method.

Scala

```
import com.mapr.db.spark.sql._

df.write.saveToMapRDB("/tmp/userInfo")
```

For EEP 4.1.0 and later, you can directly invoke the `saveToMapRDB` method on the `DataFrame` object:

```
def saveToMapRDB(tableName: String,
idFieldPath : String = "_id",
createTable: Boolean = false,
bulkInsert:Boolean = false): Unit

import
org.apache.spark.sql.SparkSession
import com.mapr.db.spark.sql._

val df = spark.loadFromMapRDB("/tmp/
user_profiles")
df.saveToMapRDB(tableName,
createTable = true)
```

Java

For saving a DataFrame (Dataset<Row>), apply the following method on a `MapRDBJavaSession` object:

```
def saveToMapRDB[T](df: DataFrame[T],
tableName: String, idFieldPath:
String, createTable: Boolean,
bulkInsert: Boolean): Unit
```

```
import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;

MapRDBJavaSession maprSession = new
MapRDBJavaSession(sparkSession);
Dataset<Row> ds =
maprSession.loadFromMapRDB("/tmp/
user_profiles");

maprSession.saveToMapRDB(ds, "/tmp/
userInfo");
```

Python

For saving a DataFrame, apply the following method on a Dataframe:

```
def saveToMapRDB(dataframe,
table_name, id_field_path =
default_id_field, create_table =
False, bulk_insert = False)

from pyspark.sql import SparkSession

df = spark.loadFromMapRDB("/tmp/
user_profiles")

sparkSession.saveToMapRDB(df,
table_name, create_table=True)
```

Inserting an Apache Spark DataFrame into a MapR Database JSON Table

Starting in the EEP 4.1.0 release, you can use the `insertToMapRDB` API to insert an Apache Spark DataFrame into a MapR Database JSON table in Python. The `insertToMapRDB` API throws an exception if a row with the same ID already exists.

PySpark supports only DataFrame(Dataset<Row>):

Python

```
sparkSession.insertToMapRDB(df,
tableName, idFieldPath, bulkInsert)
```

Using Alternate Write Modes for MapR Database OJAI Connector

You can use alternate write modes supported by MapR Database OJAI Connector for Apache Spark to save an Apache Spark DataFrame to a MapR Database JSON table.

Normally, the Apache Spark `DataFrameWriter` class supports the following write modes:

- Append
- Overwrite
- ErrorIfExists
- Ignore

The MapR Database OJAI Connector for Apache Spark returns an `OperationNotSupported` exception if you attempt to use one of these modes. The following example returns the error:

Scala

```
import org.apache.spark.sql.SaveMode
import com.mapr.db.spark.sql._
```

```
df.write.mode(SaveMode.Append).saveToM
apRDB("/tmp/userInfo")
```

The MapR Database OJAI Connector for Apache Spark provides the following alternative modes:

Insert	Inserts the data into the MapR Database table. Throws a `DBException` if a row with same `_id` value already exists in the table.
Overwrite	Overwrites the data in the table with the current DataFrame data. This operation drops the table and creates a new table with the data.
ErrorIfExists	Returns an exception (`TableExistsException`) if the table already exists. Otherwise, creates the table and inserts the data.
Ignore	Ignores the data in the table if the table already exists. Otherwise, creates the table and inserts the data.
InsertOrReplace	Replaces the row with the row in the DataFrame, if a row with the same `_id` already exists in the table. Otherwise, inserts the new row.

You cannot specify these modes using the Apache Spark `SaveMode` method. Doing so results in the same `OperationNotSupported` exception noted earlier. To use these modes, you must call the `option` method on a `DataFrameWriter` object. The following example sets the `Insert` mode:

Scala

```
df.write.option("Operation",
"Insert").saveToMapRDB("/tmp/
usersInfo")
```

Note: The `UPDATE` mode for MapR Database OJAI Connector is not supported and it results in an `OperationNotSupported` exception.

Saving an Apache Spark DStream to a MapR Database JSON Table

The MapR Database OJAI Connector for Apache Spark enables you to use MapR Database as a sink for Apache Spark DStreams.

Note: Saving of Apache Spark DStream to MapR Database JSON table is currently only supported in Scala.

The following API saves a `DStream[OJAIDocument]` object to a MapR Database table:

Scala

```
def saveToMapRDB(tablename: String,
createTable: Boolean,
          bulkInsert: Boolean,
idFieldPath: String): Unit
```

The parameters are as follows:

Parameter	Default	Description
`tableName`	Not applicable	The name of the MapR Database table to which you are saving the DStream.
`createTable`	`false`	Creates the table before saving the DStream. Note that if the table already exists and `createTable` is set to true, the API throws an exception.
`idFieldPath`	`_id`	Specifies the key to be used for the DStream.
`bulkInsert`	`false`	Loads a group of streams simultaneously. `bulkInsert` is similar to a bulk load in MapReduce.

Note: The only required parameter for this function is tableName. All the others are optional.

The following example creates a DStream object, converts it to a `DStream[OJAIDocument]` object, and then stores it in MapR Database:

Scala

```
val clicksStream: DStream[String] =
createKafkaStream(…)
clicksStream.map(MapRDBSpark.newDocume
nt()).saveToMapRDB("/clicks",
createTable=true)
```

Note: You must use the `map(MapRDBSpark.newDocument())` API to convert the DStream object to a `DStream[OJAIDocument]` object.

If `clicksStream` is a DStream of Strings, it can be saved to MapR Database using the `saveToMapRDB` API:

Scala

```
clicksStream.map(MapRDBSpark.newDocume
nt(_)).saveToMapRDB("/clicks",
createTable = true);
```

Note: To use the `saveToMapRDB` API, you need to transform the DStream object to a `DStream[OJAIDocument]` by using the Apache Spark Map API.

Saving an Apache Spark Dataset to a MapR Database JSON Table

Starting in the EEP 4.1.0 release, the MapR Database OJAI Connector for Apache Spark provides the following API to save a Dataset to a MapR Database table:

Scala

For saving a Dataset, apply the following method on a `Spark` object:

```
def saveToMapRDB(tableName: String,
idFieldPath : String = "_id",
           createTable: Boolean =
false, bulkInsert:Boolean = false):
Unit
```


last-updated: Apr 11, 2022

```
import
org.apache.spark.sql.SparkSession
import com.mapr.db.spark.sql._

val ds = spark.loadFromMapRDB("/tmp/
user_profiles")
ds.saveToMapRDB(tableName,
createTable = true)
```

Java

For saving a Dataset, apply the following method on a `MapRDBJavaSession` object:

```
def saveToMapRDB[T](ds: Dataset[T],
tableName: String, idFieldPath:
String,
            createTable:oolean,
bulkInsert: Boolean): Unit

import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;
import
org.apache.spark.sql.SparkSession;

MapRDBJavaSession maprSession = new
MapRDBJavaSession(spark);
Dataset<Row> ds =
maprSession.loadFromMapRDB("/tmp/
user_profiles");
maprSession.saveToMapRDB(ds, true);
```

The MapR Database OJAI Connector for Apache Spark also provides the following API to insert a Dataset into a MapR Database table:

Scala

```
import com.mapr.db.spark._

ds.insertToMapRDB(tableName,
idFieldPath, bulkInsert)
```

Java

```
import
com.mapr.db.spark.sql.api.java.MapRDBJ
avaSession;

maprSession.insertToMapRDB(ds,
tableName, idFieldPath, bulkInsert)
```

Note: The `insertToMapRDB` API throws an exception if a row with the same ID already exists.

Word Count Example Using MapR Database OJAI Connector

Scala

```


/*
 * Licensed to the Apache Software
Foundation (ASF) under one or more
 * contributor license agreements.
See the NOTICE file distributed with
```

```
 * this work for additional
information regarding copyright
ownership.
 * The ASF licenses this file to You
under the Apache License, Version 2.0
 * (the "License"); you may not use
this file except in compliance with
 * the License.  You may obtain a
copy of the License at
 *
 *    http://www.apache.org/licenses/
LICENSE-2.0
 *
 * Unless required by applicable law
or agreed to in writing, software
 * distributed under the License is
distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS
OF ANY KIND, either express or
implied.
 * See the License for the specific
language governing permissions and
 * limitations under the License.
 */

// scalastyle:off println
package
org.apache.spark.examples.maprdbconnec
tor

import
org.apache.spark.sql.SparkSession

import com.mapr.db.spark.sql._

object MaprDBJsonConnectorWordCount {

  def main(args: Array[String]): Unit
= {

    parseArgs(args)

    val pathToFileWithData = args(0)
    val tableName = args(1)
    val tableNameWithResult = args(2)

    val spark = SparkSession
      .builder()
      .appName("OJAI MaprDB connector
wordcount example")
      .getOrCreate()

    import spark.implicits._
    val wordSequenceDS =
importDataIntoSeq(pathToFileWithData).
toDS()


wordSequenceDS.saveToMapRDB(tableName,
 createTable = true)

    val dfWithDataFromMaprDB =
```

```
spark.loadFromMapRDB(tableName)
      .flatMap(line =>
line.getAs[String](1).split(" "))
      .groupBy("value")
      .count()

    println("Dataset with counted
words:")
    dfWithDataFromMaprDB.show()


dfWithDataFromMaprDB.withColumn("_id",
 $"value")
      .saveToMapRDB(tableNameWithResul
t, createTable = true)
    println("Dataset with counted
words was saved into the MaprDB
table.")

    spark.stop()
  }

  private def parseArgs(args:
Array[String]): Unit = {
    if (args.length != 3) {
      printUsage()
      System.exit(1)
    }
  }

  private def printUsage(): Unit = {
    val usage =
      """OJAI MaprDB connector
wordcount example
        |Usage:
        |1) path to the file with
data (words.txt can be used for the
test);
        |2) name of the MaprDB table
where data from file will be saved;
        |3) name of the MaprDB table
where result will be saved;
        |""".stripMargin

    println(usage)
  }

  private def
importDataIntoSeq(filePath: String):
Seq[Word] = {
    scala.io.Source.fromURL(filePath)
      .getLines
      .map(line => {
        val wordWithId = line.split("
")
        Word(wordWithId(0),
wordWithId.drop(1).mkString(" "))
      }).toSeq
  }

  private case class Word(_id:
String, words: String)
```

```
}
```

Using Serialization with the MapR Database OJAI Connector for Apache Spark
In the context of the MapR Database OJAI Connector for Apache Spark, serialization refers to the methods that read and write objects into bytes. This section describes how to configure your application to use a more efficient serializer.

The Apache Spark cluster framework requires serialization to exchange objects between driver and cluster executors. This type of serialization has nothing to do with the way MapR Database serializes the objects onto the disk.

Because classes used in Spark transformations or actions must be serializable, classes created for the MapR Database OJAI Connector for Apache Spark are serializable.

Spark uses Java serialization by default, but it can alternatively use Kyro Serialization. A new Kyro registrator is introduced so you can avoid using the default Java serialization. Kyro serialization provides better performance than Java serialization.

The following example shows how to set the new Kryo registrator in `sparkconf`:

Scala

```
new sparkconf()
   .set("spark.serializer",
"org.apache.spark.serializer.KryoSeria
lizer")
   .set("spark.kryo.registrator",
"com.mapr.db.spark.OJAIKryoRegistrator
")
```

A JSON document can use both complex and primitive value types. Java can serialize the primitive types, but for complex types (such as `Map`, `Array`, and `Binary`), you must use wrappers to achieve serialization. See Working with Complex JSON Document Types on page 4085 for details about these wrappers.

Time-related data types, such as `ODate`, `OInterval`, `OTime`, and `OTimeStamp`, use Java serialization by default. For efficiency, new serializers and comparators have been created for these data types.

Here are the new serializers and the type which each serializer applies:

Serializer	Type
`ODateSerializer`	`ODate` type
`OTimeSerializer`	`OTime`
`OTimeStampSerializer`	`OTimeStamp`
`OIntervalSerializer` -	`OInterval`
`DBBinaryValueSerializer`	`ByteBuffer`

MapR Database Binary Connector for Apache Spark
This section describes the three main interaction points between Spark and HBase APIs and provides examples for each interaction point.

The interaction points are:

Basic Spark	You can have an HBase Connection at any point in your Spark DAG.
Spark Streaming	You can have an HBase Connection at any point in your Spark Streaming application.

last-updated: Apr 11, 2022

Spark Structured Streaming	Using Spark structured streaming to write data to a MapR Database binary table is currently not supported.
Spark Bulk Load	This option is currently not supported for MapR Database.
SparkSQL/DataFrames	You can write SparkSQL that draws on tables that are represented in HBase.

The following pages provide examples of each of these interaction points.

Configuring the MapR Database Binary Connector for Apache Spark

Use these steps to configure the MapR Database Binary Connector for Apache Spark:

1. Verify that the `mapr-hbase` package is installed. For more information, refer to the HBase release notes.
2. Copy the `HBASE_HOME/conf/hbase-site.xml` file to `SPARK_HOME/conf/`.
3. Specify the `hbase-site.xml` file in the `SPARK_HOME/conf/spark-defaults.conf`:

```
spark.yarn.dist.files   SPARK_HOME/conf/hbase-site.xml
```

MapR Database Binary Connector for Apache Spark Integration with Basic Spark
This page describes integration between Apache Spark and HBase APIs.

This section describes Spark integration with HBase APIs at the lowest and simplest levels. All other interaction points are built upon the concepts described here.

At the root of all integration with Spark and HBase APIs is the `HBaseContext`. The `HBaseContext` takes in HBase configurations and pushes them to the Spark executors. This allows you to have an `HBase Connection` per Spark executor in a static location.

HBaseContext Usage Example

This example shows how `HBaseContext` can be used to do a `foreachPartition` on an RDD in Scala:

```
val sc = new SparkContext("local", "test")
val config = new HbaseConfiguration()
...
val hbaseContext = new HBaseContext(sc, config)

rdd.hbaseForeachPartition(hbaseContext, (it, conn) => {
 val bufferedMutator = conn.getBufferedMutator(TableName.valueOf("/apps/
my_table"))
 it.foreach((putRecord) => {
    val put = new Put(putRecord._1)
    putRecord._2.foreach((putValue) =>
        put.addColumn(putValue._1,
        putValue._2, putValue._3))
    bufferedMutator.mutate(put)
 })
 bufferedMutator.flush()
 bufferedMutator.close()
})
Here is the same example implemented in Java:
JavaSparkContext jsc = new JavaSparkContext(sparkConf);

try {
  List<byte[]> list = new ArrayList<>();
  list.add(Bytes.toBytes("1"));
  ...
  list.add(Bytes.toBytes("5"));
```

```
  JavaRDD<byte[]> rdd = jsc.parallelize(list);
  Configuration conf = HBaseConfiguration.create();
  JavaHBaseContext hbaseContext = new JavaHBaseContext(jsc, conf);

  hbaseContext.foreachPartition(
    rdd,
    new VoidFunction<Tuple2<Iterator<byte[]>, Connection>>() {
   public void call(Tuple2<Iterator<byte[]>, Connection> t) throws
Exception {
    Table table = t._2().getTable(TableName.valueOf(tableName));
    BufferedMutator mutator =
t._2().getBufferedMutator(TableName.valueOf(tableName));
    while (t._1().hasNext()) {
      byte[] b = t._1().next();
      Result r = table.get(new Get(b));
      if (r.getExists()) {
       mutator.mutate(new Put(b));
      }
    }

    mutator.flush();
    mutator.close();
    table.close();
   }
  });
} finally {
  jsc.stop();
}
```

All functionality between Spark and HBase Client is supported both in Scala and in Java, with the exception of SparkSQL, which supports any language that is supported by Spark. This section focuses on Scala examples.

The example here shows how to do a `foreachPartition` with a connection. A number of other Spark base functions are supported out of the box:

`bulkPut`	Enables massively parallel sending of puts to HBase.
`bulkDelete`	Enables massively parallel sending of deletes to HBase.
`bulkGet`	Enables massively parallel sending of gets to HBase to create a new RDD.
`mapPartition`	Enables the Spark Map function with a Connection object to allow full access to HBase.
`hBaseRDD`	Simplifies a distributed scan to create an RDD.

You can see examples of these commands in the source code of the HBase-Spark Module.

MapR Database Binary Connector for Apache Spark Integration with Spark Streaming

Spark Streaming is a micro-batching, stream-processing framework built on top of Spark. HBase APIs and Spark Streaming make great companions. When used alongside Spark Streaming, HBase APIs can serve as:

- A place to grab reference data or profile data on the fly.
- A place to store counts or aggregates in a way that supports the Spark Streaming promise of only once processing.

The MapR Database Binary Connector for Apache Spark integration points with Spark Streaming are similar to its normal Spark integration points. You can use the following commands straight off a Spark Streaming DStream:

`bulkPut`	Enables massively parallel sending of puts to HBase APIs.
`bulkDelete`	Enables massively parallel sending of deletes to HBase APIs.
`bulkGet`	Enables massively parallel sending of gets to HBase APIs to create a new RDD.
`mapPartition`	Enables the Spark Map function with a `Connection` object to allow full access to HBase APIs.
`hBaseRDD`	Simplifies a distributed scan to create an RDD.

`bulkPut` Example with DStreams

The following example shows a bulkPut with DStreams. It is similar to the RDD bulk put.

Note: To invoke the `hbaseBulkPut` method, make sure you import the `HBaseDStreamFunctions` class.

```
import org.apache.hadoop.hbase.spark.HBaseDStreamFunctions._

val sc = new SparkContext("local", "test")
val config = new HBaseConfiguration()

val hbaseContext = new HBaseContext(sc, config)
val ssc = new StreamingContext(sc, Milliseconds(200))

val rdd1 = ...
val rdd2 = ...
val queue = mutable.Queue[
    RDD[(Array[Byte],
    Array[(Array[Byte],
    Array[Byte],
    Array[Byte])])]]()

queue += rdd1
queue += rdd2

val dStream = ssc.queueStream(queue)

dStream.hbaseBulkPut(
  hbaseContext,
  TableName.valueOf(tableName),
  (putRecord) => {
   val put = new Put(putRecord._1)
   putRecord._2.foreach((putValue) =>
    put.addColumn(putValue._1, putValue._2, putValue._3))
   put
  })
```

The `hbaseBulkPut` function has three inputs:

- The `hbaseContext` that carries the configuration broadcast information link to the HBase Connections in the executors.
- The table name of the table you are putting data into.
- A function that will convert a record in the DStream into an HBase `Put` object.

The code snippet above has been extracted from https://github.com/mapr/hbase/blob/1.1.8-mapr-1703/hbase-spark/src/test/scala/org/apache/hadoop/hbase/spark/HBaseDStreamFunctionsSuite.scala.

Bulk Loading Data into HBase with Spark

There are two options for bulk loading data into HBase with Spark:

Note: The bulk load operation is currently not supported for MapR Database.

Basic bulk load functionality	The basic bulk load functionality works for cases where your rows have millions of columns and cases where your columns are not consolidated.
Thin-record bulk load option	The thin-record bulk load option with Spark is designed for tables that have fewer then 10,000 columns per row. The advantage of this option is higher throughput and less overall load on the Spark shuffle operation.

Both implementations work more or less like the MapReduce bulk load process. A partitioner partitions the RowKeys based on region splits, and the RowKeys are sent to the reducers in order, so that HFiles can be written directly from the reduce phase.

In Spark terms, the bulk load is implemented around a `SparkrepartitionAndSortWithinPartitions` followed by a Spark `foreachPartition`. Here is an example of using the basic bulk load functionality:

Bulk Loading Example

Note: Before executing the following example by using Spark Shell, you must create a table in HBase Shell. Run the code in `:paste` mode.

```
import org.apache.hadoop.fs.Path
import org.apache.hadoop.hbase.mapreduce.{LoadIncrementalHFiles,
TableInputFormat}
import org.apache.hadoop.hbase.spark._
import org.apache.hadoop.hbase.spark.HBaseRDDFunctions._
import org.apache.hadoop.hbase.util.Bytes._
import org.apache.hadoop.hbase.{HBaseConfiguration, TableName}
import org.apache.spark.sql.SparkSession
import org.apache.hadoop.hbase.client.{HBaseAdmin, HConnectionManager}
 val tableName = "table1"
    val stagingFolder = "/home/mapr"
    val columnFamily1 = "cf1"
    @transient val conf = HBaseConfiguration.create()
    val hbaseContext = new HBaseContext(sc, conf)
    conf.set(TableInputFormat.INPUT_TABLE, tableName)
    conf.set("hbase.zookeeper.quorum", "node1.cluster.com")
    conf.setInt("hbase.zookeeper.property.clientPort", 5181)
    val rdd = sc.parallelize(Array(
      (toBytes("1"), (toBytes(columnFamily1), toBytes("a"),
toBytes("foo1"))),
      (toBytes("3"), (toBytes(columnFamily1), toBytes("b"),
toBytes("foo2.b")))
    ))
    rdd.hbaseBulkLoad(hbaseContext,
      TableName.valueOf(tableName),
      t => {
        val rowKey = t._1
        val family: Array[Byte] = t._2._1
        val qualifier = t._2._2
        val value: Array[Byte] = t._2._3
        val keyFamilyQualifier= new KeyFamilyQualifier(rowKey, family,
qualifier)
        Seq((keyFamilyQualifier, value)).iterator
      },
      stagingFolder)
    val connection = HConnectionManager.createConnection(conf)
    val table = connection.getTable(TableName.valueOf(tableName))
```

```
    val load = new LoadIncrementalHFiles(conf)
    load.doBulkLoad(
      new Path(stagingFolder),
      connection.getAdmin,
      table,
      connection.getRegionLocator(TableName.valueOf(tableName)))
```

Required Parameters for Bulk Loading with Spark

The `hbaseBulkLoad` function takes three required parameters:

- The name of the table you intend to bulk load to.
- A function that converts a record in the RDD to a tuple key-value pair, with the tuple key being a `KeyFamilyQualifer` object and the value being the cell value. The `KeyFamilyQualifer` object holds the RowKey, Column Family, and Column Qualifier. The shuffle partitions on the RowKey but sorts by all three values.
- The temporary path for the HFile to be written out to. Following the Spark bulk load command, use the HBase `LoadIncrementalHFiles` object to load the newly created HFiles into HBase.

Additional Parameters for Bulk Loading with Spark

You can set the following attributes with additional parameter options on `hbaseBulkLoad`:

- Max file size of the HFiles
- A flag to exclude HFiles from compactions
- Column Family settings for compression, bloomType, blockSize, and dataBlockEncoding

The following example shows the use of additional parameters:

Note: Before executing the following example by using Spark Shell, you must create a table in HBase Shell. Run the code in `:paste` mode.

```
import org.apache.hadoop.fs.Path
import org.apache.hadoop.hbase.client.HConnectionManager
import org.apache.hadoop.hbase.mapreduce.{LoadIncrementalHFiles,
TableInputFormat}
import org.apache.hadoop.hbase.spark.HBaseRDDFunctions._
import org.apache.hadoop.hbase.spark.{FamilyHFileWriteOptions,
HBaseContext, KeyFamilyQualifier}
import org.apache.hadoop.hbase.util.Bytes
import org.apache.hadoop.hbase.{HBaseConfiguration, HConstants, TableName}
import org.apache.spark.sql.SparkSession
  val tableName = "table2"
    val stagingFolder = "/home/mapr"
    val columnFamily1 = "cf1"
    val sc = spark.sparkContext
    @transient val conf = HBaseConfiguration.create()
    conf.set(TableInputFormat.INPUT_TABLE, tableName)
    conf.set("hbase.zookeeper.quorum", "node1.cluster.com")
    conf.setInt("hbase.zookeeper.property.clientPort", 5181)
    val hbaseContext = new HBaseContext(sc, conf)
    val rdd = sc.parallelize(Array(
      (Bytes.toBytes("1"),
        (Bytes.toBytes(columnFamily1),
          Bytes.toBytes("a"), Bytes.toBytes("foo1"))),
      (Bytes.toBytes("3"),
        (Bytes.toBytes(columnFamily1),
```

```
          Bytes.toBytes("b"),
          Bytes.toBytes("foo2.b")))))
    val familyHBaseWriterOptions =
      new java.util.HashMap[Array[Byte], FamilyHFileWriteOptions]
    val f1Options = new FamilyHFileWriteOptions("GZ", "ROW", 128, "PREFIX")
    familyHBaseWriterOptions.put(Bytes.toBytes(columnFamily1), f1Options)
    rdd.hbaseBulkLoad(hbaseContext,
      TableName.valueOf(tableName),
      t => {
        val rowKey = t._1
        val family:Array[Byte] = t._2._1
        val qualifier = t._2._2
        val value = t._2._3
        val keyFamilyQualifier= new KeyFamilyQualifier(rowKey, family,
qualifier)
        Seq((keyFamilyQualifier, value)).iterator
      },
      stagingFolder,
      familyHBaseWriterOptions,
      compactionExclude = false,
      HConstants.DEFAULT_MAX_FILE_SIZE)
    val connection = HConnectionManager.createConnection(conf)
    val table = connection.getTable(TableName.valueOf(tableName))
    val load = new LoadIncrementalHFiles(conf)
    load.doBulkLoad(new Path(stagingFolder),
      connection.getAdmin, table,
connection.getRegionLocator(TableName.valueOf(tableName)))
```

Thin-Record Bulk Load Example

The following example shows how to call the thin-record bulk load implementation:

Note: Before executing the following example by using Spark Shell, you must create a table in HBase Shell. Run the code in `:paste` mode.

```
import org.apache.hadoop.fs.Path
import org.apache.hadoop.hbase.client.HConnectionManager
import org.apache.hadoop.hbase.mapreduce.{LoadIncrementalHFiles,
TableInputFormat}
import org.apache.hadoop.hbase.spark.HBaseRDDFunctions._
import org.apache.hadoop.hbase.spark.{HBaseContext, _}
import org.apache.hadoop.hbase.util.Bytes
import org.apache.hadoop.hbase.{HBaseConfiguration, TableName}
import org.apache.spark.sql.SparkSession
val tableName = "table3"
    val stagingFolder = "/home/mapr"
    val columnFamily1 = "cf1"
    @transient val conf = HBaseConfiguration.create()
    val hbaseContext = new HBaseContext(sc, conf)
    conf.set(TableInputFormat.INPUT_TABLE, tableName)
    conf.set("hbase.zookeeper.quorum", "node1.cluster.com")
    conf.setInt("hbase.zookeeper.property.clientPort", 5181)
    val rdd = sc.parallelize(Array(
      ("1",  List(Bytes.toBytes(columnFamily1), Bytes.toBytes("a"),
Bytes.toBytes("foo1"))),
      ("3", List(Bytes.toBytes(columnFamily1), Bytes.toBytes("b"),
Bytes.toBytes("foo2.b")))))
    rdd.hbaseBulkLoadThinRows(hbaseContext,
      TableName.valueOf(tableName),
      t => {
        val rowKey = t._1
        val familyQualifiersValues = new FamiliesQualifiersValues
```

```
        val q = t._2
        val family:Array[Byte] = q.head
        val qualifier = q(1)
        val value:Array[Byte] = q(2)
        println(s"family: $family")
        println(s"qualifier: $qualifier")
        println(s"value: $value")
        familyQualifiersValues +=(family, qualifier, value)
        (new ByteArrayWrapper(Bytes.toBytes(rowKey)),
familyQualifiersValues)}, stagingFolder, new java.util.HashMap[Array[Byte],
FamilyHFileWriteOptions], compactionExclude = false, 20)
    val connection = HConnectionManager.createConnection(conf)
    val table = connection.getTable(TableName.valueOf(tableName))
    val load = new LoadIncrementalHFiles(conf)
    load.doBulkLoad(
      new Path(stagingFolder),
      connection.getAdmin,
      table,
      connection.getRegionLocator(TableName.valueOf(tableName)))
```

The big difference in using bulk load for thin rows is that the function returns a tuple with the first value being the RowKey and the second value being an object of `FamiliesQualifiersValues`. `FamiliesQualifiersValues` contains all the values for this row for all column families.

SparkSQL and DataFrames

The MapR Database Binary Connector for Apache Spark leverages DataSource API (SPARK-3247) introduced in Spark-1.2.0. The connector bridges the gap between simple HBase KV store and complex relational SQL queries and enables users to perform complex data analytical work on top of MapR Database binary tables using Spark. HBase Dataframe is a standard Spark Dataframe, and is able to interact with any other data sources, such as Hive, Orc, Parquet, JSON, and others. The MapR Database Binary Connector for Apache Spark applies critical techniques such as partition pruning, column pruning, predicate pushdown and data locality.

To use the MapR Database Binary Connector for Apache Spark, you need to define the Catalog for the schema mapping between MapR Database binary tables and Spark tables, prepare the data and populate the MapR Database binary table, then load the HBase DataFrame. After that, users can do integrated query and access records in a MapR Database binary table with SQL query. The following examples illustrate the basic procedure.

Define Catalog Example

The `catalog` defines a mapping between MapR Database binary tables and Spark tables. There are two critical parts of this catalog. One is the rowkey definition. The other is the mapping between the table column in Spark and the column family and column qualifier in MapR Database binary table. The following example defines a schema for a MapR Database binary table with name as `my_table`, row key as `key` and a number of columns (`col1 - col8`). Note that the rowkey also has to be defined in details as a column (`col0`), which has a specific `cf (rowkey)`.

```
def catalog = s"""{
       |"table":{"namespace":"default", "name":"/path_to/my_table"},
       |"rowkey":"key",
       |"columns":{
         |"col0":{"cf":"rowkey", "col":"key", "type":"string"},
         |"col1":{"cf":"cf1", "col":"col1", "type":"boolean"},
         |"col2":{"cf":"cf2", "col":"col2", "type":"double"},
         |"col3":{"cf":"cf3", "col":"col3", "type":"float"},
         |"col4":{"cf":"cf4", "col":"col4", "type":"int"},
         |"col5":{"cf":"cf5", "col":"col5", "type":"bigint"},
         |"col6":{"cf":"cf6", "col":"col6", "type":"smallint"},
         |"col7":{"cf":"cf7", "col":"col7", "type":"string"},
```

```
          |"col8":{"cf":"cf8", "col":"col8", "type":"tinyint"}
        |}
     |}""".stripMargin
```

Save the DataFrame Example

Data prepared by the user is a local Scala collection that has 256 HBaseRecord objects. The `sc.parallelize(data)` function distributes data to form an RDD. `toDF` returns a DataFrame. `writefunction` returns a DataFrameWriter used to write the DataFrame to external storage systems (e.g. MapR Database here). Given a DataFrame with a specified schema catalog, the `save` function creates a MapR Database binary table with five (5) regions and saves the DataFrame inside.

```
case class HBaseRecord(
   col0: String,
   col1: Boolean,
   col2: Double,
   col3: Float,
   col4: Int,
   col5: Long,
   col6: Short,
   col7: String,
   col8: Byte)

object HBaseRecord
{

   def apply(i: Int, t: String): HBaseRecord = {
      val s = s"""row${"%03d".format(i)}"""
      HBaseRecord(s,
      i % 2 == 0,
      i.toDouble,
      i.toFloat,
      i,
      i.toLong,
      i.toShort,
      s"String$i: $t",
      i.toByte)
  }
}

val data = (0 to 255).map { i =>  HBaseRecord(i, "extra")}

sc.parallelize(data).toDF.write.options(Map(
    HBaseTableCatalog.tableCatalog -> catalog,
    HBaseTableCatalog.newTable -> "5")
).format("org.apache.hadoop.hbase.spark")
 .save()
```

Load the DataFrame Example

In the `withCatalog` function, `sqlContext` is a variable of `SQLContext`, which is the entry point for working with structured data (rows and columns) in Spark. `read` returns a DataFrameReader that can be used to read data in a DataFrame. The `option` function adds input options for the underlying data source to the DataFrameReader. The `format` function specifies the input data source format for the DataFrameReader. The `load()` function loads input as a DataFrame. The data frame `df` returned by the `withCatalog` function can be used to access the MapR Database binary table, as shown in the Language Integrated Query and SQL Query examples.

```
def withCatalog(cat: String): DataFrame = {
  sqlContext
```

```
  .read
  .options(Map(HBaseTableCatalog.tableCatalog->cat))
  .format("org.apache.hadoop.hbase.spark")
  .load()
}
val df = withCatalog(catalog)
```

Language Integrated Query Example

DataFrame can do various operations, such as `join`, `sort`, `select`, `filter`, `orderBy`, and so on. In the following example, `df.filter` filters rows using the given SQL expression. `select` selects a set of columns: `col0`, `col1` and `col4`.

```
val s = df.filter(($"col0" <= "row050" && $"col0" > "row040") ||
  $"col0" === "row005" ||
  $"col0" <= "row005")
  .select("col0", "col1", "col4")
s.show
```

SQL Query Example

`registerTempTable` registers `df` DataFrame as a temporary table using the table name `table1`. The lifetime of this temporary table is tied to the SQLContext that was used to create `df`. `sqlContext.sqlfunction` allows the user to execute SQL queries.

```
df.registerTempTable("table1")
sqlContext.sql("select count(col1) from table1").show
```

Query with Different Timestamps

In `HBaseSparkConf`, you can set four parameters related to timestamp:

- `TIMESTAMP`
- `MIN_TIMESTAMP`
- `MAX_TIMESTAMP`
- `MAX_VERSIONS`

With `MIN_TIMESTAMP` and `MAX_TIMESTAMP`, you can query records with different timestamps or time ranges. In the meantime, use a concrete value instead of `tsSpecified` and `oldMs` in the following examples. The first example shows how to load `df` DataFrame with different timestamps. `tsSpecified` is specified by the user. `HBaseTableCatalog` defines the HBase and Relation relation schema. `writeCatalog` defines the catalog for the schema mapping.

```
val df = sqlContext.read
      .options(Map(
        HBaseTableCatalog.tableCatalog -> writeCatalog,
        HBaseSparkConf.TIMESTAMP -> tsSpecified.toString)
    ).format("org.apache.hadoop.hbase.spark")
      .load()
```

The following example shows how to load `df` DataFrame with different time ranges. `oldMs` is specified by the user.

```
val df = sqlContext.read
      .options(Map(
```


last-updated: Apr 11, 2022

```
        HBaseTableCatalog.tableCatalog -> writeCatalog,
        HBaseSparkConf.MIN_TIMESTAMP -> "0",
        HBaseSparkConf.MAX_TIMESTAMP -> oldMs.toString)
    ).format("org.apache.hadoop.hbase.spark")
      .load()
After loading df DataFrame, users can query data.
df.registerTempTable("table")
sqlContext.sql("select count(col1) from table").show
```

Native Avro Support

The MapR Database Binary Connector for Apache Spark supports different data formats such as Avro, JSON, and others. The following use case shows how Spark supports Avro. You can persist the Avro record into MapR Database binary tables directly. Internally, the Avro schema is converted to a native Spark Catalyst data type automatically. Note that both key-value parts in a MapR Database binary table can be defined in Avro format.

1. Define the `catalog` for schema mapping. `catalog` is a schema for a MapR Database binary table named `Avrotable`, a row key as `key`, and one column `col1`. The rowkey also has to be defined in details as a column (`col0`), which has a specific `cf` (`rowkey`).

   ```
   def catalog = s"""{
                        |"table":{"namespace":"default", "name":"/path_to/
   avro_table"},
                        |"rowkey":"key",
                        |"columns":{
                        |"col0":{"cf":"rowkey", "col":"key",
   "type":"string"},
                        |"col1":{"cf":"cf1", "col":"col1", "type":"binary"}
                        |}
                        |}""".stripMargin
   ```

2. Prepare the data. `schemaString` is defined first. Then it is parsed to get `avroSchema`. `avroSchema` is used to generate `AvroHBaseRecord`. `data prepared by users` is a local Scala collection that has 256 AvroHBaseRecord objects.

```
object AvroHBaseRecord {
   val schemaString =
     s"""{"namespace": "example.avro",
         |   "type": "record",      "name": "User",
         |    "fields": [
         |        {"name": "name", "type": "string"},
         |        {"name": "favorite_number",  "type": ["int", "null"]},
         |        {"name": "favorite_color", "type": ["string", "null"]},
         |        {"name": "favorite_array", "type": {"type": "array",
"items": "string"}},
         |        {"name": "favorite_map", "type": {"type": "map",
"values": "int"}}
         |      ]    }""".stripMargin

   val avroSchema: Schema = {
     val p = new Schema.Parser
     p.parse(schemaString)
   }

   def apply(i: Int): AvroHBaseRecord = {
     val user = new GenericData.Record(avroSchema);
     user.put("name", s"name${"%03d".format(i)}")
     user.put("favorite_number", i)
     user.put("favorite_color", s"color${"%03d".format(i)}")
     val favoriteArray = new GenericData.Array[String](
    2,
    avroSchema.getField("favorite_array").schema())
     favoriteArray.add(s"number${i}")
     favoriteArray.add(s"number${i+1}")
     user.put("favorite_array", favoriteArray)
     import collection.JavaConverters._
     val favoriteMap = Map[String, Int](("key1" -> i), ("key2" ->
(i+1))).asJava
     user.put("favorite_map", favoriteMap)
     val avroByte = AvroSedes.serialize(user, avroSchema)
     AvroHBaseRecord(s"name${"%03d".format(i)}", avroByte)
   }
 }

 val data = (0 to 255).map { i =>
    AvroHBaseRecord(i)
 }
```

3. Save the DataFrame. Given a data frame with the specified schema catalog, the following example creates a MapR Database binary table with five (5) regions and saves the data frame inside.

```
sc.parallelize(data).toDF.write.options(
     Map(
    HBaseTableCatalog.tableCatalog -> catalog,
    HBaseTableCatalog.newTable -> "5")
    ).format("org.apache.spark.sql.execution.datasources.hbase")
     .save()
```


last-updated: Apr 11, 2022

4. Load the DataFrame. In the `withCatalog` function, `read` returns a DataFrameReader that can be used to read data in as a DataFrame. The `option` function adds input options for the underlying data source to the DataFrameReader. There are two options: one is to set `avroSchema` as `AvroHBaseRecord.schemaString`. The other option is to set `HBaseTableCatalog.tableCatalog` as `avroCatalog`. The `load()` function loads input in as a DataFrame. The date frame df returned by the `withCatalog` function can be used to access the MapR Database binary table.

```
def avroCatalog = s"""{
            |"table":{"namespace":"default", "name":"avrotable"},
            |"rowkey":"key",
            |"columns":{
              |"col0":{"cf":"rowkey", "col":"key", "type":"string"},
              |"col1":{"cf":"cf1", "col":"col1", "avro":"avroSchema"}
            |}
          |}""".stripMargin

 def withCatalog(cat: String): DataFrame = {
     sqlContext
         .read
         .options(Map(
        "avroSchema" -> AvroHBaseRecord.schemaString,
        HBaseTableCatalog.tableCatalog -> avroCatalog)
         ).format("org.apache.spark.sql.execution.datasources.hbase")
         .load()
 }
 val df = withCatalog(catalog)
```

5. Query data using SQL. After loading `df DataFrame`, you can query data. `registerTempTable` registers `df DataFrame` as a temporary table using the table name `avrotable`. The `sqlContext.sql` function allows you to execute SQL queries.

```
df.registerTempTable("avrotable")
val c = sqlContext.sql("select count(1) from avrotable"
```

Integrating Spark

This section includes the following topics about configuring Spark to work with other ecosystem components.

Integrate Spark-SQL (Spark 2.3.1 and later) with Avro

You integrate Spark-SQL with Avro when you want to read and write Avro data. This information is for Spark 2.3.0 or later users.

Note: For Spark 2.2.1 and 2.3.1 versions, use the 4.0.0 avro version of `com.databricks:spark-avro_2.11`.

Use the following steps to perform the integration. Previous versions of Spark do not require these steps.

1. Download the Avro 1.7.7 JAR file to the Spark `jars` (`opt/mapr/spark/spark-<version>/jars`) directory.

   You can download the file from the maven repository: http://mvnrepository.com/artifact/org.apache.avro/avro/1.7.7

last-updated: Apr 11, 2022

2. Add the following properties in `spark-defaults.conf`:

```
spark.driver.extraClassPath /opt/mapr/spark/spark-<spark_version>/jars/
avro-1.7.7.jar
spark.executor.extraClassPath /opt/mapr/spark/spark-<spark_version>/jars/
avro-1.7.7.jar
```

Integrate Spark-SQL (Spark 1.6.1) with Avro

You integrate Spark-SQL with Avro when you want to read and write Avro data. This information is for Spark 1.6.1 or earlier users.

Use the following steps to perform the integration. Previous versions of Spark do not require these steps.

1. Download the Avro 1.7.7 JAR file to the Spark lib (`opt/mapr/spark/spark-<version>/lib`) directory.

   You can download the file from the maven repository: http://mvnrepository.com/artifact/org.apache.avro/avro/1.7.7

2. Use one of the following methods to add the Avro 1.7.7 JAR to the classpath:

   - Prepend the Avro 1.7.7 JAR file to the spark.executor.extraClassPath and spark.driver.extraClassPath in the spark-defaults.conf (`/opt/mapr/spark/spark-<version>/conf/spark-defaults.conf`) file:

     ```
     spark.executor.extraClassPath  /opt/mapr/spark/
     spark-<spark_version>/lib/avro-1.7.7.jar:<rest_of_path>
     spark.driver.extraClassPath  /opt/mapr/spark/spark-<spark_version>/lib/
     avro-1.7.7.jar:<rest_of_path>
     ```

   - Specify the Avro 1.7.7 JAR files with command line arguments on the spark shell:

     ```
     /opt/mapr/spark/spark-<version>/bin/spark-shell \
     --packages com.databricks:spark-avro_2.10:2.0.1 \
     --driver-class-path /opt/mapr/spark/spark-<version>/lib/avro-1.7.7.jar
     \
     --conf spark.executor.extraClassPath=/opt/mapr/spark/
     spark-<version>/lib/avro-1.7.7.jar --master <master-url>
     ```

Integrate Spark with HBase

Integrate Spark with HBase or MapR Database when you want to run Spark jobs on HBase or MapR Database tables.

If you installed Spark with the MapR Installer, these steps are not required.

1. Configure the HBase version in the `/opt/mapr/spark/spark-<version>/mapr-util/compatibility.version` file:

   ```
   hbase_versions=<version>
   ```

   The HBase version depends on the current EEP and MapR version that you are running.

2. If you want to create HBase tables with Spark, add the following property to `hbase-site.xml`:

   ```
   <property>
   hbase.table.sanity.checks</name>
   <value>false</value>
   </property>
   ```

3. On each Spark node, copy the `hbase-site.xml` to the `{SPARK_HOME}/conf/` directory.

   Tip: Starting in the EEP 7.0.0 release, you do not have to complete step 3. Running `configure.sh` copies the `hbase-site.xml` file to the Spark directory automatically.

4. Specify the `hbase-site.xml` file in the `SPARK_HOME/conf/spark-defaults.conf` file:

```
spark.yarn.dist.files SPARK_HOME/conf/hbase-site.xml
```

5. To verify the integration, complete the following steps:
   a) Create an HBase or MapR Database table:

```
create '<table_name>' , '<column_family>'
```

   b) Run the following command as the `mapr` user or as a user that `mapr` impersonates:

```
/opt/mapr/spark/spark-<spark_version>/bin/spark-submit --master
<master> [--deploy-mode <deploy-mode>]  --class
org.apache.hadoop.hbase.spark.example.rdd.HBaseBulkPutExample /opt/
mapr/hbase/hbase-<hbase_versrion>/lib/
hbase-spark-<hbase_version>-mapr.jar  <table_name>  <column_family>
```

   The master URL for the cluster is either `spark://<host>:7077`, yarn, or local (without deploy-mode). The deploy-mode is either `client` or `cluster`.

   c) Check the data in the HBase or MapR-DB table:

```
hbase(main):001:0> scan '<table_name>'
```

Integrate Spark-SQL (Spark 2.0.1 and later) with Hive
You integrate Spark-SQL with Hive when you want to run Spark-SQL queries on Hive tables. This information is for Spark 2.0.1 or later users.

For information about Spark-SQL and Hive support, see Spark Feature Support.

Note: If you installed Spark with the MapR Installer, the following steps are not required.

1. Copy the `hive-site.xml` file into the `SPARK_HOME/conf` directory so that Spark and Spark-SQL recognize the Hive Metastore configuration. Do not create a symbolic link instead of copying the file. You may need to edit the file with settings that are specific to the Spark Thrift server.

2. Add `644` permission to the `hive-site.xml` using the following command:

```
sudo chmod 644 /opt/mapr/spark/spark-<sparkVersion>/conf/hive-site.xml
```

3. If Hive is configured on Tez (not on MR), you must remove the Tez property from the Spark conf directory hive-site.xml. Delete this entry:

```
<property>
  <name>hive.execution.engine</name>
  <value>tez</value>
</property>
```


last-updated: Apr 11, 2022

4. If Hive is configured on PAM, set "`hive.metastore.sasl.enabled = true`" in the `hive-site.xml` located in the Spark conf directory.

5. Add the following additional properties to the `/opt/mapr/spark/spark-<version>/conf/spark-defaults.conf` file:

Property	Configuration Requirements
spark.yarn.dist.files	For Spark on YARN, specify the location of the `hive-site.xml` file: `/opt/mapr/spark/spark-<spark-version>/conf/hive-site.xml`
spark.sql.hive.metastore.version	Specify the Hive version that you are using. **Note:** If you are using Hive Metastore 2.1, set the version to 1.2.1.

6. Depending on whether you plan to run with impersonation, perform one of the following:
   - Configure user impersonation. See Hive User Impersonation for the steps to configure impersonation in the Spark Thrift server.
   - Set `hive.server2.enable.doAs` to `false` in the `hive-site.xml` file.

7. To verify the integration, run the following command as the mapr user or as a user that mapr impersonates:

```
<spark-home>/bin/run-example --master <master> [--deploy-mode
<deploy-mode>] sql.hive.SparkHiveExample
```

   The master URL for the cluster is either spark://<host>:7077 or yarn. The deploy-mode is either client or cluster.

Note: The default port for both HiveServer 2 and the Spark Thrift server is 10000. Therefore, before you start the Spark Thrift server on a node where HiveServer 2 is running, verify that there is no port conflict.

Note: If you plan to access Hive tables that store data in MapR Database, you need to copy the Hive HBase handler jar into the Spark jars directory. For example:

```
cp /opt/mapr/hive/hive-2.1/lib/hive-hbase-handler-2.1.1-mapr-1707.jar /opt/
mapr/spark/spark-2.1.0/jars/
```

Integrate Spark-SQL (Spark 1.6.1) with Hive

You integrate Spark-SQL with Hive when you want to run Spark-SQL queries on Hive tables. This information is for Spark 1.6.1 or earlier users.

For information about Spark-SQL and Hive support, see Spark Feature Support.

Note: If you installed Spark with the MapR Installer, the following steps are not required.

1. Copy the `hive-site.xml` file into the `SPARK_HOME/conf` directory so that Spark and Spark-SQL recognize the Hive Metastore configuration.

2. Configure the Hive version in the `/opt/mapr/spark/spark-<version>/mapr-util/compatibility.version` file:

```
hive_versions=<version>
```

3. Add the following additional properties to the `/opt/mapr/spark/spark-<version>/conf/spark-defaults.conf` file:

Property	Configuration Requirements

spark.yarn.dist.files	Option 1: For Spark on YARN, specify the location of the hive-site.xml and the datanucleus JARs:
	`/opt/mapr/hive/hive-<hive-version>/conf/hive-site.xml,/opt/mapr/hive/<version>/lib/datanucleus-api-jdo-<version>.jar,/opt/mapr/hive/<version>/lib/datanucleus-core-<version>.jar,/opt/mapr/hive/hive-1.2/lib/datanucleus-rdbms-<version>.jar`
	Option 2: For Spark on YARN, store hive-site.xml and datanucleus JARs on MapR File System, and use the following syntax:
	`maprfs:///<path to hive-site.xml>,maprfs:///<path to datanucleus jar files>`

spark.sql.hive.metastore.version	Specify the Hive version that you are using. For example, for Hive 1.2.x, set the value to 1.2.0.

spark.sql.hive.metastore.jars	Specify the classpath to JARs for Hive, Hive dependencies, and Hadoop. These files must be available on the node from which you submit Spark jobs:
	`/opt/mapr/hadoop/hadoop-<hadoop-version>/etc/hadoop:/opt/mapr/hadoop/hadoop-<hadoop-version>/share/hadoop/common/lib/*:<rest of hadoop classpath>:/opt/mapr/hive/hive-<version>/lib/accumulo-core-<version>.jar:/opt/mapr/hive/hive-<version>/lib/hive-contrib-<version>.jar:<rest of hive classpath>`
	For example, when you run with Hive 1.2, you can set the following classpath:
	`/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop:/opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/common/lib/*:/opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/common/*:/opt/mapr/hadoop/./hadoop-2.7.0/share/hadoop/mapreduce/*:/opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/yarn/*:/opt/mapr/hive/hive-1.2/lib/accumulo-core-1.6.0.jar:/opt/mapr/hive/hive-1.2/lib/hive-contrib-1.2.0-mapr-1607.jar:/opt/mapr/hive/hive-1.2/lib/*`
	For more information, see the Apache Spark documentation.

last-updated: Apr 11, 2022

4. To verify the integration, run the following command as the mapr user or as a user that mapr impersonates:

```
MASTER=<master-url> <spark-home>/bin/run-example sql.hive.HiveFromSpark
```

   The master URL for the cluster is either spark://<host>:7077, yarn-client, or yarn-cluster.

Note: The default port for both HiveServer 2 and the Spark Thrift server is 10000. Therefore, before you start the Spark Thrift server on a node where HiveServer 2 is running, verify that there is no port conflict.

Integrate Spark with MapR Event Store For Apache Kafka
Integrate Spark with MapR Streams to enable Spark to query MapR Event Store For Apache Kafka for new messages at a given interval, process any new messages that are available, and also publish messages into MapR Event Store For Apache Kafka.

You can use Spark to access MapR Event Store For Apache Kafka through Spark's receiver-less, direct approach.

For more information, see the Apache Spark documentation.

Note: Before you integrate Spark with MapR Event Store For Apache Kafka, verify that the Streams Client is install on all Spark nodes. For more information, see the Installing MapR and MapR Ecosystem Components on page 128.

Configure Spark 2.2.1 and later to Consume MapR Event Store For Apache Kafka Messages
Using the Kafka 0.9 API, you can configure a Spark application to query MapR Event Store For Apache Kafka for new messages at a given interval. This information is for Spark 2.2.1 and later users.

1. Install the MapR core Kafka package, if you have not already done so.
2. Copy the Kafka client jar into the Spark jars directory as shown below:

```
cp /opt/mapr/lib/kafka-clients-<version>.jar SPARK_HOME/jars
```

3. Add the following dependency:

```
groupId = org.apache.spark
artifactId = spark-streaming-kafka-0-9_2.11
version = <spark_version>-mapr-<mapr_eco_version>
```

   Note: If you would like to use Streaming Producer Examples, you must add the appropriate Spark streaming Kafka producer jar from the MapR Maven repository to the Spark classpath (`/opt/mapr/spark/spark-<spark_version>/jars/`.

4. Consider the following when you write the Spark application:
   a) Verify that it meets the following requirements:
      - Imports and use classes from `org.apache.spark.streaming.kafka09`. The following code snippet imports three classes.

```
import org.apache.spark.streaming.kafka09.{ConsumerStrategies,
KafkaUtils, LocationStrategies}
```

- Defines key and value deserializers in the kafkaParams map.

```
val kafkaParams = Map[String, String](
   ConsumerConfig.GROUP_ID_CONFIG -> groupId,
   ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG ->
      "org.apache.kafka.common.serialization.StringDeserializer",
   ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG ->
      "org.apache.kafka.common.serialization.StringDeserializer",
   ConsumerConfig.AUTO_OFFSET_RESET_CONFIG -> offsetReset)
```

- Does not configure a broker address or Zookeeper as these are not required for MapR Event Store For Apache Kafka.

b) Optionally, define a value for `spark.streaming.kafka.consumer.poll.ms` in the Spark configuration.

Note: You can configure the poll timeout using Spark option `spark.streaming.kafka.consumer.poll.ms`. If you do not configure `spark.streaming.kafka.consumer.poll.ms`, the `spark.network.timeout` property is used. If `spark.network.timeout` is empty, the default is 120 seconds.

```
val sparkConf = new SparkConf()
      .setAppName("v09DirectKafkaWordCount")
      .set("spark.streaming.kafka.consumer.poll.ms", pollTimeout)
```

Example:

https://github.com/mapr/spark/blob/2.2.1-mapr-1803/examples/src/main/scala/org/apache/spark/examples/streaming/V09DirectKafkaWordCount.scala is a sample consumer program.

The `KafkaUtils.createDirectStream` method creates an input stream to read MapR Event Store For Apache Kafka messages. The `ConsumerStrategies.Subscribe` method creates the `consumerStrategy` that will limit the set of topics the stream subscribes to. This is derived from the `topics` parameter passed into the program. Using `LocationStategies.PreferConsistent` will distribute partitions evenly across available executors.

```
val consumerStrategy = ConsumerStrategies.Subscribe[String, String]
(topicsSet, kafkaParams)
val messages = KafkaUtils.createDirectStream[String, String](
   ssc, LocationStrategies.PreferConsistent, consumerStrategy)
```

Configure Spark to Produce MapR Event Store For Apache Kafka Messages
Using the Kafka 0.9 API, you can configure a Spark application to produce MapR Streams messages.

1. Add the following dependency:

```
groupId = org.apache.spark
artifactId = spark-streaming-kafka-producer_2.11
version = <spark_version>-mapr-<mapr_eco_version>
```

Note: If you would like to use Streaming Producer Examples, you must add the appropriate Spark streaming Kafka producer jar from the MapR Maven repository to the Spark classpath (`/opt/mapr/spark/spark-<spark_version>/jars/`.

last-updated: Apr 11, 2022

2. When you write the Spark program, import and use classes from `org.apache.spark.streaming.kafka.producer._` and `org.apache.spark.streaming.dstream`.

   The import of `org.apache.spark.streaming.stream.DStream` adds the following method from `DStream`:

   ```
   sendToKafka(topic: String, conf: ProducerConf)
   ```

   In the code below, calling `sendToKafka` will send `numMessages` messages to the set of topics specified by the `topics` parameter.

   ```
   val producerConf = new ProducerConf(bootstrapServers =
   kafkaBrokers.split(",").toList)
      .withKeySerializer("org.apache.kafka.common.serialization.ByteArraySer
   ializer")
      .withValueSerializer("org.apache.kafka.common.serialization.StringSeri
   alizer")

   val items = (0 until numMessages.toInt).map(i => Item(i, i).toString)
   val defaultRDD: RDD[String] = ssc.sparkContext.parallelize(items)
   val dStream: DStream[String] = new ConstantInputDStream[String](ssc,
   defaultRDD)

   dStream.foreachRDD(_.sendToKafka(topics, producerConf))
   dStream.count().print()
   ```

   The `org.apache.kafka.common.serialization.ByteArraySerializer` and `org.apache.kafka.common.serialization.StringSerializer` properties are used by default, and in case you do not want to use another serializer, `withKeySerializer` and `withValueSerializer` methods are not necessary.

Source code for a sample producer program can be found at https://github.com/mapr/spark/blob/2.2.1-mapr-1803/examples/src/main/scala/org/apache/spark/examples/streaming/KafkaProducerExample.scala

Integrate Spark with R

You integrate Spark with R when you want to run R programs as Spark jobs.

1. On each node that will submit Spark jobs, install R 3.2.2 or greater:

   - On Ubuntu:

     ```
     apt-get install r-base-dev
     ```

   - On CentOS/RedHat:

     ```
     yum install R
     ```

   For more information about installing R, see the R documentation.

2. To verify the integration, run the following commands as the mapr user or as a user that mapr impersonates:

   a) Start Spark R:

- On Spark 2.0.1, 2.1.0, and later:

```
/opt/mapr/spark/spark-<version>/bin/sparkR --master <master>
[--deploy-mode <deploy-mode>]
```

- On Spark 1.6.1:

```
/opt/mapr/spark/spark-<version>/bin/sparkR --master <master-url>
```

b) Run the following command to create a DataFrame using sample data:
On Spark 1.6.1:

```
people <- read.df(sqlContext, "file:///opt/mapr/spark/spark-<version>/
examples/src/main/resources/people.json", "json")
```

On Spark 2.0.1, 2.1.0, and later:

```
people <- read.df(spark, "file:///opt/mapr/spark/spark-<version>/
examples/src/main/resources/people.json", "json")
```

c) Run the following command to display the data from the DataFrame that you just created:

```
head(people)
```

Integrate Spark with Kafka
From EEP-5.0.0, Spark can be integrated with Kafka-1.0. You can configure a Spark application to produce Kafka messages.

1. Add the following dependency:

```
groupId = org.apache.spark
artifactId = spark-streaming-kafka-producer_2.11
version = <spark_version>-mapr-<mapr_eco_version>
```

2. When you write the Spark program, import and use classes from:

```
org.apache.spark.streaming.kafka.producer._
org.apache.spark.streaming.dstream.
```

The import of `org.apache.spark.streaming.stream.DStream` adds the following method from DStream:

```
sendToKafka(topic: String, conf: ProducerConf)
```

3. In the code below, calling `sendToKafka` will send `numMessages` messages to the set of topics specified by the topics parameter:

```
val producerConf = new ProducerConf(
bootstrapServers = kafkaBrokers.split(",").toList)

val items = (0 until numMessages.toInt).map(i => Item(i, i).toString)
val defaultRDD: RDD[String] = ssc.sparkContext.parallelize(items)
val dStream: DStream[String] = new ConstantInputDStream[String](ssc,
defaultRDD)

dStream.foreachRDD(_.sendToKafka(topics, producerConf))
dStream.count().print()
```

Source code for a sample producer program can be found at https://github.com/mapr/spark/blob/2.2.1-mapr-1803/examples/src/main/scala/org/apache/spark/examples/streaming/KafkaProducerExample.scala

Spark JDBC and ODBC Drivers

MapR provides JDBC and ODBC drivers so you can write SQL queries that access the Apache Spark data-processing engine. This section describes how to download the drivers, and install and configure them.

You can download the Spark JDBC driver from https://package.mapr.hpe.com/tools/MapR-JDBC/MapR_Spark/ and the Spark ODBC driver from https://package.mapr.hpe.com/tools/MapR-ODBC/MapR_Spark/.

After downloading the driver, refer to the documentation at Spark JDBC Driver to install and configure the JDBC driver and Spark ODBC Driver for the ODBC driver. A copy of the documentation also is available in each download package. The following table describes the driver versions available for various EEP releases:

	EEP version	Driver version	Driver link	Documentation link
JDBC version	EEP 6.0.0+	2.6.3	Spark JDBC Driver for version 2.6.3	Spark JDBC Documentation for version 2.6.3
	EEP 3.0.1+	1.1.8	Spark JDBC Driver for version 1.1.8	Spark JDBC Documentation for version 1.1.8
ODBC version	EEP 6.0.0+	2.6.1	Spark ODBC Driver for version 2.6.1	Spark ODBC Documentation for version 2.6.1
	EEP 2.0.2+	1.2.5	Spark ODBC Driver for version 1.2.5	Spark ODBC Documentation for version 1.2.5

Note: When connecting to the Spark Thrift Server using `beeline` and the JDBC driver, you might encounter the following error:

```
"Unsupported transaction isolation level: 4"
```

To avoid this error, pass the `isolation` parameter to `beeline` as follows:

```
bin/beeline --isolation=default
```

Spark API Changes
This topic describes the public API changes that occurred for specific Spark versions.

Spark 2.3.1 API Changes
EEP 6.0.0 supports Spark version 2.3.1.

For more information about Spark 2.3.1, see the Spark 2.3.1-1808 (EEP 6.0.0) Release Notes on page 6385 and the Spark 2.3.1 API Documentation.

For a complete list of all new and changed APIs, refer to the open source documentation.

Spark 2.1.0 API Changes
This topic describes the public API changes that occurred between Apache Spark 2.0.1 and Spark 2.1.0.

For more information about Spark 2.1.0, see the Spark Release Notes and the Spark 2.1.0 API Documentation.

New API

- The `DataType` API is now mostly stable. Please see `InterfaceStability` annotations for the classes you need.
- Add the `from_json` and `to_json` functions to SQL.
- `StructType` now accepts Python Dictionaries.
- New ML algorithms have been added for Spark R.
- `SparkContext.addFile` is now supported for SparkR.
- SparkR now supports multinomial logistics regression.
- MLlib supports MLR in DataFrames, LSH.
- MLlib model loading is now backward-compatible with Spark 1.6.

Changed API

- Parquet-MR is bumped to 1.8.1.
- `spark.sql.warehouse.dir` now needs to be set before SparkSession creation and is shared between multiple SparkSessions.
- Values generated by non-deterministic functions will not change after coalesce or union.
- The default `Locale` for `DateFormat/NumberFormat` is now `Locale.US`.
- Function `SIZE` returns -1 when its input parameter is null.

Spark 2.0.1-1703 API Changes
This release does not introduce any changes to public Spark API apart from Structured Streaming, which is not supported by the MapR platform.

Spark 2.0.1-1611 API Changes
This topic describes the public API changes that occurred between Apache Spark 1.6.1 and Spark 2.0.1.

Removed Methods

The following items have been removed from Apache Spark 2.0.1:

- Bagel (the Spark implementation of Google Pregel)

- Most of the deprecated methods from Spark 1.x, including:

Category	Subcategory	Instead of this removed API...	Use...
GraphX		`mapReduceTriplets`	`aggregateMessages`
		`runSVDPlusPlus`	`run`
		`GraphKryoRegistrator`	
SQL	DataType	`DataType.fromCaseClassString`	`DataType.fromJson`
	DecimalType	`DecimalType()`	`DecimalType(precision, scale)` `precision explicitly`
		`DecimalType(Option[PrecisionInfo])`	`DecimalType(precision scale)`
		`PrecisionInfo`	`DecimalType(precision, scale)`
		`precisionInfo`	`precision` and `scale`
		`Unlimited`	(No longer supported)
	Column	`Column.in()`	`isin()`
	DataFrame	`toSchemaRDD`	`toDF`
		`createJDBCTable`	`write.jdbc()`
		`saveAsParquetFile`	`write.parquet()`
		`saveAsTable`	`write.saveAsTable()`
		`save`	`write.save()`
		`insertInto`	`write.mode(SaveMode.Append).s`
	DataframeReader	`DataFrameReader.load(path)`	`option("path", path).load()`
	Functions	`cumeDist`	`cume_dist`
		`denseRank`	`dense_rank`
		`percentRank`	`percent_rank`
		`rowNumber`	`row_number`
		`inputFileName`	`input_file_name`
		`isNaN`	`isnan`
		`sparkPartitionId`	`spark_partition_id`
		`callUDF`	`udf`
Core	SparkContext	Constructors no longer take `prefferedNodeLocationData` param	
		`tachyonFolderName`	`externalBlockStoreFolderName`
		`initLocalProperties`, `clearFiles`, `clearJars`	(No longer needed)
		`runJob` method no longer takes `allowLocal` param	
		`defaultMinSplits`	`defaultMinPartitions`
		`[Double, Int, Long, Float]AccumulatorParam`	implicit objects from `AccumulatorPar`
		`rddTo[Pair, Async, Sequence, Ordered]RDDFunctions`	implicit functions from RDD

last-updated: Apr 11, 2022

Category	Subcategory	Instead of this removed API...	Use...
		`[double, numeric]RDDToDoubleRDDFunctions`	implicit functions from RDD
		`intToIntWritable, longToLongWritable, floatToFloatWritable, doubleToDoubleWritable, boolToBoolWritable, bytesToBytesWritable, stringToText`	implicit functions from `WriteableFact`
		`[int, long, double, float, boolean, bytes, string, writable]WritableConverter`	implicit functions from `WritableConve`
	TaskContext	`runningLocally`	`isRunningLocally`
		`addOnCompleteCallback`	`addTaskCompletionListener`
		`attemptId`	`attemptNumber`
	JavaRDDLike	`splits`	`partitions`
		`toArray`	`collect`
	JavaSparkContext	`defaultMinSplits`	`defaultMinPartitions`
		`clearJars, clearFiles`	(No longer needed)
	PairRDDFunctions	`PairRDDFunctions.reduceByKeyToDriver`	`reduceByKeyLocally`
	RDD	`mapPartitionsWithContext`	`Taskcontext.get`
		`mapPartitionsWithSplit`	`mapPartitionsWithIndex`
		`mapWith`	`mapPartitionsWithIndex`
		`flatMapWith`	`mapPartitionsWithIndex` and `flat`
		`foreachWith`	`mapPartitionsWithIndex` and `fore`
		`filterWith`	`mapPartitionsWithIndex` and `filt`
		`toArray`	`collect`
	TaskInfo	`TaskInfo.attempt`	`TaskInfo.attemptNumber`
	Guava Optional	`Guava Optional`	`org.apache.spark.api.java.Opt`
	Vector	`Vector, VectorSuite`	
Configuration options and params		`--name`	
		`--driver-memory`	`spark.driver.memory`
		`--driver-cores`	`spark.driver.cores`
		`--executor-memory`	`spark.executor.memory`
		`--executor-cores`	`spark.executor.cores`
		`--queue`	`spark.yarn.queue`
		`--files`	`spark.yarn.dist.files`
		`--archives`	`spark.yarn.dist.archives`
		`--addJars`	`spark.yarn.dist.jars`
		`--py-files`	`spark.submit.pyFiles`

last-updated: Apr 11, 2022

Note also the following deprecated configuration options and parameters:

- Methods from Python `DataFrame` that returned `RDD` have been moved to `dataframe.rdd`. For example, `df.map` is now `df.rdd.map`.
- Some streaming connectors (Twitter, Akka, MQTT, and ZeroMQ) have been removed.
- `org.apache.spark.shuffle.hash.HashShuffleManager` no longer exists. `SortShuffleManager` is the default since Spark 1.2.
- `DataFrame` is no longer a class. It is a subtype of `DataSet`.

Behavior Changes

Spark 2.0.1 implements the following behavior changes:

- Spark 2.0.1 uses Scala 2.11 instead of 2.10.
- Floating literals in SQL are now parsed as decimal type instead of double type.
- The Kryo version is now 3.0.
- Jersey version is now 2.
- Java `RDD flatMap` and `mapPartitions` functions now require functions that return Java `iterator` instead of `Iterable`.
- Java `RDD countByKey` and `countApproxDistinctByKey` now return `Map[K => Long]` instead of `Map[K => Object]`.
- When writing Parquet files, the summary files are no longer written (set `parquet.enable.summary-metadata` to `true` to re-enable).
- Lots were changed in MLLib. Follow the Apache Spark Migration Guide.
- `Sparkcontext.emptyRDD` now returns `RDD` instead of `EmptyRDD`.
- Spark Standalone Master no longer serves the jobs history.
- org.apache.spark.api.java.JavaPairRDD methods were changed:
  - `countByKey` and `countApproxDistinctByKey` now return `java.lang.Long` instead of `scala.Long`.
  - `sampleByKey` and `sampleByKeyExact` now return `java.lang.Double` instead of `scala.Double`.
- The Old Application History format that created folders for each application has been removed.
- `org.apache.spark.Logging` is now private. You can use `slf4j` directly instead.

Other Deprecated Items

- Java 7 is now deprecated.
- Python 2.6 is now deprecated.
- TaskContext.isRunningLocally now is always false, as there is no more local execution of *yarn-client* and `yarn-cluster` as masters. Use `--master yarn` and `--deploy-mode client/cluster`.

- Instead of `HiveContext`, use `SparkSession.builder.enableHiveSupport`.
- Instead of `SQLContext`, use `SparkSession.builder`.
- Some methods related to `Accumulators`, `ShuffleWriteMetrics`, `SparklLoop`, `DataSet`, and `SQLContext` are now deprecated. You will see warnings in your application logs if you use them.

Structured Streaming in Spark
Starting in EEP 5.0.0, structured streaming is supported in Spark.

Related Links

Spark streaming is integrated with MapR Event Store For Apache Kafka for Apache Kafka.

- MapR Event Store For Apache Kafka Clients and Tools

Prerequisites for Using Structured Streaming in Spark
To deploy a structured streaming application in Spark, you must create a MapR Streams topic and install a Kafka client on all nodes in your cluster.
Creating a MapR Streams Topic

- Create a MapR Streams topic consisting of the stream path and topic name separated by a colon (:); for example, `/test_stream:topic1`.

Installing a Kafka Client

- Install a kafka-client on all nodes of your cluster or copy the `kafka-clients.jar` file from `/opt/mapr/lib/kafka-clients-<version>mapr<release>.jar` to `/opt/mapr/spark/spark-<version>/jars/`.

Using Structured Streaming to Create a Word Count Application
The example in this section creates a dataset representing a stream of input lines from Kafka and prints out a running word count of the input lines to the console.
Using Apache Kafka

Scala

```
val spark = SparkSession
       .builder
       .appName("StructuredKafkaWordCo
unt")
       .getOrCreate()

import spark.implicits._
//Create a DataSet representing the
stream of input lines from Kafka
val lines = spark
       .readStream
       .format("kafka")
       .option("kafka.bootstrap.server
s", bootstrapServers)
       .option(subscribeType, topics)
       .load()
       .selectExpr("CAST(value AS
STRING)")
       .as[String]
//Generate a running word count
val wordCounts =
lines.flatMap(_.split("
")).groupBy("value").count()
//Run the query that prints the
running counts to the console
```

```
val query = wordCounts.writeStream
       .outputMode("complete")
       .format("console")
       .option("checkpointLocation",
checkpointLocation)
       .start()

query.awaitTermination()
```

Java

```
SparkSession spark = SparkSession
              .builder()
              .appName("JavaStructured
KafkaWordCount")
              .getOrCreate();
//Create a DataSet representing the
stream of input lines from Kafka
Dataset<String> lines = spark
              .readStream()
              .format("kafka")
              .option("kafka.bootstrap
.servers", bootstrapServers)
              .option(subscribeType,
topics)
              .load()
              .selectExpr("CAST(value
AS STRING)")
              .as(Encoders.STRING());
//Generate a running word count
Dataset<Row> wordCounts =
lines.flatMap(
(FlatMapFunction<String, String>)
x -> Arrays.asList(x.split("
")).iterator(),
Encoders.STRING()).groupBy("value").co
unt();

//Run the query that prints the
running counts to the console
StreamingQuery query =
wordCounts.writeStream()
              .outputMode("complete")
              .format("console")
              .start();

query.awaitTermination();
```

Python

```
spark = SparkSession\
          .builder\
          .appName("StructuredKafkaWor
dCount")\
          .getOrCreate()

#Create a DataSet representing the
stream of input lines from Kafka
lines = spark\
          .readStream\
          .format("kafka")\
          .option("kafka.bootstrap.ser
vers", bootstrapServers)\
          .option(subscribeType,
```

```
topics)\
          .load()\
          .selectExpr("CAST(value AS
STRING)")

#Split the lines into words
words = lines.select(
#explode turns each item in an array
into a separate row
explode(
        split(lines.value, ' ')
       ).alias('word')
     )

#Generate a running word count
wordCounts =
words.groupBy('word').count()

#Run the query that prints the
running counts to the console
query = wordCounts\
           .writeStream\
           .outputMode('complete')\
           .format('console')\
           .start()

query.awaitTermination()
```

Using MapR Event Store for Apache Kafka

For MapR Event Store, the topic name consists of the stream name and topic, and the bootstrap servers are not used. For example:

```
var topic: String = "/user/mapr/stream:reviews"
val df1 = spark.readStream.format("kafka").option("kafka.bootstrap.servers",
   "maprdemo:9092").option("subscribe", topic).option("group.id",
   "testgroup").option("startingOffsets",
"earliest").option("failOnDataLoss",
          false).option("maxOffsetsPerTrigger", 1000).load()
```

Writing a Structured Spark Stream to MapR Database JSON Table

The example in this section writes a structured stream in Spark to MapR Database JSON table.

To write a structured Spark stream to MapR Database JSON table, use `MapRDBSourceConfig.Format` for Java and Scala and `com.mapr.db.spark.streaming` for Python to format the `tablePath`, `idFieldPath`, `createTable`, `bulkMode`, and `sampleSize` parameters.

Scala

```
import
com.mapr.db.spark.streaming.MapRDBSour
ceConfig
import org.apache.spark.sql.streaming.
{DataStreamReader, DataStreamWriter}
import org.apache.spark.sql.
{DataFrame, Row, SparkSession}

def dataStreamWriter(spark:
SparkSession, df: DataFrame):
DataStreamWriter[Row] = {
import spark.implicits._
```

```
df.select($"value" as "_id")
  .writeStream
  .format(MapRDBSourceConfig.Format)
  .option(MapRDBSourceConfig.TablePath
Option, "/table/path")
  .option(MapRDBSourceConfig.IdFieldPa
thOption, "value")
  .option(MapRDBSourceConfig.CreateTab
leOption, true)
  .option(MapRDBSourceConfig.BulkModeO
ption, true)
  .option(MapRDBSourceConfig.SampleSiz
eOption, 1000)
  .outputMode("append")
}
```

Java

```
import
com.mapr.db.spark.streaming.MapRDBSour
ceConfig;
import org.apache.spark.sql.Dataset;
import org.apache.spark.sql.Row;
import
org.apache.spark.sql.SparkSession;
import
org.apache.spark.sql.streaming.DataStr
eamReader;
import
org.apache.spark.sql.streaming.DataStr
eamWriter;
import
org.apache.spark.sql.streaming.Streami
ngQueryException;

DataStreamWriter<Row>
dataStreamWriter(Dataset<Row> df) {
   return df.selectExpr("CAST(value
AS STRING) as _id")
            .writeStream()
            .format(MapRDBSourceConfig
.Format())
            .option(MapRDBSourceConfig
.TablePathOption(), "/table/path")
            .option(MapRDBSourceConfig
.IdFieldPathOption(), "value")
            .option(MapRDBSourceConfig
.CreateTableOption(), true)
            .option(MapRDBSourceConfig
.BulkModeOption(), true)
            .option(MapRDBSourceConfig
.SampleSizeOption(), 1000)
            .outputMode("append");
 }
```

Python

```
from pyspark.sql import *

def data_stream_writer_func(df,
checkpoint_dir, table_path):
  return df.selectExpr("CAST(value AS
STRING) as _id") \
           .writeStream \
```

```
            .format("com.mapr.db.spark.
streaming") \
            .option("checkpointLocation
", checkpoint_dir) \
            .option("tablePath",
table_path) \
            .option("idFieldPath",
"value") \
            .option("createTable",
True) \
            .option("bulkMode", True) \
            .option("sampleSize", 1000)
```

Writing a Spark Stream Word Count Application to MapR Database
The example in this section writes a Spark stream word count application to MapR Database.

Scala

```
val spark = SparkSession
      .builder
      .appName("StructuredKafkaWordCou
nt")
      .getOrCreate()

import spark.implicits._
//Create a DataSet representing the
stream of input lines from Kafka
val lines = spark
      .readStream
      .format("kafka")
      .option("kafka.bootstrap.servers
", bootstrapServers)
      .option(subscribeType, topics)
      .load()
      .selectExpr("CAST(value AS
STRING)")
      .as[String]

//Generate a running word count
val wordCounts =
lines.flatMap(_.split("
")).groupBy("value").count()

//Run the query that saves the result
to MapR-DB
val query = wordCounts.writeStream
      .format(MapRDBSourceConfig.Forma
t)
      .option(MapRDBSourceConfig.Table
PathOption, resultTable)
      .option(MapRDBSourceConfig.Creat
eTableOption, true)
      .option(MapRDBSourceConfig.IdFie
ldPathOption, "value")
      .outputMode("complete")
      .start()

query.awaitTermination()
```

Java

```
SparkSession spark = SparkSession
            .builder()
```

```
            .appName("JavaStructuredKaf
kaWordCount")
            .getOrCreate();

//Create a DataSet representing the
stream of input lines from Kafka
Dataset<String> lines = spark
             .readStream()
             .format("kafka")
             .option("kafka.bootstrap.s
ervers", bootstrapServers)
             .option(subscribeType,
topics)
             .load()
             .selectExpr("CAST(value
AS STRING)")
             .as(Encoders.STRING());

//Generate a running word count
Dataset<Row> wordCounts =
lines.flatMap(
(FlatMapFunction<String, String>)
x -> Arrays.asList(x.split("
")).iterator(),
Encoders.STRING()).groupBy("value").co
unt();

//Run the query that saves the result
to MapR-DB
StreamingQuery query =
wordCounts.writeStream()
             .format(MapRDBSourceConfig
.Format())
             .option(MapRDBSourceConfig
.TablePathOption(), resultTable)
             .option(MapRDBSourceConfig
.CreateTableOption(), true)
             .option(MapRDBSourceConfig
.IdFieldPathOption(), "value")
             .outputMode("complete");
             .start();

query.awaitTermination();
```

Python

```
spark = SparkSession\
            .builder\
            .appName("StructuredKafkaWo
rdCount")\
            .getOrCreate()

#Create a DataSet representing the
stream of input lines from Kafka
lines = spark\
           .readStream\
           .format("kafka")\
           .option("kafka.bootstrap.ser
vers", bootstrapServers)\
           .option(subscribeType,
topics)\
           .load()\
           .selectExpr("CAST(value AS
```

```
STRING)")

#Split the lines into words
words = lines.select(
#Explode turns each item in an array
into a separate row
explode(
       split(lines.value, ' ')
       ).alias('word')
)

#Generate a running word count
wordCounts =
words.groupBy('word').count()

#Run the query that saves the result
to MapR-DB
query = wordCounts\
              .writeStream\
              .format("com.mapr.db.spa
rk.streaming") \
              .option("tablePath",
table_path) \
              .option("createTable",
True) \
              .option("idFieldPath",
"value") \
              .outputMode('complete')\
              .start()

query.awaitTermination()
```

PAM Authentication for Spark
Spark supports PAM authentication on secure MapR clusters.

In EEP-5.0.0, PAM authentication and encryption is enabled by default for all Spark Web UIs. After running configure.sh, if the cluster is secure and Spark is installed, Spark will be configured using PAM.

Note: Spark PAM is available in Spark-2.2.1 from EEP-5.0 and in Spark-2.1.0 from EEP-4.1.1.

See Configuring PAM on page 1442 for information on how PAM works with MapR.

Read or Write LZO Compressed Data for Spark
This topic provides details for reading or writing LZO compressed data for Spark.

1. Install the LZO library:

```
sudo yum install lzo-devel lzo
```

2. Clone hadoop-lzo and build it:

```
[mapr@node1 ~]$ git clone https://github.com/twitter/hadoop-lzo
[mapr@node1 ~]$ cd hadoop-lzo
[mapr@node1 hadoop-lzo]$ mvn package
```

3. Copy the jar file to hadoop classpath:

```
[mapr@node1 hadoop-lzo]$ sudo
cp target/hadoop-lzo-0.4.21-SNAPSHOT.jar /opt/mapr/hadoop/hadoop-2.7.0/
share/hadoop/yarn/lib/
```

4. Add two LZO compression codes to `core-site.xml`:

```
property: io.compression.codecs
codecs:
com.hadoop.compression.lzo.LzoCodec,com.hadoop.compression.lzo.LzopCodec/
```

   It will look like this:

```
<property>
  <name>io.compression.codecs</name>

<value>org.apache.hadoop.io.compress.DefaultCodec,org.apache.hadoop.io.co
mpress.GzipCodec,org.apache.hadoop.io.compress.BZip2Codec,org.apache.hado
op.io.compress.DeflateCodec,org.apache.hadoop.io.compress.SnappyCodec,com
.hadoop.compression.lzo.LzoCodec,com.hadoop.compression.lzo.LzopCodec</
value>
</property>

<property>
 <name>io.compression.codec.lzo.class</name>
 <value>com.hadoop.compression.lzo.LzoCodec</value>
</property>
```

5. Run Spark and read LZO compressed data:

```
[mapr@node1 spark]$ ./bin/spark-shell --master yarn
spark.read.csv("/user/mapr/LzoCompressedCsv").show
```

6. Write LZO compressed data with Spark:

```
scala>
df.write.option("codec","com.hadoop.compression.lzo.LzopCodec").csv("csv1
")

[mapr@node1 spark]$ hadoop fs -ls /user/mapr/csv1
Found 2 items
-rwxr-xr-x   3 mapr mapr          0 2017-12-15 12:42 /user/mapr/csv1/
_SUCCESS
-rwxr-xr-x   3 mapr mapr     493366 2017-12-15 12:42 /user/mapr/csv1/
part-00000-256a95a9-eb9c-4048-b7ce-c95dfbef54d7.csv.lzo
```

Ports Used by Spark

To run a Spark job from a client node, ephemeral ports should be opened in the cluster for the client from which you are running the Spark job.

If you do not want to open all the ephemeral ports, you can use the configuration parameter to specify the range of ports.

To set ports to special values, use the `spark.driver.port`, `spark.blockManager.port`, and `spark.port.maxRetries` properties. The `spark.port.maxRetries` property is 16 by default.

For example, if you need to open port 200 for `spark.blockManager.port` from 40000, set `spark.blockManager.port = 40000` and `spark.port.maxRetries = 200`.

For a list of Web UIs ports dynamically used when starting spark contexts, see the open source documentation.

The default port numbers that need to be opened on the firewall behind the client and MapR cluster nodes for Spark jobs to operate in YARN client, YARN cluster, and standalone modes are as follows:

Service	Port Number
Spark Standalone Master (RPC)	7077
Spark Standalone Master (Web UI)	8580, 8980*
Spark Standalone Worker	8581, 8981*
Spark Thrift Server	2304
Spark History Server	18080,18480*
Spark External Shuffle Service (if yarn shuffle service is enabled)	7337
CLDB	7222
ZooKeeper	5181
Nodes running ResourceManager	8032
MapR Filesystem Server	5660, 5692

* refers to ports for secure clusters

ACL Configuration for Spark
Starting in the EEP 6.0 release, the ACL configuration for Spark is disabled by default.

If you are authorized by PAM, you will have access to all Spark UIs. For the Spark History Server, you can only see the logs of your own Spark jobs if PAM is enabled (regardless of ACL being enabled).

Starting in Spark-2.4.4.0, MapR Spark ACLs behave like Apache Spark ACLs. With this change, all users can log in to the Spark History Server UI and see the full list of applications. Only an application owner or the users specified in `spark.ui.view.acls` or `history.ui.admin.acls` can see application details. Users specified in `history.ui.admin.acls` can see the details for all applications.

By default on a secure cluster:

```
spark.acls.enable false
spark.admin.acls mapr
spark.admin.acls.groups mapr
spark.ui.view.acls mapruser1
```

Other Example:

```
spark.acls.enable true - ACL is enabled and restricted access to Spark
master and thriftserver UIs for other users.
spark.admin.acls mapr - Administrator or "sudoer" of ACL access.
spark.admin.acls.groups mapr - Group of administrators.
spark.ui.view.acls mapruser1 - user who can be logged in to Spark master
and thriftserver UIs.
```


last-updated: Apr 11, 2022

Sqoop

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Apache Sqoop™ is a tool designed to efficiently transfer bulk data between Apache Hadoop and structured datastores, such as relational databases.

This documentation provides information for using Sqoop and Sqoop2, but does not duplicate the Apache Sqoop™ documentation on the Apache Sqoop website.

The following table describes the differences between Sqoop1 or Sqoop2:

Feature	Sqoop1	Sqoop2
Specialized connectors for all major RDBMS	Available.	Not available. However, you can use the `generic-jdbc-connector` , which has been tested on these databases: • MySQL • Microsoft SQL Server • Oracle (Not supported in Sqoop 1.99.7) • PostgreSQL The generic JDBC connector should also work with any other JDBC-compliant database, although specialized connectors probably give better performance.
Data transfer from RDBMS to Hive	Done automatically.	Must be done manually in two stages: 1. Import data from RDBMS into MapR File System. 2. Load data into Hive using the `LOAD DATA` command **Note:** As of Sqoop 1.99.7, you can also use the `kite-connector` to load data into Hive.

last-updated: Apr 11, 2022

Feature	Sqoop1	Sqoop2
Data transfer from Hive to RDBMS	Must be done manually in two stages: 1. Extract data from Hive into MapR File System, as a text file or as an Avro file. 2. Export the output of step 1 to an RDBMS using Sqoop.	Must be done manually in two stages: 1. Extract data from Hive into MapR File System, as a text file or as an Avro file. **Note:** As of Sqoop 1.99.7, you can also use the `kite-connector` to extract data from Hive. 2. Export the output of step 1 to an RDBMS using Sqoop.
Integrated Kerberos security	Supported.	Supported.
Password encryption	Not supported.	Supported as of Sqoop 1.99.7.

Sqoop1

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Sqoop transfers data between the MapR File System and relational databases. You can use Sqoop to transfer data from a relational database management system (RDBMS), such as MySQL or Oracle, into the MapR File System and use MapReduce on the transferred data. Sqoop can export this transformed data back into an RDBMS.

The Maven repository contains JAR files for Sqoop that you can download and use for your application:

Version	Core	EEP	Maven Repository Locations
Sqoop 1.4.7	6.1.x	6.3.2	1.4.7-mapr-632/
		6.3.1	1.4.7-mapr-2009/
		6.3.0	1.4.7-mapr-1904/

For more information about Sqoop, see the Apache Sqoop 1.4.7 Documentation.

MapR Connector for Teradata

The MapR Connector for Teradata (powered by the Teradata Connector for Hadoop (TDCH)) is a connector for Sqoop1 that enables the transfer of data between a Teradata MPP database and MapR environments.

Sqoop TDCH Features

The MapR Connector for Teradata supports the following features:

- Import of data to Sqoop in these formats:
  - TextFormat, delimited
  - RCFile
  - ORCFile
  - Avro file
- Hive arguments. The MapR Connector for Teradata supports all standard Hive arguments and all data types except Union.

last-updated: Apr 11, 2022

- Export from and import to HCatalog tables.
- Automatic schema mapping to or from HCatalog.
- Import using a query.

The following features are NOT supported:

- Update table
- Compression
- Parquet format

Supported Product Versions and System Requirements

You must have the following minimum software versions to use the MapR Connector for Teradata:

- Teradata Database 15.10
- Hive 1.2
- MapR 5.2
- Sqoop 1.4.6

The Sqoop 1.4.6-1707 release supports TDCH 1.5.2. Prior releases support TDCH 1.5.1.

The MapR Connector for Teradata also requires JRE/JDK 1.7.x or 1.8.x. Make sure to include the Hive libraries in your Java classpath.

Importing and Exporting Data Using the MapR Connector for Teradata

You can specify MapR Connector for Teradata options using any of these methods:

- Configuration file
- `-D` command line option
- Sqoop options

Some Sqoop options are unsupported in the current release. For a list of unsupported Sqoop options, see Sqoop TDCH Import and Export Options.

This table shows examples for importing and exporting data:

Table

Operation	Example

Import from Hive and HCatalog	Importing from Hive and HCatalog requires that `HIVE_HOME` and `HCAT_HOME` be specified before the Sqoop command is run or using `--hive-home/--hcatalog-home` options. This example shows the environment variable setup:
	`export HIVE_HOME=/opt/mapr/hive/hive-1.2`
	`export HCAT_HOME=/opt/mapr/hive/hive-1.2/hcatalog`
	Or
	`sqoop import --connect jdbc:teradata://<hostname>/ database=test --connection-manager org.apache.connectors.td.TeradataManager --username test --password test --table test --hive-import --hive-overwrite --hive-home /opt/mapr/hive/hive-1.2/`

Import from Teradata to MapR File System	`sqoop import --connect jdbc:teradata://<hostname>/ database=test --connection-manager org.apache.connectors.td.TeradataManager --username test --password test --table test --target-dir /user/mapr/test`
Import from Teradata into a Hive Table	`sqoop import --connect jdbc:teradata://<hostname>/ database=test --connection-manager org.apache.connectors.td.TeradataManager --username test --password test --table test --hive-import --hive-overwrite --hive-home /opt/mapr/hive/hive-1.2/ --fields-terminated-by ';'`
Import from Teradata into an HCatalog Table	`sqoop import --connect jdbc:teradata://<hostname>/ database=test --connection-manager org.apache.connectors.td.TeradataManager --username test --password test --table test --hcatalog-table test`

last-updated: Apr 11, 2022

Table (Continued)

Operation	Example
Import from Teradata into a Hive Table Stored as RCFile  **Note:** If you must import to a table stored as an RCFile or ORCFile, the table must be created before data is imported.	`sqoop import -D tdch.fileformat="rcfile" --connect jdbc:teradata://<hostname>/ database=test --connection-manager org.apache.connectors.td.TeradataManager --username test --password test --table test --hive-import --hive-overwrite --hive-home /opt/mapr/hive/hive-1.2/`
Export to Teradata	`sqoop export --connect jdbc:teradata://<hostname>/ database=test --connection-manager org.apache.connectors.td.TeradataManager --username test --password test --table test --export-dir /user/mapr/test`
Export from an AVRO File to Teradata	`sqoop export -D tdch.fileformat="avrofile" --connect jdbc:teradata://<hostname>/ database=test --connection-manager org.apache.connectors.td.TeradataManager --username test --password test --table test --export-dir /user/mapr/avro`

MapR Connector Import and Export Options

Import Options

Import Option	Description

`--as-avrodatafile`, `--as-textfile`	The format of a to-be-imported data file in MapR File System. An 'hcat' or 'hive' job type supports 'rcfile', 'orcfile' and 'textfile' file formats. To set the file format, you need to use the `-D` command line option. For example:
	`sqoop import -D tdch.fileformat="orcfile" --connect jdbc:teradata://<hosname>/database=test --connection-manager org.apache.connectors.td.TeradataManager …`
	  **Note:** If import to MapR File System in Avro format fails, include `-Dmapreduce.job.user.classpath.first=true` in your command.
	`sqoop import --connect jdbc:teradata://<hostname>/ database=test --connection-manager org.apache.connectors.td.TeradataManager --as-avrodatafile ...`

`--target-dir`	MapR File System destination directory.

last-updated: Apr 11, 2022

`--num-mappers`	The number of mappers for the import job. The default value is 4.
`--query`	The SQL query to select data from a Teradata database. This option works only with the textfile and avrofile formats.
`--table`	The name of the source table in a Teradata system from which the data is imported.
`--columns`	The names of columns to import from the source table in a Teradata system, in comma-separated format. For example: `sqoop import --connect jdbc:teradata://<hostname>/ database=test --connection-manager org.apache.connectors.td.TeradataManager --username test --password test --table test --columns id,name`
`--hive-table`	The name of the target table in Hive or HCatalog.
`--fields-terminated-by`	The field separator to use with the imported files. This parameter is only applicable with the textfile file format. The default value is `\t`.
`--split-by`	The column of the table used to split work units.
`--map-column-hive`	Override mapping from SQL to Hive type for configured columns.
`--where`	WHERE clause to use during import.
`--staging-table`Footnote.	The table for staging data before insertion into the destination table. Only applicable when using `input-method split.by.partition`
`--num-partitions-for-staging-table`Footnote.	The number of partitions to create when auto-creating the staging table. Only applicable when using `input-method split.by.partition`.
`--staging-database`Footnote.	The database for creating the staging table. Only applicable when using `input-method split.by.partition.`
`--staging-force`Footnote.	Option to force the connector to create a staging table, if supported. Only applicable when using `input-method split.by.partition`.
`--input-method`Footnote.	The input method to use to transfer data from Teradata. Supported values: • `split.by.amp` • `split.by.value` • `split.by,partition` • `split.by.hash`
`--batch-size`Footnote.	The number of row processed per batch.
`--access-lock`Footnote.	Option to apply access lock on the database.
`--query-band`Footnote.	The arbitrary query bands to be set for all queries the connector runs. Specify the query bands using a semicolon-separated `key=value` format.
`--skip-xviews`Footnote.	Option to switch to the non-X version of system views to obtain metadata.
`--date-format`Footnote.	Custom format for date columns.

[1] Only available starting in Sqoop-1.4.6-1707.

last-updated: Apr 11, 2022

`--time-format`Footnote.	Custom format for time columns.
`--timestamp-format`Footnote.	Custom format for timestamp columns.

Only available starting in Sqoop-1.4.6-1707.

Use the `-D` command line option to set the file format. For example:

```
-D tdch.fileformat="fileformat"
```

The following Sqoop import options are unsupported:

- `--append`
- `--compression-codec`
- `--direct`
- `--direct-split-size`
- `--compress, -z`
- `--check-column`
- `--incremental`
- `--last-value`
- `--mysql-delimiters`
- `--optionally-enclosed-by`
- `--hive-delims-replacement`
- `--hive-drop-import-delims`
- `--hive-partition-key`
- `--hive-partition-value`
- `--column-family`
- `--hbase-create-table`
- `--hbase-row-key`
- `--hbase-table`
- `--map-column-java`
- `--fetch-size`
- `--as-sequencefile`

Export Options

Export Option	Description

`--table`	The name of the target table in a Teradata system.
`--export-dir`	The directory of to-be-exported source files in MapR File System.
`--num-mappers`	The number of mappers for the export job. The default value is 4.
`--columns`	The names of fields to export to the target table in a Teradata system, in comma-separated format. For export from MapR File System, you can only use this option with the avrofile format.
`--staging-table`	The table in which data will be staged before being inserted into the destination table.
`--keep-staging-table`Footnote.	Option specifying that the connector retain the staging table after failures.
`--staging-database`Footnote.	The database for creating the staging table.
`--staging-force`Footnote.	Option to force the connector to create a staging table, if supported.
`--output-method`Footnote.	The output method to use to transfer data to Teradata. Supported values: • `batch.insert` • `internal.fastload`
`--query-band`Footnote.	The arbitrary query bands to be set for all queries that the connector runs. Specify the query bands using a semicolon-separated `key=value` format.
`--error-table`Footnote.	Prefix name for error tables. Only applicable when using `output-method internal.fastload`.
`--error-database`Footnote.	Override for the default error database name. Only applicable when using `output-method internal.fastload`.
`--fastload-socket-hostname`Footnote.	Hostname or IP address of the host on which Sqoop runs. If not set, the connector auto-detects the host. Only applicable when using `output-method internal.fastload`.
`--fastload-socket-port`Footnote.	The host port that fastload tasks use to synchronize state. Only applicable when using `output-method internal.fastload`.
`--fastload-socket-timeout`Footnote.	The timeout value the server socket uses for fastload task connections. Only applicable for `output-method internal.fastload`.
`--skip-xviews`Footnote.	Option to switch to the non-X version of system views to obtain metadata.
`--date-format`Footnote.	Custom format for date columns.
`--time-format`Footnote.	Custom format for time columns.
`--timestamp-format`Footnote.	Custom format for timestamp columns.

Use the `-D` command line option to set the file format. For example:

```
-D tdch.fileformat="fileformat"
```

Supported export file format values are:

- textfile (default format)
- avrofile

last-updated: Apr 11, 2022

- orcfile
- refile

The following Sqoop export options are unsupported:

- `--batch`
- `--clear-staging-table`
- `--direct`
- `--update-key`
- `--update-mode`
- `--input-lines-terminated-by`
- `--input-optionally-enclosed-by`
- `--map-column-java`
- `--as-sequencefile`

YARN

YARN is a resource-management and scheduling framework that distributes resource-management and job-management duties. YARN assigns the resource-management and job-management duties as follows:

- ResourceManager: manages cluster resources and tracks resource usage and node health.
- ApplicationMaster: a framework-specific process that negotiates resources for a single application (a single job or a directed acyclic graph of jobs), which runs in the first *container* allocated for the application.
- A YARN component called the HistoryServer archives job metrics and metadata. Status on completed applications is available via REST APIs.

The ResourceManager allocates resources among all the applications running the cluster. The ResourceManager includes a pluggable scheduler, which is responsible for allocating resources according to the resource requirements of the running applications. Current MapReduce schedulers, including the Capacity Scheduler and the Fair Scheduler, can be plugged into the YARN scheduler directly.

Label-based scheduling provides job placement control on a multi-tenant Hadoop cluster. Administrators can control exactly which nodes are chosen to run jobs submitted by different users and groups. An administrator assigns node labels in a text file, then composes queue labels or job labels based on the node labels. When users run jobs, they can place them on specified nodes on a per-job basis (using a job label) or on a per-queue level (using a queue label).

The ResourceManager caches the mapping file, and checks every two minutes (the default monitoring period) for updates. If the file has been modified, the ResourceManager updates the labels for all active ApplicationMasters immediately.

Each application runs an ApplicationMaster to negotiate resources from the ResourceManager. The ApplicationMaster works with the NodeManagers to execute and monitor tasks. The duties of the ApplicationMaster are divided as follows:

- NodeManager: One instance runs on each node, to manage that node's resources.
- Container: An abstraction representing a unit of resources on a node.

The NodeManager provides containers to an application. The ResourceManager and the NodeManager provide the system for distributed management of applications and resources.

ResourceManager
Describes the role of the ResourceManager.

The ResourceManager is mainly concerned with arbitrating available resources in the cluster among competing applications, with the goal of maximum cluster utilization. The ResourceManager includes a pluggable scheduler called the YarnScheduler, which allows different policies for managing constraints such as capacity, fairness, and service level agreements.

The ResourceManager manages resources as follows:

- Each NodeManager takes instructions from the ResourceManager, reporting and handling containers on a single node
- Each ApplicationMaster requests resources from the ResourceManager, then works with containers provided by NodeManagers

The ResourceManager communicates with application clients via an interface called the ClientService. A client can submit or terminate an application and gain information about the scheduling queue or cluster statistics through the ClientService.

Administrative requests are served by a separate interface called the AdminService, through which operators can get updated information about cluster operation.

Behind the scenes, the ResourceTrackerService receives node heartbeats from the NodeManager to track new or decommissioned nodes. The NMLivelinessMonitor and NodesListManager keep an updated status of which nodes are healthy so that the scheduler and the ResourceTrackerService can allocate work appropriately.

A component called the ApplicationMasterService manages ApplicationMasters on all nodes, keeping the scheduler informed. A component called the AMLivelinessMonitor keeps a list of ApplicationMasters and their last heartbeat times, in order to let the ResourceManager know what applications are healthy on the cluster. Any ApplicationMaster that does not heartbeat within a certain interval is marked as dead and re-scheduled to run on a new container.

At the core of the ResourceManager is an interface called the ApplicationsManager, which maintains a list of applications that have been submitted, are running, or are completed. The ApplicationsManager accepts job submissions, negotiates the first container for an application (in which the ApplicationMaster will run) and restarts the ApplicationMaster if it fails.

The ResourceManager and NodeManagers communicate via heartbeats.

Configure the ResourceManager for high availability so that the failure of the ResourceManager service is a not single point of failure for the cluster. High availability of the ResourceManager is configured by default when you run `configure.sh` without specifying the `-RM` parameter.

ApplicationMaster
Describes the role of the ApplicationMaster.

The ApplicationMaster is an instance of a framework-specific library that negotiates resources from the ResourceManager and works with the NodeManager to execute and monitor the granted resources (bundled as containers) for a given application. An application can be a process or set of processes, a service, or a description of work.

The ApplicationMaster is run in a container like any other application. The ApplicationsManager, part of the ResourceManager, negotiates for the container in which an application's ApplicationMaster runs when the application is scheduled by the YarnScheduler.

While an application is running, the ApplicationMaster manages the following:

- Application life cycle

last-updated: Apr 11, 2022

- Dynamic adjustments to resource consumption
- Execution flow
- Faults
- Providing status and metrics

The ApplicationMaster is designed to support a specific framework, and can be written in any language since its communication with the NodeManagers and the ResourceManager is accomplished using extensible communication protocols. The ApplicationMaster can be customized to extend the framework or run any other code. For this reason, the ApplicationMaster is not considered trustworthy, and is not run as a trusted service.

An ApplicationMaster typically requests resources on multiple nodes to complete a job by sending the ResourceManager requests that include locality preferences and attributes of the containers. When the ResourceManager is able to allocate a resource to the ApplicationMaster, it generates a lease that the ApplicationMaster pulls on a subsequent heartbeat. A security token associated with the lease guarantees its authenticity when the ApplicationManager presents the lease to the NodeManager to gain access to the container.

The Application Master heartbeats to the ResourceManager to communicate its changing resource needs, and to let the ResourceManager know it is still alive. In response, the ResourceManager can return a lease on additional containers on other nodes, or cancel the lease on some containers. The ApplicationMaster can then adjust its execution strategy to fit the increase or decrease in available resources. When cluster resources become scarce, the ResourceManager can also request that the ApplicationMaster relinquish some resources. The ApplicationMaster can move work to other running containers in order to give up resources gracefully.

Containers

A YARN container is a result of a successful resource allocation, meaning that the ResourceManager has granted an application a lease to use a specific set of resources in certain amounts on a specific node. The ApplicationMaster presents the lease to the NodeManager on the node where the container has been allocated, thereby gaining access to the resources.

To launch the container, the ApplicationMaster must provide a container launch context (CLC) that includes the following information:

- Environment variables
- Dependencies (local resources such as data files or shared objects needed prior to launch)
- Security tokens
- The command necessary to create the process the application plans to launch

The CLC makes it possible for the ApplicationMaster to use containers to run a variety of different kinds of work, from simple shell scripts to applications to virtual machines.

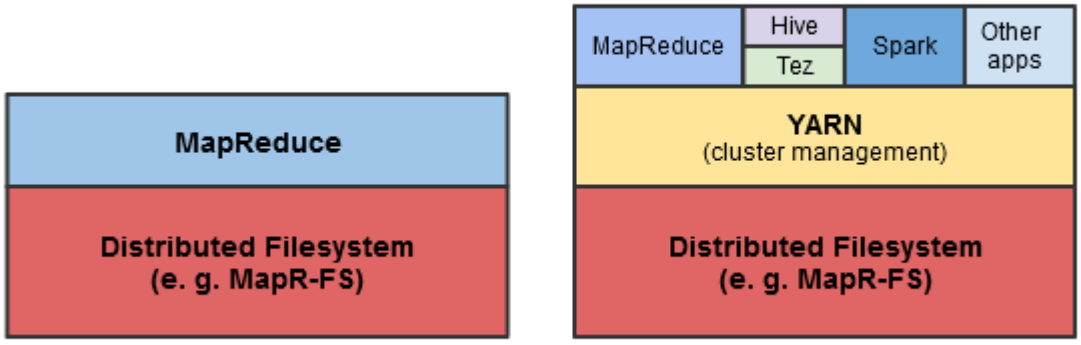
MapReduce Version 2

Provides an overview of how MapReduce works.

YARN dynamically allocates resources for applications as they execute. The MapReduce version 1 (MRv1) has been rewritten to run as an application on top of YARN; this new version is called MapReduce version 2.0 (MRv2).

Figure 2. A comparison between MapReduce 1.0 and MapReduce 2.0

last-updated: Apr 11, 2022



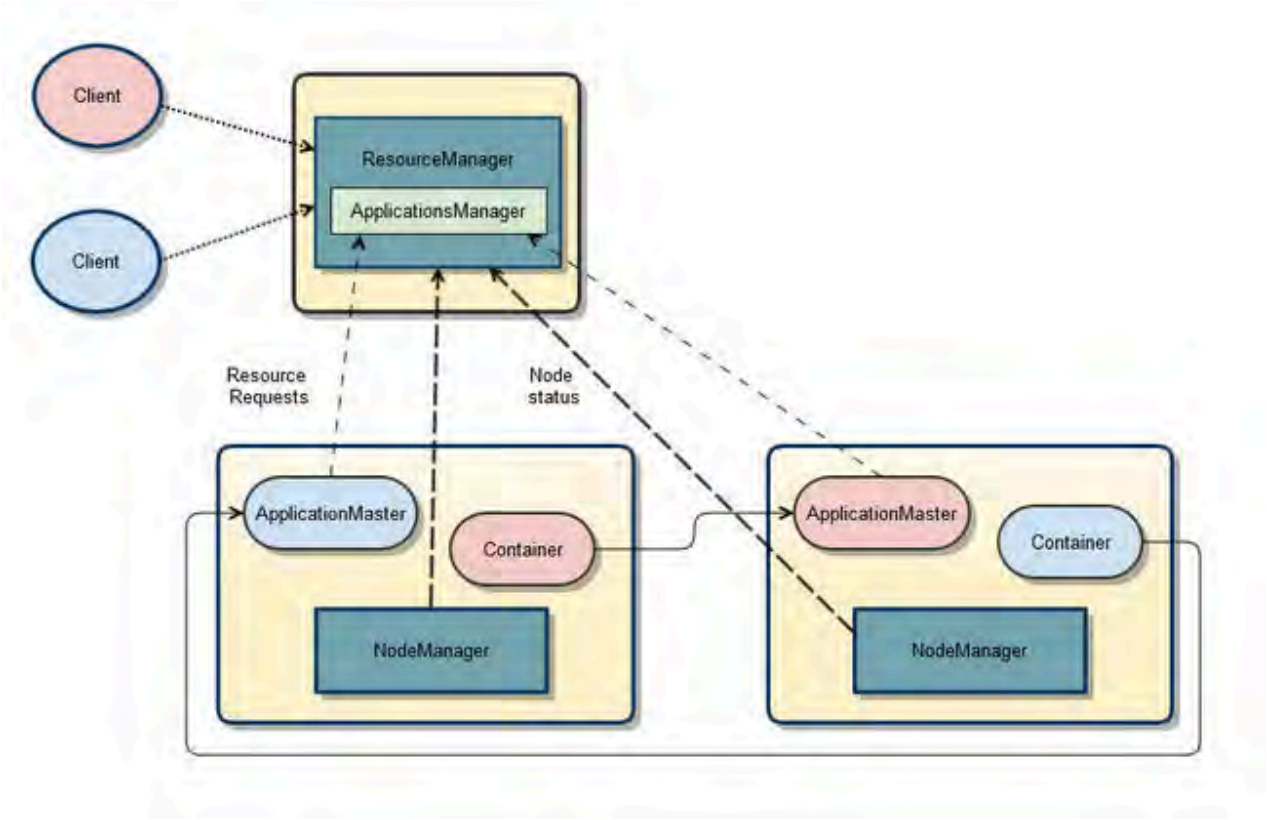
The main advancement in YARN architecture is the separation of resource management and job management, which were both handled by the same process (the JobTracker) in Hadoop 1.x. Cluster resources and job scheduling are managed by the ResourceManager, while resource negotiation and job monitoring are managed by an ApplicationMaster for each application running on the cluster. In MapReduce, each node advertises a relatively fixed number of map slots and reduce slots. This can lead to resource under-utilization, for example, when there is a heavy reduce load and map slots are available, because the map slots cannot accept reduce tasks (and vice versa).

YARN generalizes resource management for use by new engines and frameworks, allowing resources to be allocated and reallocated for different concurrent applications sharing a cluster. Existing MapReduce applications can run on YARN without any changes. At the same time, because MapReduce is now merely another application on YARN, MapReduce is free to evolve independently of the resource management infrastructure.

How Applications Work in YARN
Describes the data flow during application execution in YARN.

The following diagram and steps describe how data flows during application execution in YARN.

last-updated: Apr 11, 2022



The following steps summarize execution of the application:

1. A client submits an application to the YARN ResourceManager, including the information required for the Container Life Cycle (CLC).
2. The ApplicationsManager (in the ResourceManager) negotiates a container and bootstraps the ApplicationMaster instance for the application.
3. The ApplicationMaster registers with the ResourceManager and requests containers.
4. The ApplicationMaster communicates with NodeManagers to launch the containers it has been granted, specifying the CLC for each container.
5. The ApplicationMaster manages application execution. During execution, the application provides progress and status information to the ApplicationMaster. The client can monitor the application's status by querying the ResourceManager or by communicating directly with the ApplicationMaster.
6. The ApplicationMaster reports completion of the application to the ResourceManager.
7. The ApplicationMaster deregisters with the ResourceManager, which then cleans up the ApplicationMaster container.

Direct Shuffle on YARN
Explains the shuffle phase of a MapReduce application.

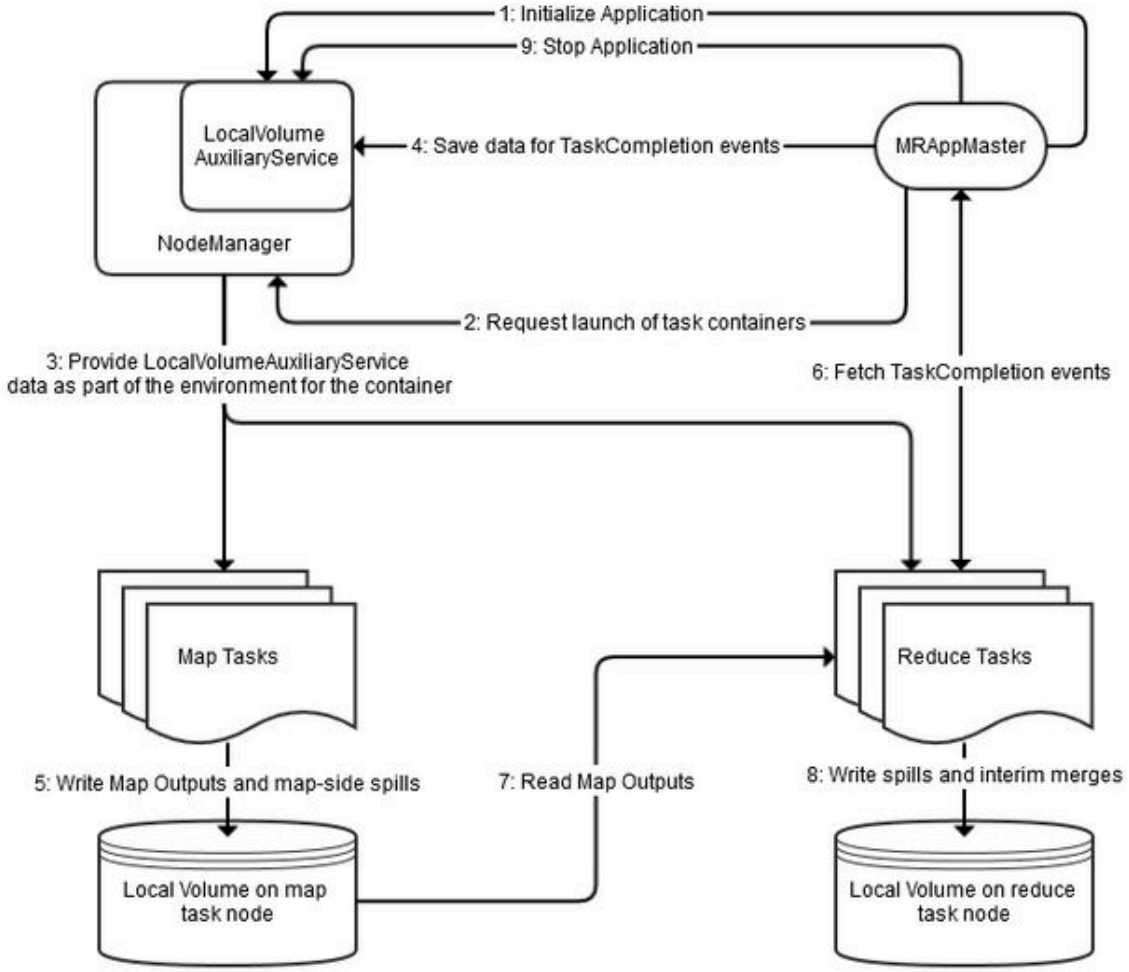
Overview of Direct Shuffle

During the shuffle phase of a MapReduce application, MapR writes to a MapR File System volume limited by its topology to the local node instead of writing intermediate data to local disks controlled by the operating system. This improves performance and reduces demand on local disk space while making the output available cluster-wide.

last-updated: Apr 11, 2022

Direct Shuffle is the default shuffle mechanism for MapR Data Platform. However, you can modify the `yarn-site.xml` and `mapred-site.xml` configuration files to enable Apache Shuffle for MapReduce applications. See Apache Shuffle on YARN.

The LocalVolumeAuxiliaryService runs in the NodeManager process. The LocalVolumeAuxiliaryService manages the local volume on each node and cleans up shuffle data after a MapReduce application has finished executing.



1. The MRAppMaster service initializes the application by calling `initializeApplication()` on the LocalVolumeAuxiliaryService.
2. The MRAppMaster service requests task containers from the ResourceManager. The ResourceManager sends the MRAppMaster information that MRAppMaster uses to request containers from the NodeManager.
3. The NodeManager on each node launches containers using information about the node's local volume from the LocalVolumeAuxiliaryService.
4. Data from map tasks is saved in MRAppMaster for later use in TaskCompletion events, which are requested by reduce tasks.
5. As map tasks complete, map outputs and map-side spills are written to the local volumes on the map task nodes, generating Task Completion events.

last-updated: Apr 11, 2022

6. ReduceTasks fetch Task Completion events from the Application Manager. The task Completion events include information on the location of map output data, enabling reduce tasks to copy data from MapOutput locations.
7. Reduce tasks read the map output information.
8. Spills and interim merges are written to local volumes on the reduce task nodes.
9. MRAppMaster calls `stopApplication()` on the LocalVolumeAuxiliaryService to clean up data on the local volume.

Configuration for Direct Shuffle

The deafult YARN parameters for Direct Shuffle are as follows:

```
<property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle,mapr_direct_shuffle</value>
    <description>shuffle service that needs to be set for Map Reduce to
run</description>
</property>
<property>
    <name>yarn.nodemanager.aux-services.mapr_direct_shuffle.class</name>
    <value>org.apache.hadoop.mapred.LocalVolumeAuxService</value>
</property>
```

The default mapred parameters for Direct Shuffle are as follows:

```
<property>
    <name>mapreduce.job.shuffle.provider.services</name>
    <value>mapr_direct_shuffle</value>
</property>
<property>
    <name>mapreduce.job.reduce.shuffle.consumer.plugin.class</name>
    <value>org.apache.hadoop.mapreduce.task.reduce.DirectShuffle</value>
</property>
<property>
    <name>mapreduce.job.map.output.collector.class</name>
    <value>org.apache.hadoop.mapred.MapRFsOutputBuffer</value>
</property>
<property>
    <name>mapred.ifile.outputstream</name>
    <value>org.apache.hadoop.mapred.MapRIFileOutputStream</value>
</property>
<property>
    <name>mapred.ifile.inputstream</name>
    <value>org.apache.hadoop.mapred.MapRIFileInputStream</value>
</property>
<property>
    <name>mapred.local.mapoutput</name>
    <value>false</value>
</property>
<property>
    <name>mapreduce.task.local.output.class</name>
    <value>org.apache.hadoop.mapred.MapRFsOutputFile</value>
</property>
```

Apache Shuffle on YARN

You can disable Direct Shuffle and enable Apache Shuffle by modifying the configuration options in the `yarn-site.xml` and `mapred-site.xml` files. This page describes how to configure Apache Shuffle for MapReduce applications.

last-updated: Apr 11, 2022

The shuffling phase in Hadoop is the process of transferring mappers intermediate output to the reducers. Direct shuffle increases the load on MapR File System disks. You can enable the Apache Shuffle to reduce the load on MapR File System disks.

Configuration for Apache Shuffle

Add the following property to `yarn-site.xml` file:

```
<property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
</property>
```

Add the following properties to `mapred-site.xml` file:

```
 <property>
    <name>mapreduce.job.shuffle.provider.services</name>
    <value>mapreduce_shuffle</value>
</property>
<property>
    <name>mapreduce.job.reduce.shuffle.consumer.plugin.class</name>
    <value>org.apache.hadoop.mapreduce.task.reduce.Shuffle</value>
</property>
<property>
    <name>mapreduce.job.map.output.collector.class</name>
    <value>org.apache.hadoop.mapred.MapTask$MapOutputBuffer</value>
</property>
<property>
    <name>mapred.ifile.outputstream</name>
    <value>org.apache.hadoop.mapred.IFileOutputStream</value>
</property>
<property>
    <name>mapred.ifile.inputstream</name>
    <value>org.apache.hadoop.mapred.IFileInputStream</value>
</property>
<property>
    <name>mapred.local.mapoutput</name>
    <value>true</value>
</property>
<property>
    <name>mapreduce.task.local.output.class</name>
    <value>org.apache.hadoop.mapred.YarnOutputFiles</value>
</property>
```

Logging Options on YARN

Describes the logging options that are available on YARN.

For YARN applications, there are various logging options to choose from based on the MapR version and the types of applications that you run. In 4.0.2 and later versions, you have the following logging options:

- For MapReduce version 2 (MRv2) applications, the default logging option is to log files on the local filesystem. However, central logging and YARN log aggregation are also available.
- For non-MapReduce applications, the default logging option is to log files on the local filesystem. However,YARN log aggregation is also available.

Centralized Logging for MRv2	Centralized logging provides an application-centric view of all the log files generated by NodeManager nodes throughout the cluster. It enables users to gain a complete picture of application execution by having

	all the logs available in a single directory, without having to navigate from node to node. The MapReduce program generates three types of log output: • Standard output stream: captured in the `stdout` file • Standard error stream: captured in the `stderr` file • Log4j logs: captured in the `syslog` file Centralized logs are available cluster-wide as they are written to the following local volume on the MapR filesystem: / `var/mapr/local/<NodeManager node>/logs/yarn/userlogs` Since the log files are stored in a local volume directory that is associated with each NodeManager node, you run the `maprcli job linklogs` command to create symbolic links for all the logs in a single directory. You can then use tools such as `grep` and `awk` to analyze them from an NFS mount point. You can also view the entire set of logs for a particular application using the HistoryServer UI.
YARN Log Aggregation	The YARN log aggregation option aggregates logs from the local filesystem and moves log files for completed applications from the local filesystem to the MapR filesystem. This allows users to view the entire set of logs for a particular application using the HistoryServer UI or by running the `yarn logs` command.

Support for ADLS
Starting with MapR 6.1, you can use Azure Data Lake Store (ADLS) as a data source or destination for all applications.

Prerequisites for Using ADLS
Setting up Azure Data Lake Store (ADLS) on the Azure portal enables you to access ADLS from any application.

- Create an account on the Azure portal.
- Create an Azure Data Lake Store (get started with Azure Data Lake Storage).

Authenticating ADLS Account
To access data stored in Azure Data Lake Store (ADLS), you must first authenticate your ADLS account using your ADLS credentials.

1. Obtain the following properties from your Azure application:
   - `dfs.adls.oauth2.access.token.provider.type`

     `ClientCredential`, `Refresh Tokens`, or `Client Keys` to obtain the authentication type.
   - `dfs.adls.oauth2.client.id`

     Create an Azure Active Directory application and get your application ID and authentication key.

- `dfs.adls.oauth2.refresh.url`

  Navigate to Azure Active Directory and click on `Endpoints`. Use the `OAUTH 2.0 TOKEN ENDPOINT` value.

- `dfs.adls.oauth2.credential`

  Obtain the access token key value from `App Registrations` in your Azure account.

2. Add the properties obtained in step 1 to the `core-site.xml` file:

```
<!--ADL-->
<property>
   <name>dfs.adls.oauth2.access.token.provider.type</name>
   <value>ClientCredential</value>
</property>

<property>
   <name>dfs.adls.oauth2.client.id</name>
   <value>f377fab9-c0a3-4531-a1c9-77345105</value>
</property>

<property>
   <name>dfs.adls.oauth2.refresh.url</name>
   <value>https://login.microsoftonline.com/25735fb/oauth2/token</
value>
</property>

<property>
   <name>dfs.adls.oauth2.credential</name>
   <value>WTkn4xS0ISsqyzo4R6bu/OW2oPyGNMzWRw/d2z2CGiw=</value>
</property>
```

   Note: The `core-site.xml` file can be overwritten using the command line. You can also specify these properties at runtime. The syntax for overwriting ADLS properties at runtime using the command line is as follows:

```
yarn jar /opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/mapreduce/
hadoop-mapreduce-examples-2.7.0-mapr-1710-SNAPSHOT.jar wordcount
-Ddfs.adls.oauth2.access.token.provider.type=ClientCredential
-Ddfs.adls.oauth2.client.id=f377fab9-c0a3-4531-a1c9-77345105
-Ddfs.adls.oauth2.refresh.url=https://login.microsoftonline.com/
25735fb/oauth2/token
-Ddfs.adls.oauth2.credential=WTkn4xS0ISsqyzo4R6bu/OW2oPyGNMzWRw/
d2z2CGiw= adl://testhue.azuredatalakestore.net/some_folder/testfile
adl://testhue.azuredatalakestore.net/some_folder/wordcountout
```

   To provide your ADLS credentials securely, see Securely Providing ADLS Credentials on page 42.

3. Provide your application with file access.

4. For secure clusters, MapR-SASL (Simple Authentication and Security Layer), and Kerberos, import the required CA certificate.

- Open source documentation
- Azure documentation
- Azure documentation on authorization and access control

last-updated: Apr 11, 2022

-

Securely Providing ADLS Credentials
You can provide your ADLS credentials securely by hiding the open, readable configuration on the command line using the Hadoop credential provider.

1. Generate a `jceks` file for ADLS authorization:

```
hadoop credential create dfs.adls.oauth2.client.id -provider jceks://
hdfs/user/USER_NAME/adlskeyfile.jceks -value client ID
hadoop credential create dfs.adls.oauth2.credential -provider jceks://
hdfs/user/USER_NAME/adlskeyfile.jceks -value client secret
hadoop credential create dfs.adls.oauth2.refresh.url -provider jceks://
hdfs/user/USER_NAME/adlskeyfile.jceks -value refresh URL
```

2. Run the `DistCp` example using the `jceks` file:

```
hadoop distcp
[-D hadoop.security.credential.provider.path=localjceks://hdfs/user/
USER_NAME/adlskeyfile.jceks]
hdfs://<NameNode Hostname>:9001/user/foo/007020615
adl://<Account Name>.azuredatalakestore.net/testDir/
```

3. Configure the `core-site.xml` file to use the `jceks` file:

```
<property>
  <name>hadoop.security.credential.provider.path</name>
  <value>localjceks://hdfs/user/USER_NAME/adlskeyfile.jceks</value>
  <description>Path to interrogate for protected credentials.</
description>
</property>
```

Using ADLS for Data Input or Output
You can use Azure Data Lake Store (ADLS) as a source or destination for your application data.

For general information about the features of ADLS, refer to the Azure Data Lake Store documentation.

For information about configuring ADLS as storage for a Hadoop cluster, refer to the official Apache documentation.

The Azure Data Lake Storage access path syntax is:

```
adl://<Account Name>.azuredatalakestore.net/
```

You can use ADLS the same way as you use MapR File System, substituting an `adl` scheme instead of `maprfs`, `hdfs`, `webhdfs`, and so on.

1. Create a directory and read data:

```
[mapr@node4 ~]$ hadoop fs -mkdir adl://<username>.azuredatalakestore.net/
testdir

[mapr@node4 ~]$ hadoop fs -ls adl://<username>.azuredatalakestore.net/

Found 1 items
drwxr-xr-x - 9d3f4f74-8337-4dae-ad77-f63459438553
331c9f66-6875-4e13-a74f-458dd23e4bde 0 2018-04-16 09:09
adl://<username>.azuredatalakestore.net/testdir
```

2. Put data into ADLS from your local MapR File System:

```
[mapr@node4 ~]$ hadoop fs -put testfile adl://
<username>.azuredatalakestore.net/testdir

[mapr@node4 ~]$ hadoop fs -ls adl://<username>.azuredatalakestore.net/
testdir

Found 1 itemsrw-rr- 1 9d3f4f74-8337-4dae-ad77-f63459438553
331c9f66-6875-4e13-a74f-458dd23e4bde 0 2018-04-16 09:10
adl://<username>.azuredatalakestore.net/testdir/testfile
```

3. Delete data from ADLS:

```
[mapr@node4 ~]$ hadoop fs -rm -r adl://<username>.azuredatalakestore.net/
testdir

[mapr@node4 ~]$ hadoop fs -ls adl://<username>.azuredatalakestore.net/
```

4. Run YARN jobs with your input and output stored in ADLS:

```
yarn jar /opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/mapreduce/
hadoop-mapreduce-examples-2.7.0-mapr-1710-SNAPSHOT.jar wordcount
adl://<username>.azuredatalakestore.net/testdir/testfile adl://
<username>.azuredatalakestore.net/wordcountout
```

Deleting Data from ADLS

You can delete your data from Azure Data Lake Store (ADLS).

- To delete data from ADLS:

```
[mapr@node4 ~]$ hadoop fs -rm -r adl://<username>.azuredatalakestore.net/
testdir

[mapr@node4 ~]$ hadoop fs -ls adl://<username>.azuredatalakestore.net/
```

List of YARN Enhancements for MapR 6.0.1

MapR 6.0.1 runs the 2.7.0 version of Hadoop. A number of new features were incorporated into the version of Hadoop that MapR Converged Data Platform 6.0.1 uses.

You can enable or disable additional debug logs. See yarn debugcontrol on page 5352

The aws-java-sdk version for hadoop was updated to 1.11.199

Maven and MapR

This section discusses topics associated with Maven and MapR.

Maven Artifacts for MapR

Maven artifacts can be used for dependency management when developing applications based on the MapR platform.

You can access the MapR Maven repository by browsing Nexus or as follows:

```
<repositories>
  <repository> <id>mapr-releases</id>
```

```
    <url>https://repository.mapr.com/maven/</url>
    <snapshots><enabled>false</enabled></snapshots>
    <releases><enabled>true</enabled></releases>
  </repository>
</repositories>
```

Table

Group Id	Artifact Id	Version	Maven Coordinate
com.mapr.util	baseutils	6.1.0-mapr	`<dependency> <groupId>com.mapr.util</groupId> <artifactId>baseutils</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.util	central-logging	6.1.0-mapr	`<dependency> <groupId>com.mapr.util</groupId> <artifactId>central-logging</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.cldb	cldb	6.1.0-mapr	`<dependency> <groupId>com.mapr.cldb</groupId> <artifactId>cldb</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.cliframework	cliframework	6.1.0-mapr	`<dependency> <groupId>com.mapr.cliframework</groupId> <artifactId>cliframework</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.external	external	6.1.0-mapr	`<dependency> <groupId>com.mapr.external</groupId> <artifactId>external</artifactId> <version>6.1.0-mapr</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
com.mapr.gateway	gateway	6.1.0-mapr	`<dependency> <groupId>com.mapr.gateway</groupId> <artifactId>gateway</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.hadoop	hadoop2	6.1.0-mapr	`<dependency> <groupId>com.mapr.hadoop</groupId> <artifactId>hadoop2</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.fs	kvstore	6.1.0-mapr	`<dependency> <groupId>com.mapr.fs</groupId> <artifactId>kvstore</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.fs	libprotodefs	6.1.0-mapr	`<dependency> <groupId>com.mapr.fs</groupId> <artifactId>libprotodefs</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.fs	libprotodefs-full	6.1.0-mapr	`<dependency> <groupId>com.mapr.fs</groupId> <artifactId>libprotodefs-full</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.admin	mapr-apiserver	6.1.0-mapr	`<dependency> <groupId>com.mapr.admin</groupId> <artifactId>mapr-apiserver</artifactId> <version>6.1.0-mapr</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
com.mapr.fs	maprbuildversion	6.1.0-mapr	`<dependency> <groupId>com.mapr.fs</groupId> <artifactId>maprbuildversion</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.cli	maprcli	6.1.0-mapr	`<dependency> <groupId>com.mapr.cli</groupId> <artifactId>maprcli</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr	mapr-client-security	6.1.0-mapr	`<dependency> <groupId>com.mapr</groupId> <artifactId>mapr-client-security</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.db	maprdb	6.1.0-mapr	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.db	maprdb-cdc	6.1.0-mapr	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-cdc</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.db	maprdb-java	6.1.0-mapr	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-java</artifactId> <version>6.1.0-mapr</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
com.mapr.db	maprdb-mapreduce	6.1.0-mapr	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-mapreduce</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.db	maprdb-parent	6.1.0-mapr	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-parent</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.db	maprdb-shell	6.1.0-mapr	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-shell</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.db	maprdb-spark	2.3.1-mapr-1808	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-spark</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
com.mapr.hadoop	maprfs	6.1.0-mapr	`<dependency> <groupId>com.mapr.hadoop</groupId> <artifactId>maprfs</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.hadoop	maprfs-core	6.1.0-mapr	`<dependency> <groupId>com.mapr.hadoop</groupId> <artifactId>maprfs-core</artifactId> <version>6.1.0-mapr</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
com.mapr.hadoop	maprfs-diagnostic-tools	6.1.0-mapr	`<dependency> <groupId>com.mapr.hadoop</groupId> <artifactId>maprfs-diagnostic-tools</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.hadoop	maprfs-jni	6.1.0-mapr	`<dependency> <groupId>com.mapr.hadoop</groupId> <artifactId>maprfs-jni</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.fs	mapr-hbase	6.1.0-mapr	`<dependency> <groupId>com.mapr.fs</groupId> <artifactId>mapr-hbase</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr	mapr-java-utils	6.1.0-mapr	`<dependency> <groupId>com.mapr</groupId> <artifactId>mapr-java-utils</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.fs.native	mapr-mac-x86_64	6.1.0-mapr	`<dependency> <groupId>com.mapr.fs.native</groupId> <artifactId>mapr-mac-x86_64</artifactId> <version>6.1.0-mapr</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
com.mapr.fs.native-stubjni	mapr-mac-x86_64	6.1.0-mapr	`<dependency> <groupId>com.mapr.fs.native-stubjni</groupId> <artifactId>mapr-mac-x86_64</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.ojai	mapr-ojai-driver	6.1.0-mapr	`<dependency> <groupId>com.mapr.ojai</groupId> <artifactId>mapr-ojai-driver</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr	mapr-release	6.1.0-mapr	`<dependency> <groupId>com.mapr</groupId> <artifactId>mapr-release</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.fs.native	mapr-rhel-x86_64	6.1.0-mapr	`<dependency> <groupId>com.mapr.fs.native</groupId> <artifactId>mapr-rhel-x86_64</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.fs.native-stubjni	mapr-rhel-x86_64	6.1.0-mapr	`<dependency> <groupId>com.mapr.fs.native-stubjni</groupId> <artifactId>mapr-rhel-x86_64</artifactId> <version>6.1.0-mapr</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
com.mapr	mapr-root	6.1.0-mapr	`<dependency> <groupId>com.mapr</groupId> <artifactId>mapr-root</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.security	mapr-security-web	6.1.0-mapr	`<dependency> <groupId>com.mapr.security</groupId> <artifactId>mapr-security-web</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.streams	mapr-streams	6.1.0-mapr	`<dependency> <groupId>com.mapr.streams</groupId> <artifactId>mapr-streams</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.streams	mapr-streams-mapreduce	6.1.0-mapr	`<dependency> <groupId>com.mapr.streams</groupId> <artifactId>mapr-streams-mapreduce</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr	mapr-test-annotations	6.1.0-mapr	`<dependency> <groupId>com.mapr</groupId> <artifactId>mapr-test-annotations</artifactId> <version>6.1.0-mapr</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
com.mapr.tools	mapr-tools	6.1.0-mapr	`<dependency> <groupId>com.mapr.tools</groupId> <artifactId>mapr-tools</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.fs.native	mapr-ubuntu-x86_64	6.1.0-mapr	`<dependency> <groupId>com.mapr.fs.native</groupId> <artifactId>mapr-ubuntu-x86_64</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.fs.native-stubjni	mapr-ubuntu-x86_64	6.1.0-mapr	`<dependency> <groupId>com.mapr.fs.native-stubjni</groupId> <artifactId>mapr-ubuntu-x86_64</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.util	maprutil	6.1.0-mapr	`<dependency> <groupId>com.mapr.util</groupId> <artifactId>maprutil</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.fs.native	mapr-windows-x86_64	6.1.0-mapr	`<dependency> <groupId>com.mapr.fs.native</groupId> <artifactId>mapr-windows-x86_64</artifactId> <version>6.1.0-mapr</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
com.mapr.fs.native-stubjni	mapr-windows-x86_64	6.1.0-mapr	`<dependency> <groupId>com.mapr.fs.native-stubjni</groupId> <artifactId>mapr-windows-x86_64</artifactId> <version>6.1.0-mapr</version> </dependency>`
com.mapr.db	ycsb-driver	6.1.0-mapr	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>ycsb-driver</artifactId> <version>6.1.0-mapr</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-annotations	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-annotations</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-ant	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-ant</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-archives	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-archives</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-assemblies	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-assemblies</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-auth	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-auth</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-aws	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-aws</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-azure	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-azure</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-azure-datalake	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-azure-datalake</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-client	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-client</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-common	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-common</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-datajoin	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-datajoin</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-distcp	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-distcp</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-dist-osx	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-dist-osx</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-dist-redhat	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-dist-redhat</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-dist-ubuntu	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-dist-ubuntu</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-dist-windows	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-dist-windows</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-extras	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-extras</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-gridmix	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-gridmix</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-hdfs	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-hdfs</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop.contrib	hadoop-hdfs-bkjournal	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop.contrib</groupId> <artifactId>hadoop-hdfs-bkjournal</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-hdfs-httpfs	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-hdfs-httpfs</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-hdfs-nfs	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-hdfs-nfs</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-hdfs-sources-osx	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-hdfs-sources-osx</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-hdfs-sources-redhat	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-hdfs-sources-redhat</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-hdfs-sources-ubuntu	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-hdfs-sources-ubuntu</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-hdfs-sources-windows	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-hdfs-sources-windows</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-kms	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-kms</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-main	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-main</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-client	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-mapreduce-client-app	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-app</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-client-common	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-common</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-client-contrib	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-contrib</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-client-core	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-core</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-client-hs	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-hs</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-mapreduce-client-hs-plugins	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-hs-plugins</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-client-jobclient	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-jobclient</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-client-shuffle	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-shuffle</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-examples	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-examples</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-maven-plugins	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-maven-plugins</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-minicluster	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-minicluster</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-minikdc	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-minikdc</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-nfs	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-nfs</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-openstack	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-openstack</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-pipes	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-pipes</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-project	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-project</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-project-dist	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-project-dist</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-rumen	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-rumen</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-sls	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-sls</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-streaming	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-streaming</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-yarn	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-yarn-api	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-api</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-yarn-applications	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-applications</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-yarn-applications-distributedshell	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-applications-distributedshell</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-yarn-applications-unmanaged-am-launcher	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-applications-unmanaged-am-launcher</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-yarn-client	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-client</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-yarn-common	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-common</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-yarn-project	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-project</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-yarn-registry	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-registry</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-yarn-server	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-server</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-yarn-server-applicationhistoryservice	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-server-applicationhistoryservice</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-yarn-server-common	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-server-common</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-yarn-server-nodemanager	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-server-nodemanager</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-yarn-server-resourcemanager	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-server-resourcemanager</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-yarn-server-sharedcachemanager	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-server-sharedcachemanager</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-yarn-server-tests	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-server-tests</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`
org.apache.hadoop	hadoop-yarn-server-web-proxy	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-server-web-proxy</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-yarn-site	2.7.0-mapr-1808	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-site</artifactId> <version>2.7.0-mapr-1808</version> </dependency>`

Maven Artifacts for EEP 8.1.0

Listed are all Maven artifacts for EEP 8.1.0 components.

Table

com.mapr.db	maprdb-spark_2.12	3.2.0.0-eep-810 Browse	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-spark_2.12</artifactId> <version>3.2.0.0-eep-810</version> </dependency>`

Table

org.apache.drill.contrib	drill-auth-mechanism-maprsasl	1.16.1.400-eep-810 Browse	`<dependency> <groupId>org.apache.drill.contrib</groupId> <artifactId>drill-auth-mechanism-maprsasl</artifactId> <version>1.16.1.400-eep-810</version> </dependency>`
org.apache.drill	drill-client	1.16.1.400-eep-810 Browse	`<dependency> <groupId>org.apache.drill</groupId> <artifactId>drill-client</artifactId> <version>1.16.1.400-eep-810</version> </dependency>`
org.apache.drill	drill-common	1.16.1.400-eep-810 Browse	`<dependency> <groupId>org.apache.drill</groupId> <artifactId>drill-common</artifactId> <version>1.16.1.400-eep-810</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.drill.tools	drill-fmpp-maven-plugin	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.tools</groupId>`
			`<artifactId>drill-fmpp-maven-plugin</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill.contrib	drill-format-ltsv	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-format-ltsv</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill.contrib	drill-format-mapr	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-format-mapr</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill.contrib	drill-format-syslog	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-format-syslog</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill.exec	drill-java-exec	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.exec</groupId>`
			`<artifactId>drill-java-exec</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill.exec	drill-jdbc	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.exec</groupId>`
			`<artifactId>drill-jdbc</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

Table (Continued)

org.apache.drill.exec	drill-jdbc-all	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.exec</groupId>`
			`<artifactId>drill-jdbc-all</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill.contrib	drill-jdbc-storage	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-jdbc-storage</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill.contrib	drill-kudu-storage	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-kudu-storage</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill	drill-logical	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill</groupId>`
			`<artifactId>drill-logical</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill.memory	drill-memory-base	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.memory</groupId>`
			`<artifactId>drill-memory-base</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill.contrib	drill-mongo-storage	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-mongo-storage</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.drill.contrib	drill-opentsdb-storage	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-opentsdb-storage</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill	drill-protocol	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill</groupId>`
			`<artifactId>drill-protocol</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill.exec	drill-rpc	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.exec</groupId>`
			`<artifactId>drill-rpc</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill.contrib	drill-storage-hbase	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-storage-hbase</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill.contrib	drill-storage-kafka	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-storage-kafka</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

org.apache.drill.contrib	drill-udfs	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-udfs</artifactId>`
			`<version>1.16.1.400-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.drill	drill-yarn	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill<`
			`/groupId>`
			`<artifactId>drill-yarn</`
			`artifactId>`
			`<version>1.16.1.400-eep-81`
			`0</version>`
			`</dependency>`

org.apache.drill.exec	vector	1.16.1.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.`
			`exec</groupId>`
			`<artifactId>vector</`
			`artifactId>`
			`<version>1.16.1.400-eep-81`
			`0</version>`
			`</dependency>`

Table

org.apache.hadoop	hadoop-annotations	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop`
			`</groupId>`
			`<artifactId>hadoop-annotat`
			`ions</artifactId>`
			`<version>2.7.6.200-eep-810`
			`</version>`
			`</dependency>`

org.apache.hadoop	hadoop-ant	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop`
			`</groupId>`
			`<artifactId>hadoop-ant</`
			`artifactId>`
			`<version>2.7.6.200-eep-810`
			`</version>`
			`</dependency>`

org.apache.hadoop	hadoop-archives	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop`
			`</groupId>`
			`<artifactId>hadoop-archive`
			`s</artifactId>`
			`<version>2.7.6.200-eep-810`
			`</version>`
			`</dependency>`

org.apache.hadoop	hadoop-assemblies	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop`
			`</groupId>`
			`<artifactId>hadoop-assembl`
			`ies</artifactId>`
			`<version>2.7.6.200-eep-810`
			`</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hadoop	hadoop-auth	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-auth</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-aws	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-aws</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-azure	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-azure</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-azure-datalake	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-azure-datalake</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-client	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-client</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-common	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-common</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

Table (Continued)

org.apache.hadoop	hadoop-datajoin	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-datajoin</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-distcp	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-distcp</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-extras	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-extras</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-gridmix	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-gridmix</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-hdfs	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop.contrib	hadoop-hdfs-bkjournal	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop.contrib</groupId>`
			`<artifactId>hadoop-hdfs-bkjournal</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hadoop	hadoop-hdfs-nfs	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs-nfs</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-hdfs-sources-redhat	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs-sources-redhat</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-hdfs-sources-ubuntu	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs-sources-ubuntu</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-app	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-app</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-common	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-common</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-contrib	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-contrib</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

Table (Continued)

org.apache.hadoop	hadoop-mapreduce-client-core	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-core</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-hs	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-hs</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-hs-plugins	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-hs-plugins</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-jobclient	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-jobclient</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-shuffle	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-shuffle</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-examples	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-examples</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hadoop	hadoop-maven-plugins	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-maven-plugins</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-minicluster	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-minicluster</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-minikdc	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-minikdc</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-nfs	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-nfs</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-openstack	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-openstack</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-rumen	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-rumen</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

Table (Continued)

org.apache.hadoop	hadoop-sls	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-sls</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-streaming	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-streaming</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-api	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-api</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-applications-distributedshell	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-applications-distributedshell</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-applications-unmanaged-am-launcher	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-applications-unmanaged-am-launcher</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-client	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-client</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hadoop	hadoop-yarn-common	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-common</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-registry	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-registry</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-applicationhistoryservice	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-applicationhistoryservice</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-common	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-common</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-nodemanager	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-nodemanager</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-resourcemanager	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-resourcemanager</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

Table (Continued)

org.apache.hadoop	hadoop-yarn-server-sharedcachemanager	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-sharedcachemanager</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-tests	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-tests</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-web-proxy	2.7.6.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-web-proxy</artifactId>`
			`<version>2.7.6.200-eep-810</version>`
			`</dependency>`

Table

org.apache.hbase	hbase-annotations	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-annotations</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-checkstyle	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-checkstyle</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-client	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hbase	hbase-client-project	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client-project</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-common	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-common</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-examples	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-examples</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-external-blockcache	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-external-blockcache</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-hadoop-compat	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-hadoop-compat</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-hadoop2-compat	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-hadoop2-compat</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

Table (Continued)

org.apache.hbase	hbase-hbtop	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-hbtop</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-it	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-it</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-metrics	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-metrics</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-metrics-api	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-metrics-api</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-prefix-tree	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-prefix-tree</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-procedure	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-procedure</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hbase	hbase-protocol	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-protocol</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-resource-bundle	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-resource-bundle</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-rest	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-rest</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-rsgroup	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-rsgroup</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-server	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-server</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-shaded-client	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-client</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

Table (Continued)

org.apache.hbase	hbase-shaded-client-project	1.4.13.200-eep-810 Browse	<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-client-project</artifactId> <version>1.4.13.200-eep-810</version> </dependency>
org.apache.hbase	hbase-shaded-guava	1.4.13.200-eep-810 Browse	<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-guava</artifactId> <version>1.4.13.200-eep-810</version> </dependency>
org.apache.hbase	hbase-shaded-htrace	1.4.13.200-eep-810 Browse	<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-htrace</artifactId> <version>1.4.13.200-eep-810</version> </dependency>
org.apache.hbase	hbase-shaded-server	1.4.13.200-eep-810 Browse	<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-server</artifactId> <version>1.4.13.200-eep-810</version> </dependency>
org.apache.hbase	hbase-shaded-testing-util	1.4.13.200-eep-810 Browse	<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-testing-util</artifactId> <version>1.4.13.200-eep-810</version> </dependency>
org.apache.hbase	hbase-shaded-testing-util-tester	1.4.13.200-eep-810 Browse	<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-testing-util-tester</artifactId> <version>1.4.13.200-eep-810</version> </dependency>

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hbase	hbase-shell	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shell</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-spark	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-spark</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-testing-util	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-testing-util</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

org.apache.hbase	hbase-thrift	1.4.13.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-thrift</artifactId>`
			`<version>1.4.13.200-eep-810</version>`
			`</dependency>`

Table

org.apache.hive	hive-accumulo-handler	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-beeline	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-beeline</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

Table (Continued)

org.apache.hive	hive-cli	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-cli</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-common	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-common</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-contrib	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-contrib</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-druid-handler	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-druid-handler</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-exec	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-exec</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-hbase-handler	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hive.hcatalog	hive-hcatalog-core	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

Table (Continued)

org.apache.hive	hive-jdbc-handler	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc-handler</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-llap-common	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-llap-tez	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hive	hive-maprdb-json-common	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-common</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-metastore	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-metastore</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-serde	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-serde</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-service	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-service-rpc	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service-rpc</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hive	hive-shims	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-shims</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-0.23	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.23</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-common	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	hive-vector-code-gen	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-vector-code-gen</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hive.hcatalog	hive-webhcat	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive	mapr-log4j-slf4j-impl	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>mapr-log4j-slf4j-impl</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

Table (Continued)

org.apache.hive	spark-client	2.3.9.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.3.9.0-eep-810</version>`
			`</dependency>`

Table

com.mapr.kafka	kafka-eventstreams	0.1.0.100-eep-810
		Browse
			`<dependency>`
			`<groupId>com.mapr.kafka</groupId>`
			`<artifactId>kafka-eventstreams</artifactId>`
			`<version>0.1.0.100-eep-810</version>`
			`</dependency>`

Table

org.apache.kafka	connect-api	2.6.1.100-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-api</artifactId>`
			`<version>2.6.1.100-eep-810</version>`
			`</dependency>`

org.apache.kafka	connect-json	2.6.1.100-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-json</artifactId>`
			`<version>2.6.1.100-eep-810</version>`
			`</dependency>`

org.apache.kafka	connect-runtime	2.6.1.100-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-runtime</artifactId>`
			`<version>2.6.1.100-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.kafka	connect-transforms	2.6.1.100-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-transforms</artifactId>`
			`<version>2.6.1.100-eep-810</version>`
			`</dependency>`

org.apache.kafka	kafka-clients	2.6.1.100-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-clients</artifactId>`
			`<version>2.6.1.100-eep-810</version>`
			`</dependency>`

org.apache.kafka	kafka-log4j-appender	2.6.1.100-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-log4j-appender</artifactId>`
			`<version>2.6.1.100-eep-810</version>`
			`</dependency>`

org.apache.kafka	kafka-streams	2.6.1.100-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-streams</artifactId>`
			`<version>2.6.1.100-eep-810</version>`
			`</dependency>`

org.apache.kafka	kafka-streams-test-utils	2.6.1.100-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-streams-test-utils</artifactId>`
			`<version>2.6.1.100-eep-810</version>`
			`</dependency>`

org.apache.kafka	kafka-tools	2.6.1.100-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-tools</artifactId>`
			`<version>2.6.1.100-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.kafka	kafka_2.12	2.6.1.100-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka<`
			`/groupId>`
			`<artifactId>kafka_2.12</`
			`artifactId>`
			`<version>2.6.1.100-eep-810`
			`</version>`
			`</dependency>`

org.apache.kafka	kafka_2.13	2.6.1.100-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka<`
			`/groupId>`
			`<artifactId>kafka_2.13</`
			`artifactId>`
			`<version>2.6.1.100-eep-810`
			`</version>`
			`</dependency>`

org.apache.kafka	mapr-eco-tools	2.6.1.100-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka<`
			`/groupId>`
			`<artifactId>mapr-eco-tools`
			`</artifactId>`
			`<version>2.6.1.100-eep-810`
			`</version>`
			`</dependency>`

Table

org.apache.oozie	oozie-client	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie<`
			`/groupId>`
			`<artifactId>oozie-client</`
			`artifactId>`
			`<version>5.2.1.200-eep-810`
			`</version>`
			`</dependency>`

org.apache.oozie	oozie-core	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie<`
			`/groupId>`
			`<artifactId>oozie-core</`
			`artifactId>`
			`<version>5.2.1.200-eep-810`
			`</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie<`
			`/groupId>`
			`<artifactId>oozie-examples`
			`</artifactId>`
			`<version>5.2.1.200-eep-810`
			`</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.oozie	oozie-fluent-job-api	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-fluent-job-api</artifactId>`
			`<version>5.2.1.200-eep-810</version>`
			`</dependency>`

org.apache.oozie	oozie-fluent-job-client	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-fluent-job-client</artifactId>`
			`<version>5.2.1.200-eep-810</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>5.2.1.200-eep-810</version>`
			`</dependency>`

org.apache.oozie	oozie-server	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-server</artifactId>`
			`<version>5.2.1.200-eep-810</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-distcp	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>5.2.1.200-eep-810</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-git	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-git</artifactId>`
			`<version>5.2.1.200-eep-810</version>`
			`</dependency>`

Table (Continued)

org.apache.oozie	oozie-sharelib-hcatalog	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>5.2.1.200-eep-810</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>5.2.1.200-eep-810</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>5.2.1.200-eep-810</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>5.2.1.200-eep-810</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>5.2.1.200-eep-810</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>5.2.1.200-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.oozie	oozie-tools	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>5.2.1.200-eep-810</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	5.2.1.200-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>5.2.1.200-eep-810</version>`
			`</dependency>`

Table

org.apache.spark	classpath-filter_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-avro_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-avro_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-core_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

Table (Continued)

org.apache.spark	spark-graphx_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-hive_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-kvstore_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-kvstore_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-mesos_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.spark	spark-mllib-local_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.spark	spark-sketch_2.12	3.2.0.0-eep-810 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.12</artifactId> <version>3.2.0.0-eep-810</version> </dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.12	3.2.0.0-eep-810 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.12</artifactId> <version>3.2.0.0-eep-810</version> </dependency>`
org.apache.spark	spark-sql_2.12	3.2.0.0-eep-810 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.12</artifactId> <version>3.2.0.0-eep-810</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.12	3.2.0.0-eep-810 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.12</artifactId> <version>3.2.0.0-eep-810</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.12	3.2.0.0-eep-810 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.12</artifactId> <version>3.2.0.0-eep-810</version> </dependency>`
org.apache.spark	spark-streaming_2.12	3.2.0.0-eep-810 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.12</artifactId> <version>3.2.0.0-eep-810</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.spark	spark-tags_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-token-provider-kafka-0-10_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-token-provider-kafka-0-10_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.12	3.2.0.0-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.12</artifactId>`
			`<version>3.2.0.0-eep-810</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.9.2.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.9.2.400-eep-810</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	hadoop-shim-2.7	0.9.2.400-eep-810 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim-2.7</artifactId> <version>0.9.2.400-eep-810</version> </dependency>`
org.apache.tez.conftool	mapr-tez-conf-tool	0.9.2.400-eep-810 Browse	`<dependency> <groupId>org.apache.tez.conftool</groupId> <artifactId>mapr-tez-conf-tool</artifactId> <version>0.9.2.400-eep-810</version> </dependency>`
org.apache.tez	tez-api	0.9.2.400-eep-810 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.9.2.400-eep-810</version> </dependency>`
org.apache.tez	tez-aux-services	0.9.2.400-eep-810 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-aux-services</artifactId> <version>0.9.2.400-eep-810</version> </dependency>`
org.apache.tez	tez-build-tools	0.9.2.400-eep-810 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-build-tools</artifactId> <version>0.9.2.400-eep-810</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-common	0.9.2.400-eep-810 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.9.2.400-eep-810</version> </dependency>`
org.apache.tez	tez-dag	0.9.2.400-eep-810 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.9.2.400-eep-810</version> </dependency>`
org.apache.tez	tez-examples	0.9.2.400-eep-810 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.9.2.400-eep-810</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.9.2.400-eep-810 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.9.2.400-eep-810</version> </dependency>`
org.apache.tez	tez-job-analyzer	0.9.2.400-eep-810 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-job-analyzer</artifactId> <version>0.9.2.400-eep-810</version> </dependency>`
org.apache.tez	tez-mapreduce	0.9.2.400-eep-810 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-mapreduce</artifactId> <version>0.9.2.400-eep-810</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-protobuf-history-plugin	0.9.2.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-protobuf-history-plugin</artifactId>`
			`<version>0.9.2.400-eep-810</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.9.2.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.9.2.400-eep-810</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.9.2.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.9.2.400-eep-810</version>`
			`</dependency>`

org.apache.tez	tez-tests	0.9.2.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-tests</artifactId>`
			`<version>0.9.2.400-eep-810</version>`
			`</dependency>`

org.apache.tez	tez-ui	0.9.2.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui</artifactId>`
			`<version>0.9.2.400-eep-810</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-yarn-timeline-history	0.9.2.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.9.2.400-eep-810</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.2.400-eep-810
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.9.2.400-eep-810</version>`
			`</dependency>`

Maven Artifacts for EEP 8.0.0

Listed are all Maven artifacts for EEP 8.0.0 components.

Table

com.mapr.db	maprdb-spark_2.12	3.1.2.0-eep-800
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark_2.12</artifactId>`
			`<version>3.1.2.0-eep-800</version>`
			`</dependency>`

Table

org.apache.hadoop	hadoop-annotations	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-annotations</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-ant	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-ant</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hadoop	hadoop-archives	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-archives</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-assemblies	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-assemblies</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-auth	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-auth</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-aws	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-aws</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-azure	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-azure</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-azure-datalake	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-azure-datalake</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hadoop	hadoop-client	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-client</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-common	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-common</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-datajoin	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-datajoin</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-distcp	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-distcp</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-extras	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-extras</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-gridmix	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-gridmix</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

Table (Continued)

org.apache.hadoop	hadoop-hdfs	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop.contrib	hadoop-hdfs-bkjournal	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop.contrib</groupId>`
			`<artifactId>hadoop-hdfs-bkjournal</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-hdfs-nfs	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs-nfs</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-hdfs-sources-redhat	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs-sources-redhat</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-hdfs-sources-ubuntu	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs-sources-ubuntu</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-app	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-app</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hadoop	hadoop-mapreduce-client-common	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-common</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-contrib	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-contrib</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-core	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-core</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-hs	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-hs</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-hs-plugins	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-hs-plugins</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-jobclient	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-jobclient</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hadoop	hadoop-mapreduce-client-shuffle	2.7.6.100-eep-800 Browse	<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-shuffle</artifactId> <version>2.7.6.100-eep-800</version> </dependency>
org.apache.hadoop	hadoop-mapreduce-examples	2.7.6.100-eep-800 Browse	<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-examples</artifactId> <version>2.7.6.100-eep-800</version> </dependency>
org.apache.hadoop	hadoop-maven-plugins	2.7.6.100-eep-800 Browse	<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-maven-plugins</artifactId> <version>2.7.6.100-eep-800</version> </dependency>
org.apache.hadoop	hadoop-minicluster	2.7.6.100-eep-800 Browse	<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-minicluster</artifactId> <version>2.7.6.100-eep-800</version> </dependency>
org.apache.hadoop	hadoop-minikdc	2.7.6.100-eep-800 Browse	<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-minikdc</artifactId> <version>2.7.6.100-eep-800</version> </dependency>
org.apache.hadoop	hadoop-nfs	2.7.6.100-eep-800 Browse	<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-nfs</artifactId> <version>2.7.6.100-eep-800</version> </dependency>

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hadoop	hadoop-openstack	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-openstack</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-rumen	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-rumen</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-sls	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-sls</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-streaming	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-streaming</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-api	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-api</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-applications-distributedshell	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-applications-distributedshell</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

Table (Continued)

org.apache.hadoop	hadoop-yarn-applications-unmanaged-am-launcher	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-applications-unmanaged-am-launcher</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-client	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-client</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-common	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-common</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-registry	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-registry</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-applicationhistoryservice	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-applicationhistoryservice</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-common	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-common</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hadoop	hadoop-yarn-server-nodemanager	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-nodemanager</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-resourcemanager	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-resourcemanager</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-sharedcachemanager	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-sharedcachemanager</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-tests	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-tests</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-web-proxy	2.7.6.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-web-proxy</artifactId>`
			`<version>2.7.6.100-eep-800</version>`
			`</dependency>`

Table

org.apache.kafka	connect-api	2.6.1.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-api</artifactId>`
			`<version>2.6.1.0-eep-800</version>`
			`</dependency>`

org.apache.kafka	connect-json	2.6.1.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-json</artifactId>`
			`<version>2.6.1.0-eep-800</version>`
			`</dependency>`

org.apache.kafka	connect-runtime	2.6.1.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-runtime</artifactId>`
			`<version>2.6.1.0-eep-800</version>`
			`</dependency>`

org.apache.kafka	connect-transforms	2.6.1.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-transforms</artifactId>`
			`<version>2.6.1.0-eep-800</version>`
			`</dependency>`

org.apache.kafka	kafka-clients	2.6.1.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-clients</artifactId>`
			`<version>2.6.1.0-eep-800</version>`
			`</dependency>`

org.apache.kafka	kafka-log4j-appender	2.6.1.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-log4j-appender</artifactId>`
			`<version>2.6.1.0-eep-800</version>`
			`</dependency>`

Table (Continued)

org.apache.kafka	kafka-streams	2.6.1.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka<`
			`/groupId>`
			`<artifactId>kafka-streams<`
			`/artifactId>`
			`<version>2.6.1.0-eep-800</`
			`version>`
			`</dependency>`

org.apache.kafka	kafka-tools	2.6.1.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka<`
			`/groupId>`
			`<artifactId>kafka-tools</`
			`artifactId>`
			`<version>2.6.1.0-eep-800</`
			`version>`
			`</dependency>`

org.apache.kafka	kafka_2.12	2.6.1.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka<`
			`/groupId>`
			`<artifactId>kafka_2.12</`
			`artifactId>`
			`<version>2.6.1.0-eep-800</`
			`version>`
			`</dependency>`

org.apache.kafka	kafka_2.13	2.6.1.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka<`
			`/groupId>`
			`<artifactId>kafka_2.13</`
			`artifactId>`
			`<version>2.6.1.0-eep-800</`
			`version>`
			`</dependency>`

org.apache.kafka	mapr-eco-tools	2.6.1.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka<`
			`/groupId>`
			`<artifactId>mapr-eco-tools`
			`</artifactId>`
			`<version>2.6.1.0-eep-800</`
			`version>`
			`</dependency>`

Table

org.apache.oozie	oozie-client	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie<`
			`/groupId>`
			`<artifactId>oozie-client</`
			`artifactId>`
			`<version>5.2.1.100-eep-800`
			`</version>`
			`</dependency>`

Table (Continued)

org.apache.oozie	oozie-core	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie	oozie-fluent-job-api	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-fluent-job-api</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie	oozie-fluent-job-client	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-fluent-job-client</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie	oozie-server	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-server</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.oozie	oozie-sharelib-distcp	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-git	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-git</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

Table (Continued)

org.apache.oozie	oozie-sharelib-pig	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	5.2.1.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>5.2.1.100-eep-800</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

org.apache.pig	pig	0.17.0.100-eep-800 Browse	<dependency> <groupId>org.apache.pig</groupId> <artifactId>pig</artifactId> <version>0.17.0.100-eep-800</version> </dependency>
org.apache.pig	piggybank	0.17.0.100-eep-800 Browse	<dependency> <groupId>org.apache.pig</groupId> <artifactId>piggybank</artifactId> <version>0.17.0.100-eep-800</version> </dependency>
org.apache.pig	pigsmoke	0.17.0.100-eep-800 Browse	<dependency> <groupId>org.apache.pig</groupId> <artifactId>pigsmoke</artifactId> <version>0.17.0.100-eep-800</version> </dependency>
org.apache.pig	pigunit	0.17.0.100-eep-800 Browse	<dependency> <groupId>org.apache.pig</groupId> <artifactId>pigunit</artifactId> <version>0.17.0.100-eep-800</version> </dependency>

Table

org.apache.spark	classpath-filter_2.12	3.1.2.0-eep-800 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>classpath-filter_2.12</artifactId> <version>3.1.2.0-eep-800</version> </dependency>
org.apache.spark	spark-avro_2.12	3.1.2.0-eep-800 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-avro_2.12</artifactId> <version>3.1.2.0-eep-800</version> </dependency>

Table (Continued)

org.apache.spark	spark-catalyst_2.12	3.1.2.0-eep-800 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-catalyst_2.12</artifactId> <version>3.1.2.0-eep-800</version> </dependency>`
org.apache.spark	spark-core_2.12	3.1.2.0-eep-800 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.12</artifactId> <version>3.1.2.0-eep-800</version> </dependency>`
org.apache.spark	spark-graphx_2.12	3.1.2.0-eep-800 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.12</artifactId> <version>3.1.2.0-eep-800</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.12	3.1.2.0-eep-800 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.12</artifactId> <version>3.1.2.0-eep-800</version> </dependency>`
org.apache.spark	spark-hive_2.12	3.1.2.0-eep-800 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.12</artifactId> <version>3.1.2.0-eep-800</version> </dependency>`
org.apache.spark	spark-kvstore_2.12	3.1.2.0-eep-800 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-kvstore_2.12</artifactId> <version>3.1.2.0-eep-800</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.spark	spark-launcher_2.12	3.1.2.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.12</artifactId>`
			`<version>3.1.2.0-eep-800</version>`
			`</dependency>`

org.apache.spark	spark-mesos_2.12	3.1.2.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.12</artifactId>`
			`<version>3.1.2.0-eep-800</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.12	3.1.2.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.12</artifactId>`
			`<version>3.1.2.0-eep-800</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.12	3.1.2.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.12</artifactId>`
			`<version>3.1.2.0-eep-800</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.12	3.1.2.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.12</artifactId>`
			`<version>3.1.2.0-eep-800</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.12	3.1.2.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.12</artifactId>`
			`<version>3.1.2.0-eep-800</version>`
			`</dependency>`

Table (Continued)

org.apache.spark	spark-network-yarn_2.12	3.1.2.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.12</artifactId>`
			`<version>3.1.2.0-eep-800</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.12	3.1.2.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.12</artifactId>`
			`<version>3.1.2.0-eep-800</version>`
			`</dependency>`

org.apache.spark	spark-sketch_2.12	3.1.2.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.12</artifactId>`
			`<version>3.1.2.0-eep-800</version>`
			`</dependency>`

org.apache.spark	spark-sql-kafka-0-10_2.12	3.1.2.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.12</artifactId>`
			`<version>3.1.2.0-eep-800</version>`
			`</dependency>`

org.apache.spark	spark-sql_2.12	3.1.2.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.12</artifactId>`
			`<version>3.1.2.0-eep-800</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.12	3.1.2.0-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.12</artifactId>`
			`<version>3.1.2.0-eep-800</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.spark	spark-streaming-kafka-0-10_2.12	3.1.2.0-eep-800 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.12</artifactId> <version>3.1.2.0-eep-800</version> </dependency>
org.apache.spark	spark-streaming_2.12	3.1.2.0-eep-800 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.12</artifactId> <version>3.1.2.0-eep-800</version> </dependency>
org.apache.spark	spark-tags_2.12	3.1.2.0-eep-800 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.12</artifactId> <version>3.1.2.0-eep-800</version> </dependency>
org.apache.spark	spark-token-provider-kafka-0-10_2.12	3.1.2.0-eep-800 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-token-provider-kafka-0-10_2.12</artifactId> <version>3.1.2.0-eep-800</version> </dependency>
org.apache.spark	spark-unsafe_2.12	3.1.2.0-eep-800 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-unsafe_2.12</artifactId> <version>3.1.2.0-eep-800</version> </dependency>
org.apache.spark	spark-yarn_2.12	3.1.2.0-eep-800 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-yarn_2.12</artifactId> <version>3.1.2.0-eep-800</version> </dependency>

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.7.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.7.100-eep-800</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.7.100-eep-800
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.7.100-eep-800</version>`
			`</dependency>`

Maven Artifacts for EEP 7.1.1

Maven artifacts for EEP 7.1.1 are unchanged from the Maven artifacts for EEP 7.1.0.

See Maven Artifacts for EEP 7.1.0 on page 4233.

Maven Artifacts for EEP 7.1.0

Listed are all Maven artifacts for EEP 7.1.0 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

com.mapr.db	maprdb-spark_2.12	2.4.7.100-mapr-710
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark_2.12</artifactId>`
			`<version>2.4.7.100-mapr-710</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-annotations	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-annotations</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-ant	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-ant</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`
org.apache.hadoop	hadoop-archives	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-archives</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`
org.apache.hadoop	hadoop-assemblies	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-assemblies</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`
org.apache.hadoop	hadoop-auth	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-auth</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`
org.apache.hadoop	hadoop-aws	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-aws</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`
org.apache.hadoop	hadoop-azure	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-azure</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-azure-datalake	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-azure-datalake</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`
org.apache.hadoop	hadoop-client	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-client</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`
org.apache.hadoop	hadoop-common	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-common</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`
org.apache.hadoop	hadoop-datajoin	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-datajoin</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`
org.apache.hadoop	hadoop-distcp	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-distcp</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`
org.apache.hadoop	hadoop-extras	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-extras</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-gridmix	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-gridmix</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`
org.apache.hadoop	hadoop-hdfs	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-hdfs</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`
org.apache.hadoop.contrib	hadoop-hdfs-bkjournal	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop.contrib</groupId> <artifactId>hadoop-hdfs-bkjournal</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`
org.apache.hadoop	hadoop-hdfs-nfs	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-hdfs-nfs</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`
org.apache.hadoop	hadoop-hdfs-sources-redhat	2.7.5.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-hdfs-sources-redhat</artifactId> <version>2.7.5.0-mapr-710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-hdfs-sources-ubuntu	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs-sources-ubuntu</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-app	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-app</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-common	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-common</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-contrib	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-contrib</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-core	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-core</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-mapreduce-client-hs	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-hs</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-hs-plugins	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-hs-plugins</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-jobclient	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-jobclient</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-shuffle	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-shuffle</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-examples	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-examples</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-maven-plugins	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-maven-plugins</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-minicluster	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-minicluster</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-minikdc	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-minikdc</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-nfs	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-nfs</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-openstack	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-openstack</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-rumen	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-rumen</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-sls	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-sls</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-streaming	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-streaming</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-api	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-api</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-applications-distributedshell	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-applications-distributedshell</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-yarn-applications-unmanaged-am-launcher	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-applications-unmanaged-am-launcher</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-client	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-client</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-common	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-common</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-registry	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-registry</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-applicationhistoryservice	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-applicationhistoryservice</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-yarn-server-common	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-common</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-nodemanager	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-nodemanager</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-resourcemanager	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-resourcemanager</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-sharedcachemanager	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-sharedcachemanager</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-tests	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-tests</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-yarn-server-web-proxy	2.7.5.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-web-proxy</artifactId>`
			`<version>2.7.5.0-mapr-710</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-annotations	1.4.13.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-annotations</artifactId>`
			`<version>1.4.13.0-mapr-710</version>`
			`</dependency>`

org.apache.hbase	hbase-checkstyle	1.4.13.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-checkstyle</artifactId>`
			`<version>1.4.13.0-mapr-710</version>`
			`</dependency>`

org.apache.hbase	hbase-client	1.4.13.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client</artifactId>`
			`<version>1.4.13.0-mapr-710</version>`
			`</dependency>`

org.apache.hbase	hbase-client-project	1.4.13.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client-project</artifactId>`
			`<version>1.4.13.0-mapr-710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-common	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-common</artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`
org.apache.hbase	hbase-examples	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-examples</artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`
org.apache.hbase	hbase-external-blockcache	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-external-blockcache</artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`
org.apache.hbase	hbase-hadoop-compat	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop-compat</artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`
org.apache.hbase	hbase-hadoop2-compat	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop2-compat</artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-hbtop	1.4.13.0-mapr-710 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-hbtop</artifactId><version>1.4.13.0-mapr-710</version></dependency>`
org.apache.hbase	hbase-it	1.4.13.0-mapr-710 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-it</artifactId><version>1.4.13.0-mapr-710</version></dependency>`
org.apache.hbase	hbase-metrics	1.4.13.0-mapr-710 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-metrics</artifactId><version>1.4.13.0-mapr-710</version></dependency>`
org.apache.hbase	hbase-metrics-api	1.4.13.0-mapr-710 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-metrics-api</artifactId><version>1.4.13.0-mapr-710</version></dependency>`
org.apache.hbase	hbase-prefix-tree	1.4.13.0-mapr-710 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-prefix-tree</artifactId><version>1.4.13.0-mapr-710</version></dependency>`
org.apache.hbase	hbase-procedure	1.4.13.0-mapr-710 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-procedure</artifactId><version>1.4.13.0-mapr-710</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-protocol	1.4.13.0-mapr-710 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-protocol</artifactId><version>1.4.13.0-mapr-710</version></dependency>`
org.apache.hbase	hbase-resource-bundle	1.4.13.0-mapr-710 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-resource-bundle</artifactId><version>1.4.13.0-mapr-710</version></dependency>`
org.apache.hbase	hbase-rest	1.4.13.0-mapr-710 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-rest</artifactId><version>1.4.13.0-mapr-710</version></dependency>`
org.apache.hbase	hbase-rsgroup	1.4.13.0-mapr-710 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-rsgroup</artifactId><version>1.4.13.0-mapr-710</version></dependency>`
org.apache.hbase	hbase-server	1.4.13.0-mapr-710 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-server</artifactId><version>1.4.13.0-mapr-710</version></dependency>`
org.apache.hbase	hbase-shaded-client	1.4.13.0-mapr-710 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-client</artifactId><version>1.4.13.0-mapr-710</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-shaded-client-project	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-client-project</artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`
org.apache.hbase	hbase-shaded-guava	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-guava</artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`
org.apache.hbase	hbase-shaded-htrace	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-htrace</artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`
org.apache.hbase	hbase-shaded-server	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-server</artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`
org.apache.hbase	hbase-shaded-testing-util	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-testing-util</artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-shaded-testing-util-tester	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-testing-util-tester</artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`
org.apache.hbase	hbase-shell	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shell</artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`
org.apache.hbase	hbase-spark	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-spark</artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`
org.apache.hbase	hbase-testing-util	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-testing-util</ artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`
org.apache.hbase	hbase-thrift	1.4.13.0-mapr-710 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-thrift</artifactId> <version>1.4.13.0-mapr-710</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-accumulo-handler</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`
org.apache.hive	hive-beeline	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`
org.apache.hive	hive-cli	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`
org.apache.hive	hive-common	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`
org.apache.hive	hive-contrib	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`
org.apache.hive	hive-druid-handler	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-druid-handler</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-exec	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-exec</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-hbase-handler	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-core	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-jdbc-handler	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc-handler</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-common	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`
org.apache.hive	hive-llap-ext-client	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-ext-client</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`
org.apache.hive	hive-llap-server	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-server</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`
org.apache.hive	hive-llap-tez	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`
org.apache.hive	hive-maprdb-json-common	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-common</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-maprdb-json-handler	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-metastore	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-metastore</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-serde	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-serde</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-service	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-service-rpc	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service-rpc</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-shims	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-shims</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.shims	hive-shims-0.23	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`
org.apache.hive.shims	hive-shims-common	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`
org.apache.hive.shims	hive-shims-scheduler	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-scheduler</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`
org.apache.hive	hive-testutils	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-testutils</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`
org.apache.hive	hive-vector-code-gen	2.3.8-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-vector-code-gen</artifactId> <version>2.3.8-mapr-2104</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-webhcat	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive	mapr-log4j-slf4j-impl	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>mapr-log4j-slf4j-impl</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

org.apache.hive	spark-client	2.3.8-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.3.8-mapr-2104</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.kafka	connect-api	2.1.1.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-api</artifactId>`
			`<version>2.1.1.200-mapr-710</version>`
			`</dependency>`

org.apache.kafka	connect-json	2.1.1.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-json</artifactId>`
			`<version>2.1.1.200-mapr-710</version>`
			`</dependency>`

org.apache.kafka	connect-runtime	2.1.1.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-runtime</artifactId>`
			`<version>2.1.1.200-mapr-710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.kafka	connect-transforms	2.1.1.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-transforms</artifactId>`
			`<version>2.1.1.200-mapr-710</version>`
			`</dependency>`

org.apache.kafka	kafka-clients	2.1.1.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-clients</artifactId>`
			`<version>2.1.1.200-mapr-710</version>`
			`</dependency>`

org.apache.kafka	kafka-log4j-appender	2.1.1.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-log4j-appender</artifactId>`
			`<version>2.1.1.200-mapr-710</version>`
			`</dependency>`

org.apache.kafka	kafka-streams	2.1.1.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-streams</artifactId>`
			`<version>2.1.1.200-mapr-710</version>`
			`</dependency>`

org.apache.kafka	kafka-tools	2.1.1.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-tools</artifactId>`
			`<version>2.1.1.200-mapr-710</version>`
			`</dependency>`

org.apache.kafka	kafka_2.11	2.1.1.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka_2.11</artifactId>`
			`<version>2.1.1.200-mapr-710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.kafka	kafka_2.12	2.1.1.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka_2.12</artifactId>`
			`<version>2.1.1.200-mapr-710</version>`
			`</dependency>`

org.apache.kafka	mapr-eco-tools	2.1.1.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>mapr-eco-tools</artifactId>`
			`<version>2.1.1.200-mapr-710</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

org.apache.oozie	oozie-core	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-fluent-job-api	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-fluent-job-api</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

org.apache.oozie	oozie-fluent-job-client	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-fluent-job-client</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

org.apache.oozie	oozie-server	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-server</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-distcp	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-git	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-git</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-pig	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	5.2.1.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>5.2.1.0-mapr-710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-webapp	5.2.1.0-mapr-710 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-webapp</artifactId> <version>5.2.1.0-mapr-710</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	classpath-filter_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>classpath-filter_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-avro_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-avro_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-catalyst_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-catalyst_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-core_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-graphx_2.12	2.4.7.100-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.12</artifactId>`
			`<version>2.4.7.100-mapr-710</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.12	2.4.7.100-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.12</artifactId>`
			`<version>2.4.7.100-mapr-710</version>`
			`</dependency>`

org.apache.spark	spark-hive_2.12	2.4.7.100-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.12</artifactId>`
			`<version>2.4.7.100-mapr-710</version>`
			`</dependency>`

org.apache.spark	spark-kvstore_2.12	2.4.7.100-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-kvstore_2.12</artifactId>`
			`<version>2.4.7.100-mapr-710</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.12	2.4.7.100-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.12</artifactId>`
			`<version>2.4.7.100-mapr-710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-mesos_2.12	2.4.7.100-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.12</artifactId>`
			`<version>2.4.7.100-mapr-710</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.12	2.4.7.100-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.12</artifactId>`
			`<version>2.4.7.100-mapr-710</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.12	2.4.7.100-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.12</artifactId>`
			`<version>2.4.7.100-mapr-710</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.12	2.4.7.100-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.12</artifactId>`
			`<version>2.4.7.100-mapr-710</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.12	2.4.7.100-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.12</artifactId>`
			`<version>2.4.7.100-mapr-710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-network-yarn_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-repl_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-sketch_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-sql_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-flume-assembly_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-streaming-flume_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-streaming_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`
org.apache.spark	spark-tags_2.12	2.4.7.100-mapr-710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.12</artifactId> <version>2.4.7.100-mapr-710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-unsafe_2.12	2.4.7.100-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.12</artifactId>`
			`<version>2.4.7.100-mapr-710</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.12	2.4.7.100-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.12</artifactId>`
			`<version>2.4.7.100-mapr-710</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.7.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.7.0-mapr-710</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.7.0-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.7.0-mapr-710</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.9.2.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.9.2.200-mapr-710</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim-2.7	0.9.2.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim-2.7</artifactId>`
			`<version>0.9.2.200-mapr-710</version>`
			`</dependency>`

org.apache.tez.conftool	mapr-tez-conf-tool	0.9.2.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez.conftool</groupId>`
			`<artifactId>mapr-tez-conf-tool</artifactId>`
			`<version>0.9.2.200-mapr-710</version>`
			`</dependency>`

org.apache.tez	tez-api	0.9.2.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-api</artifactId>`
			`<version>0.9.2.200-mapr-710</version>`
			`</dependency>`

org.apache.tez	tez-aux-services	0.9.2.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-aux-services</artifactId>`
			`<version>0.9.2.200-mapr-710</version>`
			`</dependency>`

org.apache.tez	tez-build-tools	0.9.2.200-mapr-710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-build-tools</artifactId>`
			`<version>0.9.2.200-mapr-710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-common	0.9.2.200-mapr-710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.9.2.200-mapr-710</version> </dependency>`
org.apache.tez	tez-dag	0.9.2.200-mapr-710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.9.2.200-mapr-710</version> </dependency>`
org.apache.tez	tez-examples	0.9.2.200-mapr-710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.9.2.200-mapr-710</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.9.2.200-mapr-710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.9.2.200-mapr-710</version> </dependency>`
org.apache.tez	tez-job-analyzer	0.9.2.200-mapr-710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-job-analyzer</artifactId> <version>0.9.2.200-mapr-710</version> </dependency>`
org.apache.tez	tez-mapreduce	0.9.2.200-mapr-710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-mapreduce</artifactId> <version>0.9.2.200-mapr-710</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-protobuf-history-plugin	0.9.2.200-mapr-710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-protobuf-history-plugin</artifactId> <version>0.9.2.200-mapr-710</version> </dependency>`
org.apache.tez	tez-runtime-internals	0.9.2.200-mapr-710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-internals</artifactId> <version>0.9.2.200-mapr-710</version> </dependency>`
org.apache.tez	tez-runtime-library	0.9.2.200-mapr-710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-library</artifactId> <version>0.9.2.200-mapr-710</version> </dependency>`
org.apache.tez	tez-tests	0.9.2.200-mapr-710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-tests</artifactId> <version>0.9.2.200-mapr-710</version> </dependency>`
org.apache.tez	tez-ui	0.9.2.200-mapr-710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui</artifactId> <version>0.9.2.200-mapr-710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-yarn-timeline-history	0.9.2.200-mapr-710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history</artifactId> <version>0.9.2.200-mapr-710</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.2.200-mapr-710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history-with-acls</artifactId> <version>0.9.2.200-mapr-710</version> </dependency>`

Maven Artifacts for EEP 7.0.1

Listed are all Maven artifacts for EEP 7.0.1 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate
com.mapr.db	maprdb-spark_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-spark_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-annotations	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-annotations</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-ant	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-ant</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

org.apache.hadoop	hadoop-archives	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-archives</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

org.apache.hadoop	hadoop-assemblies	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-assemblies</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

org.apache.hadoop	hadoop-auth	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-auth</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

org.apache.hadoop	hadoop-aws	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-aws</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

org.apache.hadoop	hadoop-azure	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-azure</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-azure-datalake	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-azure-datalake</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-client	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-client</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-common	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-common</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-datajoin	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-datajoin</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-distcp	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-distcp</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-extras	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-extras</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-gridmix	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-gridmix</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

org.apache.hadoop	hadoop-hdfs	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

org.apache.hadoop.contrib	hadoop-hdfs-bkjournal	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop.contrib</groupId>`
			`<artifactId>hadoop-hdfs-bkjournal</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

org.apache.hadoop	hadoop-hdfs-nfs	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs-nfs</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

org.apache.hadoop	hadoop-hdfs-sources-redhat	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs-sources-redhat</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-hdfs-sources-ubuntu	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-hdfs-sources-ubuntu</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-kms	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-kms</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-client-app	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-app</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-client-common	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-common</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-client-contrib	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-contrib</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-mapreduce-client-core	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-core</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-client-hs	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-hs</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-client-hs-plugins	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-hs-plugins</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-client-jobclient	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-jobclient</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-mapreduce-client-shuffle	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-client-shuffle</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-mapreduce-examples	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-mapreduce-examples</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-maven-plugins	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-maven-plugins</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-minicluster	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-minicluster</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-minikdc	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-minikdc</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-nfs	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-nfs</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-openstack	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-openstack</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-rumen	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-rumen</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-sls	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-sls</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-streaming	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-streaming</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-yarn-api	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-api</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-yarn-applications-distributedshell	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-applications-distributedshell</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-yarn-applications-unmanaged-am-launcher	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-applications-unmanaged-am-launcher</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-yarn-client	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-client</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-yarn-common	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-common</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`
org.apache.hadoop	hadoop-yarn-registry	2.7.4.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-yarn-registry</artifactId> <version>2.7.4.100-mapr-701</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-yarn-server-applicationhistoryservice	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-applicationhistoryservice</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-common	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-common</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-nodemanager	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-nodemanager</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-resourcemanager	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-resourcemanager</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-sharedcachemanager	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-sharedcachemanager</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-yarn-server-tests	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-tests</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-web-proxy	2.7.4.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-web-proxy</artifactId>`
			`<version>2.7.4.100-mapr-701</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-annotations	1.4.12.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-annotations</artifactId>`
			`<version>1.4.12.100-mapr-701</version>`
			`</dependency>`

org.apache.hbase	hbase-checkstyle	1.4.12.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-checkstyle</artifactId>`
			`<version>1.4.12.100-mapr-701</version>`
			`</dependency>`

org.apache.hbase	hbase-client	1.4.12.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client</artifactId>`
			`<version>1.4.12.100-mapr-701</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-client-project	1.4.12.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client-project</artifactId>`
			`<version>1.4.12.100-mapr-701</version>`
			`</dependency>`

org.apache.hbase	hbase-common	1.4.12.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-common</artifactId>`
			`<version>1.4.12.100-mapr-701</version>`
			`</dependency>`

org.apache.hbase	hbase-examples	1.4.12.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-examples</artifactId>`
			`<version>1.4.12.100-mapr-701</version>`
			`</dependency>`

org.apache.hbase	hbase-external-blockcache	1.4.12.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-external-blockcache</artifactId>`
			`<version>1.4.12.100-mapr-701</version>`
			`</dependency>`

org.apache.hbase	hbase-hadoop-compat	1.4.12.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-hadoop-compat</artifactId>`
			`<version>1.4.12.100-mapr-701</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-hadoop2-compat	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop2-compat</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`
org.apache.hbase	hbase-hbtop	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hbtop</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`
org.apache.hbase	hbase-it	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-it</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`
org.apache.hbase	hbase-metrics	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-metrics</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`
org.apache.hbase	hbase-metrics-api	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-metrics-api</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-prefix-tree	1.4.12.100-mapr-701 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-prefix-tree</artifactId><version>1.4.12.100-mapr-701</version></dependency>`
org.apache.hbase	hbase-procedure	1.4.12.100-mapr-701 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-procedure</artifactId><version>1.4.12.100-mapr-701</version></dependency>`
org.apache.hbase	hbase-protocol	1.4.12.100-mapr-701 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-protocol</artifactId><version>1.4.12.100-mapr-701</version></dependency>`
org.apache.hbase	hbase-resource-bundle	1.4.12.100-mapr-701 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-resource-bundle</artifactId><version>1.4.12.100-mapr-701</version></dependency>`
org.apache.hbase	hbase-rest	1.4.12.100-mapr-701 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-rest</artifactId><version>1.4.12.100-mapr-701</version></dependency>`
org.apache.hbase	hbase-rsgroup	1.4.12.100-mapr-701 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-rsgroup</artifactId><version>1.4.12.100-mapr-701</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-server	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-server</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`
org.apache.hbase	hbase-shaded-client	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-client</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`
org.apache.hbase	hbase-shaded-client-project	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-client-project</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`
org.apache.hbase	hbase-shaded-guava	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-guava</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`
org.apache.hbase	hbase-shaded-server	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-server</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-shaded-testing-util	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-testing-util</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`
org.apache.hbase	hbase-shaded-testing-util-tester	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-testing-util-tester</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`
org.apache.hbase	hbase-shell	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shell</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`
org.apache.hbase	hbase-spark	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-spark</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`
org.apache.hbase	hbase-testing-util	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-testing-util</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-thrift	1.4.12.100-mapr-701 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-thrift</artifactId> <version>1.4.12.100-mapr-701</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-accumulo-handler</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive	hive-beeline	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive	hive-cli	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive	hive-common	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-contrib	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive	hive-druid-handler	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-druid-handler</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive	hive-exec	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-jdbc-handler	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc-handler</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-llap-common	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-tez	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive	hive-maprdb-json-common	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-common</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive	hive-metastore	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive	hive-serde	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive	hive-service	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-service-rpc	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive	hive-shims	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive.shims	hive-shims-common	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`
org.apache.hive.shims	hive-shims-scheduler	2.3.7-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-scheduler</artifactId> <version>2.3.7-mapr-2101</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-testutils	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-vector-code-gen	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-vector-code-gen</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

org.apache.hive	spark-client	2.3.7-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.3.7-mapr-2101</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.kafka	connect-api	2.1.1.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-api</artifactId>`
			`<version>2.1.1.100-mapr-701</version>`
			`</dependency>`

org.apache.kafka	connect-json	2.1.1.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-json</artifactId>`
			`<version>2.1.1.100-mapr-701</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.kafka	connect-runtime	2.1.1.100-mapr-701 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>connect-runtime</artifactId> <version>2.1.1.100-mapr-701</version> </dependency>`
org.apache.kafka	connect-transforms	2.1.1.100-mapr-701 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>connect-transforms</artifactId> <version>2.1.1.100-mapr-701</version> </dependency>`
org.apache.kafka	kafka-clients	2.1.1.100-mapr-701 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-clients</artifactId> <version>2.1.1.100-mapr-701</version> </dependency>`
org.apache.kafka	kafka-log4j-appender	2.1.1.100-mapr-701 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-log4j-appender</artifactId> <version>2.1.1.100-mapr-701</version> </dependency>`
org.apache.kafka	kafka-streams	2.1.1.100-mapr-701 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-streams</artifactId> <version>2.1.1.100-mapr-701</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.kafka	kafka-tools	2.1.1.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-tools</artifactId>`
			`<version>2.1.1.100-mapr-701</version>`
			`</dependency>`

org.apache.kafka	kafka_2.11	2.1.1.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka_2.11</artifactId>`
			`<version>2.1.1.100-mapr-701</version>`
			`</dependency>`

org.apache.kafka	kafka_2.12	2.1.1.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka_2.12</artifactId>`
			`<version>2.1.1.100-mapr-701</version>`
			`</dependency>`

org.apache.kafka	mapr-eco-tools	2.1.1.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>mapr-eco-tools</artifactId>`
			`<version>2.1.1.100-mapr-701</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	5.2.0.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>5.2.0.100-mapr-701</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-core	5.2.0.100-mapr-701 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>5.2.0.100-mapr-701</version> </dependency>`
org.apache.oozie	oozie-examples	5.2.0.100-mapr-701 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>5.2.0.100-mapr-701</version> </dependency>`
org.apache.oozie	oozie-fluent-job-api	5.2.0.100-mapr-701 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-api</artifactId> <version>5.2.0.100-mapr-701</version> </dependency>`
org.apache.oozie	oozie-fluent-job-client	5.2.0.100-mapr-701 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-client</artifactId> <version>5.2.0.100-mapr-701</version> </dependency>`
org.apache.oozie.test	oozie-mini	5.2.0.100-mapr-701 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>5.2.0.100-mapr-701</version> </dependency>`
org.apache.oozie	oozie-server	5.2.0.100-mapr-701 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-server</artifactId> <version>5.2.0.100-mapr-701</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-distcp	5.2.0.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>5.2.0.100-mapr-701</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-git	5.2.0.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-git</artifactId>`
			`<version>5.2.0.100-mapr-701</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	5.2.0.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>5.2.0.100-mapr-701</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	5.2.0.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>5.2.0.100-mapr-701</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	5.2.0.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>5.2.0.100-mapr-701</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-oozie	5.2.0.100-mapr-701 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>5.2.0.100-mapr-701</version> </dependency>`
org.apache.oozie	oozie-sharelib-pig	5.2.0.100-mapr-701 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>5.2.0.100-mapr-701</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	5.2.0.100-mapr-701 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>5.2.0.100-mapr-701</version> </dependency>`
org.apache.oozie	oozie-sharelib-sqoop	5.2.0.100-mapr-701 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-sqoop</artifactId> <version>5.2.0.100-mapr-701</version> </dependency>`
org.apache.oozie	oozie-sharelib-streaming	5.2.0.100-mapr-701 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-streaming</artifactId> <version>5.2.0.100-mapr-701</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-tools	5.2.0.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>5.2.0.100-mapr-701</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	5.2.0.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>5.2.0.100-mapr-701</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.12	2.4.7.0-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.12</artifactId>`
			`<version>2.4.7.0-mapr-701</version>`
			`</dependency>`

org.apache.spark	spark-avro_2.12	2.4.7.0-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-avro_2.12</artifactId>`
			`<version>2.4.7.0-mapr-701</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.12	2.4.7.0-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.12</artifactId>`
			`<version>2.4.7.0-mapr-701</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-core_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-graphx_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-hive_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-kvstore_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-kvstore_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-launcher_2.12	2.4.7.0-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.12</artifactId>`
			`<version>2.4.7.0-mapr-701</version>`
			`</dependency>`

org.apache.spark	spark-mesos_2.12	2.4.7.0-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.12</artifactId>`
			`<version>2.4.7.0-mapr-701</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.12	2.4.7.0-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.12</artifactId>`
			`<version>2.4.7.0-mapr-701</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.12	2.4.7.0-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.12</artifactId>`
			`<version>2.4.7.0-mapr-701</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.12	2.4.7.0-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.12</artifactId>`
			`<version>2.4.7.0-mapr-701</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-network-shuffle_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-network-yarn_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-repl_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-sketch_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sql_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-streaming-flume_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-10_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`
org.apache.spark	spark-streaming_2.12	2.4.7.0-mapr-701 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.12</artifactId> <version>2.4.7.0-mapr-701</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-tags_2.12	2.4.7.0-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.12</artifactId>`
			`<version>2.4.7.0-mapr-701</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.12	2.4.7.0-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.12</artifactId>`
			`<version>2.4.7.0-mapr-701</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.12	2.4.7.0-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.12</artifactId>`
			`<version>2.4.7.0-mapr-701</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.7-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.7-mapr-701</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.7-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.7-mapr-701</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	hadoop-shim	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`
org.apache.tez	hadoop-shim-2.7	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim-2.7</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`
org.apache.tez.conftool	mapr-tez-conf-tool	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez.conftool</groupId> <artifactId>mapr-tez-conf-tool</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`
org.apache.tez	tez-api	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`
org.apache.tez	tez-aux-services	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-aux-services</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-build-tools	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-build-tools</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`
org.apache.tez	tez-common	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`
org.apache.tez	tez-dag	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`
org.apache.tez	tez-examples	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`
org.apache.tez	tez-job-analyzer	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-job-analyzer</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-mapreduce	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-mapreduce</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`
org.apache.tez	tez-protobuf-history-plugin	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-protobuf-history-plugin</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`
org.apache.tez	tez-runtime-internals	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-internals</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`
org.apache.tez	tez-runtime-library	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-library</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`
org.apache.tez	tez-tests	0.9.2.100-mapr-701 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-tests</artifactId> <version>0.9.2.100-mapr-701</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ui	0.9.2.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui</artifactId>`
			`<version>0.9.2.100-mapr-701</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history	0.9.2.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.9.2.100-mapr-701</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.2.100-mapr-701
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.9.2.100-mapr-701</version>`
			`</dependency>`

Maven Artifacts for EEP 7.0.0

Listed are all Maven artifacts for EEP 7.0.0 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

com.mapr.db	maprdb-spark_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-annotations	2.7.4.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-annotations</artifactId> <version>2.7.4.0-mapr-700</version> </dependency>`
org.apache.hadoop	hadoop-ant	2.7.4.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-ant</artifactId> <version>2.7.4.0-mapr-700</version> </dependency>`
org.apache.hadoop	hadoop-archives	2.7.4.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-archives</artifactId> <version>2.7.4.0-mapr-700</version> </dependency>`
org.apache.hadoop	hadoop-assemblies	2.7.4.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-assemblies</artifactId> <version>2.7.4.0-mapr-700</version> </dependency>`
org.apache.hadoop	hadoop-auth	2.7.4.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-auth</artifactId> <version>2.7.4.0-mapr-700</version> </dependency>`
org.apache.hadoop	hadoop-aws	2.7.4.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-aws</artifactId> <version>2.7.4.0-mapr-700</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-azure	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-azure</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-azure-datalake	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-azure-datalake</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-client	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-client</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-common	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-common</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-datajoin	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-datajoin</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-distcp	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-distcp</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-extras	2.7.4.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-extras</artifactId> <version>2.7.4.0-mapr-700</version> </dependency>`
org.apache.hadoop	hadoop-gridmix	2.7.4.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-gridmix</artifactId> <version>2.7.4.0-mapr-700</version> </dependency>`
org.apache.hadoop	hadoop-hdfs	2.7.4.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-hdfs</artifactId> <version>2.7.4.0-mapr-700</version> </dependency>`
org.apache.hadoop.contrib	hadoop-hdfs-bkjournal	2.7.4.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hadoop.contrib</groupId> <artifactId>hadoop-hdfs-bkjournal</artifactId> <version>2.7.4.0-mapr-700</version> </dependency>`
org.apache.hadoop	hadoop-hdfs-nfs	2.7.4.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hadoop</groupId> <artifactId>hadoop-hdfs-nfs</artifactId> <version>2.7.4.0-mapr-700</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-hdfs-sources-osx	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs-sources-osx</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-hdfs-sources-redhat	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs-sources-redhat</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-hdfs-sources-ubuntu	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs-sources-ubuntu</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-hdfs-sources-windows	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-hdfs-sources-windows</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-kms	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-kms</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-mapreduce-client-app	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-app</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-common	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-common</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-contrib	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-contrib</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-core	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-core</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-hs	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-hs</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-mapreduce-client-hs-plugins	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-hs-plugins</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-jobclient	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-jobclient</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-client-shuffle	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-client-shuffle</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-mapreduce-examples	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-mapreduce-examples</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-maven-plugins	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-maven-plugins</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hadoop	hadoop-minicluster	2.7.4.0-mapr-700 Browse	`<dependency><groupId>org.apache.hadoop</groupId><artifactId>hadoop-minicluster</artifactId><version>2.7.4.0-mapr-700</version></dependency>`
org.apache.hadoop	hadoop-minikdc	2.7.4.0-mapr-700 Browse	`<dependency><groupId>org.apache.hadoop</groupId><artifactId>hadoop-minikdc</artifactId><version>2.7.4.0-mapr-700</version></dependency>`
org.apache.hadoop	hadoop-nfs	2.7.4.0-mapr-700 Browse	`<dependency><groupId>org.apache.hadoop</groupId><artifactId>hadoop-nfs</artifactId><version>2.7.4.0-mapr-700</version></dependency>`
org.apache.hadoop	hadoop-openstack	2.7.4.0-mapr-700 Browse	`<dependency><groupId>org.apache.hadoop</groupId><artifactId>hadoop-openstack</artifactId><version>2.7.4.0-mapr-700</version></dependency>`
org.apache.hadoop	hadoop-rumen	2.7.4.0-mapr-700 Browse	`<dependency><groupId>org.apache.hadoop</groupId><artifactId>hadoop-rumen</artifactId><version>2.7.4.0-mapr-700</version></dependency>`
org.apache.hadoop	hadoop-sls	2.7.4.0-mapr-700 Browse	`<dependency><groupId>org.apache.hadoop</groupId><artifactId>hadoop-sls</artifactId><version>2.7.4.0-mapr-700</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-streaming	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-streaming</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-api	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-api</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-applications-distributedshell	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-applications-distributedshell</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-applications-unmanaged-am-launcher	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-applications-unmanaged-am-launcher</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-client	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-client</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-yarn-common	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-common</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-registry	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-registry</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-applicationhistoryservice	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-applicationhistoryservice</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-common	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-common</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-nodemanager	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-nodemanager</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hadoop	hadoop-yarn-server-resourcemanager	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-resourcemanager</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-sharedcachemanager	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-sharedcachemanager</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-tests	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-tests</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

org.apache.hadoop	hadoop-yarn-server-web-proxy	2.7.4.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hadoop</groupId>`
			`<artifactId>hadoop-yarn-server-web-proxy</artifactId>`
			`<version>2.7.4.0-mapr-700</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-annotations	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-annotations</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-checkstyle	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-checkstyle</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

org.apache.hbase	hbase-client	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

org.apache.hbase	hbase-client-project	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client-project</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

org.apache.hbase	hbase-common	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-common</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

org.apache.hbase	hbase-examples	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-examples</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-external-blockcache	1.4.12.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-external-blockcache</artifactId> <version>1.4.12.0-mapr-700</version> </dependency>`
org.apache.hbase	hbase-hadoop-compat	1.4.12.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop-compat</artifactId> <version>1.4.12.0-mapr-700</version> </dependency>`
org.apache.hbase	hbase-hadoop2-compat	1.4.12.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop2-compat</artifactId> <version>1.4.12.0-mapr-700</version> </dependency>`
org.apache.hbase	hbase-hbtop	1.4.12.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hbtop</artifactId> <version>1.4.12.0-mapr-700</version> </dependency>`
org.apache.hbase	hbase-it	1.4.12.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-it</artifactId> <version>1.4.12.0-mapr-700</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-metrics	1.4.12.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-metrics</artifactId> <version>1.4.12.0-mapr-700</version> </dependency>`
org.apache.hbase	hbase-metrics-api	1.4.12.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-metrics-api</artifactId> <version>1.4.12.0-mapr-700</version> </dependency>`
org.apache.hbase	hbase-prefix-tree	1.4.12.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-prefix-tree</artifactId> <version>1.4.12.0-mapr-700</version> </dependency>`
org.apache.hbase	hbase-procedure	1.4.12.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-procedure</artifactId> <version>1.4.12.0-mapr-700</version> </dependency>`
org.apache.hbase	hbase-protocol	1.4.12.0-mapr-700 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-protocol</artifactId> <version>1.4.12.0-mapr-700</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-resource-bundle	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-resource-bundle</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

org.apache.hbase	hbase-rest	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-rest</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

org.apache.hbase	hbase-rsgroup	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-rsgroup</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

org.apache.hbase	hbase-server	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-server</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

org.apache.hbase	hbase-shaded-client	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-client</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-shaded-client-project	1.4.12.0-mapr-700 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-client-project</artifactId><version>1.4.12.0-mapr-700</version></dependency>`
org.apache.hbase	hbase-shaded-guava	1.4.12.0-mapr-700 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-guava</artifactId><version>1.4.12.0-mapr-700</version></dependency>`
org.apache.hbase	hbase-shaded-server	1.4.12.0-mapr-700 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-server</artifactId><version>1.4.12.0-mapr-700</version></dependency>`
org.apache.hbase	hbase-shaded-testing-util	1.4.12.0-mapr-700 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-testing-util</artifactId><version>1.4.12.0-mapr-700</version></dependency>`
org.apache.hbase	hbase-shaded-testing-util-tester	1.4.12.0-mapr-700 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-testing-util-tester</artifactId><version>1.4.12.0-mapr-700</version></dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-shell	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shell</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

org.apache.hbase	hbase-spark	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-spark</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

org.apache.hbase	hbase-testing-util	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-testing-util</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

org.apache.hbase	hbase-thrift	1.4.12.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-thrift</artifactId>`
			`<version>1.4.12.0-mapr-700</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	2.3.7-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.3.7-mapr-2009</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-beeline	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-cli	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-common	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-contrib	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-druid-handler	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-druid-handler</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-exec	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-hbase-handler	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-pig-adapter</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-server-extensions</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-streaming</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-hplsql	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hplsql</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-jdbc	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-jdbc-handler	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc-handler</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-llap-client	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-client</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-llap-common	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-ext-client	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-ext-client</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-llap-server	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-server</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-llap-tez	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-maprdb-json-common	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-common</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-metastore	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-serde	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-service	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-service-rpc	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	hive-shims	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.shims	hive-shims-common	2.3.7-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.3.7-mapr-2009</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.3.7-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.3.7-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.3.7-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.3.7-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-vector-code-gen	2.3.7-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-vector-code-gen</artifactId>`
			`<version>2.3.7-mapr-2009</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.3.7-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.3.7-mapr-2009</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat-java-client</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive.conftool	mapr-conf-tool	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.conftool</groupId> <artifactId>mapr-conf-tool</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.encryptiontool</groupId> <artifactId>mapr-encryption-tool</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.maprminicluster</groupId> <artifactId>mapr-mini-cluster</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`
org.apache.hive	spark-client	2.3.7-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>spark-client</artifactId> <version>2.3.7-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.kafka	connect-api	2.1.1.0-mapr-700 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>connect-api</artifactId> <version>2.1.1.0-mapr-700</version> </dependency>`
org.apache.kafka	connect-json	2.1.1.0-mapr-700 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>connect-json</artifactId> <version>2.1.1.0-mapr-700</version> </dependency>`
org.apache.kafka	connect-runtime	2.1.1.0-mapr-700 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>connect-runtime</artifactId> <version>2.1.1.0-mapr-700</version> </dependency>`
org.apache.kafka	connect-transforms	2.1.1.0-mapr-700 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>connect-transforms</artifactId> <version>2.1.1.0-mapr-700</version> </dependency>`
org.apache.kafka	kafka-clients	2.1.1.0-mapr-700 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-clients</artifactId> <version>2.1.1.0-mapr-700</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.kafka	kafka-log4j-appender	2.1.1.0-mapr-700 Browse	`<dependency><groupId>org.apache.kafka</groupId><artifactId>kafka-log4j-appender</artifactId><version>2.1.1.0-mapr-700</version></dependency>`
org.apache.kafka	kafka-streams	2.1.1.0-mapr-700 Browse	`<dependency><groupId>org.apache.kafka</groupId><artifactId>kafka-streams</artifactId><version>2.1.1.0-mapr-700</version></dependency>`
org.apache.kafka	kafka-tools	2.1.1.0-mapr-700 Browse	`<dependency><groupId>org.apache.kafka</groupId><artifactId>kafka-tools</artifactId><version>2.1.1.0-mapr-700</version></dependency>`
org.apache.kafka	kafka_2.11	2.1.1.0-mapr-700 Browse	`<dependency><groupId>org.apache.kafka</groupId><artifactId>kafka_2.11</artifactId><version>2.1.1.0-mapr-700</version></dependency>`
org.apache.kafka	kafka_2.12	2.1.1.0-mapr-700 Browse	`<dependency><groupId>org.apache.kafka</groupId><artifactId>kafka_2.12</artifactId><version>2.1.1.0-mapr-700</version></dependency>`
org.apache.kafka	mapr-eco-tools	2.1.1.0-mapr-700 Browse	`<dependency><groupId>org.apache.kafka</groupId><artifactId>mapr-eco-tools</artifactId><version>2.1.1.0-mapr-700</version></dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-client	5.2.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-client</artifactId> <version>5.2.0.0-mapr-700</version> </dependency>`
org.apache.oozie	oozie-core	5.2.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>5.2.0.0-mapr-700</version> </dependency>`
org.apache.oozie	oozie-examples	5.2.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>5.2.0.0-mapr-700</version> </dependency>`
org.apache.oozie	oozie-fluent-job-api	5.2.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-api</artifactId> <version>5.2.0.0-mapr-700</version> </dependency>`
org.apache.oozie	oozie-fluent-job-client	5.2.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-client</artifactId> <version>5.2.0.0-mapr-700</version> </dependency>`
org.apache.oozie.test	oozie-mini	5.2.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>5.2.0.0-mapr-700</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-server	5.2.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-server</artifactId> <version>5.2.0.0-mapr-700</version> </dependency>`
org.apache.oozie	oozie-sharelib-distcp	5.2.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>5.2.0.0-mapr-700</version> </dependency>`
org.apache.oozie	oozie-sharelib-git	5.2.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-git</artifactId> <version>5.2.0.0-mapr-700</version> </dependency>`
org.apache.oozie	oozie-sharelib-hcatalog	5.2.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>5.2.0.0-mapr-700</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive	5.2.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>5.2.0.0-mapr-700</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-hive2	5.2.0.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>5.2.0.0-mapr-700</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	5.2.0.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>5.2.0.0-mapr-700</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-pig	5.2.0.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>5.2.0.0-mapr-700</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	5.2.0.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>5.2.0.0-mapr-700</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	5.2.0.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>5.2.0.0-mapr-700</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-streaming	5.2.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-streaming</artifactId> <version>5.2.0.0-mapr-700</version> </dependency>`
org.apache.oozie	oozie-tools	5.2.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-tools</artifactId> <version>5.2.0.0-mapr-700</version> </dependency>`
org.apache.oozie	oozie-webapp	5.2.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-webapp</artifactId> <version>5.2.0.0-mapr-700</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.pig	pig	0.17.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>pig</artifactId> <version>0.17.0.0-mapr-700</version> </dependency>`
org.apache.pig	piggybank	0.17.0.0-mapr-700 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>piggybank</artifactId> <version>0.17.0.0-mapr-700</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.pig	pigsmoke	0.17.0.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigsmoke</artifactId>`
			`<version>0.17.0.0-mapr-700</version>`
			`</dependency>`

org.apache.pig	pigunit	0.17.0.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigunit</artifactId>`
			`<version>0.17.0.0-mapr-700</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

org.apache.spark	spark-avro_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-avro_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-core_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`
org.apache.spark	spark-graphx_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`
org.apache.spark	spark-hive_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`
org.apache.spark	spark-kvstore_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-kvstore_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-launcher_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

org.apache.spark	spark-mesos_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-network-shuffle_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`
org.apache.spark	spark-network-yarn_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`
org.apache.spark	spark-repl_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`
org.apache.spark	spark-sketch_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sql_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`
org.apache.spark	spark-streaming-flume_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.12	2.4.5.0-mapr-700 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.12</artifactId> <version>2.4.5.0-mapr-700</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-0-10_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9-assembly_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9-assembly_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-producer_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

org.apache.spark	spark-streaming_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-tags_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.12	2.4.5.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.12</artifactId>`
			`<version>2.4.5.0-mapr-700</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.7-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.7-mapr-2009</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.7-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.7-mapr-2009</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	hadoop-shim	0.9.2.0-mapr-700 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim</artifactId> <version>0.9.2.0-mapr-700</version> </dependency>`
org.apache.tez	hadoop-shim-2.7	0.9.2.0-mapr-700 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim-2.7</artifactId> <version>0.9.2.0-mapr-700</version> </dependency>`
org.apache.tez.conftool	mapr-tez-conf-tool	0.9.2.0-mapr-700 Browse	`<dependency> <groupId>org.apache.tez.conftool</groupId> <artifactId>mapr-tez-conf-tool</artifactId> <version>0.9.2.0-mapr-700</version> </dependency>`
org.apache.tez	tez-api	0.9.2.0-mapr-700 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.9.2.0-mapr-700</version> </dependency>`
org.apache.tez	tez-aux-services	0.9.2.0-mapr-700 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-aux-services</artifactId> <version>0.9.2.0-mapr-700</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-build-tools	0.9.2.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-build-tools</artifactId>`
			`<version>0.9.2.0-mapr-700</version>`
			`</dependency>`

org.apache.tez	tez-common	0.9.2.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-common</artifactId>`
			`<version>0.9.2.0-mapr-700</version>`
			`</dependency>`

org.apache.tez	tez-dag	0.9.2.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-dag</artifactId>`
			`<version>0.9.2.0-mapr-700</version>`
			`</dependency>`

org.apache.tez	tez-examples	0.9.2.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-examples</artifactId>`
			`<version>0.9.2.0-mapr-700</version>`
			`</dependency>`

org.apache.tez	tez-ext-service-tests	0.9.2.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ext-service-tests</artifactId>`
			`<version>0.9.2.0-mapr-700</version>`
			`</dependency>`

org.apache.tez	tez-job-analyzer	0.9.2.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.9.2.0-mapr-700</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-mapreduce	0.9.2.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.9.2.0-mapr-700</version>`
			`</dependency>`

org.apache.tez	tez-protobuf-history-plugin	0.9.2.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-protobuf-history-plugin</artifactId>`
			`<version>0.9.2.0-mapr-700</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.9.2.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.9.2.0-mapr-700</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.9.2.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.9.2.0-mapr-700</version>`
			`</dependency>`

org.apache.tez	tez-tests	0.9.2.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-tests</artifactId>`
			`<version>0.9.2.0-mapr-700</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ui	0.9.2.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui</artifactId>`
			`<version>0.9.2.0-mapr-700</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history	0.9.2.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.9.2.0-mapr-700</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.2.0-mapr-700
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.9.2.0-mapr-700</version>`
			`</dependency>`

Maven Artifacts for EEP 6.3.6

Listed are all Maven artifacts for EEP 6.3.6 components.

Table

org.apache.drill.contrib	drill-auth-mechanism-maprsasl	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-auth-mechanism-maprsasl</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill	drill-client	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill</groupId>`
			`<artifactId>drill-client</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.drill	drill-common	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill</groupId>`
			`<artifactId>drill-common</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.tools	drill-fmpp-maven-plugin	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.tools</groupId>`
			`<artifactId>drill-fmpp-maven-plugin</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.contrib	drill-format-ltsv	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-format-ltsv</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.contrib	drill-format-mapr	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-format-mapr</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.contrib	drill-format-syslog	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-format-syslog</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.contrib	drill-hbase-test-shaded	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-hbase-test-shaded</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

Table (Continued)

org.apache.drill.exec	drill-java-exec	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.exec</groupId>`
			`<artifactId>drill-java-exec</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.exec	drill-jdbc	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.exec</groupId>`
			`<artifactId>drill-jdbc</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.exec	drill-jdbc-all	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.exec</groupId>`
			`<artifactId>drill-jdbc-all</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.contrib	drill-jdbc-storage	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-jdbc-storage</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.contrib	drill-kudu-storage	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-kudu-storage</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill	drill-logical	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill</groupId>`
			`<artifactId>drill-logical</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.drill.memory	drill-memory-base	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.memory</groupId>`
			`<artifactId>drill-memory-base</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.contrib	drill-mongo-storage	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-mongo-storage</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.contrib	drill-opentsdb-storage	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-opentsdb-storage</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill	drill-protocol	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill</groupId>`
			`<artifactId>drill-protocol</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.exec	drill-rpc	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.exec</groupId>`
			`<artifactId>drill-rpc</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.contrib	drill-storage-hbase	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-storage-hbase</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.drill.contrib	drill-storage-kafka	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-storage-kafka</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.contrib	drill-udfs	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-udfs</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill	drill-yarn	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill</groupId>`
			`<artifactId>drill-yarn</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

org.apache.drill.exec	vector	1.16.0.400-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.exec</groupId>`
			`<artifactId>vector</artifactId>`
			`<version>1.16.0.400-mapr-636</version>`
			`</dependency>`

Table

org.apache.hbase	hbase-annotations	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-annotations</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

org.apache.hbase	hbase-checkstyle	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-checkstyle</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hbase	hbase-client	1.1.13.500-mapr-636 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-client</artifactId> <version>1.1.13.500-mapr-636</version> </dependency>`
org.apache.hbase	hbase-common	1.1.13.500-mapr-636 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-common</artifactId> <version>1.1.13.500-mapr-636</version> </dependency>`
org.apache.hbase	hbase-examples	1.1.13.500-mapr-636 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-examples</artifactId> <version>1.1.13.500-mapr-636</version> </dependency>`
org.apache.hbase	hbase-hadoop-compat	1.1.13.500-mapr-636 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop-compat</artifactId> <version>1.1.13.500-mapr-636</version> </dependency>`
org.apache.hbase	hbase-hadoop2-compat	1.1.13.500-mapr-636 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop2-compat</artifactId> <version>1.1.13.500-mapr-636</version> </dependency>`
org.apache.hbase	hbase-it	1.1.13.500-mapr-636 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-it</artifactId> <version>1.1.13.500-mapr-636</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hbase	hbase-native-client	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-native-client</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

org.apache.hbase	hbase-prefix-tree	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-prefix-tree</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

org.apache.hbase	hbase-procedure	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-procedure</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

org.apache.hbase	hbase-protocol	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-protocol</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

org.apache.hbase	hbase-resource-bundle	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-resource-bundle</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

org.apache.hbase	hbase-rest	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-rest</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hbase	hbase-server	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-server</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

org.apache.hbase	hbase-shaded-client	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-client</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

org.apache.hbase	hbase-shaded-htrace	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-htrace</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

org.apache.hbase	hbase-shaded-server	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-server</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

org.apache.hbase	hbase-shell	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shell</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

org.apache.hbase	hbase-spark	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-spark</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hbase	hbase-testing-util	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-testing-util</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

org.apache.hbase	hbase-thrift	1.1.13.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-thrift</artifactId>`
			`<version>1.1.13.500-mapr-636</version>`
			`</dependency>`

Table

org.apache.hive	hive-accumulo-handler	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-beeline	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-beeline</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-cli	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-cli</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-common	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-common</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hive	hive-contrib	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-contrib</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-druid-handler	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-druid-handler</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-exec	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-exec</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-hbase-handler	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-core	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

Table (Continued)

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.6-mapr-2201 Browse	<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-server-extensions</artifactId> <version>2.3.6-mapr-2201</version> </dependency>
org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.6-mapr-2201 Browse	<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-streaming</artifactId> <version>2.3.6-mapr-2201</version> </dependency>
org.apache.hive	hive-hplsql	2.3.6-mapr-2201 Browse	<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hplsql</artifactId> <version>2.3.6-mapr-2201</version> </dependency>
org.apache.hive	hive-jdbc	2.3.6-mapr-2201 Browse	<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc</artifactId> <version>2.3.6-mapr-2201</version> </dependency>
org.apache.hive	hive-jdbc-handler	2.3.6-mapr-2201 Browse	<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc-handler</artifactId> <version>2.3.6-mapr-2201</version> </dependency>
org.apache.hive	hive-llap-client	2.3.6-mapr-2201 Browse	<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-client</artifactId> <version>2.3.6-mapr-2201</version> </dependency>

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hive	hive-llap-common	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-llap-tez	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-common	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-common</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

Table (Continued)

org.apache.hive	hive-metastore	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-metastore</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-serde	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-serde</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-service	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-service-rpc	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service-rpc</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-shims	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-shims</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-0.23	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.23</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hive.shims	hive-shims-common	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	hive-vector-code-gen	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-vector-code-gen</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hive.conftool	mapr-conf-tool	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	mapr-log4j-slf4j-impl	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>mapr-log4j-slf4j-impl</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

org.apache.hive	spark-client	2.3.6-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.3.6-mapr-2201</version>`
			`</dependency>`

Table

org.apache.kafka	connect-api	1.1.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-api</artifactId>`
			`<version>1.1.1-mapr-2201</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.kafka	connect-json	1.1.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-json</artifactId>`
			`<version>1.1.1-mapr-2201</version>`
			`</dependency>`

org.apache.kafka	connect-runtime	1.1.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-runtime</artifactId>`
			`<version>1.1.1-mapr-2201</version>`
			`</dependency>`

org.apache.kafka	connect-transforms	1.1.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-transforms</artifactId>`
			`<version>1.1.1-mapr-2201</version>`
			`</dependency>`

org.apache.kafka	kafka-clients	1.1.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-clients</artifactId>`
			`<version>1.1.1-mapr-2201</version>`
			`</dependency>`

org.apache.kafka	kafka-log4j-appender	1.1.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-log4j-appender</artifactId>`
			`<version>1.1.1-mapr-2201</version>`
			`</dependency>`

org.apache.kafka	kafka-streams	1.1.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-streams</artifactId>`
			`<version>1.1.1-mapr-2201</version>`
			`</dependency>`

Table (Continued)

org.apache.kafka	kafka-tools	1.1.1-mapr-2201 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-tools</artifactId> <version>1.1.1-mapr-2201</version> </dependency>`
org.apache.kafka	kafka_2.11	1.1.1-mapr-2201 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka_2.11</artifactId> <version>1.1.1-mapr-2201</version> </dependency>`
org.apache.kafka	kafka_2.12	1.1.1-mapr-2201 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka_2.12</artifactId> <version>1.1.1-mapr-2201</version> </dependency>`

Table

org.apache.oozie	oozie-client	5.1.0.800-mapr-636 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-client</artifactId> <version>5.1.0.800-mapr-636</version> </dependency>`
org.apache.oozie	oozie-core	5.1.0.800-mapr-636 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>5.1.0.800-mapr-636</version> </dependency>`
org.apache.oozie	oozie-examples	5.1.0.800-mapr-636 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>5.1.0.800-mapr-636</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.oozie	oozie-fluent-job-api	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-fluent-job-api</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

org.apache.oozie	oozie-fluent-job-client	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-fluent-job-client</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

org.apache.oozie	oozie-server	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-server</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-distcp	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-git	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-git</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.oozie	oozie-sharelib-hcatalog	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-pig	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.oozie	oozie-sharelib-sqoop	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	5.1.0.800-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>5.1.0.800-mapr-636</version>`
			`</dependency>`

Table

org.apache.spark	classpath-filter_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.spark	spark-avro_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-avro_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-graphx_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-hive_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.spark	spark-kvstore_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-kvstore_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-mesos_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

Table (Continued)

org.apache.spark	spark-network-shuffle_2.11	2.4.4.500-mapr-636 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.11</artifactId> <version>2.4.4.500-mapr-636</version> </dependency>
org.apache.spark	spark-network-yarn_2.11	2.4.4.500-mapr-636 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.11</artifactId> <version>2.4.4.500-mapr-636</version> </dependency>
org.apache.spark	spark-repl_2.11	2.4.4.500-mapr-636 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.4.4.500-mapr-636</version> </dependency>
org.apache.spark	spark-sketch_2.11	2.4.4.500-mapr-636 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.4.4.500-mapr-636</version> </dependency>
org.apache.spark	spark-sql-kafka-0-10_2.11	2.4.4.500-mapr-636 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.11</artifactId> <version>2.4.4.500-mapr-636</version> </dependency>
org.apache.spark	spark-sql_2.11	2.4.4.500-mapr-636 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.4.4.500-mapr-636</version> </dependency>

last-updated: Apr 11, 2022

Table (Continued)

org.apache.spark	spark-streaming-flume-assembly_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

Table (Continued)

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-producer_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-streaming_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

org.apache.spark	spark-tags_2.11	2.4.4.500-mapr-636
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.11</artifactId>`
			`<version>2.4.4.500-mapr-636</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.spark	spark-unsafe_2.11	2.4.4.500-mapr-636 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-unsafe_2.11</artifactId> <version>2.4.4.500-mapr-636</version> </dependency>`
org.apache.spark	spark-yarn_2.11	2.4.4.500-mapr-636 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-yarn_2.11</artifactId> <version>2.4.4.500-mapr-636</version> </dependency>`

Table

org.apache.sqoop	sqoop	1.4.7-mapr-636 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop</artifactId> <version>1.4.7-mapr-636</version> </dependency>`
org.apache.sqoop	sqoop-test	1.4.7-mapr-636 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-test</artifactId> <version>1.4.7-mapr-636</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	hadoop-shim	0.9.1-mapr-2201 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim</artifactId> <version>0.9.1-mapr-2201</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim-2.7	0.9.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim-2.7</artifactId>`
			`<version>0.9.1-mapr-2201</version>`
			`</dependency>`

org.apache.tez.conftool	mapr-tez-conf-tool	0.9.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.tez.conftool</groupId>`
			`<artifactId>mapr-tez-conf-tool</artifactId>`
			`<version>0.9.1-mapr-2201</version>`
			`</dependency>`

org.apache.tez	tez-api	0.9.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-api</artifactId>`
			`<version>0.9.1-mapr-2201</version>`
			`</dependency>`

org.apache.tez	tez-aux-services	0.9.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-aux-services</artifactId>`
			`<version>0.9.1-mapr-2201</version>`
			`</dependency>`

org.apache.tez	tez-common	0.9.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-common</artifactId>`
			`<version>0.9.1-mapr-2201</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-dag	0.9.1-mapr-2201 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.9.1-mapr-2201</version> </dependency>`
org.apache.tez	tez-examples	0.9.1-mapr-2201 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.9.1-mapr-2201</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.9.1-mapr-2201 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.9.1-mapr-2201</version> </dependency>`
org.apache.tez	tez-job-analyzer	0.9.1-mapr-2201 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-job-analyzer</artifactId> <version>0.9.1-mapr-2201</version> </dependency>`
org.apache.tez	tez-mapreduce	0.9.1-mapr-2201 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-mapreduce</artifactId> <version>0.9.1-mapr-2201</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-runtime-internals	0.9.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.9.1-mapr-2201</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.9.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.9.1-mapr-2201</version>`
			`</dependency>`

org.apache.tez	tez-tests	0.9.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-tests</artifactId>`
			`<version>0.9.1-mapr-2201</version>`
			`</dependency>`

org.apache.tez	tez-ui	0.9.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui</artifactId>`
			`<version>0.9.1-mapr-2201</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history	0.9.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.9.1-mapr-2201</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.1-mapr-2201
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.9.1-mapr-2201</version>`
			`</dependency>`

Maven Artifacts for EEP 6.3.5

Listed are all Maven artifacts for EEP 6.3.5 components.

Table

org.apache.drill.contrib	drill-auth-mechanism-maprsasl	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-auth-mechanism-maprsasl</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

org.apache.drill	drill-client	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill</groupId>`
			`<artifactId>drill-client</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

org.apache.drill	drill-common	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill</groupId>`
			`<artifactId>drill-common</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

org.apache.drill.tools	drill-fmpp-maven-plugin	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.tools</groupId>`
			`<artifactId>drill-fmpp-maven-plugin</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.drill.contrib	drill-format-ltsv	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-format-ltsv</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

org.apache.drill.contrib	drill-format-mapr	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-format-mapr</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

org.apache.drill.contrib	drill-format-syslog	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-format-syslog</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

org.apache.drill.exec	drill-java-exec	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.exec</groupId>`
			`<artifactId>drill-java-exec</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

org.apache.drill.exec	drill-jdbc	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.exec</groupId>`
			`<artifactId>drill-jdbc</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

org.apache.drill.exec	drill-jdbc-all	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.exec</groupId>`
			`<artifactId>drill-jdbc-all</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.drill.contrib	drill-jdbc-storage	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-jdbc-storage</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

org.apache.drill.contrib	drill-kudu-storage	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-kudu-storage</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

org.apache.drill	drill-logical	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill</groupId>`
			`<artifactId>drill-logical</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

org.apache.drill.memory	drill-memory-base	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.memory</groupId>`
			`<artifactId>drill-memory-base</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

org.apache.drill.contrib	drill-mongo-storage	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-mongo-storage</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

org.apache.drill.contrib	drill-opentsdb-storage	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.contrib</groupId>`
			`<artifactId>drill-opentsdb-storage</artifactId>`
			`<version>1.16.0.300-mapr-635</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.drill	drill-protocol	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill<`
			`/groupId>`
			`<artifactId>drill-protocol`
			`</artifactId>`
			`<version>1.16.0.300-mapr-6`
			`35</version>`
			`</dependency>`

org.apache.drill.exec	drill-rpc	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.`
			`exec</groupId>`
			`<artifactId>drill-rpc</`
			`artifactId>`
			`<version>1.16.0.300-mapr-6`
			`35</version>`
			`</dependency>`

org.apache.drill.contrib	drill-storage-hbase	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.`
			`contrib</groupId>`
			`<artifactId>drill-storag`
			`e-hbase</artifactId>`
			`<version>1.16.0.300-mapr-6`
			`35</version>`
			`</dependency>`

org.apache.drill.contrib	drill-storage-kafka	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.`
			`contrib</groupId>`
			`<artifactId>drill-storag`
			`e-kafka</artifactId>`
			`<version>1.16.0.300-mapr-6`
			`35</version>`
			`</dependency>`

org.apache.drill.contrib	drill-udfs	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill.`
			`contrib</groupId>`
			`<artifactId>drill-udfs</`
			`artifactId>`
			`<version>1.16.0.300-mapr-6`
			`35</version>`
			`</dependency>`

org.apache.drill	drill-yarn	1.16.0.300-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.drill<`
			`/groupId>`
			`<artifactId>drill-yarn</`
			`artifactId>`
			`<version>1.16.0.300-mapr-6`
			`35</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.drill.exec	vector	1.16.0.300-mapr-635 Browse	<dependency> <groupId>org.apache.drill.exec</groupId> <artifactId>vector</artifactId> <version>1.16.0.300-mapr-635</version> </dependency>

Table

org.apache.hbase	hbase-annotations	1.1.13.400-mapr-635 Browse	<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-annotations</artifactId> <version>1.1.13.400-mapr-635</version> </dependency>
org.apache.hbase	hbase-checkstyle	1.1.13.400-mapr-635 Browse	<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-checkstyle</artifactId> <version>1.1.13.400-mapr-635</version> </dependency>
org.apache.hbase	hbase-client	1.1.13.400-mapr-635 Browse	<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-client</artifactId> <version>1.1.13.400-mapr-635</version> </dependency>
org.apache.hbase	hbase-common	1.1.13.400-mapr-635 Browse	<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-common</artifactId> <version>1.1.13.400-mapr-635</version> </dependency>
org.apache.hbase	hbase-examples	1.1.13.400-mapr-635 Browse	<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-examples</artifactId> <version>1.1.13.400-mapr-635</version> </dependency>

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hbase	hbase-hadoop-compat	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-hadoop-compat</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

org.apache.hbase	hbase-hadoop2-compat	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-hadoop2-compat</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

org.apache.hbase	hbase-it	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-it</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

org.apache.hbase	hbase-native-client	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-native-client</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

org.apache.hbase	hbase-prefix-tree	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-prefix-tree</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

org.apache.hbase	hbase-procedure	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-procedure</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hbase	hbase-protocol	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-protocol</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

org.apache.hbase	hbase-resource-bundle	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-resource-bundle</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

org.apache.hbase	hbase-rest	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-rest</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

org.apache.hbase	hbase-server	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-server</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

org.apache.hbase	hbase-shaded-client	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-client</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

org.apache.hbase	hbase-shaded-htrace	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-htrace</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hbase	hbase-shaded-server	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-server</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

org.apache.hbase	hbase-shell	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shell</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

org.apache.hbase	hbase-spark	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-spark</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

org.apache.hbase	hbase-testing-util	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-testing-util</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

org.apache.hbase	hbase-thrift	1.1.13.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-thrift</artifactId>`
			`<version>1.1.13.400-mapr-635</version>`
			`</dependency>`

Table

org.apache.hive	hive-accumulo-handler	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hive	hive-beeline	2.3.6-mapr-2110 Browse	<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.3.6-mapr-2110</version> </dependency>
org.apache.hive	hive-cli	2.3.6-mapr-2110 Browse	<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.3.6-mapr-2110</version> </dependency>
org.apache.hive	hive-common	2.3.6-mapr-2110 Browse	<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.3.6-mapr-2110</version> </dependency>
org.apache.hive	hive-contrib	2.3.6-mapr-2110 Browse	<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.3.6-mapr-2110</version> </dependency>
org.apache.hive	hive-druid-handler	2.3.6-mapr-2110 Browse	<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-druid-handler</artifactId> <version>2.3.6-mapr-2110</version> </dependency>
org.apache.hive	hive-exec	2.3.6-mapr-2110 Browse	<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.3.6-mapr-2110</version> </dependency>

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hive	hive-hbase-handler	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-core	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hive	hive-jdbc	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive	hive-jdbc-handler	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc-handler</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive	hive-llap-common	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

Table (Continued)

org.apache.hive	hive-llap-tez	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-common	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-common</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive	hive-metastore	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-metastore</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive	hive-serde	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-serde</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive	hive-service	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hive	hive-service-rpc	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service-rpc</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive	hive-shims	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-shims</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-0.23	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.23</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-common	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hive	hive-vector-code-gen	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-vector-code-gen</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.6-mapr-2110
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.3.6-mapr-2110</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.hive	spark-client	2.3.6-mapr-2110 Browse	<dependency> <groupId>org.apache.hive</groupId> <artifactId>spark-client</artifactId> <version>2.3.6-mapr-2110</version> </dependency>

Table

org.apache.oozie	oozie-client	5.1.0.700-mapr-635 Browse	<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-client</artifactId> <version>5.1.0.700-mapr-635</version> </dependency>
org.apache.oozie	oozie-core	5.1.0.700-mapr-635 Browse	<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>5.1.0.700-mapr-635</version> </dependency>
org.apache.oozie	oozie-examples	5.1.0.700-mapr-635 Browse	<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>5.1.0.700-mapr-635</version> </dependency>
org.apache.oozie	oozie-fluent-job-api	5.1.0.700-mapr-635 Browse	<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-api</artifactId> <version>5.1.0.700-mapr-635</version> </dependency>
org.apache.oozie	oozie-fluent-job-client	5.1.0.700-mapr-635 Browse	<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-client</artifactId> <version>5.1.0.700-mapr-635</version> </dependency>

last-updated: Apr 11, 2022

Table (Continued)

org.apache.oozie.test	oozie-mini	5.1.0.700-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>5.1.0.700-mapr-635</version>`
			`</dependency>`

org.apache.oozie	oozie-server	5.1.0.700-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-server</artifactId>`
			`<version>5.1.0.700-mapr-635</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-distcp	5.1.0.700-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>5.1.0.700-mapr-635</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-git	5.1.0.700-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-git</artifactId>`
			`<version>5.1.0.700-mapr-635</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	5.1.0.700-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>5.1.0.700-mapr-635</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	5.1.0.700-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>5.1.0.700-mapr-635</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.oozie	oozie-sharelib-hive2	5.1.0.700-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>5.1.0.700-mapr-635</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	5.1.0.700-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>5.1.0.700-mapr-635</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-pig	5.1.0.700-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>5.1.0.700-mapr-635</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	5.1.0.700-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>5.1.0.700-mapr-635</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	5.1.0.700-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>5.1.0.700-mapr-635</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	5.1.0.700-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>5.1.0.700-mapr-635</version>`
			`</dependency>`

Table (Continued)

org.apache.oozie	oozie-tools	5.1.0.700-mapr-635 Browse	<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-tools</artifactId> <version>5.1.0.700-mapr-635</version> </dependency>
org.apache.oozie	oozie-webapp	5.1.0.700-mapr-635 Browse	<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-webapp</artifactId> <version>5.1.0.700-mapr-635</version> </dependency>

Table

org.apache.pig	pig	0.16.0-mapr-2110 Browse	<dependency> <groupId>org.apache.pig</groupId> <artifactId>pig</artifactId> <version>0.16.0-mapr-2110</version> </dependency>
org.apache.pig	piggybank	0.16.0-mapr-2110 Browse	<dependency> <groupId>org.apache.pig</groupId> <artifactId>piggybank</artifactId> <version>0.16.0-mapr-2110</version> </dependency>
org.apache.pig	pigsmoke	0.16.0-mapr-2110 Browse	<dependency> <groupId>org.apache.pig</groupId> <artifactId>pigsmoke</artifactId> <version>0.16.0-mapr-2110</version> </dependency>
org.apache.pig	pigunit	0.16.0-mapr-2110 Browse	<dependency> <groupId>org.apache.pig</groupId> <artifactId>pigunit</artifactId> <version>0.16.0-mapr-2110</version> </dependency>

last-updated: Apr 11, 2022

Table

org.apache.spark	classpath-filter_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

org.apache.spark	spark-avro_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-avro_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

org.apache.spark	spark-graphx_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

Table (Continued)

org.apache.spark	spark-hive-thriftserver_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>
org.apache.spark	spark-hive_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>
org.apache.spark	spark-kvstore_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-kvstore_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>
org.apache.spark	spark-launcher_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-launcher_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>
org.apache.spark	spark-mesos_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mesos_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>
org.apache.spark	spark-mllib-local_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib-local_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>

last-updated: Apr 11, 2022

Table (Continued)

org.apache.spark	spark-mllib_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

org.apache.spark	spark-sketch_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.spark	spark-sql-kafka-0-10_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

org.apache.spark	spark-sql_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-assembly_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.4.4.400-mapr-635
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.4.4.400-mapr-635</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>

Table (Continued)

org.apache.spark	spark-streaming-kafka-producer_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>
org.apache.spark	spark-streaming_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>
org.apache.spark	spark-tags_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>
org.apache.spark	spark-unsafe_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-unsafe_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>
org.apache.spark	spark-yarn_2.11	2.4.4.400-mapr-635 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-yarn_2.11</artifactId> <version>2.4.4.400-mapr-635</version> </dependency>

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sqoop	sqoop	1.4.7-mapr-635 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop</artifactId> <version>1.4.7-mapr-635</version> </dependency>`
org.apache.sqoop	sqoop-test	1.4.7-mapr-635 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-test</artifactId> <version>1.4.7-mapr-635</version> </dependency>`

Maven Artifacts for EEP 6.3.4

Listed are all Maven artifacts for EEP 6.3.4 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-annotations	1.1.13.300-mapr-634 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-annotations</artifactId> <version>1.1.13.300-mapr-634</version> </dependency>`
org.apache.hbase	hbase-checkstyle	1.1.13.300-mapr-634 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-checkstyle</artifactId> <version>1.1.13.300-mapr-634</version> </dependency>`
org.apache.hbase	hbase-client	1.1.13.300-mapr-634 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-client</artifactId> <version>1.1.13.300-mapr-634</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-common	1.1.13.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-common</artifactId>`
			`<version>1.1.13.300-mapr-634</version>`
			`</dependency>`

org.apache.hbase	hbase-examples	1.1.13.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-examples</artifactId>`
			`<version>1.1.13.300-mapr-634</version>`
			`</dependency>`

org.apache.hbase	hbase-hadoop-compat	1.1.13.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-hadoop-compat</artifactId>`
			`<version>1.1.13.300-mapr-634</version>`
			`</dependency>`

org.apache.hbase	hbase-hadoop2-compat	1.1.13.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-hadoop2-compat</artifactId>`
			`<version>1.1.13.300-mapr-634</version>`
			`</dependency>`

org.apache.hbase	hbase-it	1.1.13.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-it</artifactId>`
			`<version>1.1.13.300-mapr-634</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-native-client	1.1.13.300-mapr-634 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-native-client</artifactId><version>1.1.13.300-mapr-634</version></dependency>`
org.apache.hbase	hbase-prefix-tree	1.1.13.300-mapr-634 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-prefix-tree</artifactId><version>1.1.13.300-mapr-634</version></dependency>`
org.apache.hbase	hbase-procedure	1.1.13.300-mapr-634 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-procedure</artifactId><version>1.1.13.300-mapr-634</version></dependency>`
org.apache.hbase	hbase-protocol	1.1.13.300-mapr-634 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-protocol</artifactId><version>1.1.13.300-mapr-634</version></dependency>`
org.apache.hbase	hbase-resource-bundle	1.1.13.300-mapr-634 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-resource-bundle</artifactId><version>1.1.13.300-mapr-634</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-rest	1.1.13.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-rest</artifactId>`
			`<version>1.1.13.300-mapr-634</version>`
			`</dependency>`

org.apache.hbase	hbase-server	1.1.13.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-server</artifactId>`
			`<version>1.1.13.300-mapr-634</version>`
			`</dependency>`

org.apache.hbase	hbase-shaded-client	1.1.13.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-client</artifactId>`
			`<version>1.1.13.300-mapr-634</version>`
			`</dependency>`

org.apache.hbase	hbase-shaded-htrace	1.1.13.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-htrace</artifactId>`
			`<version>1.1.13.300-mapr-634</version>`
			`</dependency>`

org.apache.hbase	hbase-shaded-server	1.1.13.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-server</artifactId>`
			`<version>1.1.13.300-mapr-634</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-shell	1.1.13.300-mapr-634 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shell</artifactId> <version>1.1.13.300-mapr-634</version> </dependency>`
org.apache.hbase	hbase-spark	1.1.13.300-mapr-634 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-spark</artifactId> <version>1.1.13.300-mapr-634</version> </dependency>`
org.apache.hbase	hbase-testing-util	1.1.13.300-mapr-634 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-testing-util</artifactId> <version>1.1.13.300-mapr-634</version> </dependency>`
org.apache.hbase	hbase-thrift	1.1.13.300-mapr-634 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-thrift</artifactId> <version>1.1.13.300-mapr-634</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-accumulo-handler</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-beeline	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive	hive-cli	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive	hive-common	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive	hive-contrib	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive	hive-druid-handler	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-druid-handler</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive	hive-exec	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-hbase-handler	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-core	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-hplsql	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hplsql</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive	hive-jdbc	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive	hive-jdbc-handler	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc-handler</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive	hive-llap-client	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-client</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive	hive-llap-common	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-llap-ext-client	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-llap-tez	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-common	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-common</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-metastore	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive	hive-serde	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive	hive-service	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive	hive-service-rpc	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive	hive-shims	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.shims	hive-shims-common	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

org.apache.hive	hive-vector-code-gen	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-vector-code-gen</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.3.6-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.3.6-mapr-2104</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat-java-client</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive.conftool	mapr-conf-tool	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive.conftool</groupId> <artifactId>mapr-conf-tool</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive.encryptiontool</groupId> <artifactId>mapr-encryption-tool</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive.maprminicluster</groupId> <artifactId>mapr-mini-cluster</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`
org.apache.hive	spark-client	2.3.6-mapr-2104 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>spark-client</artifactId> <version>2.3.6-mapr-2104</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.kafka	connect-api	1.1.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>connect-api</artifactId> <version>1.1.1-mapr-2104</version> </dependency>`
org.apache.kafka	connect-json	1.1.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>connect-json</artifactId> <version>1.1.1-mapr-2104</version> </dependency>`
org.apache.kafka	connect-runtime	1.1.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>connect-runtime</artifactId> <version>1.1.1-mapr-2104</version> </dependency>`
org.apache.kafka	connect-transforms	1.1.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>connect-transforms</artifactId> <version>1.1.1-mapr-2104</version> </dependency>`
org.apache.kafka	kafka-clients	1.1.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-clients</artifactId> <version>1.1.1-mapr-2104</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.kafka	kafka-log4j-appender	1.1.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-log4j-appender</artifactId> <version>1.1.1-mapr-2104</version> </dependency>`
org.apache.kafka	kafka-streams	1.1.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-streams</artifactId> <version>1.1.1-mapr-2104</version> </dependency>`
org.apache.kafka	kafka-tools	1.1.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-tools</artifactId> <version>1.1.1-mapr-2104</version> </dependency>`
org.apache.kafka	kafka_2.11	1.1.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka_2.11</artifactId> <version>1.1.1-mapr-2104</version> </dependency>`
org.apache.kafka	kafka_2.12	1.1.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka_2.12</artifactId> <version>1.1.1-mapr-2104</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-client	5.1.0.600-mapr-634 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-client</artifactId> <version>5.1.0.600-mapr-634</version> </dependency>`
org.apache.oozie	oozie-core	5.1.0.600-mapr-634 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>5.1.0.600-mapr-634</version> </dependency>`
org.apache.oozie	oozie-examples	5.1.0.600-mapr-634 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>5.1.0.600-mapr-634</version> </dependency>`
org.apache.oozie	oozie-fluent-job-api	5.1.0.600-mapr-634 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-api</artifactId> <version>5.1.0.600-mapr-634</version> </dependency>`
org.apache.oozie	oozie-fluent-job-client	5.1.0.600-mapr-634 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-client</artifactId> <version>5.1.0.600-mapr-634</version> </dependency>`
org.apache.oozie.test	oozie-mini	5.1.0.600-mapr-634 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>5.1.0.600-mapr-634</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-server	5.1.0.600-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-server</artifactId>`
			`<version>5.1.0.600-mapr-634</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-distcp	5.1.0.600-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>5.1.0.600-mapr-634</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-git	5.1.0.600-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-git</artifactId>`
			`<version>5.1.0.600-mapr-634</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	5.1.0.600-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>5.1.0.600-mapr-634</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	5.1.0.600-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>5.1.0.600-mapr-634</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-hive2	5.1.0.600-mapr-634 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>5.1.0.600-mapr-634</version> </dependency>`
org.apache.oozie	oozie-sharelib-oozie	5.1.0.600-mapr-634 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>5.1.0.600-mapr-634</version> </dependency>`
org.apache.oozie	oozie-sharelib-pig	5.1.0.600-mapr-634 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>5.1.0.600-mapr-634</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	5.1.0.600-mapr-634 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>5.1.0.600-mapr-634</version> </dependency>`
org.apache.oozie	oozie-sharelib-sqoop	5.1.0.600-mapr-634 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-sqoop</artifactId> <version>5.1.0.600-mapr-634</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-streaming	5.1.0.600-mapr-634 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-streaming</artifactId> <version>5.1.0.600-mapr-634</version> </dependency>`
org.apache.oozie	oozie-tools	5.1.0.600-mapr-634 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-tools</artifactId> <version>5.1.0.600-mapr-634</version> </dependency>`
org.apache.oozie	oozie-webapp	5.1.0.600-mapr-634 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-webapp</artifactId> <version>5.1.0.600-mapr-634</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	classpath-filter_2.11	2.4.4.300-mapr-634 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>classpath-filter_2.11</artifactId> <version>2.4.4.300-mapr-634</version> </dependency>`
com.mapr.db	maprdb-spark	2.4.4.300-mapr-634 Browse	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-spark</artifactId> <version>2.4.4.300-mapr-634</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-avro_2.11	2.4.4.300-mapr-634 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-avro_2.11</artifactId> <version>2.4.4.300-mapr-634</version> </dependency>`
org.apache.spark	spark-catalyst_2.11	2.4.4.300-mapr-634 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-catalyst_2.11</artifactId> <version>2.4.4.300-mapr-634</version> </dependency>`
org.apache.spark	spark-core_2.11	2.4.4.300-mapr-634 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.11</artifactId> <version>2.4.4.300-mapr-634</version> </dependency>`
org.apache.spark	spark-graphx_2.11	2.4.4.300-mapr-634 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.11</artifactId> <version>2.4.4.300-mapr-634</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.11	2.4.4.300-mapr-634 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.4.4.300-mapr-634</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-hive_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-kvstore_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-kvstore_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-mesos_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-mllib_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-sketch_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-sql-kafka-0-10_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-sql_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-assembly_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-flume_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.4.4.300-mapr-634 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.4.4.300-mapr-634</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.4.4.300-mapr-634 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.4.4.300-mapr-634</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.4.4.300-mapr-634 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.4.4.300-mapr-634</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.4.4.300-mapr-634 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.4.4.300-mapr-634</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.4.4.300-mapr-634 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.4.4.300-mapr-634</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-unsafe_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.4.4.300-mapr-634
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.4.4.300-mapr-634</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.9.1-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.9.1-mapr-2104</version>`
			`</dependency>`

org.apache.tez	hadoop-shim-2.7	0.9.1-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim-2.7</artifactId>`
			`<version>0.9.1-mapr-2104</version>`
			`</dependency>`

org.apache.tez.conftool	mapr-tez-conf-tool	0.9.1-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.tez.conftool</groupId>`
			`<artifactId>mapr-tez-conf-tool</artifactId>`
			`<version>0.9.1-mapr-2104</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-api	0.9.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.9.1-mapr-2104</version> </dependency>`
org.apache.tez	tez-aux-services	0.9.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-aux-services</artifactId> <version>0.9.1-mapr-2104</version> </dependency>`
org.apache.tez	tez-common	0.9.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.9.1-mapr-2104</version> </dependency>`
org.apache.tez	tez-dag	0.9.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.9.1-mapr-2104</version> </dependency>`
org.apache.tez	tez-examples	0.9.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.9.1-mapr-2104</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.9.1-mapr-2104 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.9.1-mapr-2104</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-job-analyzer	0.9.1-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.9.1-mapr-2104</version>`
			`</dependency>`

org.apache.tez	tez-mapreduce	0.9.1-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.9.1-mapr-2104</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.9.1-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.9.1-mapr-2104</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.9.1-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.9.1-mapr-2104</version>`
			`</dependency>`

org.apache.tez	tez-tests	0.9.1-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-tests</artifactId>`
			`<version>0.9.1-mapr-2104</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ui	0.9.1-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui</artifactId>`
			`<version>0.9.1-mapr-2104</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history	0.9.1-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.9.1-mapr-2104</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.1-mapr-2104
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.9.1-mapr-2104</version>`
			`</dependency>`

Maven Artifacts for EEP 6.3.3

Listed are all Maven artifacts for EEP 6.3.3 components.

For EEP 6.3.3 Maven information, see Maven Artifacts for EEP 6.3.2 on page 4431.

Maven Artifacts for EEP 6.3.2

Listed are all Maven artifacts for EEP 6.3.2 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-annotations	1.1.13.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-annotations</artifactId>`
			`<version>1.1.13.200-mapr-632</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-checkstyle	1.1.13.200-mapr-632 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-checkstyle</artifactId><version>1.1.13.200-mapr-632</version></dependency>`
org.apache.hbase	hbase-client	1.1.13.200-mapr-632 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-client</artifactId><version>1.1.13.200-mapr-632</version></dependency>`
org.apache.hbase	hbase-common	1.1.13.200-mapr-632 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-common</artifactId><version>1.1.13.200-mapr-632</version></dependency>`
org.apache.hbase	hbase-examples	1.1.13.200-mapr-632 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-examples</artifactId><version>1.1.13.200-mapr-632</version></dependency>`
org.apache.hbase	hbase-hadoop-compat	1.1.13.200-mapr-632 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-hadoop-compat</artifactId><version>1.1.13.200-mapr-632</version></dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-hadoop2-compat	1.1.13.200-mapr-632 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop2-compat</artifactId> <version>1.1.13.200-mapr-632</version> </dependency>`
org.apache.hbase	hbase-it	1.1.13.200-mapr-632 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-it</artifactId> <version>1.1.13.200-mapr-632</version> </dependency>`
org.apache.hbase	hbase-native-client	1.1.13.200-mapr-632 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-native-client</artifactId> <version>1.1.13.200-mapr-632</version> </dependency>`
org.apache.hbase	hbase-prefix-tree	1.1.13.200-mapr-632 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-prefix-tree</artifactId> <version>1.1.13.200-mapr-632</version> </dependency>`
org.apache.hbase	hbase-procedure	1.1.13.200-mapr-632 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-procedure</artifactId> <version>1.1.13.200-mapr-632</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-protocol	1.1.13.200-mapr-632 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-protocol</artifactId> <version>1.1.13.200-mapr-632</version> </dependency>`
org.apache.hbase	hbase-resource-bundle	1.1.13.200-mapr-632 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-resource-bundle</artifactId> <version>1.1.13.200-mapr-632</version> </dependency>`
org.apache.hbase	hbase-rest	1.1.13.200-mapr-632 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-rest</artifactId> <version>1.1.13.200-mapr-632</version> </dependency>`
org.apache.hbase	hbase-server	1.1.13.200-mapr-632 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-server</artifactId> <version>1.1.13.200-mapr-632</version> </dependency>`
org.apache.hbase	hbase-shaded-client	1.1.13.200-mapr-632 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-client</artifactId> <version>1.1.13.200-mapr-632</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-shaded-guava	1.1.13.200-mapr-632 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-guava</artifactId><version>1.1.13.200-mapr-632</version></dependency>`
org.apache.hbase	hbase-shaded-server	1.1.13.200-mapr-632 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-server</artifactId><version>1.1.13.200-mapr-632</version></dependency>`
org.apache.hbase	hbase-shell	1.1.13.200-mapr-632 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shell</artifactId><version>1.1.13.200-mapr-632</version></dependency>`
org.apache.hbase	hbase-spark	1.1.13.200-mapr-632 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-spark</artifactId><version>1.1.13.200-mapr-632</version></dependency>`
org.apache.hbase	hbase-testing-util	1.1.13.200-mapr-632 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-testing-util</artifactId><version>1.1.13.200-mapr-632</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-thrift	1.1.13.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-thrift</artifactId>`
			`<version>1.1.13.200-mapr-632</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-beeline	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-beeline</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-cli	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-cli</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-common	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-common</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-contrib	2.3.6-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.3.6-mapr-2101</version> </dependency>`
org.apache.hive	hive-druid-handler	2.3.6-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-druid-handler</artifactId> <version>2.3.6-mapr-2101</version> </dependency>`
org.apache.hive	hive-exec	2.3.6-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.3.6-mapr-2101</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.3.6-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.3.6-mapr-2101</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	2.3.6-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>2.3.6-mapr-2101</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-jdbc-handler	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc-handler</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-llap-common	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-tez	2.3.6-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.3.6-mapr-2101</version> </dependency>`
org.apache.hive	hive-maprdb-json-common	2.3.6-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-common</artifactId> <version>2.3.6-mapr-2101</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler	2.3.6-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler</artifactId> <version>2.3.6-mapr-2101</version> </dependency>`
org.apache.hive	hive-metastore	2.3.6-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.3.6-mapr-2101</version> </dependency>`
org.apache.hive	hive-serde	2.3.6-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.3.6-mapr-2101</version> </dependency>`
org.apache.hive	hive-service	2.3.6-mapr-2101 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.3.6-mapr-2101</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-service-rpc	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service-rpc</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-shims	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-shims</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-0.23	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.23</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-common	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-testutils	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive	hive-vector-code-gen	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-vector-code-gen</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

org.apache.hive	spark-client	2.3.6-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.3.6-mapr-2101</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.kafka	connect-api	1.1.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-api</artifactId>`
			`<version>1.1.1-mapr-2101</version>`
			`</dependency>`

org.apache.kafka	connect-json	1.1.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-json</artifactId>`
			`<version>1.1.1-mapr-2101</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.kafka	connect-runtime	1.1.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>connect-runtime</artifactId> <version>1.1.1-mapr-2101</version> </dependency>`
org.apache.kafka	connect-transforms	1.1.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>connect-transforms</artifactId> <version>1.1.1-mapr-2101</version> </dependency>`
org.apache.kafka	kafka-clients	1.1.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-clients</artifactId> <version>1.1.1-mapr-2101</version> </dependency>`
org.apache.kafka	kafka-log4j-appender	1.1.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-log4j-appender</artifactId> <version>1.1.1-mapr-2101</version> </dependency>`
org.apache.kafka	kafka-streams	1.1.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-streams</artifactId> <version>1.1.1-mapr-2101</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.kafka	kafka-tools	1.1.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-tools</artifactId>`
			`<version>1.1.1-mapr-2101</version>`
			`</dependency>`

org.apache.kafka	kafka_2.11	1.1.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka_2.11</artifactId>`
			`<version>1.1.1-mapr-2101</version>`
			`</dependency>`

org.apache.kafka	kafka_2.12	1.1.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka_2.12</artifactId>`
			`<version>1.1.1-mapr-2101</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	5.1.0.500-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>5.1.0.500-mapr-632</version>`
			`</dependency>`

org.apache.oozie	oozie-core	5.1.0.500-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>5.1.0.500-mapr-632</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-examples	5.1.0.500-mapr-632 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>5.1.0.500-mapr-632</version> </dependency>`
org.apache.oozie	oozie-fluent-job-api	5.1.0.500-mapr-632 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-api</artifactId> <version>5.1.0.500-mapr-632</version> </dependency>`
org.apache.oozie	oozie-fluent-job-client	5.1.0.500-mapr-632 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-client</artifactId> <version>5.1.0.500-mapr-632</version> </dependency>`
org.apache.oozie.test	oozie-mini	5.1.0.500-mapr-632 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>5.1.0.500-mapr-632</version> </dependency>`
org.apache.oozie	oozie-server	5.1.0.500-mapr-632 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-server</artifactId> <version>5.1.0.500-mapr-632</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-distcp	5.1.0.500-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>5.1.0.500-mapr-632</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-git	5.1.0.500-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-git</artifactId>`
			`<version>5.1.0.500-mapr-632</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	5.1.0.500-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>5.1.0.500-mapr-632</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	5.1.0.500-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>5.1.0.500-mapr-632</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	5.1.0.500-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>5.1.0.500-mapr-632</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-oozie	5.1.0.500-mapr-632 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>5.1.0.500-mapr-632</version> </dependency>`
org.apache.oozie	oozie-sharelib-pig	5.1.0.500-mapr-632 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>5.1.0.500-mapr-632</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	5.1.0.500-mapr-632 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>5.1.0.500-mapr-632</version> </dependency>`
org.apache.oozie	oozie-sharelib-sqoop	5.1.0.500-mapr-632 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-sqoop</artifactId> <version>5.1.0.500-mapr-632</version> </dependency>`
org.apache.oozie	oozie-sharelib-streaming	5.1.0.500-mapr-632 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-streaming</artifactId> <version>5.1.0.500-mapr-632</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-tools	5.1.0.500-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>5.1.0.500-mapr-632</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	5.1.0.500-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>5.1.0.500-mapr-632</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.4.4.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.4.4.200-mapr-632</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.4.4.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.4.4.200-mapr-632</version>`
			`</dependency>`

org.apache.spark	spark-avro_2.11	2.4.4.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-avro_2.11</artifactId>`
			`<version>2.4.4.200-mapr-632</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-catalyst_2.11	2.4.4.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.4.4.200-mapr-632</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.4.4.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.4.4.200-mapr-632</version>`
			`</dependency>`

org.apache.spark	spark-graphx_2.11	2.4.4.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.11</artifactId>`
			`<version>2.4.4.200-mapr-632</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.11	2.4.4.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.4.4.200-mapr-632</version>`
			`</dependency>`

org.apache.spark	spark-hive_2.11	2.4.4.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.4.4.200-mapr-632</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-kvstore_2.11	2.4.4.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-kvstore_2.11</artifactId>`
			`<version>2.4.4.200-mapr-632</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.4.4.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.4.4.200-mapr-632</version>`
			`</dependency>`

org.apache.spark	spark-mesos_2.11	2.4.4.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.11</artifactId>`
			`<version>2.4.4.200-mapr-632</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.4.4.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.4.4.200-mapr-632</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.4.4.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.4.4.200-mapr-632</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-network-common_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-common_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-network-shuffle_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-network-yarn_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-repl_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-sketch_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sql-kafka-0-10_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-sql_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-streaming-flume_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`
org.apache.spark	spark-unsafe_2.11	2.4.4.200-mapr-632 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-unsafe_2.11</artifactId> <version>2.4.4.200-mapr-632</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-yarn_2.11	2.4.4.200-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.4.4.200-mapr-632</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.7-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.7-mapr-632</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.7-mapr-632
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.7-mapr-632</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.9.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.9.1-mapr-2101</version>`
			`</dependency>`

org.apache.tez	hadoop-shim-2.7	0.9.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim-2.7</artifactId>`
			`<version>0.9.1-mapr-2101</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez.conftool	mapr-tez-conf-tool	0.9.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.tez.conftool</groupId> <artifactId>mapr-tez-conf-tool</artifactId> <version>0.9.1-mapr-2101</version> </dependency>`
org.apache.tez	tez-api	0.9.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.9.1-mapr-2101</version> </dependency>`
org.apache.tez	tez-aux-services	0.9.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-aux-services</artifactId> <version>0.9.1-mapr-2101</version> </dependency>`
org.apache.tez	tez-common	0.9.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.9.1-mapr-2101</version> </dependency>`
org.apache.tez	tez-dag	0.9.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.9.1-mapr-2101</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-examples	0.9.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-examples</artifactId>`
			`<version>0.9.1-mapr-2101</version>`
			`</dependency>`

org.apache.tez	tez-ext-service-tests	0.9.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ext-service-tests</artifactId>`
			`<version>0.9.1-mapr-2101</version>`
			`</dependency>`

org.apache.tez	tez-job-analyzer	0.9.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.9.1-mapr-2101</version>`
			`</dependency>`

org.apache.tez	tez-mapreduce	0.9.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.9.1-mapr-2101</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.9.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.9.1-mapr-2101</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-runtime-library	0.9.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-library</artifactId> <version>0.9.1-mapr-2101</version> </dependency>`
org.apache.tez	tez-tests	0.9.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-tests</artifactId> <version>0.9.1-mapr-2101</version> </dependency>`
org.apache.tez	tez-ui	0.9.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui</artifactId> <version>0.9.1-mapr-2101</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history	0.9.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history</artifactId> <version>0.9.1-mapr-2101</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history-with-acls</artifactId> <version>0.9.1-mapr-2101</version> </dependency>`

Maven Artifacts for EEP 6.3.1

Listed are all Maven artifacts for EEP 6.3.1 components.

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-annotations	1.1.13.100-mapr-631 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-annotations</artifactId> <version>1.1.13.100-mapr-631</version> </dependency>`
org.apache.hbase	hbase-checkstyle	1.1.13.100-mapr-631 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-checkstyle</artifactId> <version>1.1.13.100-mapr-631</version> </dependency>`
org.apache.hbase	hbase-client	1.1.13.100-mapr-631 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-client</artifactId> <version>1.1.13.100-mapr-631</version> </dependency>`
org.apache.hbase	hbase-common	1.1.13.100-mapr-631 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-common</artifactId> <version>1.1.13.100-mapr-631</version> </dependency>`
org.apache.hbase	hbase-examples	1.1.13.100-mapr-631 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-examples</artifactId> <version>1.1.13.100-mapr-631</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-hadoop-compat	1.1.13.100-mapr-631 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop-compat</artifactId> <version>1.1.13.100-mapr-631</version> </dependency>`
org.apache.hbase	hbase-hadoop2-compat	1.1.13.100-mapr-631 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop2-compat</artifactId> <version>1.1.13.100-mapr-631</version> </dependency>`
org.apache.hbase	hbase-it	1.1.13.100-mapr-631 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-it</artifactId> <version>1.1.13.100-mapr-631</version> </dependency>`
org.apache.hbase	hbase-native-client	1.1.13.100-mapr-631 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-native-client</artifactId> <version>1.1.13.100-mapr-631</version> </dependency>`
org.apache.hbase	hbase-prefix-tree	1.1.13.100-mapr-631 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-prefix-tree</artifactId> <version>1.1.13.100-mapr-631</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-procedure	1.1.13.100-mapr-631 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-procedure</artifactId><version>1.1.13.100-mapr-631</version></dependency>`
org.apache.hbase	hbase-protocol	1.1.13.100-mapr-631 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-protocol</artifactId><version>1.1.13.100-mapr-631</version></dependency>`
org.apache.hbase	hbase-resource-bundle	1.1.13.100-mapr-631 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-resource-bundle</artifactId><version>1.1.13.100-mapr-631</version></dependency>`
org.apache.hbase	hbase-rest	1.1.13.100-mapr-631 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-rest</artifactId><version>1.1.13.100-mapr-631</version></dependency>`
org.apache.hbase	hbase-server	1.1.13.100-mapr-631 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-server</artifactId><version>1.1.13.100-mapr-631</version></dependency>`
org.apache.hbase	hbase-shaded-client	1.1.13.100-mapr-631 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-client</artifactId><version>1.1.13.100-mapr-631</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-shaded-server	1.1.13.100-mapr-631
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-server</artifactId>`
			`<version>1.1.13.100-mapr-631</version>`
			`</dependency>`

org.apache.hbase	hbase-shell	1.1.13.100-mapr-631
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shell</artifactId>`
			`<version>1.1.13.100-mapr-631</version>`
			`</dependency>`

org.apache.hbase	hbase-spark	1.1.13.100-mapr-631
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-spark</artifactId>`
			`<version>1.1.13.100-mapr-631</version>`
			`</dependency>`

org.apache.hbase	hbase-testing-util	1.1.13.100-mapr-631
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-testing-util</artifactId>`
			`<version>1.1.13.100-mapr-631</version>`
			`</dependency>`

org.apache.hbase	hbase-thrift	1.1.13.100-mapr-631
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-thrift</artifactId>`
			`<version>1.1.13.100-mapr-631</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.3.6-mapr-2009 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-accumulo-handler</artifactId><version>2.3.6-mapr-2009</version></dependency>`
org.apache.hive	hive-beeline	2.3.6-mapr-2009 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-beeline</artifactId><version>2.3.6-mapr-2009</version></dependency>`
org.apache.hive	hive-cli	2.3.6-mapr-2009 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-cli</artifactId><version>2.3.6-mapr-2009</version></dependency>`
org.apache.hive	hive-common	2.3.6-mapr-2009 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-common</artifactId><version>2.3.6-mapr-2009</version></dependency>`
org.apache.hive	hive-contrib	2.3.6-mapr-2009 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-contrib</artifactId><version>2.3.6-mapr-2009</version></dependency>`
org.apache.hive	hive-druid-handler	2.3.6-mapr-2009 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-druid-handler</artifactId><version>2.3.6-mapr-2009</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-exec	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-exec</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-hbase-handler	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-core	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-streaming</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`
org.apache.hive	hive-hplsql	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hplsql</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`
org.apache.hive	hive-jdbc	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`
org.apache.hive	hive-jdbc-handler	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc-handler</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`
org.apache.hive	hive-llap-client	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-client</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-llap-common	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-llap-tez	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-commmon	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-common</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-maprdb-json-handler	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-metastore	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-metastore</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-serde	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-serde</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-service	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-service-rpc	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service-rpc</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-shims	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-shims</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.shims	hive-shims-0.23	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`
org.apache.hive.shims	hive-shims-common	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`
org.apache.hive.shims	hive-shims-scheduler	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-scheduler</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`
org.apache.hive	hive-testutils	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-testutils</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`
org.apache.hive	hive-vector-code-gen	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-vector-code-gen</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.hcatalog	hive-webhcat	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat-java-client</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`
org.apache.hive.conftool	mapr-conf-tool	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.conftool</groupId> <artifactId>mapr-conf-tool</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`
org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.encryptiontool</groupId> <artifactId>mapr-encryption-tool</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`
org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.6-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.maprminicluster</groupId> <artifactId>mapr-mini-cluster</artifactId> <version>2.3.6-mapr-2009</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	spark-client	2.3.6-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.3.6-mapr-2009</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.kafka	connect-api	1.1.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-api</artifactId>`
			`<version>1.1.1-mapr-2009</version>`
			`</dependency>`

org.apache.kafka	connect-json	1.1.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-json</artifactId>`
			`<version>1.1.1-mapr-2009</version>`
			`</dependency>`

org.apache.kafka	connect-runtime	1.1.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-runtime</artifactId>`
			`<version>1.1.1-mapr-2009</version>`
			`</dependency>`

org.apache.kafka	connect-transforms	1.1.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-transforms</artifactId>`
			`<version>1.1.1-mapr-2009</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.kafka	kafka-clients	1.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-clients</artifactId> <version>1.1.1-mapr-2009</version> </dependency>`
org.apache.kafka	kafka-log4j-appender	1.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-log4j-appender</artifactId> <version>1.1.1-mapr-2009</version> </dependency>`
org.apache.kafka	kafka-streams	1.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-streams</artifactId> <version>1.1.1-mapr-2009</version> </dependency>`
org.apache.kafka	kafka-tools	1.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-tools</artifactId> <version>1.1.1-mapr-2009</version> </dependency>`
org.apache.kafka	kafka_2.11	1.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka_2.11</artifactId> <version>1.1.1-mapr-2009</version> </dependency>`
org.apache.kafka	kafka_2.12	1.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka_2.12</artifactId> <version>1.1.1-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-client	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-client</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie	oozie-core	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie	oozie-examples	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie	oozie-fluent-job-api	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-api</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie	oozie-fluent-job-client	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-client</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie.test	oozie-mini	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-server	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-server</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie	oozie-sharelib-distcp	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie	oozie-sharelib-git	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-git</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie	oozie-sharelib-hcatalog	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-hive2	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie	oozie-sharelib-oozie	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie	oozie-sharelib-pig	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie	oozie-sharelib-sqoop	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-sqoop</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-streaming	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-streaming</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie	oozie-tools	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-tools</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`
org.apache.oozie	oozie-webapp	5.1.0.400-mapr-631 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-webapp</artifactId> <version>5.1.0.400-mapr-631</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.pig	pig	0.16.0-mapr-2009 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>pig</artifactId> <version>0.16.0-mapr-2009</version> </dependency>`
org.apache.pig	piggybank	0.16.0-mapr-2009 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>piggybank</artifactId> <version>0.16.0-mapr-2009</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.pig	pigsmoke	0.16.0-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigsmoke</artifactId>`
			`<version>0.16.0-mapr-2009</version>`
			`</dependency>`

org.apache.pig	pigunit	0.16.0-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigunit</artifactId>`
			`<version>0.16.0-mapr-2009</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.4.4.100-mapr-631
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.4.4.100-mapr-631</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.4.4.100-mapr-631
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.4.4.100-mapr-631</version>`
			`</dependency>`

org.apache.spark	spark-avro_2.11	2.4.4.100-mapr-631
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-avro_2.11</artifactId>`
			`<version>2.4.4.100-mapr-631</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-catalyst_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-catalyst_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-core_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-graphx_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-hive_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-kvstore_2.11	2.4.4.100-mapr-631
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-kvstore_2.11</artifactId>`
			`<version>2.4.4.100-mapr-631</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.4.4.100-mapr-631
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.4.4.100-mapr-631</version>`
			`</dependency>`

org.apache.spark	spark-mesos_2.11	2.4.4.100-mapr-631
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.11</artifactId>`
			`<version>2.4.4.100-mapr-631</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.4.4.100-mapr-631
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.4.4.100-mapr-631</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.4.4.100-mapr-631
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.4.4.100-mapr-631</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-network-common_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-common_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-network-shuffle_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-network-yarn_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-repl_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-sketch_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sql-kafka-0-10_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-sql_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-streaming-flume_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`
org.apache.spark	spark-unsafe_2.11	2.4.4.100-mapr-631 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-unsafe_2.11</artifactId> <version>2.4.4.100-mapr-631</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-yarn_2.11	2.4.4.100-mapr-631
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.4.4.100-mapr-631</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.7-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.7-mapr-2009</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.7-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.7-mapr-2009</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.9.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.9.1-mapr-2009</version>`
			`</dependency>`

org.apache.tez	hadoop-shim-2.7	0.9.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim-2.7</artifactId>`
			`<version>0.9.1-mapr-2009</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez.conftool	mapr-tez-conf-tool	0.9.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez.conftool</groupId>`
			`<artifactId>mapr-tez-conf-tool</artifactId>`
			`<version>0.9.1-mapr-2009</version>`
			`</dependency>`

org.apache.tez	tez-api	0.9.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-api</artifactId>`
			`<version>0.9.1-mapr-2009</version>`
			`</dependency>`

org.apache.tez	tez-aux-services	0.9.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-aux-services</artifactId>`
			`<version>0.9.1-mapr-2009</version>`
			`</dependency>`

org.apache.tez	tez-common	0.9.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-common</artifactId>`
			`<version>0.9.1-mapr-2009</version>`
			`</dependency>`

org.apache.tez	tez-dag	0.9.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-dag</artifactId>`
			`<version>0.9.1-mapr-2009</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-examples	0.9.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.9.1-mapr-2009</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.9.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.9.1-mapr-2009</version> </dependency>`
org.apache.tez	tez-job-analyzer	0.9.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-job-analyzer</artifactId> <version>0.9.1-mapr-2009</version> </dependency>`
org.apache.tez	tez-mapreduce	0.9.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-mapreduce</artifactId> <version>0.9.1-mapr-2009</version> </dependency>`
org.apache.tez	tez-runtime-internals	0.9.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-internals</artifactId> <version>0.9.1-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-runtime-library	0.9.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-library</artifactId> <version>0.9.1-mapr-2009</version> </dependency>`
org.apache.tez	tez-tests	0.9.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-tests</artifactId> <version>0.9.1-mapr-2009</version> </dependency>`
org.apache.tez	tez-ui	0.9.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui</artifactId> <version>0.9.1-mapr-2009</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history	0.9.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history</artifactId> <version>0.9.1-mapr-2009</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history-with-acls</artifactId> <version>0.9.1-mapr-2009</version> </dependency>`

Maven Artifacts for EEP 6.3.0

Listed are all Maven artifacts for EEP 6.3.0 components.

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-annotations	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-annotations</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

org.apache.hbase	hbase-checkstyle	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-checkstyle</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

org.apache.hbase	hbase-client	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

org.apache.hbase	hbase-common	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-common</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

org.apache.hbase	hbase-examples	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-examples</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-hadoop-compat	1.1.13.0-mapr-630 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-hadoop-compat</artifactId><version>1.1.13.0-mapr-630</version></dependency>`
org.apache.hbase	hbase-hadoop2-compat	1.1.13.0-mapr-630 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-hadoop2-compat</artifactId><version>1.1.13.0-mapr-630</version></dependency>`
org.apache.hbase	hbase-it	1.1.13.0-mapr-630 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-it</artifactId><version>1.1.13.0-mapr-630</version></dependency>`
org.apache.hbase	hbase-native-client	1.1.13.0-mapr-630 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-native-client</artifactId><version>1.1.13.0-mapr-630</version></dependency>`
org.apache.hbase	hbase-prefix-tree	1.1.13.0-mapr-630 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-prefix-tree</artifactId><version>1.1.13.0-mapr-630</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-procedure	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-procedure</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

org.apache.hbase	hbase-protocol	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-protocol</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

org.apache.hbase	hbase-resource-bundle	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-resource-bundle</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

org.apache.hbase	hbase-rest	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-rest</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

org.apache.hbase	hbase-server	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-server</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

org.apache.hbase	hbase-shaded-client	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-client</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-shaded-server	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-server</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

org.apache.hbase	hbase-shell	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shell</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

org.apache.hbase	hbase-spark	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-spark</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

org.apache.hbase	hbase-testing-util	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-testing-util</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

org.apache.hbase	hbase-thrift	1.1.13.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-thrift</artifactId>`
			`<version>1.1.13.0-mapr-630</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-accumulo-handler</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`
org.apache.hive	hive-beeline	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`
org.apache.hive	hive-cli	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`
org.apache.hive	hive-common	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`
org.apache.hive	hive-contrib	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`
org.apache.hive	hive-druid-handler	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-druid-handler</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-exec	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-exec</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-hbase-handler	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-core	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-jdbc-handler	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc-handler</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-llap-common	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-llap-tez	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-commmon	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-common</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-maprdb-json-handler	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-metastore	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-metastore</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-serde	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-serde</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-service	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-service-rpc	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service-rpc</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-shims	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-shims</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.shims	hive-shims-0.23	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`
org.apache.hive.shims	hive-shims-common	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`
org.apache.hive.shims	hive-shims-scheduler	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-scheduler</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`
org.apache.hive	hive-testutils	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-testutils</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`
org.apache.hive	hive-vector-code-gen	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-vector-code-gen</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.hcatalog	hive-webhcat	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat-java-client</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`
org.apache.hive.conftool	mapr-conf-tool	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive.conftool</groupId> <artifactId>mapr-conf-tool</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`
org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive.encryptiontool</groupId> <artifactId>mapr-encryption-tool</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`
org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.6-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive.maprminicluster</groupId> <artifactId>mapr-mini-cluster</artifactId> <version>2.3.6-mapr-1912</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	spark-client	2.3.6-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.3.6-mapr-1912</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.kafka	connect-api	1.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-api</artifactId>`
			`<version>1.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.kafka	connect-json	1.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-json</artifactId>`
			`<version>1.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.kafka	connect-runtime	1.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-runtime</artifactId>`
			`<version>1.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.kafka	connect-transforms	1.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-transforms</artifactId>`
			`<version>1.1.1-mapr-1912</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.kafka	kafka-clients	1.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-clients</artifactId> <version>1.1.1-mapr-1912</version> </dependency>`
org.apache.kafka	kafka-log4j-appender	1.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-log4j-appender</artifactId> <version>1.1.1-mapr-1912</version> </dependency>`
org.apache.kafka	kafka-streams	1.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-streams</artifactId> <version>1.1.1-mapr-1912</version> </dependency>`
org.apache.kafka	kafka-tools	1.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-tools</artifactId> <version>1.1.1-mapr-1912</version> </dependency>`
org.apache.kafka	kafka_2.11	1.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka_2.11</artifactId> <version>1.1.1-mapr-1912</version> </dependency>`
org.apache.kafka	kafka_2.12	1.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka_2.12</artifactId> <version>1.1.1-mapr-1912</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate

com.mapr.ojai	mapr-ojai-driver-thin	1.0.0-mapr
		Browse
			`<dependency>`
			`<groupId>com.mapr.ojai</groupId>`
			`<artifactId>mapr-ojai-driver-thin</artifactId>`
			`<version>1.0.0-mapr</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	5.1.0.300-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>5.1.0.300-mapr-630</version>`
			`</dependency>`

org.apache.oozie	oozie-core	5.1.0.300-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>5.1.0.300-mapr-630</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	5.1.0.300-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>5.1.0.300-mapr-630</version>`
			`</dependency>`

org.apache.oozie	oozie-fluent-job-api	5.1.0.300-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-fluent-job-api</artifactId>`
			`<version>5.1.0.300-mapr-630</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-fluent-job-client	5.1.0.300-mapr-630 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-client</artifactId> <version>5.1.0.300-mapr-630</version> </dependency>`
org.apache.oozie.test	oozie-mini	5.1.0.300-mapr-630 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>5.1.0.300-mapr-630</version> </dependency>`
org.apache.oozie	oozie-server	5.1.0.300-mapr-630 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-server</artifactId> <version>5.1.0.300-mapr-630</version> </dependency>`
org.apache.oozie	oozie-sharelib-distcp	5.1.0.300-mapr-630 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>5.1.0.300-mapr-630</version> </dependency>`
org.apache.oozie	oozie-sharelib-git	5.1.0.300-mapr-630 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-git</artifactId> <version>5.1.0.300-mapr-630</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-hcatalog	5.1.0.300-mapr-630 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>5.1.0.300-mapr-630</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive	5.1.0.300-mapr-630 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>5.1.0.300-mapr-630</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive2	5.1.0.300-mapr-630 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>5.1.0.300-mapr-630</version> </dependency>`
org.apache.oozie	oozie-sharelib-oozie	5.1.0.300-mapr-630 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>5.1.0.300-mapr-630</version> </dependency>`
org.apache.oozie	oozie-sharelib-pig	5.1.0.300-mapr-630 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>5.1.0.300-mapr-630</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-spark	5.1.0.300-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>5.1.0.300-mapr-630</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	5.1.0.300-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>5.1.0.300-mapr-630</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	5.1.0.300-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>5.1.0.300-mapr-630</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	5.1.0.300-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>5.1.0.300-mapr-630</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.pig	pig	0.16.0-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pig</artifactId>`
			`<version>0.16.0-mapr-1912</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.pig	piggybank	0.16.0-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>piggybank</artifactId>`
			`<version>0.16.0-mapr-1912</version>`
			`</dependency>`

org.apache.pig	pigsmoke	0.16.0-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigsmoke</artifactId>`
			`<version>0.16.0-mapr-1912</version>`
			`</dependency>`

org.apache.pig	pigunit	0.16.0-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigunit</artifactId>`
			`<version>0.16.0-mapr-1912</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-avro_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-avro_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-graphx_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-hive_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-kvstore_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-kvstore_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-mesos_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-mllib_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-sketch_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-sql-kafka-0-10_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-sql_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-assembly_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-flume_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.4.4.0-mapr-630 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.4.4.0-mapr-630</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.4.4.0-mapr-630 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.4.4.0-mapr-630</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.4.4.0-mapr-630 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.4.4.0-mapr-630</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.4.4.0-mapr-630 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.4.4.0-mapr-630</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.4.4.0-mapr-630 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.4.4.0-mapr-630</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-unsafe_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.4.4.0-mapr-630
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.4.4.0-mapr-630</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.9.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.9.1-mapr-1912</version>`
			`</dependency>`

org.apache.tez	hadoop-shim-2.7	0.9.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim-2.7</artifactId>`
			`<version>0.9.1-mapr-1912</version>`
			`</dependency>`

org.apache.tez.conftool	mapr-tez-conf-tool	0.9.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.tez.conftool</groupId>`
			`<artifactId>mapr-tez-conf-tool</artifactId>`
			`<version>0.9.1-mapr-1912</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-api	0.9.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.9.1-mapr-1912</version> </dependency>`
org.apache.tez	tez-aux-services	0.9.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-aux-services</artifactId> <version>0.9.1-mapr-1912</version> </dependency>`
org.apache.tez	tez-common	0.9.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.9.1-mapr-1912</version> </dependency>`
org.apache.tez	tez-dag	0.9.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.9.1-mapr-1912</version> </dependency>`
org.apache.tez	tez-examples	0.9.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.9.1-mapr-1912</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.9.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.9.1-mapr-1912</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-job-analyzer	0.9.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.9.1-mapr-1912</version>`
			`</dependency>`

org.apache.tez	tez-mapreduce	0.9.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.9.1-mapr-1912</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.9.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.9.1-mapr-1912</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.9.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.9.1-mapr-1912</version>`
			`</dependency>`

org.apache.tez	tez-tests	0.9.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-tests</artifactId>`
			`<version>0.9.1-mapr-1912</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ui	0.9.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui</artifactId>`
			`<version>0.9.1-mapr-1912</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history	0.9.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.9.1-mapr-1912</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.9.1-mapr-1912</version>`
			`</dependency>`

Maven Artifacts for EEP 6.2.0

Listed are all Maven artifacts for EEP 6.2.0 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-legacy-sources	flume-avro-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-avro-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume	flume-checkstyle	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-checkstyle</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-dataset-sink	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-dataset-sink</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`
org.apache.flume.flume-ng-channels	flume-file-channel	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-file-channel</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-hdfs-sink	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-hdfs-sink</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-hive-sink	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-hive-sink</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-http-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-http-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-irc-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-irc-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-jdbc-channel	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-jdbc-channel</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-jms-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-jms-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-kafka-channel	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-kafka-channel</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-kafka-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-auth	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-auth</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-configuration	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-configuration</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-core	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-core</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-elasticsearch-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-elasticsearch-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume	flume-ng-embedded-agent	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-embedded-agent</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-hbase-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-hbase-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-kafka-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-clients	flume-ng-log4jappender	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-clients</groupId>`
			`<artifactId>flume-ng-log4jappender</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-morphline-solr-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-morphline-solr-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume	flume-ng-node	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-node</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-sdk	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-sdk</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-tests	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-tests</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-scribe-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-scribe-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-shared	flume-shared-kafka-test	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-shared</groupId>`
			`<artifactId>flume-shared-kafka-test</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-channels	flume-spillable-memory-channel	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-spillable-memory-channel</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-taildir-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-taildir-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-legacy-sources	flume-thrift-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-thrift-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-tools	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-tools</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-twitter-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-twitter-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-annotations	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-annotations</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-checkstyle	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-checkstyle</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-client	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-common	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-common</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-examples	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-examples</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-hadoop-compat	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-hadoop-compat</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-hadoop2-compat	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-hadoop2-compat</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-it	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-it</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-native-client	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-native-client</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-prefix-tree	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-prefix-tree</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-procedure	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-procedure</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-protocol	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-protocol</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-resource-bundle	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-resource-bundle</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-rest	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-rest</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-server	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-server</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-shaded-client	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-client</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-shaded-server	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-server</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-shell	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shell</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-spark	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-spark</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-testing-util	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-testing-util</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-thrift	1.1.8-mapr-mep-6.2-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-thrift</artifactId>`
			`<version>1.1.8-mapr-mep-6.2-1904</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-beeline	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-beeline</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-cli	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-cli</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-common	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-common</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-contrib	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-druid-handler	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-druid-handler</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-exec	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-jdbc-handler	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc-handler</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-common	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-llap-tez	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-metastore	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-metastore</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-serde	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-serde</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-service	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-service-rpc	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service-rpc</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-shims	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive.shims	hive-shims-common	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive.shims	hive-shims-scheduler	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-scheduler</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-testutils	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-testutils</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-vector-code-gen	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-vector-code-gen</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	spark-client	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	5.1.0.200-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>5.1.0.200-mapr-620</version>`
			`</dependency>`

org.apache.oozie	oozie-core	5.1.0.200-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>5.1.0.200-mapr-620</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	5.1.0.200-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>5.1.0.200-mapr-620</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-fluent-job-api	5.1.0.200-mapr-620 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-api</artifactId> <version>5.1.0.200-mapr-620</version> </dependency>`
org.apache.oozie	oozie-fluent-job-client	5.1.0.200-mapr-620 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-client</artifactId> <version>5.1.0.200-mapr-620</version> </dependency>`
org.apache.oozie.test	oozie-mini	5.1.0.200-mapr-620 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>5.1.0.200-mapr-620</version> </dependency>`
org.apache.oozie	oozie-server	5.1.0.200-mapr-620 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-server</artifactId> <version>5.1.0.200-mapr-620</version> </dependency>`
org.apache.oozie	oozie-sharelib-distcp	5.1.0.200-mapr-620 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>5.1.0.200-mapr-620</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-git	5.1.0.200-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-git</artifactId>`
			`<version>5.1.0.200-mapr-620</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	5.1.0.200-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>5.1.0.200-mapr-620</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	5.1.0.200-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>5.1.0.200-mapr-620</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	5.1.0.200-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>5.1.0.200-mapr-620</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	5.1.0.200-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>5.1.0.200-mapr-620</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-pig	5.1.0.200-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>5.1.0.200-mapr-620</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	5.1.0.200-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>5.1.0.200-mapr-620</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	5.1.0.200-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>5.1.0.200-mapr-620</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	5.1.0.200-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>5.1.0.200-mapr-620</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	5.1.0.200-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>5.1.0.200-mapr-620</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-webapp	5.1.0.200-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>5.1.0.200-mapr-620</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-avro_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-avro_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-core_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-graphx_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-hive_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-kvstore_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-kvstore_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-launcher_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-mesos_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-network-shuffle_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-sketch_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-sql-kafka-0-10_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-sql_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-assembly_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.4.0.0-mapr-620 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.4.0.0-mapr-620</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.4.0.0-mapr-620 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.4.0.0-mapr-620</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.4.0.0-mapr-620 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.4.0.0-mapr-620</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.4.0.0-mapr-620 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.4.0.0-mapr-620</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.4.0.0-mapr-620 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.4.0.0-mapr-620</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-producer_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-streaming_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-tags_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.4.0.0-mapr-620
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.4.0.0-mapr-620</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sqoop	sqoop	1.4.7-mapr-1904 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop</artifactId> <version>1.4.7-mapr-1904</version> </dependency>`
org.apache.sqoop	sqoop-test	1.4.7-mapr-1904 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-test</artifactId> <version>1.4.7-mapr-1904</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	hadoop-shim	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	hadoop-shim-2.7	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim-2.7</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez.conftool	mapr-tez-conf-tool	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez.conftool</groupId> <artifactId>mapr-tez-conf-tool</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-api	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-aux-services	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-aux-services</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-common	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-dag	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-examples	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-job-analyzer	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

org.apache.tez	tez-mapreduce	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

org.apache.tez	tez-tests	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-tests</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ui	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

Maven Artifacts for EEP 6.1.1

Listed are all Maven artifacts for EEP 6.1.1 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-legacy-sources	flume-avro-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-avro-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume	flume-checkstyle	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-checkstyle</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-dataset-sink	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-dataset-sink</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`
org.apache.flume.flume-ng-channels	flume-file-channel	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-file-channel</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-hdfs-sink	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-hdfs-sink</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-hive-sink	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-hive-sink</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-http-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-http-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-irc-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-irc-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-jdbc-channel	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-jdbc-channel</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-jms-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-jms-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-kafka-channel	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-kafka-channel</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-kafka-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-auth	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-auth</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-configuration	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-configuration</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-core	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-core</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-elasticsearch-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-elasticsearch-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume	flume-ng-embedded-agent	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-embedded-agent</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-hbase-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-hbase-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-kafka-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-clients	flume-ng-log4jappender	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-clients</groupId>`
			`<artifactId>flume-ng-log4jappender</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-morphline-solr-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-morphline-solr-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume	flume-ng-node	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-node</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-sdk	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-sdk</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-tests	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-tests</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-scribe-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-scribe-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-shared	flume-shared-kafka-test	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-shared</groupId>`
			`<artifactId>flume-shared-kafka-test</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-channels	flume-spillable-memory-channel	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-spillable-memory-channel</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`
org.apache.flume.flume-ng-sources	flume-taildir-source	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sources</groupId> <artifactId>flume-taildir-source</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`
org.apache.flume.flume-ng-legacy-sources	flume-thrift-source	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-legacy-sources</groupId> <artifactId>flume-thrift-source</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`
org.apache.flume	flume-tools	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-tools</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`
org.apache.flume.flume-ng-sources	flume-twitter-source	1.8.0-mapr-mep-6.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sources</groupId> <artifactId>flume-twitter-source</artifactId> <version>1.8.0-mapr-mep-6.x-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-annotations	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-annotations</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-checkstyle	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-checkstyle</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-client	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-client</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-common	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-common</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-examples	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-examples</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-hadoop-compat	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop-compat</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-hadoop2-compat	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop2-compat</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-it	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-it</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-native-client	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-native-client</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-prefix-tree	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-prefix-tree</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-procedure	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-procedure</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-protocol	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-protocol</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-resource-bundle	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-resource-bundle</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-rest	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-rest</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-server	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-server</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-shaded-client	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-client</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-shaded-server	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-server</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-shell	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shell</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-spark	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-spark</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-testing-util	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-testing-util</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-thrift	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-thrift</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-accumulo-handler</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-beeline	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-cli	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-common	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-contrib	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-druid-handler	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-druid-handler</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-exec	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-exec</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-hbase-handler	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-core	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-jdbc-handler	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc-handler</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-llap-common	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-tez	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-metastore	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-serde	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-service	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-service-rpc	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-shims	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.shims	hive-shims-common	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-vector-code-gen	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-vector-code-gen</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	spark-client	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-client	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-client</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie	oozie-core	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie	oozie-examples	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie	oozie-fluent-job-api	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-api</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie	oozie-fluent-job-client	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-fluent-job-client</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie.test	oozie-mini	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-server	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-server</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie	oozie-sharelib-distcp	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie	oozie-sharelib-git	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-git</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie	oozie-sharelib-hcatalog	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-hive2	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie	oozie-sharelib-oozie	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie	oozie-sharelib-pig	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie	oozie-sharelib-sqoop	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-sqoop</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-streaming	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-streaming</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie	oozie-tools	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-tools</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`
org.apache.oozie	oozie-webapp	5.1.0.100-mapr-611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-webapp</artifactId> <version>5.1.0.100-mapr-611</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	classpath-filter_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>classpath-filter_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
com.mapr.db	maprdb-spark	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-spark</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-catalyst_2.11	2.3.3.100-mapr-611 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-catalyst_2.11</artifactId><version>2.3.3.100-mapr-611</version></dependency>`
org.apache.spark	spark-core_2.11	2.3.3.100-mapr-611 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-core_2.11</artifactId><version>2.3.3.100-mapr-611</version></dependency>`
org.apache.spark	spark-graphx_2.11	2.3.3.100-mapr-611 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-graphx_2.11</artifactId><version>2.3.3.100-mapr-611</version></dependency>`
org.apache.spark	spark-hive-thriftserver_2.11	2.3.3.100-mapr-611 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-hive-thriftserver_2.11</artifactId><version>2.3.3.100-mapr-611</version></dependency>`
org.apache.spark	spark-hive_2.11	2.3.3.100-mapr-611 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-hive_2.11</artifactId><version>2.3.3.100-mapr-611</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-kvstore_2.11	2.3.3.100-mapr-611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-kvstore_2.11</artifactId>`
			`<version>2.3.3.100-mapr-611</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.3.3.100-mapr-611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.3.3.100-mapr-611</version>`
			`</dependency>`

org.apache.spark	spark-mesos_2.11	2.3.3.100-mapr-611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.11</artifactId>`
			`<version>2.3.3.100-mapr-611</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.3.3.100-mapr-611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.3.3.100-mapr-611</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.3.3.100-mapr-611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.3.3.100-mapr-611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-network-common_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-common_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-network-shuffle_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-network-yarn_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-repl_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-sketch_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sql-kafka-0-10_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-sql_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-streaming-flume_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`
org.apache.spark	spark-unsafe_2.11	2.3.3.100-mapr-611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-unsafe_2.11</artifactId> <version>2.3.3.100-mapr-611</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-yarn_2.11	2.3.3.100-mapr-611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.3.3.100-mapr-611</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.7-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.7-mapr-1904</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.7-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.7-mapr-1904</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

org.apache.tez	hadoop-shim-2.7	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim-2.7</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez.conftool	mapr-tez-conf-tool	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez.conftool</groupId> <artifactId>mapr-tez-conf-tool</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-api	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-aux-services	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-aux-services</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-common	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-dag	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-examples	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-job-analyzer	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-job-analyzer</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-mapreduce	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-mapreduce</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-runtime-internals	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-internals</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-runtime-library	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-library</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-tests	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-tests</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-ui	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history-with-acls</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`

Maven Artifacts for EEP 6.1.0

Listed are all Maven artifacts for EEP 6.1.0 components.

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-legacy-sources	flume-avro-source	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-legacy-sources</groupId> <artifactId>flume-avro-source</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume	flume-checkstyle	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-checkstyle</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-dataset-sink	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-dataset-sink</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-channels	flume-file-channel	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-file-channel</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-hdfs-sink	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-hdfs-sink</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-sinks	flume-hive-sink	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-hive-sink</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-http-sink	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-http-sink</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-irc-sink	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-irc-sink</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-channels	flume-jdbc-channel	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-jdbc-channel</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-sources	flume-jms-source	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sources</groupId> <artifactId>flume-jms-source</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-channels	flume-kafka-channel	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-kafka-channel</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-kafka-source	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume	flume-ng-auth	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-auth</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume	flume-ng-configuration	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-configuration</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume	flume-ng-core	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-core</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-sinks	flume-ng-elasticsearch-sink	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-elasticsearch-sink</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume	flume-ng-embedded-agent	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-ng-embedded-agent</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-hbase-sink	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-hbase-sink</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-kafka-sink</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-clients	flume-ng-log4jappender	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-clients</groupId> <artifactId>flume-ng-log4jappender</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-ng-morphline-solr-sink	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-morphline-solr-sink</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume	flume-ng-node	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-node</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume	flume-ng-sdk	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-sdk</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume	flume-ng-tests	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-tests</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-scribe-source	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-scribe-source</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-shared	flume-shared-kafka-test	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-shared</groupId>`
			`<artifactId>flume-shared-kafka-test</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-spillable-memory-channel	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-spillable-memory-channel</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-taildir-source	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-taildir-source</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume.flume-ng-legacy-sources	flume-thrift-source	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-thrift-source</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume	flume-tools	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-tools</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-twitter-source	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-twitter-source</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-annotations	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-annotations</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

org.apache.hbase	hbase-checkstyle	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-checkstyle</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

org.apache.hbase	hbase-client	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

org.apache.hbase	hbase-common	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-common</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-examples	1.1.8-mapr-1901 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-examples</artifactId> <version>1.1.8-mapr-1901</version> </dependency>`
org.apache.hbase	hbase-hadoop-compat	1.1.8-mapr-1901 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop-compat</artifactId> <version>1.1.8-mapr-1901</version> </dependency>`
org.apache.hbase	hbase-hadoop2-compat	1.1.8-mapr-1901 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop2-compat</artifactId> <version>1.1.8-mapr-1901</version> </dependency>`
org.apache.hbase	hbase-it	1.1.8-mapr-1901 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-it</artifactId> <version>1.1.8-mapr-1901</version> </dependency>`
org.apache.hbase	hbase-native-client	1.1.8-mapr-1901 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-native-client</artifactId> <version>1.1.8-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-prefix-tree	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-prefix-tree</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-procedure	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-procedure</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-protocol	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-protocol</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-resource-bundle	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-resource-bundle</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-rest	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-rest</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-server	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-server</artifactId><version>1.1.8-mapr-1901</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-shaded-client	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-client</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-shaded-server	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-server</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-shell	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shell</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-spark	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-spark</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-testing-util	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-testing-util</artifactId><version>1.1.8-mapr-1901</version></dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-thrift	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-thrift</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-beeline	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-beeline</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-cli	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-cli</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-common	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-common</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-contrib	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-druid-handler	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-druid-handler</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-exec	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-jdbc-handler	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc-handler</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-common	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-llap-tez	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-metastore	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-metastore</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-serde	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-serde</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-service	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-service-rpc	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service-rpc</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-shims	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive.shims	hive-shims-common	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive.shims	hive-shims-scheduler	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-scheduler</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-testutils	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-testutils</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-vector-code-gen	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-vector-code-gen</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	spark-client	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.kafka	connect-api	1.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-api</artifactId>`
			`<version>1.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.kafka	connect-json	1.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-json</artifactId>`
			`<version>1.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.kafka	connect-runtime	1.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-runtime</artifactId>`
			`<version>1.1.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.kafka	connect-transforms	1.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-transforms</artifactId>`
			`<version>1.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.kafka	kafka-clients	1.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-clients</artifactId>`
			`<version>1.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.kafka	kafka-log4j-appender	1.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-log4j-appender</artifactId>`
			`<version>1.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.kafka	kafka-streams	1.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-streams</artifactId>`
			`<version>1.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.kafka	kafka-tools	1.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka-tools</artifactId>`
			`<version>1.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.kafka	kafka_2.11	1.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka_2.11</artifactId>`
			`<version>1.1.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.kafka	kafka_2.12	1.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka_2.12</artifactId>`
			`<version>1.1.1-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	5.1.0.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>5.1.0.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-core	5.1.0.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>5.1.0.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	5.1.0.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>5.1.0.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-fluent-job-api	5.1.0.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-fluent-job-api</artifactId>`
			`<version>5.1.0.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-fluent-job-client	5.1.0.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-fluent-job-client</artifactId>`
			`<version>5.1.0.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	5.1.0.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>5.1.0.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-server	5.1.0.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-server</artifactId>`
			`<version>5.1.0.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-distcp	5.1.0.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>5.1.0.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-git	5.1.0.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-git</artifactId>`
			`<version>5.1.0.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-hcatalog	5.1.0.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>5.1.0.0-mapr-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive	5.1.0.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>5.1.0.0-mapr-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive2	5.1.0.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>5.1.0.0-mapr-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-oozie	5.1.0.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>5.1.0.0-mapr-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-pig	5.1.0.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>5.1.0.0-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-spark	5.1.0.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>5.1.0.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	5.1.0.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>5.1.0.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	5.1.0.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>5.1.0.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	5.1.0.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>5.1.0.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	5.1.0.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>5.1.0.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.pig	pig	0.16.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pig</artifactId>`
			`<version>0.16.0-mapr-1901</version>`
			`</dependency>`

org.apache.pig	piggybank	0.16.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>piggybank</artifactId>`
			`<version>0.16.0-mapr-1901</version>`
			`</dependency>`

org.apache.pig	pigsmoke	0.16.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigsmoke</artifactId>`
			`<version>0.16.0-mapr-1901</version>`
			`</dependency>`

org.apache.pig	pigunit	0.16.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigunit</artifactId>`
			`<version>0.16.0-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-hive-common	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-common</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sentry	sentry-binding-hive-v2	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-binding-hive-v2</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`
org.apache.sentry	sentry-binding-kafka	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-binding-kafka</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`
org.apache.sentry	sentry-binding-solr	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-binding-solr</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`
org.apache.sentry	sentry-binding-sqoop	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-binding-sqoop</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`
org.apache.sentry	sentry-core-common	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-core-common</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-db	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-kafka	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-kafka</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-sqoop	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sentry	sentry-dist	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-dist</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`
org.apache.sentry	sentry-policy-common	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-policy-common</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`
org.apache.sentry	sentry-policy-db	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-policy-db</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`
org.apache.sentry	sentry-policy-indexer	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-policy-indexer</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`
org.apache.sentry	sentry-policy-kafka	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-policy-kafka</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-search	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-db	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-file	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-core	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-core</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-handlers	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-catalyst_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-graphx_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-hive_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-kvstore_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-kvstore_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-network-shuffle_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-network-yarn_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-repl_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-sketch_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sql_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-unsafe_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	hadoop-shim-2.7	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim-2.7</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-api	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-api</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-aux-services	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-aux-services</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-common	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-common</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-dag	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-dag</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-examples	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-examples</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-ext-service-tests	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ext-service-tests</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-job-analyzer	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-mapreduce	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-tests	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-tests</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ui	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

Maven Artifacts for EEP 6.0.2

Listed are all Maven artifacts for EEP 6.0.2 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-legacy-sources	flume-avro-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-avro-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume	flume-checkstyle	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-checkstyle</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-dataset-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-dataset-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-file-channel	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-file-channel</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-hdfs-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-hdfs-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-hive-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-hive-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-http-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-http-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-irc-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-irc-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-jdbc-channel	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-jdbc-channel</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-jms-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-jms-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-kafka-channel	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-kafka-channel</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-kafka-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-auth	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-auth</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-configuration	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-configuration</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-core	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-core</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-elasticsearch-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-elasticsearch-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume	flume-ng-embedded-agent	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-embedded-agent</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-hbase-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-hbase-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-kafka-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-clients	flume-ng-log4jappender	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-clients</groupId>`
			`<artifactId>flume-ng-log4jappender</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-morphline-solr-sink	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-morphline-solr-sink</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume	flume-ng-node	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-node</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-sdk	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-sdk</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-tests	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-tests</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-scribe-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-scribe-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-shared	flume-shared-kafka-test	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-shared</groupId>`
			`<artifactId>flume-shared-kafka-test</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-channels	flume-spillable-memory-channel	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-spillable-memory-channel</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-taildir-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-taildir-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-legacy-sources	flume-thrift-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-thrift-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-tools	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-tools</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-twitter-source	1.8.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-twitter-source</artifactId>`
			`<version>1.8.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-annotations	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-annotations</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-checkstyle	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-checkstyle</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-client	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-client</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-common	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-common</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-examples	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-examples</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-hadoop-compat	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop-compat</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-hadoop2-compat	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop2-compat</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-it	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-it</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-native-client	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-native-client</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-prefix-tree	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-prefix-tree</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-procedure	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-procedure</artifactId><version>1.1.8-mapr-1904</version></dependency>`
org.apache.hbase	hbase-protocol	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-protocol</artifactId><version>1.1.8-mapr-1904</version></dependency>`
org.apache.hbase	hbase-resource-bundle	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-resource-bundle</artifactId><version>1.1.8-mapr-1904</version></dependency>`
org.apache.hbase	hbase-rest	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-rest</artifactId><version>1.1.8-mapr-1904</version></dependency>`
org.apache.hbase	hbase-server	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-server</artifactId><version>1.1.8-mapr-1904</version></dependency>`
org.apache.hbase	hbase-shaded-client	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-client</artifactId><version>1.1.8-mapr-1904</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-shaded-server	1.1.8-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-server</artifactId>`
			`<version>1.1.8-mapr-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-shell	1.1.8-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shell</artifactId>`
			`<version>1.1.8-mapr-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-spark	1.1.8-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-spark</artifactId>`
			`<version>1.1.8-mapr-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-testing-util	1.1.8-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-testing-util</artifactId>`
			`<version>1.1.8-mapr-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-thrift	1.1.8-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-thrift</artifactId>`
			`<version>1.1.8-mapr-1904</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-accumulo-handler</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-beeline	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-cli	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-common	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-contrib	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-druid-handler	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-druid-handler</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-exec	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-exec</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-hbase-handler	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-core	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-jdbc-handler	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc-handler</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-common	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-llap-ext-client	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-ext-client</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-llap-server	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-server</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-llap-tez	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-metastore	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-serde	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-service	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-service-rpc	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive	hive-shims	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	2.3.3-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.3.3-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.shims	hive-shims-common	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-vector-code-gen	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-vector-code-gen</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

org.apache.hive	spark-client	2.3.3-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.3.3-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-client	4.3.0-mapr-mep-6.x-1904 Browse	<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-client</artifactId> <version>4.3.0-mapr-mep-6.x-1904</version> </dependency>
org.apache.oozie	oozie-core	4.3.0-mapr-mep-6.x-1904 Browse	<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>4.3.0-mapr-mep-6.x-1904</version> </dependency>
org.apache.oozie	oozie-examples	4.3.0-mapr-mep-6.x-1904 Browse	<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>4.3.0-mapr-mep-6.x-1904</version> </dependency>
org.apache.oozie.test	oozie-mini	4.3.0-mapr-mep-6.x-1904 Browse	<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>4.3.0-mapr-mep-6.x-1904</version> </dependency>
org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-mep-6.x-1904 Browse	<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>4.3.0-mapr-mep-6.x-1904</version> </dependency>

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>4.3.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>4.3.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>4.3.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.3.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>4.3.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>4.3.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>4.3.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.3.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	4.3.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.3.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	4.3.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.3.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-mep-6.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-utils</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-6.x-1904</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-catalyst_2.11	2.3.3.0-mapr-602 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-catalyst_2.11</artifactId> <version>2.3.3.0-mapr-602</version> </dependency>`
org.apache.spark	spark-core_2.11	2.3.3.0-mapr-602 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.11</artifactId> <version>2.3.3.0-mapr-602</version> </dependency>`
org.apache.spark	spark-graphx_2.11	2.3.3.0-mapr-602 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.11</artifactId> <version>2.3.3.0-mapr-602</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.11	2.3.3.0-mapr-602 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.3.3.0-mapr-602</version> </dependency>`
org.apache.spark	spark-hive_2.11	2.3.3.0-mapr-602 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.11</artifactId> <version>2.3.3.0-mapr-602</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-kvstore_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-kvstore_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-mesos_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-network-common_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-sketch_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-sql-kafka-0-10_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-sql_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-assembly_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.3.3.0-mapr-602 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.3.3.0-mapr-602</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.3.3.0-mapr-602 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.3.3.0-mapr-602</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.3.3.0-mapr-602 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.3.3.0-mapr-602</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.3.3.0-mapr-602 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.3.3.0-mapr-602</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.3.3.0-mapr-602 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.3.3.0-mapr-602</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-0-9_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-producer_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-streaming_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-tags_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-yarn_2.11	2.3.3.0-mapr-602
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.3.3.0-mapr-602</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.7-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.7-mapr-1904</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.7-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.7-mapr-1904</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

org.apache.tez	hadoop-shim-2.7	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim-2.7</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez.conftool	mapr-tez-conf-tool	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez.conftool</groupId>`
			`<artifactId>mapr-tez-conf-tool</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

org.apache.tez	tez-api	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-api</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

org.apache.tez	tez-aux-services	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-aux-services</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

org.apache.tez	tez-common	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-common</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

org.apache.tez	tez-dag	0.9.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-dag</artifactId>`
			`<version>0.9.1-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-examples	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-job-analyzer	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-job-analyzer</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-mapreduce	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-mapreduce</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-runtime-internals	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-internals</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-runtime-library	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-library</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-tests	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-tests</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-ui	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history-with-acls</artifactId> <version>0.9.1-mapr-1904</version> </dependency>`

Maven Artifacts for EEP 6.0.1

Listed are all Maven artifacts for EEP 6.0.1 components.

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-legacy-sources	flume-avro-source	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-legacy-sources</groupId> <artifactId>flume-avro-source</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume	flume-checkstyle	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-checkstyle</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-dataset-sink	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-dataset-sink</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-channels	flume-file-channel	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-file-channel</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-hdfs-sink	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-hdfs-sink</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-hive-sink	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-hive-sink</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-http-sink	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-http-sink</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-irc-sink	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-irc-sink</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-jdbc-channel	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-jdbc-channel</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-jms-source	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-jms-source</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-channels	flume-kafka-channel	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-kafka-channel</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-kafka-source	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume	flume-ng-auth	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-auth</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume	flume-ng-configuration	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-configuration</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume	flume-ng-core	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-core</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-sinks	flume-ng-elasticsearch-sink	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-elasticsearch-sink</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume	flume-ng-embedded-agent	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-ng-embedded-agent</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-hbase-sink	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-hbase-sink</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-kafka-sink</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-clients	flume-ng-log4jappender	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-clients</groupId> <artifactId>flume-ng-log4jappender</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-sinks	flume-ng-morphline-solr-sink	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-morphline-solr-sink</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume	flume-ng-node	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-ng-node</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume	flume-ng-sdk	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-ng-sdk</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume	flume-ng-tests	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-ng-tests</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`
org.apache.flume.flume-ng-sources	flume-scribe-source	1.8.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sources</groupId> <artifactId>flume-scribe-source</artifactId> <version>1.8.0-mapr-1901</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-shared	flume-shared-kafka-test	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-shared</groupId>`
			`<artifactId>flume-shared-kafka-test</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-spillable-memory-channel	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-spillable-memory-channel</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-taildir-source	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-taildir-source</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume.flume-ng-legacy-sources	flume-thrift-source	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-thrift-source</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

org.apache.flume	flume-tools	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-tools</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-twitter-source	1.8.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-twitter-source</artifactId>`
			`<version>1.8.0-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-annotations	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-annotations</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

org.apache.hbase	hbase-checkstyle	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-checkstyle</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

org.apache.hbase	hbase-client	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

org.apache.hbase	hbase-common	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-common</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-examples	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-examples</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

org.apache.hbase	hbase-hadoop-compat	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-hadoop-compat</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

org.apache.hbase	hbase-hadoop2-compat	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-hadoop2-compat</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

org.apache.hbase	hbase-it	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-it</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

org.apache.hbase	hbase-native-client	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-native-client</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-prefix-tree	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-prefix-tree</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-procedure	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-procedure</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-protocol	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-protocol</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-resource-bundle	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-resource-bundle</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-rest	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-rest</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-server	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-server</artifactId><version>1.1.8-mapr-1901</version></dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-shaded-client	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-client</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-shaded-server	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-server</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-shell	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shell</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-spark	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-spark</artifactId><version>1.1.8-mapr-1901</version></dependency>`
org.apache.hbase	hbase-testing-util	1.1.8-mapr-1901 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-testing-util</artifactId><version>1.1.8-mapr-1901</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-thrift	1.1.8-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-thrift</artifactId>`
			`<version>1.1.8-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-beeline	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-beeline</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-cli	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-cli</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-common	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-common</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-contrib	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-druid-handler	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-druid-handler</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-exec	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-jdbc-handler	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc-handler</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-llap-client	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-client</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-llap-common	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-llap-ext-client	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-ext-client</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-llap-server	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-server</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-tez	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-metastore	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-serde	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-service	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`
org.apache.hive	hive-service-rpc	2.3.3-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.3.3-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-shims	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-shims</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-0.23	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.23</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-common	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-vector-code-gen	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-vector-code-gen</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

org.apache.hive	spark-client	2.3.3-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.3.3-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.kafka	connect-api	1.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-api</artifactId>`
			`<version>1.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.kafka	connect-json	1.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-json</artifactId>`
			`<version>1.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.kafka	connect-runtime	1.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>connect-runtime</artifactId>`
			`<version>1.1.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.kafka	connect-transforms	1.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>connect-transforms</artifactId> <version>1.1.1-mapr-1901</version> </dependency>`
org.apache.kafka	kafka-clients	1.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-clients</artifactId> <version>1.1.1-mapr-1901</version> </dependency>`
org.apache.kafka	kafka-log4j-appender	1.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-log4j-appender</artifactId> <version>1.1.1-mapr-1901</version> </dependency>`
org.apache.kafka	kafka-streams	1.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-streams</artifactId> <version>1.1.1-mapr-1901</version> </dependency>`
org.apache.kafka	kafka-tools	1.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka-tools</artifactId> <version>1.1.1-mapr-1901</version> </dependency>`
org.apache.kafka	kafka_2.11	1.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.kafka</groupId> <artifactId>kafka_2.11</artifactId> <version>1.1.1-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.kafka	kafka_2.12	1.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.kafka</groupId>`
			`<artifactId>kafka_2.12</artifactId>`
			`<version>1.1.1-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.3.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.3.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-core	4.3.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.3.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	4.3.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.3.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	4.3.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>4.3.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>4.3.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>4.3.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>4.3.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>4.3.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.3.0-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>4.3.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>4.3.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>4.3.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.3.0-mapr-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	4.3.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.3.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-webapp	4.3.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-webapp</artifactId> <version>4.3.0-mapr-1901</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-auth</artifactId> <version>hadoop-2-4.3.0-mapr-1901</version> </dependency>`
org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-distcp</artifactId> <version>hadoop-2-4.3.0-mapr-1901</version> </dependency>`
org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-utils</artifactId> <version>hadoop-2-4.3.0-mapr-1901</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.pig	pig	0.16.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>pig</artifactId> <version>0.16.0-mapr-1901</version> </dependency>`
org.apache.pig	piggybank	0.16.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>piggybank</artifactId> <version>0.16.0-mapr-1901</version> </dependency>`
org.apache.pig	pigsmoke	0.16.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>pigsmoke</artifactId> <version>0.16.0-mapr-1901</version> </dependency>`
org.apache.pig	pigunit	0.16.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>pigunit</artifactId> <version>0.16.0-mapr-1901</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sentry	sentry-binding-hive-common	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-binding-hive-common</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-hive-v2	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-v2</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-kafka	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-kafka</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-solr	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-solr</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-sqoop	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-common	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-db	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-kafka	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-kafka</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-sqoop	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sentry	sentry-dist	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-dist</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`
org.apache.sentry	sentry-policy-common	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-policy-common</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`
org.apache.sentry	sentry-policy-db	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-policy-db</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`
org.apache.sentry	sentry-policy-indexer	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-policy-indexer</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`
org.apache.sentry	sentry-policy-kafka	1.7.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-policy-kafka</artifactId> <version>1.7.0-mapr-1901</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-search	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-db	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-file	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-core	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-core</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-handlers	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-catalyst_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-graphx_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-hive_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-kvstore_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-kvstore_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-network-shuffle_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-network-yarn_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-repl_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-sketch_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sql_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.3.2.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.3.2.0-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-unsafe_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.3.2.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.3.2.0-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	hadoop-shim-2.7	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim-2.7</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-api	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-api</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-aux-services	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-aux-services</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-common	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-common</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-dag	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-dag</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-examples	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-examples</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-ext-service-tests	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ext-service-tests</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-job-analyzer	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-mapreduce	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-tests	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-tests</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ui	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.9.1-mapr-1901</version>`
			`</dependency>`

Maven Artifacts for EEP 6.0.0

Listed are all Maven artifacts for EEP 6.0.0 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.hbase	asynchbase	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.hbase</groupId>`
			`<artifactId>asynchbase</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-legacy-sources	flume-avro-source	1.8.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-legacy-sources</groupId> <artifactId>flume-avro-source</artifactId> <version>1.8.0-mapr-1808</version> </dependency>`
org.apache.flume	flume-checkstyle	1.8.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-checkstyle</artifactId> <version>1.8.0-mapr-1808</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-dataset-sink	1.8.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-dataset-sink</artifactId> <version>1.8.0-mapr-1808</version> </dependency>`
org.apache.flume.flume-ng-channels	flume-file-channel	1.8.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-file-channel</artifactId> <version>1.8.0-mapr-1808</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-hdfs-sink	1.8.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-hdfs-sink</artifactId> <version>1.8.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-hive-sink	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-hive-sink</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-http-sink	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-http-sink</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-irc-sink	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-irc-sink</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-jdbc-channel	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-jdbc-channel</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-jms-source	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-jms-source</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-channels	flume-kafka-channel	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-kafka-channel</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-kafka-source	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume	flume-ng-auth	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-auth</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume	flume-ng-configuration	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-configuration</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume	flume-ng-core	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-core</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-sinks	flume-ng-elasticsearch-sink	1.8.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-elasticsearch-sink</artifactId> <version>1.8.0-mapr-1808</version> </dependency>`
org.apache.flume	flume-ng-embedded-agent	1.8.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-ng-embedded-agent</artifactId> <version>1.8.0-mapr-1808</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-hbase-sink	1.8.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-hbase-sink</artifactId> <version>1.8.0-mapr-1808</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink	1.8.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-kafka-sink</artifactId> <version>1.8.0-mapr-1808</version> </dependency>`
org.apache.flume.flume-ng-clients	flume-ng-log4jappender	1.8.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-clients</groupId> <artifactId>flume-ng-log4jappender</artifactId> <version>1.8.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-ng-morphline-solr-sink	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-morphline-solr-sink</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume	flume-ng-node	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-node</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume	flume-ng-sdk	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-sdk</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume	flume-ng-tests	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-tests</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-scribe-source	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-scribe-source</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-shared	flume-shared-kafka-test	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-shared</groupId>`
			`<artifactId>flume-shared-kafka-test</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-spillable-memory-channel	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-spillable-memory-channel</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-taildir-source	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-taildir-source</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume.flume-ng-legacy-sources	flume-thrift-source	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-thrift-source</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

org.apache.flume	flume-tools	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-tools</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-twitter-source	1.8.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-twitter-source</artifactId>`
			`<version>1.8.0-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-annotations	1.1.8-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-annotations</artifactId>`
			`<version>1.1.8-mapr-1808</version>`
			`</dependency>`

org.apache.hbase	hbase-checkstyle	1.1.8-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-checkstyle</artifactId>`
			`<version>1.1.8-mapr-1808</version>`
			`</dependency>`

org.apache.hbase	hbase-client	1.1.8-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client</artifactId>`
			`<version>1.1.8-mapr-1808</version>`
			`</dependency>`

org.apache.hbase	hbase-common	1.1.8-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-common</artifactId>`
			`<version>1.1.8-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-examples	1.1.8-mapr-1808 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-examples</artifactId> <version>1.1.8-mapr-1808</version> </dependency>`
org.apache.hbase	hbase-hadoop-compat	1.1.8-mapr-1808 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop-compat</artifactId> <version>1.1.8-mapr-1808</version> </dependency>`
org.apache.hbase	hbase-hadoop2-compat	1.1.8-mapr-1808 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop2-compat</artifactId> <version>1.1.8-mapr-1808</version> </dependency>`
org.apache.hbase	hbase-it	1.1.8-mapr-1808 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-it</artifactId> <version>1.1.8-mapr-1808</version> </dependency>`
org.apache.hbase	hbase-native-client	1.1.8-mapr-1808 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-native-client</artifactId> <version>1.1.8-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-prefix-tree	1.1.8-mapr-1808 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-prefix-tree</artifactId><version>1.1.8-mapr-1808</version></dependency>`
org.apache.hbase	hbase-procedure	1.1.8-mapr-1808 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-procedure</artifactId><version>1.1.8-mapr-1808</version></dependency>`
org.apache.hbase	hbase-protocol	1.1.8-mapr-1808 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-protocol</artifactId><version>1.1.8-mapr-1808</version></dependency>`
org.apache.hbase	hbase-resource-bundle	1.1.8-mapr-1808 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-resource-bundle</artifactId><version>1.1.8-mapr-1808</version></dependency>`
org.apache.hbase	hbase-rest	1.1.8-mapr-1808 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-rest</artifactId><version>1.1.8-mapr-1808</version></dependency>`
org.apache.hbase	hbase-server	1.1.8-mapr-1808 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-server</artifactId><version>1.1.8-mapr-1808</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-shaded-client	1.1.8-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-client</artifactId>`
			`<version>1.1.8-mapr-1808</version>`
			`</dependency>`

org.apache.hbase	hbase-shaded-server	1.1.8-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shaded-server</artifactId>`
			`<version>1.1.8-mapr-1808</version>`
			`</dependency>`

org.apache.hbase	hbase-shell	1.1.8-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-shell</artifactId>`
			`<version>1.1.8-mapr-1808</version>`
			`</dependency>`

org.apache.hbase	hbase-spark	1.1.8-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-spark</artifactId>`
			`<version>1.1.8-mapr-1808</version>`
			`</dependency>`

org.apache.hbase	hbase-testing-util	1.1.8-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-testing-util</artifactId>`
			`<version>1.1.8-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-thrift	1.1.8-mapr-1808 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-thrift</artifactId> <version>1.1.8-mapr-1808</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-accumulo-handler</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-beeline	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-cli	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-common	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-contrib	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-druid-handler	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-druid-handler</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-exec	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-jdbc-handler	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc-handler</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-llap-client	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-client</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-llap-common	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-llap-ext-client	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-ext-client</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-llap-server	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-server</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-tez	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-metastore	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-serde	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-service	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	hive-service-rpc	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-shims	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-shims</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-0.23	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.23</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-common	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-vector-code-gen	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-vector-code-gen</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

org.apache.hive.encryptiontool	mapr-encryption-tool	2.3.3-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.3.3-mapr-1808</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.maprminicluster	mapr-mini-cluster	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.maprminicluster</groupId> <artifactId>mapr-mini-cluster</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`
org.apache.hive	spark-client	2.3.3-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>spark-client</artifactId> <version>2.3.3-mapr-1808</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-client	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-client</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-core	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-examples	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie.test	oozie-mini	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-tools	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-tools</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-webapp	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-webapp</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-auth</artifactId> <version>hadoop-2-4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-distcp</artifactId> <version>hadoop-2-4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-utils</artifactId> <version>hadoop-2-4.3.0-mapr-1808</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-hive-common	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-common</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-hive-v2	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-v2</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-kafka	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-kafka</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-solr	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-solr</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-sqoop	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-common	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-db	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-kafka	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-kafka</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-sqoop	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-dist	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-dist</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-hdfs-common	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-hdfs-common</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-hdfs-dist	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-hdfs-dist</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-hdfs-namenode-plugin	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-hdfs-namenode-plugin</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-hdfs-service	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-hdfs-service</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-common	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-common</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-db	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-db</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-indexer	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-indexer</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-kafka	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-kafka</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-search	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-db	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-file	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-core	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-core</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-handlers	1.7.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.7.0-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.3.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.3.1-mapr-1808</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.3.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.3.1-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-catalyst_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-catalyst_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-core_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-graphx_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-hive_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-kvstore_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-kvstore_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-launcher_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-launcher_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-mllib-local_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib-local_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-mllib_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-network-common_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-common_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-network-shuffle_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-network-yarn_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-repl_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-sketch_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sql_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.3.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.3.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-unsafe_2.11	2.3.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.3.1-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.3.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.3.1-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.7-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.7-mapr-1808</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.7-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.7-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.9.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.9.1-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	hadoop-shim-2.7	0.9.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim-2.7</artifactId> <version>0.9.1-mapr-1808</version> </dependency>`
org.apache.tez	tez-api	0.9.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.9.1-mapr-1808</version> </dependency>`
org.apache.tez	tez-aux-services	0.9.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-aux-services</artifactId> <version>0.9.1-mapr-1808</version> </dependency>`
org.apache.tez	tez-common	0.9.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.9.1-mapr-1808</version> </dependency>`
org.apache.tez	tez-dag	0.9.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.9.1-mapr-1808</version> </dependency>`
org.apache.tez	tez-examples	0.9.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.9.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ext-service-tests	0.9.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ext-service-tests</artifactId>`
			`<version>0.9.1-mapr-1808</version>`
			`</dependency>`

org.apache.tez	tez-job-analyzer	0.9.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.9.1-mapr-1808</version>`
			`</dependency>`

org.apache.tez	tez-mapreduce	0.9.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.9.1-mapr-1808</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.9.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.9.1-mapr-1808</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.9.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.9.1-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-tests	0.9.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-tests</artifactId>`
			`<version>0.9.1-mapr-1808</version>`
			`</dependency>`

org.apache.tez	tez-ui	0.9.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui</artifactId>`
			`<version>0.9.1-mapr-1808</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history	0.9.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.9.1-mapr-1808</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.9.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.9.1-mapr-1808</version>`
			`</dependency>`

Maven Artifacts for EEP 5.0.7
There are no changes to the Maven artifacts for EEP 5.0.7 components.

See the Maven Artifacts for EEP 5.0.6 on page 4721.

Maven Artifacts for EEP 5.0.6
Listed are all Maven artifacts for EEP 5.0.6 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-annotations	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-annotations</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`
org.apache.hbase	hbase-checkstyle	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-checkstyle</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`
org.apache.hbase	hbase-client	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-client</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`
org.apache.hbase	hbase-common	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-common</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`
org.apache.hbase	hbase-examples	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-examples</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-hadoop-compat	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop-compat</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`
org.apache.hbase	hbase-hadoop2-compat	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop2-compat</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`
org.apache.hbase	hbase-it	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-it</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`
org.apache.hbase	hbase-native-client	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-native-client</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`
org.apache.hbase	hbase-prefix-tree	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-prefix-tree</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-procedure	1.1.8-mapr-2101 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-procedure</artifactId><version>1.1.8-mapr-2101</version></dependency>`
org.apache.hbase	hbase-protocol	1.1.8-mapr-2101 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-protocol</artifactId><version>1.1.8-mapr-2101</version></dependency>`
org.apache.hbase	hbase-resource-bundle	1.1.8-mapr-2101 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-resource-bundle</artifactId><version>1.1.8-mapr-2101</version></dependency>`
org.apache.hbase	hbase-rest	1.1.8-mapr-2101 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-rest</artifactId><version>1.1.8-mapr-2101</version></dependency>`
org.apache.hbase	hbase-server	1.1.8-mapr-2101 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-server</artifactId><version>1.1.8-mapr-2101</version></dependency>`
org.apache.hbase	hbase-shaded-client	1.1.8-mapr-2101 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-client</artifactId><version>1.1.8-mapr-2101</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-shaded-server	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-server</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`
org.apache.hbase	hbase-shell	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shell</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`
org.apache.hbase	hbase-spark	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-spark</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`
org.apache.hbase	hbase-testing-util	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-testing-util</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`
org.apache.hbase	hbase-thrift	1.1.8-mapr-2101 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-thrift</artifactId> <version>1.1.8-mapr-2101</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-client	4.3.0-mapr-506 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-client</artifactId> <version>4.3.0-mapr-506</version> </dependency>`
org.apache.oozie	oozie-core	4.3.0-mapr-506 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>4.3.0-mapr-506</version> </dependency>`
org.apache.oozie	oozie-examples	4.3.0-mapr-506 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>4.3.0-mapr-506</version> </dependency>`
org.apache.oozie.test	oozie-mini	4.3.0-mapr-506 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>4.3.0-mapr-506</version> </dependency>`
org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-506 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>4.3.0-mapr-506</version> </dependency>`
org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-506 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>4.3.0-mapr-506</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-506 Browse	`<dependency><groupId>org.apache.oozie</groupId><artifactId>oozie-sharelib-hive</artifactId><version>4.3.0-mapr-506</version></dependency>`
org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-506 Browse	`<dependency><groupId>org.apache.oozie</groupId><artifactId>oozie-sharelib-hive2</artifactId><version>4.3.0-mapr-506</version></dependency>`
org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-506 Browse	`<dependency><groupId>org.apache.oozie</groupId><artifactId>oozie-sharelib-oozie</artifactId><version>4.3.0-mapr-506</version></dependency>`
org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-506 Browse	`<dependency><groupId>org.apache.oozie</groupId><artifactId>oozie-sharelib-pig</artifactId><version>4.3.0-mapr-506</version></dependency>`
org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-506 Browse	`<dependency><groupId>org.apache.oozie</groupId><artifactId>oozie-sharelib-spark</artifactId><version>4.3.0-mapr-506</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-506 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-sqoop</artifactId> <version>4.3.0-mapr-506</version> </dependency>`
org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-506 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-streaming</artifactId> <version>4.3.0-mapr-506</version> </dependency>`
org.apache.oozie	oozie-tools	4.3.0-mapr-506 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-tools</artifactId> <version>4.3.0-mapr-506</version> </dependency>`
org.apache.oozie	oozie-webapp	4.3.0-mapr-506 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-webapp</artifactId> <version>4.3.0-mapr-506</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	classpath-filter_2.11	2.2.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>classpath-filter_2.11</artifactId> <version>2.2.1-mapr-2101</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

com.mapr.db	maprdb-spark	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-graphx_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-hive_2.11	2.2.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.11</artifactId> <version>2.2.1-mapr-2101</version> </dependency>`
org.apache.spark	spark-launcher_2.11	2.2.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-launcher_2.11</artifactId> <version>2.2.1-mapr-2101</version> </dependency>`
org.apache.spark	spark-mesos_2.11	2.2.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mesos_2.11</artifactId> <version>2.2.1-mapr-2101</version> </dependency>`
org.apache.spark	spark-mllib-local_2.11	2.2.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib-local_2.11</artifactId> <version>2.2.1-mapr-2101</version> </dependency>`
org.apache.spark	spark-mllib_2.11	2.2.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib_2.11</artifactId> <version>2.2.1-mapr-2101</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-network-common_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-sketch_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-sql-kafka-0-10_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-sql_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-assembly_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.2.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.2.1-mapr-2101</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.2.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.2.1-mapr-2101</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.2.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.2.1-mapr-2101</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.2.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.2.1-mapr-2101</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.2.1-mapr-2101 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.2.1-mapr-2101</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-0-9_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-producer_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-streaming_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-tags_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-yarn_2.11	2.2.1-mapr-2101
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.2.1-mapr-2101</version>`
			`</dependency>`

Maven Artifacts for EEP 5.0.5

Listed are all Maven artifacts for EEP 5.0.5 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-annotations	1.1.8-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-annotations</artifactId>`
			`<version>1.1.8-mapr-2009</version>`
			`</dependency>`

org.apache.hbase	hbase-checkstyle	1.1.8-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-checkstyle</artifactId>`
			`<version>1.1.8-mapr-2009</version>`
			`</dependency>`

org.apache.hbase	hbase-client	1.1.8-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client</artifactId>`
			`<version>1.1.8-mapr-2009</version>`
			`</dependency>`

org.apache.hbase	hbase-common	1.1.8-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-common</artifactId>`
			`<version>1.1.8-mapr-2009</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-examples	1.1.8-mapr-2009 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-examples</artifactId> <version>1.1.8-mapr-2009</version> </dependency>`
org.apache.hbase	hbase-hadoop-compat	1.1.8-mapr-2009 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop-compat</artifactId> <version>1.1.8-mapr-2009</version> </dependency>`
org.apache.hbase	hbase-hadoop2-compat	1.1.8-mapr-2009 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop2-compat</artifactId> <version>1.1.8-mapr-2009</version> </dependency>`
org.apache.hbase	hbase-it	1.1.8-mapr-2009 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-it</artifactId> <version>1.1.8-mapr-2009</version> </dependency>`
org.apache.hbase	hbase-native-client	1.1.8-mapr-2009 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-native-client</artifactId> <version>1.1.8-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-prefix-tree	1.1.8-mapr-2009 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-prefix-tree</artifactId><version>1.1.8-mapr-2009</version></dependency>`
org.apache.hbase	hbase-procedure	1.1.8-mapr-2009 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-procedure</artifactId><version>1.1.8-mapr-2009</version></dependency>`
org.apache.hbase	hbase-protocol	1.1.8-mapr-2009 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-protocol</artifactId><version>1.1.8-mapr-2009</version></dependency>`
org.apache.hbase	hbase-resource-bundle	1.1.8-mapr-2009 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-resource-bundle</artifactId><version>1.1.8-mapr-2009</version></dependency>`
org.apache.hbase	hbase-rest	1.1.8-mapr-2009 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-rest</artifactId><version>1.1.8-mapr-2009</version></dependency>`
org.apache.hbase	hbase-server	1.1.8-mapr-2009 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-server</artifactId><version>1.1.8-mapr-2009</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-shaded-client	1.1.8-mapr-2009 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-client</artifactId><version>1.1.8-mapr-2009</version></dependency>`
org.apache.hbase	hbase-shaded-server	1.1.8-mapr-2009 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-server</artifactId><version>1.1.8-mapr-2009</version></dependency>`
org.apache.hbase	hbase-shell	1.1.8-mapr-2009 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shell</artifactId><version>1.1.8-mapr-2009</version></dependency>`
org.apache.hbase	hbase-spark	1.1.8-mapr-2009 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-spark</artifactId><version>1.1.8-mapr-2009</version></dependency>`
org.apache.hbase	hbase-testing-util	1.1.8-mapr-2009 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-testing-util</artifactId><version>1.1.8-mapr-2009</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-thrift	1.1.8-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-thrift</artifactId>`
			`<version>1.1.8-mapr-2009</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	2.1.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.1.1-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-ant	2.1.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-ant</artifactId>`
			`<version>2.1.1-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-beeline	2.1.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-beeline</artifactId>`
			`<version>2.1.1-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-cli	2.1.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-cli</artifactId>`
			`<version>2.1.1-mapr-2009</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-common	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-contrib	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-exec	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-pig-adapter</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-server-extensions</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-streaming</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-hplsql	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hplsql</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-hwi	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hwi</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-jdbc	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-llap-client	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-client</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-llap-common	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-llap-ext-client	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-ext-client</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-llap-server	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-server</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-tez	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler-test	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler-test</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-metastore	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-service	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-service-rpc	2.1.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service-rpc</artifactId>`
			`<version>2.1.1-mapr-2009</version>`
			`</dependency>`

org.apache.hive	hive-shims	2.1.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-shims</artifactId>`
			`<version>2.1.1-mapr-2009</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.23</artifactId>`
			`<version>2.1.1-mapr-2009</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-common	2.1.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.1.1-mapr-2009</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.1.1-mapr-2009</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-storage-api	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-storage-api</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	hive-testutils	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-testutils</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat-java-client</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive.conftool	mapr-conf-tool	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.conftool</groupId> <artifactId>mapr-conf-tool</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.encryptiontool	mapr-encryption-tool	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.encryptiontool</groupId> <artifactId>mapr-encryption-tool</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive.maprminicluster</groupId> <artifactId>mapr-mini-cluster</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`
org.apache.hive	spark-client	2.1.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>spark-client</artifactId> <version>2.1.1-mapr-2009</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-client	4.3.0-mapr-mep-5.x-2009 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-client</artifactId> <version>4.3.0-mapr-mep-5.x-2009</version> </dependency>`
org.apache.oozie	oozie-core	4.3.0-mapr-mep-5.x-2009 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>4.3.0-mapr-mep-5.x-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-examples	4.3.0-mapr-mep-5.x-2009 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>4.3.0-mapr-mep-5.x-2009</version> </dependency>`
org.apache.oozie.test	oozie-mini	4.3.0-mapr-mep-5.x-2009 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>4.3.0-mapr-mep-5.x-2009</version> </dependency>`
org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-mep-5.x-2009 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>4.3.0-mapr-mep-5.x-2009</version> </dependency>`
org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-mep-5.x-2009 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>4.3.0-mapr-mep-5.x-2009</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-mep-5.x-2009 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>4.3.0-mapr-mep-5.x-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-mep-5.x-2009 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>4.3.0-mapr-mep-5.x-2009</version> </dependency>`
org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-mep-5.x-2009 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>4.3.0-mapr-mep-5.x-2009</version> </dependency>`
org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-mep-5.x-2009 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>4.3.0-mapr-mep-5.x-2009</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-mep-5.x-2009 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>4.3.0-mapr-mep-5.x-2009</version> </dependency>`
org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-mep-5.x-2009 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-sqoop</artifactId> <version>4.3.0-mapr-mep-5.x-2009</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-mep-5.x-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-2009</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	4.3.0-mapr-mep-5.x-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-2009</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	4.3.0-mapr-mep-5.x-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-2009</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.pig	pig	0.16.0-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pig</artifactId>`
			`<version>0.16.0-mapr-2009</version>`
			`</dependency>`

org.apache.pig	piggybank	0.16.0-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>piggybank</artifactId>`
			`<version>0.16.0-mapr-2009</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.pig	pigsmoke	0.16.0-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigsmoke</artifactId>`
			`<version>0.16.0-mapr-2009</version>`
			`</dependency>`

org.apache.pig	pigunit	0.16.0-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigunit</artifactId>`
			`<version>0.16.0-mapr-2009</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.2.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.2.1-mapr-2009</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.2.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.2.1-mapr-2009</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.11	2.2.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.2.1-mapr-2009</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-core_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-graphx_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-hive_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-launcher_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-launcher_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-mesos_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mesos_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-mllib-local_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib-local_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-mllib_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-network-common_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-common_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-network-shuffle_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-network-yarn_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-repl_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-sketch_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-sql_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-flume-assembly_2.11	2.2.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-2009</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.2.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.2.1-mapr-2009</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume_2.11	2.2.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.2.1-mapr-2009</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.2.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-2009</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.2.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.2.1-mapr-2009</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.2.1-mapr-2009 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.2.1-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming_2.11	2.2.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.2.1-mapr-2009</version>`
			`</dependency>`

org.apache.spark	spark-tags_2.11	2.2.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.11</artifactId>`
			`<version>2.2.1-mapr-2009</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.11	2.2.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.2.1-mapr-2009</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.2.1-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.2.1-mapr-2009</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.8.4-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.8.4-mapr-2009</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	hadoop-shim-2.6	0.8.4-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim-2.6</artifactId> <version>0.8.4-mapr-2009</version> </dependency>`
org.apache.tez	tez-api	0.8.4-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.8.4-mapr-2009</version> </dependency>`
org.apache.tez	tez-common	0.8.4-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.8.4-mapr-2009</version> </dependency>`
org.apache.tez	tez-dag	0.8.4-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.8.4-mapr-2009</version> </dependency>`
org.apache.tez	tez-examples	0.8.4-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.8.4-mapr-2009</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.8.4-mapr-2009 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.8.4-mapr-2009</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-job-analyzer	0.8.4-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.8.4-mapr-2009</version>`
			`</dependency>`

org.apache.tez	tez-mapreduce	0.8.4-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.8.4-mapr-2009</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.8.4-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.8.4-mapr-2009</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.8.4-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.8.4-mapr-2009</version>`
			`</dependency>`

org.apache.tez	tez-tests	0.8.4-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-tests</artifactId>`
			`<version>0.8.4-mapr-2009</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ui	0.8.4-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui</artifactId>`
			`<version>0.8.4-mapr-2009</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history	0.8.4-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.8.4-mapr-2009</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.8.4-mapr-2009
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.8.4-mapr-2009</version>`
			`</dependency>`

Maven Artifacts for EEP 5.0.4

Listed are all Maven artifacts for EEP 5.0.4 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-ant	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-ant</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-beeline	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-cli	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-common	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-contrib	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-exec	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-hwi	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hwi</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-jdbc	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-llap-client	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-client</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-llap-common	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-ext-client</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-server	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-server</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-llap-tez	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler-test	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler-test</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-metastore	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-serde	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-service	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-service-rpc	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive	hive-shims	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`
org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>2.1.1-mapr-1912</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-storage-api	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-storage-api</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.conftool	mapr-conf-tool	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.hive.encryptiontool	mapr-encryption-tool	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

org.apache.hive	spark-client	2.1.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.1.1-mapr-1912</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.3.0-mapr-mep-5.x-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1912</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-core	4.3.0-mapr-mep-5.x-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1912</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	4.3.0-mapr-mep-5.x-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1912</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	4.3.0-mapr-mep-5.x-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1912</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-mep-5.x-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1912</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-mep-5.x-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1912</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-mep-5.x-1912 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>4.3.0-mapr-mep-5.x-1912</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-mep-5.x-1912 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>4.3.0-mapr-mep-5.x-1912</version> </dependency>`
org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-mep-5.x-1912 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>4.3.0-mapr-mep-5.x-1912</version> </dependency>`
org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-mep-5.x-1912 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>4.3.0-mapr-mep-5.x-1912</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-mep-5.x-1912 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>4.3.0-mapr-mep-5.x-1912</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-mep-5.x-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1912</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-mep-5.x-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1912</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	4.3.0-mapr-mep-5.x-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1912</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-mep-5.x-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-5.x-1912</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-mep-5.x-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-5.x-1912</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-mep-5.x-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-utils</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-5.x-1912</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-graphx_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-hive_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-mesos_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-mllib-local_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-repl_2.11	2.2.1-mapr-1912 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-repl_2.11</artifactId><version>2.2.1-mapr-1912</version></dependency>`
org.apache.spark	spark-sketch_2.11	2.2.1-mapr-1912 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-sketch_2.11</artifactId><version>2.2.1-mapr-1912</version></dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.11	2.2.1-mapr-1912 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-sql-kafka-0-10_2.11</artifactId><version>2.2.1-mapr-1912</version></dependency>`
org.apache.spark	spark-sql_2.11	2.2.1-mapr-1912 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-sql_2.11</artifactId><version>2.2.1-mapr-1912</version></dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.11	2.2.1-mapr-1912 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-streaming-flume-assembly_2.11</artifactId><version>2.2.1-mapr-1912</version></dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-flume-sink_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-producer_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

org.apache.spark	spark-streaming_2.11	2.2.1-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.2.1-mapr-1912</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-tags_2.11	2.2.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.2.1-mapr-1912</version> </dependency>`
org.apache.spark	spark-unsafe_2.11	2.2.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-unsafe_2.11</artifactId> <version>2.2.1-mapr-1912</version> </dependency>`
org.apache.spark	spark-yarn_2.11	2.2.1-mapr-1912 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-yarn_2.11</artifactId> <version>2.2.1-mapr-1912</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	hadoop-shim	0.8.4-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim</artifactId> <version>0.8.4-mapr-1912</version> </dependency>`
org.apache.tez	hadoop-shim-2.6	0.8.4-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim-2.6</artifactId> <version>0.8.4-mapr-1912</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-api	0.8.4-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.8.4-mapr-1912</version> </dependency>`
org.apache.tez	tez-common	0.8.4-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.8.4-mapr-1912</version> </dependency>`
org.apache.tez	tez-dag	0.8.4-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.8.4-mapr-1912</version> </dependency>`
org.apache.tez	tez-examples	0.8.4-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.8.4-mapr-1912</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.8.4-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.8.4-mapr-1912</version> </dependency>`
org.apache.tez	tez-job-analyzer	0.8.4-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-job-analyzer</artifactId> <version>0.8.4-mapr-1912</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-mapreduce	0.8.4-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-mapreduce</artifactId> <version>0.8.4-mapr-1912</version> </dependency>`
org.apache.tez	tez-runtime-internals	0.8.4-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-internals</artifactId> <version>0.8.4-mapr-1912</version> </dependency>`
org.apache.tez	tez-runtime-library	0.8.4-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-library</artifactId> <version>0.8.4-mapr-1912</version> </dependency>`
org.apache.tez	tez-tests	0.8.4-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-tests</artifactId> <version>0.8.4-mapr-1912</version> </dependency>`
org.apache.tez	tez-ui	0.8.4-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui</artifactId> <version>0.8.4-mapr-1912</version> </dependency>`
org.apache.tez	tez-ui2	0.8.4-mapr-1912 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui2</artifactId> <version>0.8.4-mapr-1912</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-yarn-timeline-history	0.8.4-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.8.4-mapr-1912</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.8.4-mapr-1912
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.8.4-mapr-1912</version>`
			`</dependency>`

Maven Artifacts for EEP 5.0.3

Listed are all Maven artifacts for EEP 5.0.3 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-legacy-sources	flume-avro-source	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-avro-source</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-checkstyle	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-checkstyle</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-sinks	flume-dataset-sink	1.8.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-dataset-sink</artifactId> <version>1.8.0-mapr-mep-5.x-1904</version> </dependency>`
org.apache.flume.flume-ng-channels	flume-file-channel	1.8.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-file-channel</artifactId> <version>1.8.0-mapr-mep-5.x-1904</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-hdfs-sink	1.8.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-hdfs-sink</artifactId> <version>1.8.0-mapr-mep-5.x-1904</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-hive-sink	1.8.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-hive-sink</artifactId> <version>1.8.0-mapr-mep-5.x-1904</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-http-sink	1.8.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-http-sink</artifactId> <version>1.8.0-mapr-mep-5.x-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-irc-sink	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-irc-sink</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-jdbc-channel	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-jdbc-channel</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-jms-source	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-jms-source</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-kafka-channel	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-kafka-channel</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-kafka-source	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume	flume-ng-auth	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-auth</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-configuration	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-configuration</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-core	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-core</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-elasticsearch-sink	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-elasticsearch-sink</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-embedded-agent	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-embedded-agent</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-ng-hbase-sink	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-hbase-sink</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-kafka-sink</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-clients	flume-ng-log4jappender	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-clients</groupId>`
			`<artifactId>flume-ng-log4jappender</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-morphline-solr-sink	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-morphline-solr-sink</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-node	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-node</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume	flume-ng-sdk	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-sdk</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-ng-tests	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-tests</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-scribe-source	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-scribe-source</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-shared	flume-shared-kafka-test	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-shared</groupId>`
			`<artifactId>flume-shared-kafka-test</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-spillable-memory-channel	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-spillable-memory-channel</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-taildir-source	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-taildir-source</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-legacy-sources	flume-thrift-source	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-thrift-source</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume	flume-tools	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-tools</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-twitter-source	1.8.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-twitter-source</artifactId>`
			`<version>1.8.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-annotations	1.1.8-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-annotations</artifactId>`
			`<version>1.1.8-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-checkstyle	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-checkstyle</artifactId><version>1.1.8-mapr-1904</version></dependency>`
org.apache.hbase	hbase-client	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-client</artifactId><version>1.1.8-mapr-1904</version></dependency>`
org.apache.hbase	hbase-common	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-common</artifactId><version>1.1.8-mapr-1904</version></dependency>`
org.apache.hbase	hbase-examples	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-examples</artifactId><version>1.1.8-mapr-1904</version></dependency>`
org.apache.hbase	hbase-hadoop-compat	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-hadoop-compat</artifactId><version>1.1.8-mapr-1904</version></dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-hadoop2-compat	1.1.8-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-hadoop2-compat</artifactId>`
			`<version>1.1.8-mapr-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-it	1.1.8-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-it</artifactId>`
			`<version>1.1.8-mapr-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-native-client	1.1.8-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-native-client</artifactId>`
			`<version>1.1.8-mapr-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-prefix-tree	1.1.8-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-prefix-tree</artifactId>`
			`<version>1.1.8-mapr-1904</version>`
			`</dependency>`

org.apache.hbase	hbase-procedure	1.1.8-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-procedure</artifactId>`
			`<version>1.1.8-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-protocol	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-protocol</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-resource-bundle	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-resource-bundle</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-rest	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-rest</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-server	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-server</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-shaded-client	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-client</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-shaded-server	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-server</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-shell	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shell</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-spark	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-spark</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-testing-util	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-testing-util</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-thrift	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-thrift</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-accumulo-handler</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-ant	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-ant</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-beeline	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-cli	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-common	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-contrib	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-exec	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-pig-adapter</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-server-extensions</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-hwi	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hwi</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-llap-common	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-tez	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-maprdb-json-handler-test	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler-test</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-metastore	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-service	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-service-rpc	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-shims	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-scheduler</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-storage-api	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-storage-api</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-testutils	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-testutils</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive.encryptiontool	mapr-encryption-tool	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	spark-client	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.3.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-core	4.3.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	4.3.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	4.3.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>4.3.0-mapr-mep-5.x-1904</version> </dependency>`
org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>4.3.0-mapr-mep-5.x-1904</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>4.3.0-mapr-mep-5.x-1904</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>4.3.0-mapr-mep-5.x-1904</version> </dependency>`
org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>4.3.0-mapr-mep-5.x-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>4.3.0-mapr-mep-5.x-1904</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>4.3.0-mapr-mep-5.x-1904</version> </dependency>`
org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-sqoop</artifactId> <version>4.3.0-mapr-mep-5.x-1904</version> </dependency>`
org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-streaming</artifactId> <version>4.3.0-mapr-mep-5.x-1904</version> </dependency>`
org.apache.oozie	oozie-tools	4.3.0-mapr-mep-5.x-1904 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-tools</artifactId> <version>4.3.0-mapr-mep-5.x-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-webapp	4.3.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-mep-5.x-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-utils</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-5.x-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	classpath-filter_2.11	2.2.1-mapr-1904 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>classpath-filter_2.11</artifactId><version>2.2.1-mapr-1904</version></dependency>`
com.mapr.db	maprdb-spark	2.2.1-mapr-1904 Browse	`<dependency><groupId>com.mapr.db</groupId><artifactId>maprdb-spark</artifactId><version>2.2.1-mapr-1904</version></dependency>`
org.apache.spark	spark-catalyst_2.11	2.2.1-mapr-1904 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-catalyst_2.11</artifactId><version>2.2.1-mapr-1904</version></dependency>`
org.apache.spark	spark-core_2.11	2.2.1-mapr-1904 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-core_2.11</artifactId><version>2.2.1-mapr-1904</version></dependency>`
org.apache.spark	spark-graphx_2.11	2.2.1-mapr-1904 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-graphx_2.11</artifactId><version>2.2.1-mapr-1904</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-hive-thriftserver_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-hive_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-mesos_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mesos_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-mllib_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sketch_2.11	2.2.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.2.1-mapr-1904</version> </dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.11	2.2.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.11</artifactId> <version>2.2.1-mapr-1904</version> </dependency>`
org.apache.spark	spark-sql_2.11	2.2.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.2.1-mapr-1904</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.11	2.2.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.11</artifactId> <version>2.2.1-mapr-1904</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.11	2.2.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.11</artifactId> <version>2.2.1-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-flume_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-producer_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-streaming_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-tags_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-unsafe_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.2.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.2.1-mapr-1904</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.6-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.6-mapr-1904</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.6-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.6-mapr-1904</version>`
			`</dependency>`

Maven Artifacts for EEP 5.0.2

Listed are all Maven artifacts for EEP 5.0.2 components.

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-accumulo-handler</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-ant	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-ant</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-beeline	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-cli	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-common	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-contrib	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-exec	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-exec</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-hbase-handler	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-hwi	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hwi</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-llap-common	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-tez	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-maprdb-json-handler-test	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler-test</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-metastore	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-service	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-service-rpc	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-shims	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.23</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-storage-api	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-storage-api</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat-java-client</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive.conftool	mapr-conf-tool	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.conftool</groupId> <artifactId>mapr-conf-tool</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive.encryptiontool	mapr-encryption-tool	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.encryptiontool</groupId> <artifactId>mapr-encryption-tool</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.maprminicluster</groupId> <artifactId>mapr-mini-cluster</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	spark-client	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.3.0-mapr-mep-5.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-core	4.3.0-mapr-mep-5.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	4.3.0-mapr-mep-5.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1901</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	4.3.0-mapr-mep-5.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-mep-5.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>4.3.0-mapr-mep-5.x-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-mep-5.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>4.3.0-mapr-mep-5.x-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-mep-5.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>4.3.0-mapr-mep-5.x-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-mep-5.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>4.3.0-mapr-mep-5.x-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-mep-5.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>4.3.0-mapr-mep-5.x-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-mep-5.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-mep-5.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-mep-5.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-mep-5.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	4.3.0-mapr-mep-5.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-webapp	4.3.0-mapr-mep-5.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.3.0-mapr-mep-5.x-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-mep-5.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-5.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-mep-5.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-5.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-mep-5.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-utils</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-5.x-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-hive-common	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-common</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-hive-v2	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-v2</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-kafka	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-kafka</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-solr	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-solr</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-sqoop	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-common	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-db	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-kafka	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-kafka</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-sqoop	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-dist	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-dist</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-common	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-common</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-db	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-db</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-indexer	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-indexer</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-kafka	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-kafka</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-search	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-db	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-file	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-core	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-core</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-handlers	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

com.mapr.db	maprdb-spark	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-graphx_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-hive_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-network-shuffle_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-sketch_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-sql-kafka-0-10_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sql_2.11	2.2.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.2.1-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.11	2.2.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.11</artifactId> <version>2.2.1-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.11	2.2.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.11</artifactId> <version>2.2.1-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-flume_2.11	2.2.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.11</artifactId> <version>2.2.1-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.2.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.2.1-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-producer_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-streaming_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-tags_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.2.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.2.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	hadoop-shim	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	hadoop-shim-2.6	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim-2.6</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-api	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-common	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-dag	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-examples	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ext-service-tests	0.8.4-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ext-service-tests</artifactId>`
			`<version>0.8.4-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-job-analyzer	0.8.4-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.8.4-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-mapreduce	0.8.4-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.8.4-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.8.4-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.8.4-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.8.4-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.8.4-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-tests	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-tests</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-ui	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-ui2	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui2</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history-with-acls	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history-with-acls</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`

Maven Artifacts for EEP 5.0.1
Listed are all Maven artifacts for EEP 5.0.1 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-ant	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-ant</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-beeline	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-beeline</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-cli	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-cli</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-common	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-common</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-contrib	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-contrib</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-exec	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-pig-adapter</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-server-extensions</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-hwi	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hwi</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-common	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-ext-client</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-llap-server	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-server</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-llap-tez	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-maprdb-json-handler-test	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler-test</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-metastore	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-orc	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-orc</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-service	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-service-rpc	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-shims	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-shims</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.23</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-storage-api	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-storage-api</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-testutils	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-testutils</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat-java-client</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.conftool	mapr-conf-tool	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.conftool</groupId> <artifactId>mapr-conf-tool</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.encryptiontool	mapr-encryption-tool	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.encryptiontool</groupId> <artifactId>mapr-encryption-tool</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	spark-client	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-core	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie.test	oozie-mini	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-tools	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-utils</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	classpath-filter_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>classpath-filter_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
com.mapr.db	maprdb-spark	2.2.1-mapr-1808 Browse	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-spark</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-catalyst_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-catalyst_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-core_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-graphx_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-hive-thriftserver_2.11	2.2.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.2.1-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-hive_2.11	2.2.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.2.1-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.2.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.2.1-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.2.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.2.1-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.2.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.2.1-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-network-common_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-common_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-network-shuffle_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-network-yarn_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-repl_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-sketch_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-sql-kafka-0-10_2.11	2.2.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.2.1-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-sql_2.11	2.2.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.11</artifactId>`
			`<version>2.2.1-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-assembly_2.11	2.2.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.2.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.2.1-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume_2.11	2.2.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.2.1-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`
org.apache.spark	spark-unsafe_2.11	2.2.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-unsafe_2.11</artifactId> <version>2.2.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-yarn_2.11	2.2.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.2.1-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.6-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.6-mapr-1808</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.6-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.6-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.8.4-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.8.4-mapr-1808</version>`
			`</dependency>`

org.apache.tez	hadoop-shim-2.6	0.8.4-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim-2.6</artifactId>`
			`<version>0.8.4-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-api	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-common	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-dag	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-examples	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-job-analyzer	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-job-analyzer</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-mapreduce	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-mapreduce</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-runtime-internals	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-internals</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-runtime-library	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-library</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-tests	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-tests</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-ui	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-ui2	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui2</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-yarn-timeline-history	0.8.4-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.8.4-mapr-1808</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.8.4-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.8.4-mapr-1808</version>`
			`</dependency>`

Maven Artifacts for EEP 5.0.0

Listed are all Maven artifacts for EEP 5.0.0 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-legacy-sources	flume-avro-source	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-avro-source</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume	flume-checkstyle	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-checkstyle</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-dataset-sink	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-dataset-sink</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-file-channel	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-file-channel</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-hdfs-sink	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-hdfs-sink</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-hive-sink	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-hive-sink</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-http-sink	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-http-sink</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-irc-sink	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-irc-sink</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-jdbc-channel	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-jdbc-channel</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-jms-source	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-jms-source</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-kafka-channel	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-kafka-channel</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-kafka-source	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume	flume-ng-auth	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-auth</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume	flume-ng-configuration	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-configuration</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume	flume-ng-core	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-core</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-elasticsearch-sink	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-elasticsearch-sink</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume	flume-ng-embedded-agent	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-embedded-agent</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-ng-hbase-sink	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-hbase-sink</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-kafka-sink</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume.flume-ng-clients	flume-ng-log4jappender	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-clients</groupId>`
			`<artifactId>flume-ng-log4jappender</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-morphline-solr-sink	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-morphline-solr-sink</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume	flume-ng-node	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-node</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume	flume-ng-sdk	1.8.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-ng-sdk</artifactId> <version>1.8.0-mapr-1803</version> </dependency>`
org.apache.flume	flume-ng-tests	1.8.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-ng-tests</artifactId> <version>1.8.0-mapr-1803</version> </dependency>`
org.apache.flume.flume-ng-sources	flume-scribe-source	1.8.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sources</groupId> <artifactId>flume-scribe-source</artifactId> <version>1.8.0-mapr-1803</version> </dependency>`
org.apache.flume.flume-shared	flume-shared-kafka-test	1.8.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.flume.flume-shared</groupId> <artifactId>flume-shared-kafka-test</artifactId> <version>1.8.0-mapr-1803</version> </dependency>`
org.apache.flume.flume-ng-channels	flume-spillable-memory-channel	1.8.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-spillable-memory-channel</artifactId> <version>1.8.0-mapr-1803</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-taildir-source	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-taildir-source</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume.flume-ng-legacy-sources	flume-thrift-source	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-thrift-source</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume	flume-tools	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-tools</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-twitter-source	1.8.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-twitter-source</artifactId>`
			`<version>1.8.0-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-ant	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-ant</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-beeline	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-cli	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-common	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-contrib	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-exec	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-hbase-handler	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-hplsql	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hplsql</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-hwi	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hwi</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-jdbc	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-llap-client	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-client</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-llap-common	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-ext-client</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-llap-server	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-server</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-llap-tez	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler-test	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler-test</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-metastore	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-orc	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-orc</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-service	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-service-rpc	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-shims	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.23</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-storage-api	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-storage-api</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat-java-client</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive.conftool	mapr-conf-tool	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive.conftool</groupId> <artifactId>mapr-conf-tool</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive.maprminicluster</groupId> <artifactId>mapr-mini-cluster</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	spark-client	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>spark-client</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-client	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-client</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie	oozie-core	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie	oozie-examples	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie.test	oozie-mini	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-utils</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-hive-common	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-common</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-hive-v2	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-v2</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-kafka	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-kafka</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-solr	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-solr</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-sqoop	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-core-common	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-db	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-kafka	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-kafka</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-sqoop	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-dist	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-dist</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-common	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-common</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-db	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-db</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-indexer	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-indexer</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-kafka	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-kafka</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-search	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-cache	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-db	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-file	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-core	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-core</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	solr-sentry-handlers	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-graphx_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-hive_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-mllib_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sketch_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-sql_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-flume_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-unsafe_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.6-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.6-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.6-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.6-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	connector-sdk	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>connector-sdk</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sqoop	connector-sdk-hadoop	1.99.7-mapr-1803 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>connector-sdk-hadoop</artifactId> <version>1.99.7-mapr-1803</version> </dependency>`
org.apache.sqoop	sqoop-assemblies	1.99.7-mapr-1803 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-assemblies</artifactId> <version>1.99.7-mapr-1803</version> </dependency>`
org.apache.sqoop	sqoop-client	1.99.7-mapr-1803 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-client</artifactId> <version>1.99.7-mapr-1803</version> </dependency>`
org.apache.sqoop	sqoop-common	1.99.7-mapr-1803 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-common</artifactId> <version>1.99.7-mapr-1803</version> </dependency>`
org.apache.sqoop	sqoop-common-test	1.99.7-mapr-1803 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-common-test</artifactId> <version>1.99.7-mapr-1803</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop.connector	sqoop-connector-ftp	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-ftp</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-generic-jdbc	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-generic-jdbc</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-hdfs	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-hdfs</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-kafka	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-kafka</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-kite	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-kite</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop.connector	sqoop-connector-oracle-jdbc	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-oracle-jdbc</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-sftp	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-sftp</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop	sqoop-core	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-core</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop	sqoop-docs	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-docs</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop.execution	sqoop-execution-mapreduce	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.execution</groupId>`
			`<artifactId>sqoop-execution-mapreduce</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop.repository	sqoop-repository-common	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-common</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop.repository	sqoop-repository-derby	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-derby</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop.repository	sqoop-repository-mysql	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-mysql</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop.repository	sqoop-repository-postgresql	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-postgresql</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop	sqoop-security	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-security</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop-server	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-server</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop	sqoop-shell	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-shell</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop.submission	sqoop-submission-mapreduce	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.submission</groupId>`
			`<artifactId>sqoop-submission-mapreduce</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop	sqoop-tools	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-tools</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop	test	1.99.7-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>test</artifactId>`
			`<version>1.99.7-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	hadoop-shim	0.8.4-mapr-1803 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim</artifactId> <version>0.8.4-mapr-1803</version> </dependency>`
org.apache.tez	hadoop-shim-2.6	0.8.4-mapr-1803 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim-2.6</artifactId> <version>0.8.4-mapr-1803</version> </dependency>`
org.apache.tez	tez-api	0.8.4-mapr-1803 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.8.4-mapr-1803</version> </dependency>`
org.apache.tez	tez-common	0.8.4-mapr-1803 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.8.4-mapr-1803</version> </dependency>`
org.apache.tez	tez-dag	0.8.4-mapr-1803 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.8.4-mapr-1803</version> </dependency>`
org.apache.tez	tez-examples	0.8.4-mapr-1803 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.8.4-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ext-service-tests	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ext-service-tests</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-job-analyzer	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-mapreduce	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-tests	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-tests</artifactId><version>0.8.4-mapr-1803</version></dependency>`
org.apache.tez	tez-ui	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-ui</artifactId><version>0.8.4-mapr-1803</version></dependency>`
org.apache.tez	tez-ui2	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-ui2</artifactId><version>0.8.4-mapr-1803</version></dependency>`
org.apache.tez	tez-yarn-timeline-history	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-yarn-timeline-history</artifactId><version>0.8.4-mapr-1803</version></dependency>`
org.apache.tez	tez-yarn-timeline-history-with-acls	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-yarn-timeline-history-with-acls</artifactId><version>0.8.4-mapr-1803</version></dependency>`

Maven Artifacts for EEP 4.1.4
Listed are all Maven artifacts for EEP 4.1.4 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-annotations	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-annotations</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-checkstyle	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-checkstyle</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-client	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-client</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-common	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-common</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-examples	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-examples</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-hadoop-compat	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop-compat</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-hadoop2-compat	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop2-compat</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-it	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-it</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-native-client	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-native-client</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-prefix-tree	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-prefix-tree</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-procedure	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-procedure</artifactId><version>1.1.8-mapr-1904</version></dependency>`
org.apache.hbase	hbase-protocol	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-protocol</artifactId><version>1.1.8-mapr-1904</version></dependency>`
org.apache.hbase	hbase-resource-bundle	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-resource-bundle</artifactId><version>1.1.8-mapr-1904</version></dependency>`
org.apache.hbase	hbase-rest	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-rest</artifactId><version>1.1.8-mapr-1904</version></dependency>`
org.apache.hbase	hbase-server	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-server</artifactId><version>1.1.8-mapr-1904</version></dependency>`
org.apache.hbase	hbase-shaded-client	1.1.8-mapr-1904 Browse	`<dependency><groupId>org.apache.hbase</groupId><artifactId>hbase-shaded-client</artifactId><version>1.1.8-mapr-1904</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-shaded-server	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-server</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-shell	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shell</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-spark	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-spark</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-testing-util	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-testing-util</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`
org.apache.hbase	hbase-thrift	1.1.8-mapr-1904 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-thrift</artifactId> <version>1.1.8-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-accumulo-handler</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-ant	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-ant</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-beeline	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-cli	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-common	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-contrib	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-exec	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-exec</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-hbase-handler	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-hwi	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hwi</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-llap-common	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-llap-tez	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-maprdb-json-handler-test	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler-test</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-metastore	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-service	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-service-rpc	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive	hive-shims	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.23</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-storage-api	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-storage-api</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat-java-client</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive.conftool	mapr-conf-tool	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.conftool</groupId> <artifactId>mapr-conf-tool</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive.encryptiontool	mapr-encryption-tool	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.encryptiontool</groupId> <artifactId>mapr-encryption-tool</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`
org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1904 Browse	`<dependency> <groupId>org.apache.hive.maprminicluster</groupId> <artifactId>mapr-mini-cluster</artifactId> <version>2.1.1-mapr-1904</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	spark-client	2.1.1-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.1.1-mapr-1904</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.3.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.3.0-mapr-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-core	4.3.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.3.0-mapr-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	4.3.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.3.0-mapr-1904</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	4.3.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>4.3.0-mapr-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>4.3.0-mapr-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>4.3.0-mapr-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>4.3.0-mapr-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>4.3.0-mapr-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.3.0-mapr-1904</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>4.3.0-mapr-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>4.3.0-mapr-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>4.3.0-mapr-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.3.0-mapr-1904</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	4.3.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.3.0-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-webapp	4.3.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-webapp</artifactId> <version>4.3.0-mapr-1904</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-auth</artifactId> <version>hadoop-2-4.3.0-mapr-1904</version> </dependency>`
org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-distcp</artifactId> <version>hadoop-2-4.3.0-mapr-1904</version> </dependency>`
org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-utils</artifactId> <version>hadoop-2-4.3.0-mapr-1904</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	classpath-filter_2.11	2.1.0-mapr-1904 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>classpath-filter_2.11</artifactId><version>2.1.0-mapr-1904</version></dependency>`
com.mapr.db	maprdb-spark	2.1.0-mapr-1904 Browse	`<dependency><groupId>com.mapr.db</groupId><artifactId>maprdb-spark</artifactId><version>2.1.0-mapr-1904</version></dependency>`
org.apache.spark	spark-catalyst_2.11	2.1.0-mapr-1904 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-catalyst_2.11</artifactId><version>2.1.0-mapr-1904</version></dependency>`
org.apache.spark	spark-core_2.11	2.1.0-mapr-1904 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-core_2.11</artifactId><version>2.1.0-mapr-1904</version></dependency>`
org.apache.spark	spark-graphx_2.11	2.1.0-mapr-1904 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-graphx_2.11</artifactId><version>2.1.0-mapr-1904</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-hive-thriftserver_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`
org.apache.spark	spark-hive_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`
org.apache.spark	spark-launcher_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-launcher_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`
org.apache.spark	spark-mesos_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mesos_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`
org.apache.spark	spark-mllib-local_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib-local_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-mllib_2.11	2.1.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.1.0-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.1.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.1.0-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.1.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.1.0-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.1.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.1.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.1.0-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sketch_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`
org.apache.spark	spark-sql_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-flume_2.11	2.1.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.1.0-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.1.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.1.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.1.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.1.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.1.0-mapr-1904</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.1.0-mapr-1904 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.1.0-mapr-1904</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-unsafe_2.11	2.1.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.1.0-mapr-1904</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.1.0-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1904</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.6-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.6-mapr-1904</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.6-mapr-1904
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.6-mapr-1904</version>`
			`</dependency>`

Maven Artifacts for EEP 4.1.3

Listed are all Maven artifacts for EEP 4.1.3 components.

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1901 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-accumulo-handler</artifactId><version>2.1.1-mapr-1901</version></dependency>`
org.apache.hive	hive-ant	2.1.1-mapr-1901 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-ant</artifactId><version>2.1.1-mapr-1901</version></dependency>`
org.apache.hive	hive-beeline	2.1.1-mapr-1901 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-beeline</artifactId><version>2.1.1-mapr-1901</version></dependency>`
org.apache.hive	hive-cli	2.1.1-mapr-1901 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-cli</artifactId><version>2.1.1-mapr-1901</version></dependency>`
org.apache.hive	hive-common	2.1.1-mapr-1901 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-common</artifactId><version>2.1.1-mapr-1901</version></dependency>`
org.apache.hive	hive-contrib	2.1.1-mapr-1901 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-contrib</artifactId><version>2.1.1-mapr-1901</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-exec	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-exec</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-hbase-handler	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-hwi	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hwi</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-llap-common	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-tez	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-maprdb-json-handler-test	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler-test</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-metastore	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-service	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-service-rpc	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-shims	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-scheduler</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-storage-api	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-storage-api</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-testutils	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-testutils</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat-java-client</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive.conftool	mapr-conf-tool	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.conftool</groupId> <artifactId>mapr-conf-tool</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive.encryptiontool	mapr-encryption-tool	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.encryptiontool</groupId> <artifactId>mapr-encryption-tool</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.maprminicluster</groupId> <artifactId>mapr-mini-cluster</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	spark-client	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.3.0-mapr-mep-4.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.3.0-mapr-mep-4.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-core	4.3.0-mapr-mep-4.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.3.0-mapr-mep-4.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	4.3.0-mapr-mep-4.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.3.0-mapr-mep-4.x-1901</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	4.3.0-mapr-mep-4.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>4.3.0-mapr-mep-4.x-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-mep-4.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>4.3.0-mapr-mep-4.x-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-mep-4.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>4.3.0-mapr-mep-4.x-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-mep-4.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>4.3.0-mapr-mep-4.x-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-mep-4.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>4.3.0-mapr-mep-4.x-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-mep-4.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>4.3.0-mapr-mep-4.x-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-mep-4.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>4.3.0-mapr-mep-4.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-mep-4.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>4.3.0-mapr-mep-4.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-mep-4.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>4.3.0-mapr-mep-4.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-mep-4.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.3.0-mapr-mep-4.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	4.3.0-mapr-mep-4.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.3.0-mapr-mep-4.x-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-webapp	4.3.0-mapr-mep-4.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.3.0-mapr-mep-4.x-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-mep-4.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-4.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-mep-4.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-4.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-mep-4.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-utils</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-4.x-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-hive-common	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-common</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-hive-v2	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-v2</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-kafka	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-kafka</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-solr	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-solr</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-sqoop	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-common	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-db	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-kafka	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-kafka</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-sqoop	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-dist	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-dist</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-common	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-common</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-db	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-db</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-indexer	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-indexer</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-kafka	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-kafka</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-search	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-db	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-file	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-core	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-core</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-handlers	1.7.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.7.0-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
com.mapr.db	maprdb-spark	2.1.0-mapr-1901 Browse	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-spark</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-catalyst_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-catalyst_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-core_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-graphx_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-hive_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-network-shuffle_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-sketch_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-sql-kafka-0-10_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sql_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-flume_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-producer_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-streaming_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-tags_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	hadoop-shim	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	hadoop-shim-2.6	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim-2.6</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-api	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-common	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-dag	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-examples	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ext-service-tests	0.8.4-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ext-service-tests</artifactId>`
			`<version>0.8.4-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-job-analyzer	0.8.4-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.8.4-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-mapreduce	0.8.4-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.8.4-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.8.4-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.8.4-mapr-1901</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.8.4-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.8.4-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-tests	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-tests</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-ui	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-ui2	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui2</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history-with-acls	0.8.4-mapr-1901 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history-with-acls</artifactId> <version>0.8.4-mapr-1901</version> </dependency>`

Maven Artifacts for EEP 4.1.2
Listed are all Maven artifacts for EEP 4.1.2 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-ant	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-ant</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-beeline	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-beeline</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-cli	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-cli</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-common	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-common</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-contrib	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-contrib</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-exec	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-pig-adapter</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-server-extensions</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-hwi	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hwi</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-common	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-ext-client</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-llap-server	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-server</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-llap-tez	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-maprdb-json-handler-test	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler-test</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-metastore	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-orc	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-orc</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-service	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-service-rpc	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-shims	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-scheduler</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-storage-api	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-storage-api</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-testutils	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-testutils</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat-java-client</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.conftool	mapr-conf-tool	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.conftool</groupId> <artifactId>mapr-conf-tool</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.encryptiontool	mapr-encryption-tool	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.encryptiontool</groupId> <artifactId>mapr-encryption-tool</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	spark-client	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-core	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie.test	oozie-mini	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-tools	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-utils</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	classpath-filter_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>classpath-filter_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
com.mapr.db	maprdb-spark	2.1.0-mapr-1808 Browse	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-spark</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-catalyst_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-catalyst_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-core_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-graphx_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-hive-thriftserver_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-hive_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-launcher_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-launcher_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-mllib-local_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib-local_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-mllib_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-network-common_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-common_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-network-shuffle_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-network-yarn_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-repl_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-sketch_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-sql-kafka-0-10_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-sql_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-assembly_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-unsafe_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-unsafe_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-yarn_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.6-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.6-mapr-1808</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.6-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.6-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.8.4-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.8.4-mapr-1808</version>`
			`</dependency>`

org.apache.tez	hadoop-shim-2.6	0.8.4-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim-2.6</artifactId>`
			`<version>0.8.4-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-api	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-common	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-dag	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-examples	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-job-analyzer	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-job-analyzer</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-mapreduce	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-mapreduce</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-runtime-internals	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-internals</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-runtime-library	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-library</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-tests	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-tests</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-ui	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`
org.apache.tez	tez-ui2	0.8.4-mapr-1808 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui2</artifactId> <version>0.8.4-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-yarn-timeline-history	0.8.4-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.8.4-mapr-1808</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.8.4-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.8.4-mapr-1808</version>`
			`</dependency>`

Maven Artifacts for EEP 4.1.1

Listed are all Maven artifacts for EEP 4.1.1 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-ant	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-ant</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-beeline	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-beeline</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-cli	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-cli</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-common	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-common</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-contrib	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-contrib</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-exec	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-exec</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-hbase-handler	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-hwi	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hwi</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-jdbc	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-llap-client	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-client</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-llap-common	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-ext-client</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-llap-server	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-llap-tez	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler-test	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler-test</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-metastore	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-metastore</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-orc	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-orc</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-service	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-service-rpc	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-shims	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-storage-api	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-storage-api</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	spark-client	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-core	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie	oozie-examples	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie.test	oozie-mini	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-utils</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-hive-common	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-common</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-hive-v2	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-v2</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-kafka	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-kafka</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-solr	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-solr</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-sqoop	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-core-common	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-db	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-kafka	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-kafka</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-sqoop	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-dist	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-dist</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-common	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-common</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-db	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-db</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-indexer	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-indexer</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-kafka	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-kafka</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-search	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-cache	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-db	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-file	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-core	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-core</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	solr-sentry-handlers	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-graphx_2.11	2.1.0-mapr-1803 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-graphx_2.11</artifactId><version>2.1.0-mapr-1803</version></dependency>`
org.apache.spark	spark-hive-thriftserver_2.11	2.1.0-mapr-1803 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-hive-thriftserver_2.11</artifactId><version>2.1.0-mapr-1803</version></dependency>`
org.apache.spark	spark-hive_2.11	2.1.0-mapr-1803 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-hive_2.11</artifactId><version>2.1.0-mapr-1803</version></dependency>`
org.apache.spark	spark-launcher_2.11	2.1.0-mapr-1803 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-launcher_2.11</artifactId><version>2.1.0-mapr-1803</version></dependency>`
org.apache.spark	spark-mllib-local_2.11	2.1.0-mapr-1803 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-mllib-local_2.11</artifactId><version>2.1.0-mapr-1803</version></dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-mllib_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-sketch_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-sql-kafka-0-10_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-sql_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-assembly_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-flume_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.1.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.1.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.1.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.1.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.1.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-unsafe_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.6-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.6-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.6-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.6-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	hadoop-shim-2.6	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>hadoop-shim-2.6</artifactId><version>0.8.4-mapr-1803</version></dependency>`
org.apache.tez	tez-api	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-api</artifactId><version>0.8.4-mapr-1803</version></dependency>`
org.apache.tez	tez-common	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-common</artifactId><version>0.8.4-mapr-1803</version></dependency>`
org.apache.tez	tez-dag	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-dag</artifactId><version>0.8.4-mapr-1803</version></dependency>`
org.apache.tez	tez-examples	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-examples</artifactId><version>0.8.4-mapr-1803</version></dependency>`
org.apache.tez	tez-ext-service-tests	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-ext-service-tests</artifactId><version>0.8.4-mapr-1803</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-job-analyzer	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-mapreduce	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-tests	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-tests</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ui	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-ui2	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui2</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

Maven Artifacts for EEP 4.1.0

Listed are all Maven artifacts for EEP 4.1.0 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-core	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.3.0-mapr-1801</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1801</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-1801 Browse	<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-distcp</artifactId> <version>hadoop-2-4.3.0-mapr-1801</version> </dependency>
org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-1801 Browse	<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-utils</artifactId> <version>hadoop-2-4.3.0-mapr-1801</version> </dependency>

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	classpath-filter_2.11	2.1.0-mapr-1801 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>classpath-filter_2.11</artifactId> <version>2.1.0-mapr-1801</version> </dependency>
com.mapr.db	maprdb-spark	2.1.0-mapr-1801 Browse	<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-spark</artifactId> <version>2.1.0-mapr-1801</version> </dependency>
org.apache.spark	spark-catalyst_2.11	2.1.0-mapr-1801 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-catalyst_2.11</artifactId> <version>2.1.0-mapr-1801</version> </dependency>

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-core_2.11	2.1.0-mapr-1801 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.11</artifactId> <version>2.1.0-mapr-1801</version> </dependency>`
org.apache.spark	spark-graphx_2.11	2.1.0-mapr-1801 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.11</artifactId> <version>2.1.0-mapr-1801</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.11	2.1.0-mapr-1801 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.1.0-mapr-1801</version> </dependency>`
org.apache.spark	spark-hive_2.11	2.1.0-mapr-1801 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.11</artifactId> <version>2.1.0-mapr-1801</version> </dependency>`
org.apache.spark	spark-launcher_2.11	2.1.0-mapr-1801 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-launcher_2.11</artifactId> <version>2.1.0-mapr-1801</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-mllib-local_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-repl_2.11	2.1.0-mapr-1801 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-repl_2.11</artifactId><version>2.1.0-mapr-1801</version></dependency>`
org.apache.spark	spark-sketch_2.11	2.1.0-mapr-1801 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-sketch_2.11</artifactId><version>2.1.0-mapr-1801</version></dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.11	2.1.0-mapr-1801 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-sql-kafka-0-10_2.11</artifactId><version>2.1.0-mapr-1801</version></dependency>`
org.apache.spark	spark-sql_2.11	2.1.0-mapr-1801 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-sql_2.11</artifactId><version>2.1.0-mapr-1801</version></dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.11	2.1.0-mapr-1801 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-streaming-flume-assembly_2.11</artifactId><version>2.1.0-mapr-1801</version></dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-flume-sink_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-producer_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

org.apache.spark	spark-streaming_2.11	2.1.0-mapr-1801
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.1.0-mapr-1801</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-tags_2.11	2.1.0-mapr-1801 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.1.0-mapr-1801</version> </dependency>`
org.apache.spark	spark-unsafe_2.11	2.1.0-mapr-1801 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-unsafe_2.11</artifactId> <version>2.1.0-mapr-1801</version> </dependency>`
org.apache.spark	spark-yarn_2.11	2.1.0-mapr-1801 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-yarn_2.11</artifactId> <version>2.1.0-mapr-1801</version> </dependency>`

Maven Artifacts for EEP 4.0.0

Listed are all Maven artifacts for EEP 4.0.0 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-accumulo-handler</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive	hive-ant	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-ant</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-beeline	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive	hive-cli	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive	hive-common	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive	hive-contrib	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive	hive-exec	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive	hive-hbase-handler	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-core	1.2.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>1.2.0-mapr-1710</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	1.2.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>1.2.0-mapr-1710</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	1.2.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>1.2.0-mapr-1710</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	1.2.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>1.2.0-mapr-1710</version>`
			`</dependency>`

org.apache.hive	hive-hwi	1.2.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hwi</artifactId>`
			`<version>1.2.0-mapr-1710</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-jdbc	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive	hive-metastore	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive	hive-serde	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive	hive-service	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive	hive-shims	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive.shims	hive-shims-0.20S	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.20S</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.shims	hive-shims-0.23	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive.shims	hive-shims-common	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive.shims	hive-shims-scheduler	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-scheduler</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive	hive-testutils	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-testutils</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.hcatalog	hive-webhcat-java-client	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat-java-client</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive.maprminicluster	mapr-mini-cluster	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive.maprminicluster</groupId> <artifactId>mapr-mini-cluster</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`
org.apache.hive	spark-client	1.2.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>spark-client</artifactId> <version>1.2.0-mapr-1710</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-accumulo-handler</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-ant	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-ant</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-beeline	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-cli	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-common	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-contrib	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-exec	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.1.1-mapr-1710</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.1.1-mapr-1710</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.1.1-mapr-1710</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.1.1-mapr-1710</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.1.1-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.1.1-mapr-1710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-hwi	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hwi</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-jdbc	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-llap-client	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-client</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-llap-common	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-ext-client</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-server	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-server</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-llap-tez	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-metastore	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-orc	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-orc</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-service	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-service-rpc	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive	hive-shims	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`
org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1710 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>2.1.1-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.1.1-mapr-1710</version>`
			`</dependency>`

org.apache.hive	hive-storage-api	2.1.1-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-storage-api</artifactId>`
			`<version>2.1.1-mapr-1710</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.1.1-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.1.1-mapr-1710</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.1.1-mapr-1710</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.1.1-mapr-1710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.conftool	mapr-conf-tool	2.1.1-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.1.1-mapr-1710</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.1.1-mapr-1710</version>`
			`</dependency>`

org.apache.hive	spark-client	2.1.1-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.1.1-mapr-1710</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

org.apache.oozie	oozie-core	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-examples	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.3.0-mapr-1710</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1710</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-1710 Browse	<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-utils</artifactId> <version>hadoop-2-4.3.0-mapr-1710</version> </dependency>

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	classpath-filter_2.11	2.1.0-mapr-1710 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>classpath-filter_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>
com.mapr.db	maprdb-spark	2.1.0-mapr-1710 Browse	<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-spark</artifactId> <version>2.1.0-mapr-1710</version> </dependency>
org.apache.spark	spark-catalyst_2.11	2.1.0-mapr-1710 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-catalyst_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>
org.apache.spark	spark-core_2.11	2.1.0-mapr-1710 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-graphx_2.11	2.1.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.11</artifactId>`
			`<version>2.1.0-mapr-1710</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.11	2.1.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.1.0-mapr-1710</version>`
			`</dependency>`

org.apache.spark	spark-hive_2.11	2.1.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.1.0-mapr-1710</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.1.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.1.0-mapr-1710</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.1.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.1.0-mapr-1710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-mllib_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-network-common_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-common_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-network-shuffle_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-network-yarn_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-repl_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sketch_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-sql_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-flume_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.1.0-mapr-1710 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.1.0-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-unsafe_2.11	2.1.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.1.0-mapr-1710</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.1.0-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1710</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-core_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-graphx_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-hive_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-launcher_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-launcher_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-mllib-local_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-repl_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-sketch_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-sql_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-flume-sink_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-producer_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming_2.11	2.2.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.2.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-tags_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-unsafe_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-unsafe_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`
org.apache.spark	spark-yarn_2.11	2.2.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-yarn_2.11</artifactId> <version>2.2.1-mapr-1803</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sqoop	sqoop	1.4.6-mapr-1710 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop</artifactId> <version>1.4.6-mapr-1710</version> </dependency>`
org.apache.sqoop	sqoop-test	1.4.6-mapr-1710 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-test</artifactId> <version>1.4.6-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sqoop	connector-sdk	1.99.7-mapr-1710 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>connector-sdk</artifactId> <version>1.99.7-mapr-1710</version> </dependency>`
org.apache.sqoop	connector-sdk-hadoop	1.99.7-mapr-1710 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>connector-sdk-hadoop</artifactId> <version>1.99.7-mapr-1710</version> </dependency>`
org.apache.sqoop	sqoop-assemblies	1.99.7-mapr-1710 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-assemblies</artifactId> <version>1.99.7-mapr-1710</version> </dependency>`
org.apache.sqoop	sqoop-client	1.99.7-mapr-1710 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-client</artifactId> <version>1.99.7-mapr-1710</version> </dependency>`
org.apache.sqoop	sqoop-common	1.99.7-mapr-1710 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-common</artifactId> <version>1.99.7-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop-common-test	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-common-test</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-ftp	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-ftp</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-generic-jdbc	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-generic-jdbc</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-hdfs	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-hdfs</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-kafka	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-kafka</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop.connector	sqoop-connector-kite	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-kite</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-oracle-jdbc	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-oracle-jdbc</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-sftp	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-sftp</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

org.apache.sqoop	sqoop-core	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-core</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

org.apache.sqoop	sqoop-docs	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-docs</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop.execution	sqoop-execution-mapreduce	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.execution</groupId>`
			`<artifactId>sqoop-execution-mapreduce</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

org.apache.sqoop.repository	sqoop-repository-common	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-common</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

org.apache.sqoop.repository	sqoop-repository-derby	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-derby</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

org.apache.sqoop.repository	sqoop-repository-mysql	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-mysql</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

org.apache.sqoop.repository	sqoop-repository-postgresql	1.99.7-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-postgresql</artifactId>`
			`<version>1.99.7-mapr-1710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sqoop	sqoop-security	1.99.7-mapr-1710 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-security</artifactId> <version>1.99.7-mapr-1710</version> </dependency>`
org.apache.sqoop	sqoop-server	1.99.7-mapr-1710 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-server</artifactId> <version>1.99.7-mapr-1710</version> </dependency>`
org.apache.sqoop	sqoop-shell	1.99.7-mapr-1710 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-shell</artifactId> <version>1.99.7-mapr-1710</version> </dependency>`
org.apache.sqoop.submission	sqoop-submission-mapreduce	1.99.7-mapr-1710 Browse	`<dependency> <groupId>org.apache.sqoop.submission</groupId> <artifactId>sqoop-submission-mapreduce</artifactId> <version>1.99.7-mapr-1710</version> </dependency>`
org.apache.sqoop	sqoop-tools	1.99.7-mapr-1710 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-tools</artifactId> <version>1.99.7-mapr-1710</version> </dependency>`
org.apache.sqoop	test	1.99.7-mapr-1710 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>test</artifactId> <version>1.99.7-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	hadoop-shim	0.8.4-mapr-1710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim</artifactId> <version>0.8.4-mapr-1710</version> </dependency>`
org.apache.tez	hadoop-shim-2.6	0.8.4-mapr-1710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>hadoop-shim-2.6</artifactId> <version>0.8.4-mapr-1710</version> </dependency>`
org.apache.tez	tez-api	0.8.4-mapr-1710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-api</artifactId> <version>0.8.4-mapr-1710</version> </dependency>`
org.apache.tez	tez-common	0.8.4-mapr-1710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-common</artifactId> <version>0.8.4-mapr-1710</version> </dependency>`
org.apache.tez	tez-dag	0.8.4-mapr-1710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.8.4-mapr-1710</version> </dependency>`
org.apache.tez	tez-examples	0.8.4-mapr-1710 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.8.4-mapr-1710</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ext-service-tests	0.8.4-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ext-service-tests</artifactId>`
			`<version>0.8.4-mapr-1710</version>`
			`</dependency>`

org.apache.tez	tez-job-analyzer	0.8.4-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.8.4-mapr-1710</version>`
			`</dependency>`

org.apache.tez	tez-mapreduce	0.8.4-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.8.4-mapr-1710</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.8.4-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.8.4-mapr-1710</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.8.4-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.8.4-mapr-1710</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-tests	0.8.4-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-tests</artifactId>`
			`<version>0.8.4-mapr-1710</version>`
			`</dependency>`

org.apache.tez	tez-ui	0.8.4-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui</artifactId>`
			`<version>0.8.4-mapr-1710</version>`
			`</dependency>`

org.apache.tez	tez-ui2	0.8.4-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui2</artifactId>`
			`<version>0.8.4-mapr-1710</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history	0.8.4-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.8.4-mapr-1710</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.8.4-mapr-1710
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.8.4-mapr-1710</version>`
			`</dependency>`

Maven Artifacts for EEP 3.0.5
Listed are all Maven artifacts for EEP 3.0.5 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-ant	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-ant</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-beeline	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-beeline</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-cli	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-cli</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-common	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-common</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-contrib	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-contrib</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-exec	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-pig-adapter</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-server-extensions</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-hwi	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hwi</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-llap-common	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-llap-tez	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-maprdb-json-handler-test	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler-test</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-metastore	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-service	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-service-rpc	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`
org.apache.hive	hive-shims	2.1.1-mapr-1901 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.1.1-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.23</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-storage-api	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-storage-api</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive.encryptiontool	mapr-encryption-tool	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	spark-client	2.1.1-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.1.1-mapr-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.3.0-mapr-mep-3.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.3.0-mapr-mep-3.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-core	4.3.0-mapr-mep-3.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.3.0-mapr-mep-3.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	4.3.0-mapr-mep-3.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.3.0-mapr-mep-3.x-1901</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	4.3.0-mapr-mep-3.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>4.3.0-mapr-mep-3.x-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-mep-3.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>4.3.0-mapr-mep-3.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-mep-3.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>4.3.0-mapr-mep-3.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-mep-3.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>4.3.0-mapr-mep-3.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-mep-3.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>4.3.0-mapr-mep-3.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-mep-3.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.3.0-mapr-mep-3.x-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-mep-3.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>4.3.0-mapr-mep-3.x-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-mep-3.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>4.3.0-mapr-mep-3.x-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-mep-3.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-sqoop</artifactId> <version>4.3.0-mapr-mep-3.x-1901</version> </dependency>`
org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-mep-3.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-streaming</artifactId> <version>4.3.0-mapr-mep-3.x-1901</version> </dependency>`
org.apache.oozie	oozie-tools	4.3.0-mapr-mep-3.x-1901 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-tools</artifactId> <version>4.3.0-mapr-mep-3.x-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-webapp	4.3.0-mapr-mep-3.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.3.0-mapr-mep-3.x-1901</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-mep-3.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-3.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-mep-3.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-3.x-1901</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-mep-3.x-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-utils</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-mep-3.x-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	classpath-filter_2.11	2.1.0-mapr-1901 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>classpath-filter_2.11</artifactId><version>2.1.0-mapr-1901</version></dependency>`
com.mapr.db	maprdb-spark	2.1.0-mapr-1901 Browse	`<dependency><groupId>com.mapr.db</groupId><artifactId>maprdb-spark</artifactId><version>2.1.0-mapr-1901</version></dependency>`
org.apache.spark	spark-catalyst_2.11	2.1.0-mapr-1901 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-catalyst_2.11</artifactId><version>2.1.0-mapr-1901</version></dependency>`
org.apache.spark	spark-core_2.11	2.1.0-mapr-1901 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-core_2.11</artifactId><version>2.1.0-mapr-1901</version></dependency>`
org.apache.spark	spark-graphx_2.11	2.1.0-mapr-1901 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-graphx_2.11</artifactId><version>2.1.0-mapr-1901</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-hive-thriftserver_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-hive_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-launcher_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-launcher_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-mllib-local_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib-local_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-mllib_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-network-common_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-sketch_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-sql-kafka-0-10_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-sql_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-assembly_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume_2.11	2.1.0-mapr-1901
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.1.0-mapr-1901</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.1.0-mapr-1901 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.1.0-mapr-1901 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.1.0-mapr-1901 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.1.0-mapr-1901 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.1.0-mapr-1901 Browse	<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`
org.apache.spark	spark-unsafe_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-unsafe_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-yarn_2.11	2.1.0-mapr-1901 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-yarn_2.11</artifactId> <version>2.1.0-mapr-1901</version> </dependency>`

Maven Artifacts for EEP 3.0.4

Listed are all Maven artifacts for EEP 3.0.4 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-accumulo-handler</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-ant	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-ant</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-beeline	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-cli	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-common	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-contrib	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-exec	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-hwi	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hwi</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-jdbc	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-llap-client	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-client</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-llap-common	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-tez	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-maprdb-json-handler-test	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler-test</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-metastore	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-orc	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-orc</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-service	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-service-rpc	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-shims	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-scheduler</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-storage-api	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-storage-api</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive	hive-testutils	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-testutils</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`
org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1808 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-webhcat-java-client</artifactId> <version>2.1.1-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.conftool	mapr-conf-tool	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive.encryptiontool	mapr-encryption-tool	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.encryptiontool</groupId>`
			`<artifactId>mapr-encryption-tool</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

org.apache.hive	spark-client	2.1.1-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.1.1-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.3.0-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-core	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-examples	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie.test	oozie-mini	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-distcp</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hcatalog</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-hive2</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-sqoop</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-streaming</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-tools	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-tools</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`
org.apache.oozie	oozie-webapp	4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-webapp</artifactId> <version>4.3.0-mapr-1808</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-auth</artifactId> <version>hadoop-2-4.3.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1808</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-utils</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

com.mapr.db	maprdb-spark	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>com.mapr.db</groupId>`
			`<artifactId>maprdb-spark</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-core_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-graphx_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-hive_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-launcher_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-launcher_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-mllib-local_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-repl_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-sketch_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-sql-kafka-0-10_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql-kafka-0-10_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-sql_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-flume-sink_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-flume_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.1.0-mapr-1808 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.1.0-mapr-1808</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-producer_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-streaming_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-tags_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.1.0-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1808</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.6-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.6-mapr-1808</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.6-mapr-1808
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.6-mapr-1808</version>`
			`</dependency>`

Maven Artifacts for EEP 3.0.3

Listed are all Maven artifacts for EEP 3.0.3 components.

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1803 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-accumulo-handler</artifactId><version>2.1.1-mapr-1803</version></dependency>`
org.apache.hive	hive-ant	2.1.1-mapr-1803 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-ant</artifactId><version>2.1.1-mapr-1803</version></dependency>`
org.apache.hive	hive-beeline	2.1.1-mapr-1803 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-beeline</artifactId><version>2.1.1-mapr-1803</version></dependency>`
org.apache.hive	hive-cli	2.1.1-mapr-1803 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-cli</artifactId><version>2.1.1-mapr-1803</version></dependency>`
org.apache.hive	hive-common	2.1.1-mapr-1803 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-common</artifactId><version>2.1.1-mapr-1803</version></dependency>`
org.apache.hive	hive-contrib	2.1.1-mapr-1803 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-contrib</artifactId><version>2.1.1-mapr-1803</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-exec	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-exec</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-hbase-handler	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-streaming</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-hplsql	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hplsql</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-hwi	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hwi</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-jdbc	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-llap-client	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-client</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-llap-common	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-common</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-ext-client</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-llap-server	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-server</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-llap-tez	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-llap-tez</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive	hive-maprdb-json-handler	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-maprdb-json-handler</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-maprdb-json-handler-test	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-maprdb-json-handler-test</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-metastore	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-orc	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-orc</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-service	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-service-rpc	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-shims	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-scheduler</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`
org.apache.hive	hive-storage-api	2.1.1-mapr-1803 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-storage-api</artifactId> <version>2.1.1-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-testutils	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.conftool	mapr-conf-tool	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.conftool</groupId>`
			`<artifactId>mapr-conf-tool</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	spark-client	2.1.1-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>2.1.1-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-core	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.3.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-1803 Browse	`<dependency><groupId>org.apache.oozie</groupId><artifactId>oozie-sharelib-pig</artifactId><version>4.3.0-mapr-1803</version></dependency>`
org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-1803 Browse	`<dependency><groupId>org.apache.oozie</groupId><artifactId>oozie-sharelib-spark</artifactId><version>4.3.0-mapr-1803</version></dependency>`
org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-1803 Browse	`<dependency><groupId>org.apache.oozie</groupId><artifactId>oozie-sharelib-sqoop</artifactId><version>4.3.0-mapr-1803</version></dependency>`
org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-1803 Browse	`<dependency><groupId>org.apache.oozie</groupId><artifactId>oozie-sharelib-streaming</artifactId><version>4.3.0-mapr-1803</version></dependency>`
org.apache.oozie	oozie-tools	4.3.0-mapr-1803 Browse	`<dependency><groupId>org.apache.oozie</groupId><artifactId>oozie-tools</artifactId><version>4.3.0-mapr-1803</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-webapp	4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-webapp</artifactId> <version>4.3.0-mapr-1803</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-auth</artifactId> <version>hadoop-2-4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-distcp</artifactId> <version>hadoop-2-4.3.0-mapr-1803</version> </dependency>`
org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-utils</artifactId> <version>hadoop-2-4.3.0-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-hive-common	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-common</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-hive-v2	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-v2</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-kafka	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-kafka</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-solr	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-solr</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-sqoop	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-common	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-db	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-kafka	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-kafka</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-sqoop	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-dist	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-dist</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-common	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-common</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-db	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-db</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-indexer	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-indexer</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-kafka	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-kafka</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-search	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-db	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-file	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-core	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-core</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-handlers	1.7.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.7.0-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	classpath-filter_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>classpath-filter_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
com.mapr.db	maprdb-spark	2.1.0-mapr-1803 Browse	`<dependency> <groupId>com.mapr.db</groupId> <artifactId>maprdb-spark</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`
org.apache.spark	spark-catalyst_2.11	2.1.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-catalyst_2.11</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`
org.apache.spark	spark-core_2.11	2.1.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.11</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`
org.apache.spark	spark-graphx_2.11	2.1.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.11</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.11	2.1.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-hive_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-network-shuffle_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-sketch_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-sql-kafka-0-10_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sql_2.11	2.1.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.11	2.1.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.11</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.11	2.1.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.11</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`
org.apache.spark	spark-streaming-flume_2.11	2.1.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.11</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.1.0-mapr-1803 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.1.0-mapr-1803</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-producer_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-streaming_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-tags_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.1.0-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.6-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.6-mapr-1803</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.6-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.6-mapr-1803</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	hadoop-shim-2.6	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim-2.6</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-api	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-api</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-common	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-common</artifactId><version>0.8.4-mapr-1803</version></dependency>`
org.apache.tez	tez-dag	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-dag</artifactId><version>0.8.4-mapr-1803</version></dependency>`
org.apache.tez	tez-examples	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-examples</artifactId><version>0.8.4-mapr-1803</version></dependency>`
org.apache.tez	tez-ext-service-tests	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-ext-service-tests</artifactId><version>0.8.4-mapr-1803</version></dependency>`
org.apache.tez	tez-job-analyzer	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-job-analyzer</artifactId><version>0.8.4-mapr-1803</version></dependency>`
org.apache.tez	tez-mapreduce	0.8.4-mapr-1803 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-mapreduce</artifactId><version>0.8.4-mapr-1803</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-runtime-internals	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-tests	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-tests</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-ui	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-ui2	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ui2</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-yarn-timeline-history	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.8.4-mapr-1803
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.8.4-mapr-1803</version>`
			`</dependency>`

Maven Artifacts for EEP 3.0.1

Listed are all Maven artifacts for EEP 3.0.1 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

org.apache.hive	hive-ant	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-ant</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

org.apache.hive	hive-beeline	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-beeline</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-cli	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-cli</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

org.apache.hive	hive-common	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-common</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

org.apache.hive	hive-contrib	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-contrib</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

org.apache.hive	hive-exec	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-exec</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

org.apache.hive	hive-hbase-handler	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hbase-handler</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-hwi	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hwi</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`
org.apache.hive	hive-jdbc	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`
org.apache.hive	hive-llap-client	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-client</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`
org.apache.hive	hive-llap-common	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`
org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-ext-client</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-server	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-server</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`
org.apache.hive	hive-llap-tez	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`
org.apache.hive	hive-metastore	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`
org.apache.hive	hive-orc	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-orc</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`
org.apache.hive	hive-service	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-service-rpc	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service-rpc</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`
org.apache.hive	hive-shims	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`
org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`
org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-scheduler</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-storage-api	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-storage-api</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.1.1-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	spark-client	2.1.1-mapr-1707 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>spark-client</artifactId> <version>2.1.1-mapr-1707</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-client	4.3.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-client</artifactId> <version>4.3.0-mapr-1707</version> </dependency>`
org.apache.oozie	oozie-core	4.3.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>4.3.0-mapr-1707</version> </dependency>`
org.apache.oozie	oozie-examples	4.3.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>4.3.0-mapr-1707</version> </dependency>`
org.apache.oozie.test	oozie-mini	4.3.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>4.3.0-mapr-1707</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>4.3.0-mapr-1707</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>4.3.0-mapr-1707</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>4.3.0-mapr-1707</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>4.3.0-mapr-1707</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.3.0-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>4.3.0-mapr-1707</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>4.3.0-mapr-1707</version> </dependency>`
org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-sqoop</artifactId> <version>4.3.0-mapr-1707</version> </dependency>`
org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-streaming</artifactId> <version>4.3.0-mapr-1707</version> </dependency>`
org.apache.oozie	oozie-tools	4.3.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-tools</artifactId> <version>4.3.0-mapr-1707</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-webapp	4.3.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-webapp</artifactId> <version>4.3.0-mapr-1707</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-auth</artifactId> <version>hadoop-2-4.3.0-mapr-1707</version> </dependency>`
org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-distcp</artifactId> <version>hadoop-2-4.3.0-mapr-1707</version> </dependency>`
org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-utils</artifactId> <version>hadoop-2-4.3.0-mapr-1707</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-catalyst_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-graphx_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-hive_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-launcher_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-network-yarn_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-sketch_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-sql-kafka-0-10_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-sql_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-flume-assembly_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-9_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-9_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-producer_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-tags_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.1.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1707</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.6-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.6-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop-test	1.4.6-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.6-mapr-1707</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	hadoop-shim	0.8.4-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim</artifactId>`
			`<version>0.8.4-mapr-1707</version>`
			`</dependency>`

org.apache.tez	hadoop-shim-2.6	0.8.4-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>hadoop-shim-2.6</artifactId>`
			`<version>0.8.4-mapr-1707</version>`
			`</dependency>`

org.apache.tez	tez-api	0.8.4-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-api</artifactId>`
			`<version>0.8.4-mapr-1707</version>`
			`</dependency>`

org.apache.tez	tez-common	0.8.4-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-common</artifactId>`
			`<version>0.8.4-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-dag	0.8.4-mapr-1707 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-dag</artifactId> <version>0.8.4-mapr-1707</version> </dependency>`
org.apache.tez	tez-examples	0.8.4-mapr-1707 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-examples</artifactId> <version>0.8.4-mapr-1707</version> </dependency>`
org.apache.tez	tez-ext-service-tests	0.8.4-mapr-1707 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ext-service-tests</artifactId> <version>0.8.4-mapr-1707</version> </dependency>`
org.apache.tez	tez-job-analyzer	0.8.4-mapr-1707 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-job-analyzer</artifactId> <version>0.8.4-mapr-1707</version> </dependency>`
org.apache.tez	tez-mapreduce	0.8.4-mapr-1707 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-mapreduce</artifactId> <version>0.8.4-mapr-1707</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-runtime-internals	0.8.4-mapr-1707 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-internals</artifactId> <version>0.8.4-mapr-1707</version> </dependency>`
org.apache.tez	tez-runtime-library	0.8.4-mapr-1707 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-runtime-library</artifactId> <version>0.8.4-mapr-1707</version> </dependency>`
org.apache.tez	tez-tests	0.8.4-mapr-1707 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-tests</artifactId> <version>0.8.4-mapr-1707</version> </dependency>`
org.apache.tez	tez-ui	0.8.4-mapr-1707 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui</artifactId> <version>0.8.4-mapr-1707</version> </dependency>`
org.apache.tez	tez-ui2	0.8.4-mapr-1707 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui2</artifactId> <version>0.8.4-mapr-1707</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-yarn-timeline-history	0.8.4-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history</artifactId>`
			`<version>0.8.4-mapr-1707</version>`
			`</dependency>`

org.apache.tez	tez-yarn-timeline-history-with-acls	0.8.4-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-yarn-timeline-history-with-acls</artifactId>`
			`<version>0.8.4-mapr-1707</version>`
			`</dependency>`

Maven Artifacts for EEP 3.0.0

Listed are all Maven artifacts for EEP 3.0.0 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-legacy-sources	flume-avro-source	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-avro-source</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-dataset-sink	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-dataset-sink</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-channels	flume-file-channel	1.7.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-file-channel</artifactId> <version>1.7.0-mapr-1703</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-hdfs-sink	1.7.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-hdfs-sink</artifactId> <version>1.7.0-mapr-1703</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-hive-sink	1.7.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-hive-sink</artifactId> <version>1.7.0-mapr-1703</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-irc-sink	1.7.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-irc-sink</artifactId> <version>1.7.0-mapr-1703</version> </dependency>`
org.apache.flume.flume-ng-channels	flume-jdbc-channel	1.7.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-jdbc-channel</artifactId> <version>1.7.0-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-jms-source	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-jms-source</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-kafka-channel	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-kafka-channel</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-kafka-source	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.flume	flume-ng-auth	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-auth</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.flume	flume-ng-configuration	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-configuration</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume	flume-ng-core	1.7.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-ng-core</artifactId> <version>1.7.0-mapr-1703</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-elasticsearch-sink	1.7.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-elasticsearch-sink</artifactId> <version>1.7.0-mapr-1703</version> </dependency>`
org.apache.flume	flume-ng-embedded-agent	1.7.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-ng-embedded-agent</artifactId> <version>1.7.0-mapr-1703</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-hbase-sink	1.7.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-hbase-sink</artifactId> <version>1.7.0-mapr-1703</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink	1.7.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-kafka-sink</artifactId> <version>1.7.0-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-clients	flume-ng-log4jappender	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-clients</groupId>`
			`<artifactId>flume-ng-log4jappender</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-morphline-solr-sink	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-morphline-solr-sink</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.flume	flume-ng-node	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-node</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.flume	flume-ng-sdk	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-sdk</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.flume	flume-ng-tests	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-tests</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-scribe-source	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-scribe-source</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.flume.flume-shared	flume-shared-kafka-test	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-shared</groupId>`
			`<artifactId>flume-shared-kafka-test</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-spillable-memory-channel	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-spillable-memory-channel</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-taildir-source	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-taildir-source</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.flume.flume-ng-legacy-sources	flume-thrift-source	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-thrift-source</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume	flume-tools	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-tools</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-twitter-source	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-twitter-source</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-annotations	1.1.8-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-annotations</artifactId>`
			`<version>1.1.8-mapr-1703</version>`
			`</dependency>`

org.apache.hbase	hbase-checkstyle	1.1.8-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-checkstyle</artifactId>`
			`<version>1.1.8-mapr-1703</version>`
			`</dependency>`

org.apache.hbase	hbase-client	1.1.8-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-client</artifactId>`
			`<version>1.1.8-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-common	1.1.8-mapr-1703 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-common</artifactId> <version>1.1.8-mapr-1703</version> </dependency>`
org.apache.hbase	hbase-examples	1.1.8-mapr-1703 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-examples</artifactId> <version>1.1.8-mapr-1703</version> </dependency>`
org.apache.hbase	hbase-hadoop-compat	1.1.8-mapr-1703 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop-compat</artifactId> <version>1.1.8-mapr-1703</version> </dependency>`
org.apache.hbase	hbase-hadoop2-compat	1.1.8-mapr-1703 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-hadoop2-compat</artifactId> <version>1.1.8-mapr-1703</version> </dependency>`
org.apache.hbase	hbase-it	1.1.8-mapr-1703 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-it</artifactId> <version>1.1.8-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-native-client	1.1.8-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-native-client</artifactId>`
			`<version>1.1.8-mapr-1703</version>`
			`</dependency>`

org.apache.hbase	hbase-prefix-tree	1.1.8-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-prefix-tree</artifactId>`
			`<version>1.1.8-mapr-1703</version>`
			`</dependency>`

org.apache.hbase	hbase-procedure	1.1.8-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-procedure</artifactId>`
			`<version>1.1.8-mapr-1703</version>`
			`</dependency>`

org.apache.hbase	hbase-protocol	1.1.8-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-protocol</artifactId>`
			`<version>1.1.8-mapr-1703</version>`
			`</dependency>`

org.apache.hbase	hbase-resource-bundle	1.1.8-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-resource-bundle</artifactId>`
			`<version>1.1.8-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hbase	hbase-rest	1.1.8-mapr-1703 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-rest</artifactId> <version>1.1.8-mapr-1703</version> </dependency>`
org.apache.hbase	hbase-server	1.1.8-mapr-1703 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-server</artifactId> <version>1.1.8-mapr-1703</version> </dependency>`
org.apache.hbase	hbase-shaded-client	1.1.8-mapr-1703 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-client</artifactId> <version>1.1.8-mapr-1703</version> </dependency>`
org.apache.hbase	hbase-shaded-server	1.1.8-mapr-1703 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shaded-server</artifactId> <version>1.1.8-mapr-1703</version> </dependency>`
org.apache.hbase	hbase-shell	1.1.8-mapr-1703 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-shell</artifactId> <version>1.1.8-mapr-1703</version> </dependency>`
org.apache.hbase	hbase-spark	1.1.8-mapr-1703 Browse	`<dependency> <groupId>org.apache.hbase</groupId> <artifactId>hbase-spark</artifactId> <version>1.1.8-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hbase	hbase-testing-util	1.1.8-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-testing-util</artifactId>`
			`<version>1.1.8-mapr-1703</version>`
			`</dependency>`

org.apache.hbase	hbase-thrift	1.1.8-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hbase</groupId>`
			`<artifactId>hbase-thrift</artifactId>`
			`<version>1.1.8-mapr-1703</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

org.apache.hive	hive-ant	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-ant</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

org.apache.hive	hive-beeline	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-beeline</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-cli	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`
org.apache.hive	hive-common	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`
org.apache.hive	hive-contrib	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`
org.apache.hive	hive-exec	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`
org.apache.hive	hive-hbase-handler	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-core	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-core</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

org.apache.hive	hive-hplsql	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hplsql</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-hwi	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hwi</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`
org.apache.hive	hive-jdbc	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-jdbc</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`
org.apache.hive	hive-llap-client	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-client</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`
org.apache.hive	hive-llap-common	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-common</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`
org.apache.hive	hive-llap-ext-client	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-ext-client</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-llap-server	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-server</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`
org.apache.hive	hive-llap-tez	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-llap-tez</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`
org.apache.hive	hive-metastore	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`
org.apache.hive	hive-orc	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-orc</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`
org.apache.hive	hive-serde	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`
org.apache.hive	hive-service	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-service-rpc	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service-rpc</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

org.apache.hive	hive-shims	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-shims</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-0.23	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.23</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-common	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-storage-api	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-storage-api</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

org.apache.hive	hive-testutils	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

org.apache.hive.maprminicluster	mapr-mini-cluster	2.1.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.maprminicluster</groupId>`
			`<artifactId>mapr-mini-cluster</artifactId>`
			`<version>2.1.1-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	spark-client	2.1.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>spark-client</artifactId> <version>2.1.1-mapr-1703</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-client	4.3.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-client</artifactId> <version>4.3.0-mapr-1703</version> </dependency>`
org.apache.oozie	oozie-core	4.3.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-core</artifactId> <version>4.3.0-mapr-1703</version> </dependency>`
org.apache.oozie	oozie-examples	4.3.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-examples</artifactId> <version>4.3.0-mapr-1703</version> </dependency>`
org.apache.oozie.test	oozie-mini	4.3.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.oozie.test</groupId> <artifactId>oozie-mini</artifactId> <version>4.3.0-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-distcp	4.3.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>4.3.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	4.3.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>4.3.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	4.3.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>4.3.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	4.3.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>4.3.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	4.3.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.3.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-pig	4.3.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>4.3.0-mapr-1703</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	4.3.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>4.3.0-mapr-1703</version> </dependency>`
org.apache.oozie	oozie-sharelib-sqoop	4.3.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-sqoop</artifactId> <version>4.3.0-mapr-1703</version> </dependency>`
org.apache.oozie	oozie-sharelib-streaming	4.3.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-streaming</artifactId> <version>4.3.0-mapr-1703</version> </dependency>`
org.apache.oozie	oozie-tools	4.3.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-tools</artifactId> <version>4.3.0-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-webapp	4.3.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.3.0-mapr-1703</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-2-4.3.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-distcp	hadoop-2-4.3.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-utils	hadoop-2-4.3.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-utils</artifactId>`
			`<version>hadoop-2-4.3.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.pig	pig	0.16.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pig</artifactId>`
			`<version>0.16.0-mapr-1703</version>`
			`</dependency>`

org.apache.pig	piggybank	0.16.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>piggybank</artifactId>`
			`<version>0.16.0-mapr-1703</version>`
			`</dependency>`

org.apache.pig	pigsmoke	0.16.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigsmoke</artifactId>`
			`<version>0.16.0-mapr-1703</version>`
			`</dependency>`

org.apache.pig	pigunit	0.16.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigunit</artifactId>`
			`<version>0.16.0-mapr-1703</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-hive-common	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-common</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-hive-v2	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive-v2</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-kafka	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-kafka</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-solr	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-solr</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-sqoop	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-core-common	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-db	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-kafka	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-kafka</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-sqoop	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-dist	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-dist</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-common	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-common</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-db	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-db</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-indexer	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-indexer</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-kafka	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-kafka</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-search	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-db	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-file	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-core	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-core</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-handlers	1.7.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.7.0-mapr-1703</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-catalyst_2.11	2.1.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.1.0-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.1.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.1.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-graphx_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`
org.apache.spark	spark-hive_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`
org.apache.spark	spark-launcher_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-launcher_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`
org.apache.spark	spark-mllib-local_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib-local_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-mllib_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`
org.apache.spark	spark-network-common_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-common_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`
org.apache.spark	spark-network-shuffle_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`
org.apache.spark	spark-network-yarn_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`
org.apache.spark	spark-repl_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-sketch_2.11	2.1.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.11</artifactId>`
			`<version>2.1.0-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-sql-kafka-0-10_2.11	2.1.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-sql_2.11	2.1.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.11</artifactId>`
			`<version>2.1.0-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-assembly_2.11	2.1.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.1.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.1.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-flume_2.11	2.1.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.11</artifactId>`
			`<version>2.1.0-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.1.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-10_2.11	2.1.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-10_2.11</artifactId>`
			`<version>2.1.0-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.1.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId>`
			`<version>2.1.0-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-0-8_2.11	2.1.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-0-8_2.11</artifactId>`
			`<version>2.1.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.1.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.1.0-mapr-1703</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-unsafe_2.11	2.1.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.1.0-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.1.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.1.0-mapr-1703</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.6-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.6-mapr-1703</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.6-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.6-mapr-1703</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.storm	flux-core	0.10.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>flux-core</artifactId>`
			`<version>0.10.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.storm	flux-examples	0.10.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>flux-examples</artifactId>`
			`<version>0.10.0-mapr-1703</version>`
			`</dependency>`

org.apache.storm	flux-wrappers	0.10.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>flux-wrappers</artifactId>`
			`<version>0.10.0-mapr-1703</version>`
			`</dependency>`

org.apache.storm	maven-shade-clojure-transformer	0.10.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>maven-shade-clojure-transformer</artifactId>`
			`<version>0.10.0-mapr-1703</version>`
			`</dependency>`

org.apache.storm	multilang-javascript	0.10.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>multilang-javascript</artifactId>`
			`<version>0.10.0-mapr-1703</version>`
			`</dependency>`

org.apache.storm	multilang-python	0.10.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>multilang-python</artifactId>`
			`<version>0.10.0-mapr-1703</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.storm	multilang-ruby	0.10.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>multilang-ruby</artifactId> <version>0.10.0-mapr-1703</version> </dependency>`
org.apache.storm	storm-core	0.10.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-core</artifactId> <version>0.10.0-mapr-1703</version> </dependency>`
org.apache.storm	storm-eventhubs	0.10.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-eventhubs</artifactId> <version>0.10.0-mapr-1703</version> </dependency>`
org.apache.storm	storm-hbase	0.10.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-hbase</artifactId> <version>0.10.0-mapr-1703</version> </dependency>`
org.apache.storm	storm-hdfs	0.10.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-hdfs</artifactId> <version>0.10.0-mapr-1703</version> </dependency>`
org.apache.storm	storm-hive	0.10.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-hive</artifactId> <version>0.10.0-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.storm	storm-jdbc	0.10.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-jdbc</artifactId> <version>0.10.0-mapr-1703</version> </dependency>`
org.apache.storm	storm-kafka	0.10.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-kafka</artifactId> <version>0.10.0-mapr-1703</version> </dependency>`
org.apache.storm	storm-maven-plugins	0.10.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-maven-plugins</artifactId> <version>0.10.0-mapr-1703</version> </dependency>`
org.apache.storm	storm-redis	0.10.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-redis</artifactId> <version>0.10.0-mapr-1703</version> </dependency>`
org.apache.storm	storm-starter	0.10.0-mapr-1703 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-starter</artifactId> <version>0.10.0-mapr-1703</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	hadoop-shim	0.8.4-mapr-1703 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>hadoop-shim</artifactId><version>0.8.4-mapr-1703</version></dependency>`
org.apache.tez	hadoop-shim-2.6	0.8.4-mapr-1703 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>hadoop-shim-2.6</artifactId><version>0.8.4-mapr-1703</version></dependency>`
org.apache.tez	tez-api	0.8.4-mapr-1703 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-api</artifactId><version>0.8.4-mapr-1703</version></dependency>`
org.apache.tez	tez-common	0.8.4-mapr-1703 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-common</artifactId><version>0.8.4-mapr-1703</version></dependency>`
org.apache.tez	tez-dag	0.8.4-mapr-1703 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-dag</artifactId><version>0.8.4-mapr-1703</version></dependency>`
org.apache.tez	tez-examples	0.8.4-mapr-1703 Browse	`<dependency><groupId>org.apache.tez</groupId><artifactId>tez-examples</artifactId><version>0.8.4-mapr-1703</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.tez	tez-ext-service-tests	0.8.4-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-ext-service-tests</artifactId>`
			`<version>0.8.4-mapr-1703</version>`
			`</dependency>`

org.apache.tez	tez-job-analyzer	0.8.4-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-job-analyzer</artifactId>`
			`<version>0.8.4-mapr-1703</version>`
			`</dependency>`

org.apache.tez	tez-mapreduce	0.8.4-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-mapreduce</artifactId>`
			`<version>0.8.4-mapr-1703</version>`
			`</dependency>`

org.apache.tez	tez-runtime-internals	0.8.4-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-internals</artifactId>`
			`<version>0.8.4-mapr-1703</version>`
			`</dependency>`

org.apache.tez	tez-runtime-library	0.8.4-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.tez</groupId>`
			`<artifactId>tez-runtime-library</artifactId>`
			`<version>0.8.4-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.tez	tez-tests	0.8.4-mapr-1703 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-tests</artifactId> <version>0.8.4-mapr-1703</version> </dependency>`
org.apache.tez	tez-ui	0.8.4-mapr-1703 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui</artifactId> <version>0.8.4-mapr-1703</version> </dependency>`
org.apache.tez	tez-ui2	0.8.4-mapr-1703 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-ui2</artifactId> <version>0.8.4-mapr-1703</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history	0.8.4-mapr-1703 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history</artifactId> <version>0.8.4-mapr-1703</version> </dependency>`
org.apache.tez	tez-yarn-timeline-history-with-acls	0.8.4-mapr-1703 Browse	`<dependency> <groupId>org.apache.tez</groupId> <artifactId>tez-yarn-timeline-history-with-acls</artifactId> <version>0.8.4-mapr-1703</version> </dependency>`

Maven Artifacts for EEP 2.0.2
Listed are all Maven artifacts for EEP 2.0.2 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-hive	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-solr	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-solr</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-sqoop	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-core-common	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-db	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-indexer	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-sqoop	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-dist	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-dist</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-common	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-common</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-db	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-db</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-indexer	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-indexer</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-search	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-common	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-db	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-file	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-handlers	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

Maven Artifacts for EEP 2.0.1

Listed are all Maven artifacts for EEP 2.0.1 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.pig	pig	0.16.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pig</artifactId>`
			`<version>0.16.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.pig	piggybank	0.16.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>piggybank</artifactId>`
			`<version>0.16.0-mapr-1703</version>`
			`</dependency>`

org.apache.pig	pigsmoke	0.16.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigsmoke</artifactId>`
			`<version>0.16.0-mapr-1703</version>`
			`</dependency>`

org.apache.pig	pigunit	0.16.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigunit</artifactId>`
			`<version>0.16.0-mapr-1703</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-catalyst_2.11	2.0.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.0.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.0.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.0.1-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-graphx_2.11	2.0.1-mapr-1703 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-graphx_2.11</artifactId><version>2.0.1-mapr-1703</version></dependency>`
org.apache.spark	spark-hive-thriftserver_2.11	2.0.1-mapr-1703 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-hive-thriftserver_2.11</artifactId><version>2.0.1-mapr-1703</version></dependency>`
org.apache.spark	spark-hive_2.11	2.0.1-mapr-1703 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-hive_2.11</artifactId><version>2.0.1-mapr-1703</version></dependency>`
org.apache.spark	spark-launcher_2.11	2.0.1-mapr-1703 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-launcher_2.11</artifactId><version>2.0.1-mapr-1703</version></dependency>`
org.apache.spark	spark-mllib-local_2.11	2.0.1-mapr-1703 Browse	`<dependency><groupId>org.apache.spark</groupId><artifactId>spark-mllib-local_2.11</artifactId><version>2.0.1-mapr-1703</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-mllib_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-network-common_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-common_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-network-shuffle_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-network-yarn_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-repl_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-sketch_2.11	2.0.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sketch_2.11</artifactId>`
			`<version>2.0.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-sql-kafka-0-10_2.11	2.0.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql-kafka-0-10_2.11</artifactId>`
			`<version>2.0.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-sql_2.11	2.0.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.11</artifactId>`
			`<version>2.0.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-assembly_2.11	2.0.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.11</artifactId>`
			`<version>2.0.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-sink_2.11	2.0.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.11</artifactId>`
			`<version>2.0.1-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-flume_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-streaming_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-tags_2.11	2.0.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-tags_2.11</artifactId> <version>2.0.1-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-unsafe_2.11	2.0.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.0.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.0.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.0.1-mapr-1703</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.6-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.6-mapr-1703</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.6-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.6-mapr-1703</version>`
			`</dependency>`

Maven Artifacts for EEP 2.0.0

Listed are all Maven artifacts for EEP 2.0.0 components.

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-legacy-sources	flume-avro-source	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-legacy-sources</groupId> <artifactId>flume-avro-source</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-dataset-sink	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-dataset-sink</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-channels	flume-file-channel	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-file-channel</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-hdfs-sink	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-hdfs-sink</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-hive-sink	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-hive-sink</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-sinks	flume-irc-sink	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-irc-sink</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-channels	flume-jdbc-channel	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-jdbc-channel</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-sources	flume-jms-source	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sources</groupId> <artifactId>flume-jms-source</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-channels	flume-kafka-channel	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-kafka-channel</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-channels	flume-kafka-channel-v08	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-channels</groupId> <artifactId>flume-kafka-channel-v08</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-kafka-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-kafka-source-v08	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source-v08</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-auth	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-auth</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-configuration	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-configuration</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-core	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-core</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-sinks	flume-ng-elasticsearch-sink	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-elasticsearch-sink</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume	flume-ng-embedded-agent	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-ng-embedded-agent</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-hbase-sink	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-hbase-sink</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-kafka-sink</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink-v08	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-kafka-sink-v08</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-clients	flume-ng-log4jappender	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-clients</groupId>`
			`<artifactId>flume-ng-log4jappender</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-morphline-solr-sink	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-morphline-solr-sink</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-node	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-node</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-sdk	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-sdk</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-tests	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-tests</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-scribe-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-scribe-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-spillable-memory-channel	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-spillable-memory-channel</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-legacy-sources	flume-thrift-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-thrift-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-tools	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-tools</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-twitter-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-twitter-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-accumulo-handler</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive	hive-ant	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-ant</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive	hive-beeline	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-beeline</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive	hive-cli	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-cli</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive	hive-common	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive	hive-contrib	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-exec	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive	hive-hbase-handler	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-pig-adapter</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-server-extensions	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-server-extensions</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-streaming	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive	hive-hwi	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hwi</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive	hive-metastore	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-metastore</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive	hive-serde	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-serde</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive	hive-service	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-shims	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-shims</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-0.20S	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.20S</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-0.23	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-0.23</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-common	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-testutils	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive	spark-client	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.pig	pig	0.16.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pig</artifactId>`
			`<version>0.16.0-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.pig	piggybank	0.16.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>piggybank</artifactId> <version>0.16.0-mapr-1611</version> </dependency>`
org.apache.pig	pigsmoke	0.16.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>pigsmoke</artifactId> <version>0.16.0-mapr-1611</version> </dependency>`
org.apache.pig	pigunit	0.16.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>pigunit</artifactId> <version>0.16.0-mapr-1611</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sentry	sentry-binding-hive	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-binding-hive</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.sentry	sentry-binding-solr	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-binding-solr</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-sqoop	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-core-common	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-db	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-sqoop	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-dist	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-dist</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-common	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-common</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-db	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-db</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-indexer	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-indexer</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-search	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-db	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-file	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-handlers	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-hive	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-solr	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-solr</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-sqoop	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-common	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-db	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-sqoop	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sentry	sentry-dist	1.6.0-mapr-1606 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-dist</artifactId> <version>1.6.0-mapr-1606</version> </dependency>`
org.apache.sentry	sentry-policy-common	1.6.0-mapr-1606 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-policy-common</artifactId> <version>1.6.0-mapr-1606</version> </dependency>`
org.apache.sentry	sentry-policy-db	1.6.0-mapr-1606 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-policy-db</artifactId> <version>1.6.0-mapr-1606</version> </dependency>`
org.apache.sentry	sentry-policy-indexer	1.6.0-mapr-1606 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-policy-indexer</artifactId> <version>1.6.0-mapr-1606</version> </dependency>`
org.apache.sentry	sentry-policy-search	1.6.0-mapr-1606 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-policy-search</artifactId> <version>1.6.0-mapr-1606</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-sqoop	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-db	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-file	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	solr-sentry-handlers	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-catalyst_2.11	2.0.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.11</artifactId>`
			`<version>2.0.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-core_2.11	2.0.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.11</artifactId>`
			`<version>2.0.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-graphx_2.11	2.0.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.11</artifactId>`
			`<version>2.0.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.11	2.0.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.11</artifactId>`
			`<version>2.0.1-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-hive_2.11	2.0.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.11</artifactId>`
			`<version>2.0.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.11	2.0.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.11</artifactId>`
			`<version>2.0.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-mllib-local_2.11	2.0.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib-local_2.11</artifactId>`
			`<version>2.0.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-mllib_2.11	2.0.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.11</artifactId>`
			`<version>2.0.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.11	2.0.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.11</artifactId>`
			`<version>2.0.1-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-network-shuffle_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-network-yarn_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-repl_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-sketch_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sketch_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-sql_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-flume-assembly_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-streaming-flume_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10-assembly_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10-assembly_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-10_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-10_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-kafka-0-8-assembly_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8-assembly_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-8_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-8_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9-assembly_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9-assembly_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-streaming-kafka-0-9_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-0-9_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-streaming-kafka-producer_2.11	2.0.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-producer_2.11</artifactId> <version>2.0.1-mapr-1611</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming_2.11	2.0.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.11</artifactId>`
			`<version>2.0.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-tags_2.11	2.0.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-tags_2.11</artifactId>`
			`<version>2.0.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.11	2.0.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.11</artifactId>`
			`<version>2.0.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.11	2.0.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.11</artifactId>`
			`<version>2.0.1-mapr-1611</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.6-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.6-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop-test	1.4.6-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.6-mapr-1611</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	connector-sdk	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>connector-sdk</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop	connector-sdk-hadoop	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>connector-sdk-hadoop</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop	sqoop-assemblies	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-assemblies</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop	sqoop-client	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-client</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop-common	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-common</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop	sqoop-common-test	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-common-test</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-ftp	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-ftp</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-generic-jdbc	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-generic-jdbc</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-hdfs	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-hdfs</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sqoop.connector	sqoop-connector-kafka	1.99.7-mapr-1611 Browse	`<dependency> <groupId>org.apache.sqoop.connector</groupId> <artifactId>sqoop-connector-kafka</artifactId> <version>1.99.7-mapr-1611</version> </dependency>`
org.apache.sqoop.connector	sqoop-connector-kite	1.99.7-mapr-1611 Browse	`<dependency> <groupId>org.apache.sqoop.connector</groupId> <artifactId>sqoop-connector-kite</artifactId> <version>1.99.7-mapr-1611</version> </dependency>`
org.apache.sqoop.connector	sqoop-connector-oracle-jdbc	1.99.7-mapr-1611 Browse	`<dependency> <groupId>org.apache.sqoop.connector</groupId> <artifactId>sqoop-connector-oracle-jdbc</artifactId> <version>1.99.7-mapr-1611</version> </dependency>`
org.apache.sqoop.connector	sqoop-connector-sftp	1.99.7-mapr-1611 Browse	`<dependency> <groupId>org.apache.sqoop.connector</groupId> <artifactId>sqoop-connector-sftp</artifactId> <version>1.99.7-mapr-1611</version> </dependency>`
org.apache.sqoop	sqoop-core	1.99.7-mapr-1611 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-core</artifactId> <version>1.99.7-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop-docs	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-docs</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop.execution	sqoop-execution-mapreduce	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.execution</groupId>`
			`<artifactId>sqoop-execution-mapreduce</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop.repository	sqoop-repository-common	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-common</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop.repository	sqoop-repository-derby	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-derby</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop.repository	sqoop-repository-mysql	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-mysql</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop.repository	sqoop-repository-postgresql	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-postgresql</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop	sqoop-security	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-security</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop	sqoop-server	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-server</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop	sqoop-shell	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-shell</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop.submission	sqoop-submission-mapreduce	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.submission</groupId>`
			`<artifactId>sqoop-submission-mapreduce</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop-tools	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-tools</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

org.apache.sqoop	test	1.99.7-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>test</artifactId>`
			`<version>1.99.7-mapr-1611</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.storm	flux-core	0.10.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>flux-core</artifactId>`
			`<version>0.10.0-mapr-1611</version>`
			`</dependency>`

org.apache.storm	flux-examples	0.10.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>flux-examples</artifactId>`
			`<version>0.10.0-mapr-1611</version>`
			`</dependency>`

org.apache.storm	flux-wrappers	0.10.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>flux-wrappers</artifactId>`
			`<version>0.10.0-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.storm	maven-shade-clojure-transformer	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>maven-shade-clojure-transformer</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	multilang-javascript	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>multilang-javascript</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	multilang-python	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>multilang-python</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	multilang-ruby	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>multilang-ruby</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	storm-core	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-core</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.storm	storm-eventhubs	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-eventhubs</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	storm-hbase	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-hbase</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	storm-hdfs	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-hdfs</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	storm-hive	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-hive</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	storm-jdbc	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-jdbc</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	storm-kafka	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-kafka</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.storm	storm-maven-plugins	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-maven-plugins</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	storm-redis	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-redis</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	storm-starter	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-starter</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`

Maven Artifacts for EEP 1.1.3

Listed are all Maven artifacts for EEP 1.1.3 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sentry	sentry-binding-hive	1.6.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-binding-hive</artifactId> <version>1.6.0-mapr-1707</version> </dependency>`
org.apache.sentry	sentry-binding-solr	1.6.0-mapr-1707 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-binding-solr</artifactId> <version>1.6.0-mapr-1707</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-sqoop	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-core-common	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-db	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-sqoop	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-dist	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-dist</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-common	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-common</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-db	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-db</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-indexer	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-indexer</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-search	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-db	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-file	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-handlers	1.6.0-mapr-1707
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.6.0-mapr-1707</version>`
			`</dependency>`

Maven Artifacts for EEP 1.1.2

Listed are all Maven artifacts for EEP 1.1.2 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-core	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie.test	oozie-mini	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-distcp	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-oozie	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-pig	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-pig2	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-pig2</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-spark	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-spark</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-sqoop	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-sqoop</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-streaming	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.2.0-mapr-1703</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-1-4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-1-4.2.0-mapr-1703</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-distcp	hadoop-1-4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-1-4.2.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-utils	hadoop-1-4.2.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-utils</artifactId>`
			`<version>hadoop-1-4.2.0-mapr-1703</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.pig	pig	0.15.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pig</artifactId>`
			`<version>0.15.0-mapr-1703</version>`
			`</dependency>`

org.apache.pig	piggybank	0.15.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>piggybank</artifactId>`
			`<version>0.15.0-mapr-1703</version>`
			`</dependency>`

org.apache.pig	pigsmoke	0.15.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigsmoke</artifactId>`
			`<version>0.15.0-mapr-1703</version>`
			`</dependency>`

org.apache.pig	pigunit	0.15.0-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigunit</artifactId>`
			`<version>0.15.0-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-bagel_2.10	1.6.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-bagel_2.10</artifactId> <version>1.6.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-catalyst_2.10	1.6.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-catalyst_2.10</artifactId> <version>1.6.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-core_2.10	1.6.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.10</artifactId> <version>1.6.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-docker-integration-tests_2.10	1.6.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-docker-integration-tests_2.10</artifactId> <version>1.6.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-graphx_2.10	1.6.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.10</artifactId> <version>1.6.1-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-hive-thriftserver_2.10	1.6.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.10</artifactId> <version>1.6.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-hive_2.10	1.6.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.10</artifactId> <version>1.6.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-launcher_2.10	1.6.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-launcher_2.10</artifactId> <version>1.6.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-mllib_2.10	1.6.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib_2.10</artifactId> <version>1.6.1-mapr-1703</version> </dependency>`
org.apache.spark	spark-network-common_2.10	1.6.1-mapr-1703 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-common_2.10</artifactId> <version>1.6.1-mapr-1703</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-network-shuffle_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-sql_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-sql_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume-assembly_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-assembly_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-flume-sink_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume-sink_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-flume_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-flume_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-assembly_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-assembly_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-producer_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-v09_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-v09_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-mqtt-assembly_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-mqtt-assembly_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-mqtt_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-mqtt_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-twitter_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-twitter_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-streaming-zeromq_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-zeromq_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-test-tags_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-test-tags_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

org.apache.spark	spark-yarn_2.10	1.6.1-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.10</artifactId>`
			`<version>1.6.1-mapr-1703</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.6-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.6-mapr-1703</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop-test	1.4.6-mapr-1703
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.6-mapr-1703</version>`
			`</dependency>`

Maven Artifacts for EEP 1.1.0

Listed are all Maven artifacts for EEP 1.1.0 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-legacy-sources	flume-avro-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-avro-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-dataset-sink	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-dataset-sink</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-file-channel	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-file-channel</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-hdfs-sink	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-hdfs-sink</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-hive-sink	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-hive-sink</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-irc-sink	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-irc-sink</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-jdbc-channel	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-jdbc-channel</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-jms-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-jms-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-kafka-channel	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-kafka-channel</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-channels	flume-kafka-channel-v08	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-kafka-channel-v08</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-kafka-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-kafka-source-v08	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source-v08</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-auth	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-auth</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-configuration	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-configuration</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume	flume-ng-core	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-ng-core</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-elasticsearch-sink	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-elasticsearch-sink</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume	flume-ng-embedded-agent	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-ng-embedded-agent</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-hbase-sink	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-hbase-sink</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-kafka-sink</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink-v08	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-kafka-sink-v08</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-clients	flume-ng-log4jappender	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-clients</groupId>`
			`<artifactId>flume-ng-log4jappender</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-morphline-solr-sink	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-morphline-solr-sink</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-node	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-node</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-sdk	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-sdk</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume	flume-ng-tests	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-tests</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-scribe-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-scribe-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-spillable-memory-channel	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-spillable-memory-channel</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-legacy-sources	flume-thrift-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-thrift-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-tools	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-tools</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-twitter-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-twitter-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-accumulo-handler	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-accumulo-handler</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive	hive-ant	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-ant</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive	hive-beeline	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-beeline</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive	hive-cli	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-cli</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-common	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-common</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive	hive-contrib	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-contrib</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive	hive-exec	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive	hive-hbase-handler	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-pig-adapter</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-server-extensions	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-server-extensions</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-hcatalog-streaming	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive	hive-hwi	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hwi</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-metastore	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-metastore</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive	hive-serde	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-serde</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive	hive-service	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-service</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive	hive-shims	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive.shims	hive-shims-0.20S	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.20S</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.shims	hive-shims-common	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-common</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive.shims	hive-shims-scheduler	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.shims</groupId>`
			`<artifactId>hive-shims-scheduler</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive	hive-testutils	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	1.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>1.2.0-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	spark-client	1.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>spark-client</artifactId> <version>1.2.0-mapr-1611</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.mahout	apache-mahout-distribution	0.12.0-mapr-1609 Browse	`<dependency> <groupId>org.apache.mahout</groupId> <artifactId>apache-mahout-distribution</artifactId> <version>0.12.0-mapr-1609</version> </dependency>`
org.apache.mahout	mahout-buildtools	0.12.0-mapr-1609 Browse	`<dependency> <groupId>org.apache.mahout</groupId> <artifactId>mahout-buildtools</artifactId> <version>0.12.0-mapr-1609</version> </dependency>`
org.apache.mahout	mahout-examples	0.12.0-mapr-1609 Browse	`<dependency> <groupId>org.apache.mahout</groupId> <artifactId>mahout-examples</artifactId> <version>0.12.0-mapr-1609</version> </dependency>`
org.apache.mahout	mahout-flink_2.10	0.12.0-mapr-1609 Browse	`<dependency> <groupId>org.apache.mahout</groupId> <artifactId>mahout-flink_2.10</artifactId> <version>0.12.0-mapr-1609</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.mahout	mahout-h2o_2.10	0.12.0-mapr-1609 Browse	`<dependency> <groupId>org.apache.mahout</groupId> <artifactId>mahout-h2o_2.10</artifactId> <version>0.12.0-mapr-1609</version> </dependency>`
org.apache.mahout	mahout-hdfs	0.12.0-mapr-1609 Browse	`<dependency> <groupId>org.apache.mahout</groupId> <artifactId>mahout-hdfs</artifactId> <version>0.12.0-mapr-1609</version> </dependency>`
org.apache.mahout	mahout-integration	0.12.0-mapr-1609 Browse	`<dependency> <groupId>org.apache.mahout</groupId> <artifactId>mahout-integration</artifactId> <version>0.12.0-mapr-1609</version> </dependency>`
org.apache.mahout	mahout-math	0.12.0-mapr-1609 Browse	`<dependency> <groupId>org.apache.mahout</groupId> <artifactId>mahout-math</artifactId> <version>0.12.0-mapr-1609</version> </dependency>`
org.apache.mahout	mahout-math-scala_2.10	0.12.0-mapr-1609 Browse	`<dependency> <groupId>org.apache.mahout</groupId> <artifactId>mahout-math-scala_2.10</artifactId> <version>0.12.0-mapr-1609</version> </dependency>`
org.apache.mahout	mahout-mr	0.12.0-mapr-1609 Browse	`<dependency> <groupId>org.apache.mahout</groupId> <artifactId>mahout-mr</artifactId> <version>0.12.0-mapr-1609</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.mahout	mahout-spark-shell_2.10	0.12.0-mapr-1609
		Browse
			`<dependency>`
			`<groupId>org.apache.mahout</groupId>`
			`<artifactId>mahout-spark-shell_2.10</artifactId>`
			`<version>0.12.0-mapr-1609</version>`
			`</dependency>`

org.apache.mahout	mahout-spark_2.10	0.12.0-mapr-1609
		Browse
			`<dependency>`
			`<groupId>org.apache.mahout</groupId>`
			`<artifactId>mahout-spark_2.10</artifactId>`
			`<version>0.12.0-mapr-1609</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.oozie	oozie-core	4.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	4.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.2.0-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie.test	oozie-mini	4.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>4.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-distcp	4.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>4.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	4.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>4.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	4.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>4.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	4.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>4.2.0-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-oozie	4.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-oozie</artifactId> <version>4.2.0-mapr-1611</version> </dependency>`
org.apache.oozie	oozie-sharelib-pig	4.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>4.2.0-mapr-1611</version> </dependency>`
org.apache.oozie	oozie-sharelib-pig2	4.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig2</artifactId> <version>4.2.0-mapr-1611</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	4.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>4.2.0-mapr-1611</version> </dependency>`
org.apache.oozie	oozie-sharelib-sqoop	4.2.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-sqoop</artifactId> <version>4.2.0-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-streaming	4.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-streaming</artifactId>`
			`<version>4.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.oozie	oozie-tools	4.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-tools</artifactId>`
			`<version>4.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.oozie	oozie-webapp	4.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-webapp</artifactId>`
			`<version>4.2.0-mapr-1611</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-auth	hadoop-1-4.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-auth</artifactId>`
			`<version>hadoop-1-4.2.0-mapr-1611</version>`
			`</dependency>`

org.apache.oozie	oozie-hadoop-distcp	hadoop-1-4.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-distcp</artifactId>`
			`<version>hadoop-1-4.2.0-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-hadoop-utils	hadoop-1-4.2.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-hadoop-utils</artifactId>`
			`<version>hadoop-1-4.2.0-mapr-1611</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.pig	pig	0.15.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pig</artifactId>`
			`<version>0.15.0-mapr-1611</version>`
			`</dependency>`

org.apache.pig	piggybank	0.15.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>piggybank</artifactId>`
			`<version>0.15.0-mapr-1611</version>`
			`</dependency>`

org.apache.pig	pigsmoke	0.15.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigsmoke</artifactId>`
			`<version>0.15.0-mapr-1611</version>`
			`</dependency>`

org.apache.pig	pigunit	0.15.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.pig</groupId>`
			`<artifactId>pigunit</artifactId>`
			`<version>0.15.0-mapr-1611</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sentry	sentry-binding-hive	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-binding-hive</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.sentry	sentry-binding-solr	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-binding-solr</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.sentry	sentry-binding-sqoop	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-binding-sqoop</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.sentry	sentry-core-common	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-core-common</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.sentry	sentry-core-model-db	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-core-model-db</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-indexer	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-sqoop	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-dist	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-dist</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-common	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-common</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-db	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-db</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-indexer	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-indexer</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-search	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-common	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-db	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-file	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-handlers	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-hive	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-solr	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-solr</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-sqoop	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-core-common	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-db	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-search	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-sqoop	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-dist	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-dist</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-common	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-common</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-db	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-db</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-indexer	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-indexer</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-search	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-db	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-file	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-handlers	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-bagel_2.10	1.6.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-bagel_2.10</artifactId>`
			`<version>1.6.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.10	1.6.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.10</artifactId>`
			`<version>1.6.1-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-core_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-core_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-docker-integration-tests_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-docker-integration-tests_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-graphx_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-graphx_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-hive-thriftserver_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive-thriftserver_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-hive_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-hive_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-launcher_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-launcher_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-mllib_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-mllib_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-network-common_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-common_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-network-shuffle_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-shuffle_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-network-yarn_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-network-yarn_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-repl_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-repl_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-sql_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-streaming-flume_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-assembly_2.10	1.6.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-assembly_2.10</artifactId>`
			`<version>1.6.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-producer_2.10	1.6.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.10</artifactId>`
			`<version>1.6.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-v09_2.10	1.6.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-v09_2.10</artifactId>`
			`<version>1.6.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka_2.10	1.6.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka_2.10</artifactId>`
			`<version>1.6.1-mapr-1611</version>`
			`</dependency>`

org.apache.spark	spark-streaming-mqtt-assembly_2.10	1.6.1-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-mqtt-assembly_2.10</artifactId>`
			`<version>1.6.1-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-streaming-mqtt_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-mqtt_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-streaming-twitter_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-twitter_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-streaming-zeromq_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-zeromq_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-streaming_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-test-tags_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-test-tags_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-unsafe_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-unsafe_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`
org.apache.spark	spark-yarn_2.10	1.6.1-mapr-1611 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-yarn_2.10</artifactId> <version>1.6.1-mapr-1611</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sqoop	sqoop	1.4.6-mapr-1611 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop</artifactId> <version>1.4.6-mapr-1611</version> </dependency>`
org.apache.sqoop	sqoop-test	1.4.6-mapr-1611 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>sqoop-test</artifactId> <version>1.4.6-mapr-1611</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sqoop	connector-sdk	1.99.6-mapr-1607 Browse	`<dependency> <groupId>org.apache.sqoop</groupId> <artifactId>connector-sdk</artifactId> <version>1.99.6-mapr-1607</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop-client	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-client</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-common	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-common</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-common-test	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-common-test</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-generic-jdbc	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-generic-jdbc</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-hdfs	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-hdfs</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop.connector	sqoop-connector-kafka	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-kafka</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-kite	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-kite</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-core	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-core</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-docs	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-docs</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop.execution	sqoop-execution-mapreduce	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.execution</groupId>`
			`<artifactId>sqoop-execution-mapreduce</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop.repository	sqoop-repository-common	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-common</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop.repository	sqoop-repository-derby	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-derby</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop.repository	sqoop-repository-postgresql	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-postgresql</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-security	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-security</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-server	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-server</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop-shell	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-shell</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop.submission	sqoop-submission-mapreduce	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.submission</groupId>`
			`<artifactId>sqoop-submission-mapreduce</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-tomcat	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-tomcat</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-tools	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-tools</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	test	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>test</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.storm	flux-core	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>flux-core</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	flux-examples	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>flux-examples</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	flux-wrappers	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>flux-wrappers</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	maven-shade-clojure-transformer	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>maven-shade-clojure-transformer</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	multilang-javascript	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>multilang-javascript</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.storm	multilang-python	0.10.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>multilang-python</artifactId>`
			`<version>0.10.0-mapr-1611</version>`
			`</dependency>`

org.apache.storm	multilang-ruby	0.10.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>multilang-ruby</artifactId>`
			`<version>0.10.0-mapr-1611</version>`
			`</dependency>`

org.apache.storm	storm-core	0.10.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>storm-core</artifactId>`
			`<version>0.10.0-mapr-1611</version>`
			`</dependency>`

org.apache.storm	storm-eventhubs	0.10.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>storm-eventhubs</artifactId>`
			`<version>0.10.0-mapr-1611</version>`
			`</dependency>`

org.apache.storm	storm-hbase	0.10.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>storm-hbase</artifactId>`
			`<version>0.10.0-mapr-1611</version>`
			`</dependency>`

org.apache.storm	storm-hdfs	0.10.0-mapr-1611
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>storm-hdfs</artifactId>`
			`<version>0.10.0-mapr-1611</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.storm	storm-hive	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-hive</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	storm-jdbc	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-jdbc</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	storm-kafka	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-kafka</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	storm-maven-plugins	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-maven-plugins</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	storm-redis	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-redis</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`
org.apache.storm	storm-starter	0.10.0-mapr-1611 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-starter</artifactId> <version>0.10.0-mapr-1611</version> </dependency>`

last-updated: Apr 11, 2022

Maven Artifacts for EEP 1.0.0

Listed are all Maven artifacts for EEP 1.0.0 components.

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-legacy-sources	flume-avro-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-avro-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-dataset-sink	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-dataset-sink</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-file-channel	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-file-channel</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-hdfs-sink	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-hdfs-sink</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-hive-sink	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-hive-sink</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sinks	flume-irc-sink	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-irc-sink</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-jdbc-channel	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-jdbc-channel</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-jms-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-jms-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-kafka-channel	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-kafka-channel</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-kafka-channel-v08	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-kafka-channel-v08</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-kafka-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-kafka-source-v08	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-kafka-source-v08</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-auth	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-auth</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-configuration	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-configuration</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-core	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-core</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.flume.flume-ng-sinks	flume-ng-elasticsearch-sink	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-elasticsearch-sink</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume	flume-ng-embedded-agent	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume</groupId> <artifactId>flume-ng-embedded-agent</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-hbase-sink	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-hbase-sink</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-kafka-sink</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`
org.apache.flume.flume-ng-sinks	flume-ng-kafka-sink-v08	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.flume.flume-ng-sinks</groupId> <artifactId>flume-ng-kafka-sink-v08</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-clients	flume-ng-log4jappender	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-clients</groupId>`
			`<artifactId>flume-ng-log4jappender</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sinks	flume-ng-morphline-solr-sink	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sinks</groupId>`
			`<artifactId>flume-ng-morphline-solr-sink</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-node	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-node</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-sdk	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-sdk</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-ng-tests	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-ng-tests</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.flume.flume-ng-sources	flume-scribe-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-scribe-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-channels	flume-spillable-memory-channel	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-channels</groupId>`
			`<artifactId>flume-spillable-memory-channel</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-legacy-sources	flume-thrift-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-legacy-sources</groupId>`
			`<artifactId>flume-thrift-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume	flume-tools	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume</groupId>`
			`<artifactId>flume-tools</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.flume.flume-ng-sources	flume-twitter-source	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.flume.flume-ng-sources</groupId>`
			`<artifactId>flume-twitter-source</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-accumulo-handler	1.2.0-mapr-1607 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-accumulo-handler</artifactId><version>1.2.0-mapr-1607</version></dependency>`
org.apache.hive	hive-ant	1.2.0-mapr-1607 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-ant</artifactId><version>1.2.0-mapr-1607</version></dependency>`
org.apache.hive	hive-beeline	1.2.0-mapr-1607 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-beeline</artifactId><version>1.2.0-mapr-1607</version></dependency>`
org.apache.hive	hive-cli	1.2.0-mapr-1607 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-cli</artifactId><version>1.2.0-mapr-1607</version></dependency>`
org.apache.hive	hive-common	1.2.0-mapr-1607 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-common</artifactId><version>1.2.0-mapr-1607</version></dependency>`
org.apache.hive	hive-contrib	1.2.0-mapr-1607 Browse	`<dependency><groupId>org.apache.hive</groupId><artifactId>hive-contrib</artifactId><version>1.2.0-mapr-1607</version></dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-exec	1.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-exec</artifactId> <version>1.2.0-mapr-1607</version> </dependency>`
org.apache.hive	hive-hbase-handler	1.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-hbase-handler</artifactId> <version>1.2.0-mapr-1607</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-core	1.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-core</artifactId> <version>1.2.0-mapr-1607</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-pig-adapter	1.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-pig-adapter</artifactId> <version>1.2.0-mapr-1607</version> </dependency>`
org.apache.hive.hcatalog	hive-hcatalog-server-extensions	1.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.hive.hcatalog</groupId> <artifactId>hive-hcatalog-server-extensions</artifactId> <version>1.2.0-mapr-1607</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive.hcatalog	hive-hcatalog-streaming	1.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-hcatalog-streaming</artifactId>`
			`<version>1.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.hive	hive-hwi	1.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-hwi</artifactId>`
			`<version>1.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.hive	hive-jdbc	1.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-jdbc</artifactId>`
			`<version>1.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.hive	hive-metastore	1.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-metastore</artifactId>`
			`<version>1.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.hive	hive-serde	1.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-serde</artifactId>`
			`<version>1.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.hive	hive-service	1.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-service</artifactId>`
			`<version>1.2.0-mapr-1607</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.hive	hive-shims	1.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.hive</groupId> <artifactId>hive-shims</artifactId> <version>1.2.0-mapr-1607</version> </dependency>`
org.apache.hive.shims	hive-shims-0.20S	1.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.20S</artifactId> <version>1.2.0-mapr-1607</version> </dependency>`
org.apache.hive.shims	hive-shims-0.23	1.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-0.23</artifactId> <version>1.2.0-mapr-1607</version> </dependency>`
org.apache.hive.shims	hive-shims-common	1.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-common</artifactId> <version>1.2.0-mapr-1607</version> </dependency>`
org.apache.hive.shims	hive-shims-scheduler	1.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.hive.shims</groupId> <artifactId>hive-shims-scheduler</artifactId> <version>1.2.0-mapr-1607</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.hive	hive-testutils	1.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>hive-testutils</artifactId>`
			`<version>1.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat	1.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat</artifactId>`
			`<version>1.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.hive.hcatalog	hive-webhcat-java-client	1.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.hive.hcatalog</groupId>`
			`<artifactId>hive-webhcat-java-client</artifactId>`
			`<version>1.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.hive	spark-client	1.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.hive</groupId>`
			`<artifactId>spark-client</artifactId>`
			`<version>1.2.0-mapr-1607</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.mahout	apache-mahout-distribution	0.12.0-mapr-1605
		Browse
			`<dependency>`
			`<groupId>org.apache.mahout</groupId>`
			`<artifactId>apache-mahout-distribution</artifactId>`
			`<version>0.12.0-mapr-1605</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.mahout	mahout-buildtools	0.12.0-mapr-1605 Browse	`<dependency><groupId>org.apache.mahout</groupId><artifactId>mahout-buildtools</artifactId><version>0.12.0-mapr-1605</version></dependency>`
org.apache.mahout	mahout-examples	0.12.0-mapr-1605 Browse	`<dependency><groupId>org.apache.mahout</groupId><artifactId>mahout-examples</artifactId><version>0.12.0-mapr-1605</version></dependency>`
org.apache.mahout	mahout-flink_2.10	0.12.0-mapr-1605 Browse	`<dependency><groupId>org.apache.mahout</groupId><artifactId>mahout-flink_2.10</artifactId><version>0.12.0-mapr-1605</version></dependency>`
org.apache.mahout	mahout-h2o_2.10	0.12.0-mapr-1605 Browse	`<dependency><groupId>org.apache.mahout</groupId><artifactId>mahout-h2o_2.10</artifactId><version>0.12.0-mapr-1605</version></dependency>`
org.apache.mahout	mahout-hdfs	0.12.0-mapr-1605 Browse	`<dependency><groupId>org.apache.mahout</groupId><artifactId>mahout-hdfs</artifactId><version>0.12.0-mapr-1605</version></dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.mahout	mahout-integration	0.12.0-mapr-1605
		Browse
			`<dependency>`
			`<groupId>org.apache.mahout</groupId>`
			`<artifactId>mahout-integration</artifactId>`
			`<version>0.12.0-mapr-1605</version>`
			`</dependency>`

org.apache.mahout	mahout-math	0.12.0-mapr-1605
		Browse
			`<dependency>`
			`<groupId>org.apache.mahout</groupId>`
			`<artifactId>mahout-math</artifactId>`
			`<version>0.12.0-mapr-1605</version>`
			`</dependency>`

org.apache.mahout	mahout-math-scala_2.10	0.12.0-mapr-1605
		Browse
			`<dependency>`
			`<groupId>org.apache.mahout</groupId>`
			`<artifactId>mahout-math-scala_2.10</artifactId>`
			`<version>0.12.0-mapr-1605</version>`
			`</dependency>`

org.apache.mahout	mahout-mr	0.12.0-mapr-1605
		Browse
			`<dependency>`
			`<groupId>org.apache.mahout</groupId>`
			`<artifactId>mahout-mr</artifactId>`
			`<version>0.12.0-mapr-1605</version>`
			`</dependency>`

org.apache.mahout	mahout-spark-shell_2.10	0.12.0-mapr-1605
		Browse
			`<dependency>`
			`<groupId>org.apache.mahout</groupId>`
			`<artifactId>mahout-spark-shell_2.10</artifactId>`
			`<version>0.12.0-mapr-1605</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.mahout	mahout-spark_2.10	0.12.0-mapr-1605
		Browse
			`<dependency>`
			`<groupId>org.apache.mahout</groupId>`
			`<artifactId>mahout-spark_2.10</artifactId>`
			`<version>0.12.0-mapr-1605</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-client	4.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-client</artifactId>`
			`<version>4.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.oozie	oozie-core	4.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-core</artifactId>`
			`<version>4.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.oozie	oozie-examples	4.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-examples</artifactId>`
			`<version>4.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.oozie.test	oozie-mini	4.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie.test</groupId>`
			`<artifactId>oozie-mini</artifactId>`
			`<version>4.2.0-mapr-1607</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.oozie	oozie-sharelib-distcp	4.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-distcp</artifactId>`
			`<version>4.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hcatalog	4.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hcatalog</artifactId>`
			`<version>4.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive	4.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive</artifactId>`
			`<version>4.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-hive2	4.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-hive2</artifactId>`
			`<version>4.2.0-mapr-1607</version>`
			`</dependency>`

org.apache.oozie	oozie-sharelib-oozie	4.2.0-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.oozie</groupId>`
			`<artifactId>oozie-sharelib-oozie</artifactId>`
			`<version>4.2.0-mapr-1607</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-sharelib-pig	4.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig</artifactId> <version>4.2.0-mapr-1607</version> </dependency>`
org.apache.oozie	oozie-sharelib-pig2	4.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-pig2</artifactId> <version>4.2.0-mapr-1607</version> </dependency>`
org.apache.oozie	oozie-sharelib-spark	4.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-spark</artifactId> <version>4.2.0-mapr-1607</version> </dependency>`
org.apache.oozie	oozie-sharelib-sqoop	4.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-sqoop</artifactId> <version>4.2.0-mapr-1607</version> </dependency>`
org.apache.oozie	oozie-sharelib-streaming	4.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-sharelib-streaming</artifactId> <version>4.2.0-mapr-1607</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-tools	4.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-tools</artifactId> <version>4.2.0-mapr-1607</version> </dependency>`
org.apache.oozie	oozie-webapp	4.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-webapp</artifactId> <version>4.2.0-mapr-1607</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.oozie	oozie-hadoop-auth	hadoop-1-4.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-auth</artifactId> <version>hadoop-1-4.2.0-mapr-1607</version> </dependency>`
org.apache.oozie	oozie-hadoop-distcp	hadoop-1-4.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-distcp</artifactId> <version>hadoop-1-4.2.0-mapr-1607</version> </dependency>`
org.apache.oozie	oozie-hadoop-utils	hadoop-1-4.2.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.oozie</groupId> <artifactId>oozie-hadoop-utils</artifactId> <version>hadoop-1-4.2.0-mapr-1607</version> </dependency>`

last-updated: Apr 11, 2022

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.pig	pig	0.15.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>pig</artifactId> <version>0.15.0-mapr-1607</version> </dependency>`
org.apache.pig	piggybank	0.15.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>piggybank</artifactId> <version>0.15.0-mapr-1607</version> </dependency>`
org.apache.pig	pigsmoke	0.15.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>pigsmoke</artifactId> <version>0.15.0-mapr-1607</version> </dependency>`
org.apache.pig	pigunit	0.15.0-mapr-1607 Browse	`<dependency> <groupId>org.apache.pig</groupId> <artifactId>pigunit</artifactId> <version>0.15.0-mapr-1607</version> </dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.sentry	sentry-binding-hive	1.6.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.sentry</groupId> <artifactId>sentry-binding-hive</artifactId> <version>1.6.0-mapr-1602</version> </dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-solr	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-solr</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-sqoop	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-core-common	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-db	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-search	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-sqoop	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-dist	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-dist</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-common	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-common</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-db	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-db</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-indexer	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-indexer</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-search	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-db	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-file	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-handlers	1.6.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.6.0-mapr-1602</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-hive	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-hive</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-binding-solr	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-solr</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-binding-sqoop	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-binding-sqoop</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-core-common	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-common</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-db	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-db</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-indexer	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-indexer</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-core-model-search	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-search</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-core-model-sqoop	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-core-model-sqoop</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-dist	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-dist</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-common	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-common</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-db	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-db</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-indexer	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-indexer</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-policy-search	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-search</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-policy-sqoop	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-policy-sqoop</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-cache	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-cache</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-common	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-common</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	sentry-provider-db	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-db</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sentry	sentry-provider-file	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>sentry-provider-file</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

org.apache.sentry	solr-sentry-handlers	1.6.0-mapr-1606
		Browse
			`<dependency>`
			`<groupId>org.apache.sentry</groupId>`
			`<artifactId>solr-sentry-handlers</artifactId>`
			`<version>1.6.0-mapr-1606</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-bagel_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-bagel_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-catalyst_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-catalyst_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-core_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-core_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-docker-integration-tests_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-docker-integration-tests_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-graphx_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-graphx_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-hive-thriftserver_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive-thriftserver_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-hive_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-hive_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-launcher_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-launcher_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-mllib_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-mllib_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-network-common_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-common_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-network-shuffle_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-shuffle_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-network-yarn_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-network-yarn_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-repl_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-repl_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.spark	spark-sql_2.10	1.6.1-mapr-1607 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-sql_2.10</artifactId> <version>1.6.1-mapr-1607</version> </dependency>`
org.apache.spark	spark-streaming-flume-assembly_2.10	1.6.1-mapr-1607 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-assembly_2.10</artifactId> <version>1.6.1-mapr-1607</version> </dependency>`
org.apache.spark	spark-streaming-flume-sink_2.10	1.6.1-mapr-1607 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume-sink_2.10</artifactId> <version>1.6.1-mapr-1607</version> </dependency>`
org.apache.spark	spark-streaming-flume_2.10	1.6.1-mapr-1607 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-flume_2.10</artifactId> <version>1.6.1-mapr-1607</version> </dependency>`
org.apache.spark	spark-streaming-kafka-assembly_2.10	1.6.1-mapr-1607 Browse	`<dependency> <groupId>org.apache.spark</groupId> <artifactId>spark-streaming-kafka-assembly_2.10</artifactId> <version>1.6.1-mapr-1607</version> </dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-kafka-producer_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-producer_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka-v09_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka-v09_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-streaming-kafka_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-kafka_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-streaming-mqtt-assembly_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-mqtt-assembly_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-streaming-mqtt_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-mqtt_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-streaming-twitter_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-twitter_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-streaming-zeromq_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming-zeromq_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-streaming_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-streaming_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-test-tags_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-test-tags_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

org.apache.spark	spark-unsafe_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-unsafe_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.spark	spark-yarn_2.10	1.6.1-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.spark</groupId>`
			`<artifactId>spark-yarn_2.10</artifactId>`
			`<version>1.6.1-mapr-1607</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop	1.4.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop</artifactId>`
			`<version>1.4.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-test	1.4.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-test</artifactId>`
			`<version>1.4.6-mapr-1607</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	connector-sdk	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>connector-sdk</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-client	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-client</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop	sqoop-common	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-common</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-common-test	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-common-test</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-generic-jdbc	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-generic-jdbc</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-hdfs	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-hdfs</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop.connector	sqoop-connector-kafka	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-kafka</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop.connector	sqoop-connector-kite	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.connector</groupId>`
			`<artifactId>sqoop-connector-kite</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-core	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-core</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-docs	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-docs</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop.execution	sqoop-execution-mapreduce	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.execution</groupId>`
			`<artifactId>sqoop-execution-mapreduce</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop.repository	sqoop-repository-common	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-common</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop.repository	sqoop-repository-derby	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-derby</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop.repository	sqoop-repository-postgresql	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.repository</groupId>`
			`<artifactId>sqoop-repository-postgresql</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-security	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-security</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-server	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-server</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-shell	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-shell</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.sqoop.submission	sqoop-submission-mapreduce	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop.submission</groupId>`
			`<artifactId>sqoop-submission-mapreduce</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-tomcat	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-tomcat</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	sqoop-tools	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>sqoop-tools</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

org.apache.sqoop	test	1.99.6-mapr-1607
		Browse
			`<dependency>`
			`<groupId>org.apache.sqoop</groupId>`
			`<artifactId>test</artifactId>`
			`<version>1.99.6-mapr-1607</version>`
			`</dependency>`

Table

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.storm	flux-core	0.10.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>flux-core</artifactId>`
			`<version>0.10.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.storm	flux-examples	0.10.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>flux-examples</artifactId>`
			`<version>0.10.0-mapr-1602</version>`
			`</dependency>`

org.apache.storm	flux-wrappers	0.10.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>flux-wrappers</artifactId>`
			`<version>0.10.0-mapr-1602</version>`
			`</dependency>`

org.apache.storm	maven-shade-clojure-transformer	0.10.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>maven-shade-clojure-transformer</artifactId>`
			`<version>0.10.0-mapr-1602</version>`
			`</dependency>`

org.apache.storm	multilang-javascript	0.10.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>multilang-javascript</artifactId>`
			`<version>0.10.0-mapr-1602</version>`
			`</dependency>`

org.apache.storm	multilang-python	0.10.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>multilang-python</artifactId>`
			`<version>0.10.0-mapr-1602</version>`
			`</dependency>`

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate

org.apache.storm	multilang-ruby	0.10.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>multilang-ruby</artifactId>`
			`<version>0.10.0-mapr-1602</version>`
			`</dependency>`

org.apache.storm	storm-core	0.10.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>storm-core</artifactId>`
			`<version>0.10.0-mapr-1602</version>`
			`</dependency>`

org.apache.storm	storm-eventhubs	0.10.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>storm-eventhubs</artifactId>`
			`<version>0.10.0-mapr-1602</version>`
			`</dependency>`

org.apache.storm	storm-hbase	0.10.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>storm-hbase</artifactId>`
			`<version>0.10.0-mapr-1602</version>`
			`</dependency>`

org.apache.storm	storm-hdfs	0.10.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>storm-hdfs</artifactId>`
			`<version>0.10.0-mapr-1602</version>`
			`</dependency>`

org.apache.storm	storm-hive	0.10.0-mapr-1602
		Browse
			`<dependency>`
			`<groupId>org.apache.storm</groupId>`
			`<artifactId>storm-hive</artifactId>`
			`<version>0.10.0-mapr-1602</version>`
			`</dependency>`

last-updated: Apr 11, 2022

Table (Continued)

Group Id	Artifact Id	Version	Maven Coordinate
org.apache.storm	storm-jdbc	0.10.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-jdbc</artifactId> <version>0.10.0-mapr-1602</version> </dependency>`
org.apache.storm	storm-kafka	0.10.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-kafka</artifactId> <version>0.10.0-mapr-1602</version> </dependency>`
org.apache.storm	storm-maven-plugins	0.10.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-maven-plugins</artifactId> <version>0.10.0-mapr-1602</version> </dependency>`
org.apache.storm	storm-redis	0.10.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-redis</artifactId> <version>0.10.0-mapr-1602</version> </dependency>`
org.apache.storm	storm-starter	0.10.0-mapr-1602 Browse	`<dependency> <groupId>org.apache.storm</groupId> <artifactId>storm-starter</artifactId> <version>0.10.0-mapr-1602</version> </dependency>`

Integrating the MapR GitHub and Maven Repositories

This topic provides instructions for cloning the GitHub and Maven repositories for a MapR open source project into your Eclipse IDE.

Integrating Git

1. Open the Git Repository perspective by selecting **Window**>**Open Perspective**>**Other...** then choosing **Git Repository Exploring**.

last-updated: Apr 11, 2022

2. From the Git Repository perspective, click the  button to display the **Clone Git Repository** dialog.

3. From a web browser, navigate to the MapR repository, then select the project you want to clone.

4. Copy the git URI from the project page to your clipboard by clicking the  button.

5. In the **Clone Git Repository** dialog, paste the git URI into the **URI:** field, then click **Next**. Eclipse will connect to github and download the repository metadata, then display a list of branches.

6. Select the branches you wish to clone, then click **Next**.

7. Configure the destination directory, then click **Finish**. Eclipse downloads the project from github and adds it to your view.

Integrating Maven

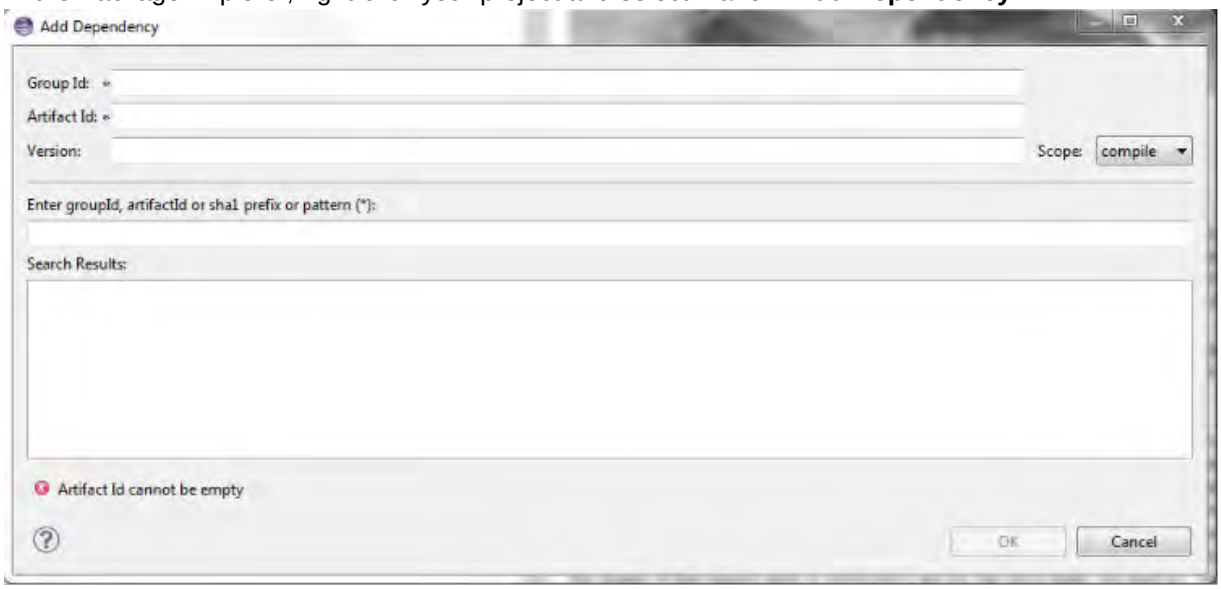
1. Start a new Maven project, or convert your current project into a Maven project if necessary.

last-updated: Apr 11, 2022

2. Select **Window**>**Show View**>**Package Explorer** to show your current Maven project.
3. Add the following lines to your project's `pom.xml` file:

```
<repositories>
              <repository>
              <id>mapr-releases</id>
              <url>https://repository.mapr.com/maven/</url>
              <snapshots><enabled>false</enabled></snapshots>
              <releases><enabled>true</enabled></releases>
              </repository>
              </repositories>
```

4. In a browser, navigate to the MapR Maven Repository and search for the Maven artifact your project depends on. You can also browse the repository.
5. In the Package Explorer, right-click your project and select **Maven**>**Add Dependency**.



6. Enter the `groupId`, `artifactId`, and `version` values for the dependency, then click **OK**.
7. Refresh the workspace by pressing F5. Your Maven dependencies download automatically.

Developer's Reference

This section contains in-depth information for the developer.

MapR Database Shell (JSON Tables)

The mapr dbshell is a tool that enables you to create and perform basic manipulation of JSON tables and documents. You run dbshell by typing `mapr dbshell` on the command line after logging into a node in a MapR Data Platform cluster.

Note: MapR Database Shell does not support MapR Event Store For Apache Kafka streams operations.

last-updated: Apr 11, 2022

Permissions

Before running dbshell, your user ID must have both the `readAce` and `writeAce` permissions on the volume. For information about these permissions, see Managing Whole Volume ACEs on page 1458.

SUSE Linux Error Messages

When you run dbshell on SUSE Linux, you might see the following messages:

```
[INFO] Unable to bind key for unsupported operation: backward-delete-word
[INFO] Unable to bind key for unsupported operation: up-history
[INFO] Unable to bind key for unsupported operation: down-history
```

To suppress these messages, edit the `/etc/inputrc` file and rename the keywords as follows:

Original name	New name
`backward-delete-word`	`backward-kill-word`
`down-history`	`next-history`
`up-history`	`previous-history`

Command Descriptions

To get a list of supported dbshell commands, run `help` at the shell prompt:

```
* ! - Allows execution of operating system (OS) commands
* // - Inline comment markers (start of line only)
* ; - Inline comment markers (start of line only)
* cat - Print the content of the specified file on the standard output
* cd - Change the current directory to the specified path.
* clear - Clears the console
* cls - Clears the console
* create - Create a json table at the given path.
* date - Displays the local date and time
* debug - Sets/shows the debug mode.
* delete - Delete a document from the table.
* desc - Describes the properties of a table.
* drop - Deletes a MapR-DB json table.
* exists - Returns true if the table exists.
* exit - Exits the shell
* find - Retrieves one or more documents from the table.
* findbyid - Retrieves a single document from the table.
* help - List all commands usage
* indexlist - Retrieves the list of indexes for the specified table.
* indexscan - Scan the index and return the document in their natural order.
* insert - Inserts or replaces a document into the table.
* jsonoptions - Sets/shows the Json output options.
* list - Lists all tables in a folder or matching the specified pattern.
* ls - Lists files and folders.
* mkdir - Create a directory at the specified path.
* pwd - Print the absolute path of the current working directory.
* quit - Exits the shell
* replace - Replace a document based on condition.
* script - Parses the specified resource file and executes its commands
* system properties - Shows the shell's properties
* tableoptions - Sets/shows the MapR-DB Table access options.
* update - Update field in a single document.
* version - Displays shell version
* whoami - Prints the current MapR-DB Shell user.
```


last-updated: Apr 11, 2022

Parameters

Various components of these commands will be either be in JSON (in case a list of key-value is required) or a single value following a switch identifying the component.

To get a list of parameters for a specific command, run `help <command>` at the prompt. For example: `help find` returns the following:

```
maprdb root:> help find
Command:                    find
Description:                Retrieves one or more documents from the table.
Options:
  *, --t, --table           Table path. [required]
  --id                      Document Id.
  --fromid                  Document Id to start from (inclusive)
  --toid                    Document Id to stop at (exclusive)
  --limit                   Maximum number of documents to return.
  --withtags, --withTags    Enables/disables printing with extended Type
Tags.
  --pretty                  Enables/disables pretty printing of the document.
  --offset                  Skip first n number of rows in the result.
  --orderby                 Sort result by the given fields.
  --c, --where              Condition in JSON format
  --f, --fields             Projections in JSON documents
  --q, --query              Query in JSON documents
Examples:
  find /tables/users
  find /tables/users --fromid user001 --toid user00a --limit 32
```

Value Types

Extended type values are shown using type tags. The scalar types are represented as follows:

- {"$binary":"AAAASw=="} for binary values
- {"$numberLong":21491} for whole numbers like long, short, byte, short
- {"$numeric":47.92} for numeric types like float and double
- {"$time":"14:35:28.981"} for time type values
- {"$date":"2017-04-24T22:35:28.981Z"} for timestamp values
- {"$dateDay":"2017-04-23"} for date type values

Bulk Operations

Currently, bulk conditional operations are not supported.

dbshell create

Describes how to use MapR Database shell to create a JSON table.

Description

To create JSON tables, run the `create <table path>` command.

When you create a JSON table using `mapr dbshell`, the default column family is created automatically. There are no commands for creating additional column families. To create column families, exit the shell and use the `maprcli table cf create` command.

last-updated: Apr 11, 2022

Note: If you are using a 5.2.x dbshell client to connect to a 6.0 (or later) server, you must set the `insertionorder` table option to false before creating your table. The `insertionorder` option is not support as of 6.0. See the following example:

```
tableoptions --insertionorder false
```

Run `tableoptions` in dbshell to see the current setting of `insertionorder`.

Parameters

create Options	Description
`--t`, `--table` `<table path>`	Table path Although the `--t` and `--table` qualifiers are optional, you must specify a `<table path>`.

Syntax

```
create <table path>
```

Example

In the following example, **data** is the volume name and **movies** is the new table name.

```
create /data/movies
```

dbshell delete

Description

The dbshell `delete` command deletes a single JSON document. To delete a document, specify the path of the table where the document is located, the ID of the document, and an optional condition. If the condition for the specified document evaluates to true, the document is deleted.

Parameters

delete Options	Description
`*`, `--t`, `--table` (Required)	Table path
`--id` (Required)	ID of the document to delete **Note:** You can specify this parameter only once.
`--c`, `--where`	OJAI condition, in JSON format The condition must qualify to perform the delete. See OJAI Query Condition Syntax on page 2606 for a description of the syntax.

Syntax

```
delete <table path> --id <row-key> --c <condition>
```


last-updated: Apr 11, 2022

Example: Delete a Document if a Condition is Met

The following example deletes the document with the `_id id1`, if the condition `(a.b[0].boolean == false && (a.c.d != 5 || a.b[1].decimal > 1))` is met:

```
delete /tbl --id id1
    --c {
        "$and":[
            {"$eq":{"a.b[0].boolean":false}},
            {"$or":[
                {"$ne":{"a.c.d":5}},
                {"$gt":{"a.b[1].decimal":1}}
            ]}
        ]}
```

Example: Delete a Document

The following example deletes a document with `_id movie0000002` from the `movies` table:

```
delete /data/movies --id movie0000002
```

dbshell drop

Description

To run the `drop` command, specify the path to the MapR Database JSON table.

Parameters

Table

drop Options	Description
*, `--t`, `--table` (Required)	Table path

Syntax

```
drop <table path>
```

Example

```
drop /data/movies
```

dbshell find or findbyid

To query JSON documents in MapR Database shell, use either the `find` or `findbyid` command. The `find` command enables you to scan complete tables and retrieve rows that satisfy projection and/or condition clauses. The `findbyid` command enables you to retrieve a single document with a given ID.

When you review the `find` examples, note that they are sometimes shown split across multiple lines for readability. You must enter the commands on a single line when you run them in dbshell.

Note: If your `find` query requires the OJAI Distributed Query Service on page 505, you must install the `mapr-drill-internal` package on the nodes where you run dbshell. The package is available in the MapR repository from which you download MapR Ecosystem Packs. See MapR Repositories and Packages on page 128 for details.

last-updated: Apr 11, 2022

Syntax

```
find <table path> <options>
```

```
findbyid <table path> --id <key-row ID>
```

Parameters

find Options	findbyid Options	Description
`*`, `--t`, `--table` (Required)	`*`, `--t`, `--table` (Required)	Table path
`--id`	`--id` (Required)	Document ID For conditions on a single document ID, you can provide the ID either by using the `--id` switch or by specifying the ID in a condition payload. For example, `--id id1` is equivalent to `--c {"$eq": {"_id":"id1"}}`. **Note:** You cannot specify multiple IDs using either syntax.
`--fromid`	n/a	Document ID to start from (inclusive)
`--toid`	n/a	Document ID to stop at (exclusive)
`--limit`	n/a	Maximum number of documents to return
`--withtags`, `--withTags`	`--withtags`, `--withTags`	Enables or disables printing with extended type tags Value: `True`|`False` Default: `True`
`--pretty`	`--pretty`	Enables or disables pretty printing of documents Value: `True`|`False` Default: `True`
`--offset`	n/a	Omits the first n number of documents in the result
`--orderby`	n/a	Sorts the result by the given fields Specify sort order as either ascending or descending using the keywords, `ASC` or `DESC`, respectively. Default: `ASC` **Note:** The keywords `ASC` and `DESC` are case insensitive. Syntax: `find <table path> --orderby <field path>:<sortorder>` See Query with --orderby on page 5296 for examples.

last-updated: Apr 11, 2022

find Options	findbyid Options	Description
`--c`, `--where`	`--c`, `--where`	Condition, in JSON format See OJAI Query Condition Syntax on page 2606 for a description of the syntax. See Return Documents Using Projection and Conditions on page 5298 for a dbshell example that uses a condition specified in JSON format.
`--f`, `--fields`	`--f`, `--fields`	Projections in JSON documents See JSON Document Field Paths on page 515 for details about how to specify field paths. See Return Documents Using Projection and Conditions on page 5298 for an example.

`--q`, `--query`	n/a	Query JSON documents
		This option accepts a query string in JSON format with predefined keywords that define the behavior of the query.
		The following examples shows a query that uses three keywords:
		`find table/test --q { "$select":"a", "$limit":2, "$offset":1 }`
		Note: The `find` command does not allow `--query` to work with other options, such as `--fields`, `--where`, and `--orderby`. For example, the following command ignores the `--f` option:
		`find table/test --f "a" --q {"$limit":2}`
		In addition, you should not enter the same keyword twice:
		`// Incorrect {"$select":"a","$select":"b"}`
		`// Correct {"$select":["a","b"]}`
		See Query with --query on page 5292 for more examples.

Query with --query

When querying JSON documents with the `find` command, you can use OJAI query syntax with the `--query` option. With this option, you can specify keywords that determine the documents and the fields from those documents that the command returns.

Syntax

```
find <table path> --query <keywords>
```

Note: The `find` command does not allow `--query` to work in tandem with other options such as `--fields`, `--where`, and `--orderby`.

For example, the following command does not return your desired results:

```
find /tbl --f a --q {"$limit":2}
```

In addition, repetition of keywords in the `--query` option is not supported. You should not enter the same keyword twice:

```
// Incorrect
{"$select":"a","$select":"b"}

// Correct
{"$select":["a","b"]}
```

Keywords for the `--query` Option

The `--query` option supports the following keywords:

--query Keywords	Equivalent `find` Option
$select	Equivalent to the `--f`, `--fields` option
$where	Equivalent to the `--c`, `--where` option
$limit	Equivalent to the `--limit` option
$offset	Equivalent to the `--offset` option
$orderby	Equivalent to the `--orderby` option
$options	No equivalent option

The following sections provide examples of each keyword. For more details, see OJAI Query Syntax on page 2603.

Sample JSON Document

The examples in this topic use the following sample JSON document:

```
{
    "_id": "id1",
    "a": {
            "b": [{"boolean":false}, {"decimal": 123.456}],
            "c":{
                  "d":10,
                  "e":"Hello"
                }
          },
    "m":"MapR wins"
}
```

$select Syntax and Example

The `$select` keyword defines the field path projections to be displayed in the result set.

The following syntax shows single and multiple field path projections:

```
// Single field path projection syntax
find <table path> --q {"$select":"<fieldpath>"}
```


last-updated: Apr 11, 2022

```
// Multiple field path projection syntax
find <table path> --q {"$select":
["<fieldpath1>","<fieldpath2>","<fieldpath3>"]}
```

The following examples show single and multiple field path projections:

```
// Single field path projection example
find /tbl --q {"$select":"a.c.d"}

// Multiple field path projection example
find /tbl --q {"$select":["a.c.d","a.c.e","m"]}
```

See OJAI Query Projection on page 2604 for more information about `$select`.

$where Syntax and Example

When using the `$where` keyword, define the condition using OJAI Query Condition Syntax on page 2606.

```
find <table path> --q {"$where":<condition>}
```

The following example performs a `find` operation with a projection and a condition:

```
find /tbl --q {"$select":"a.c.e",
            "$where":{
                    "$and":[
                            {"$eq":{"a.b[0].boolean":false}},
                            {"$or":[
                                    {"$ne":{"a.c.d":5}},
                                    {"$gt":{"a.b[1].decimal":1}},
                                    {"$lt":{"a.b[1].decimal":10}}
                                    ]
                             }
                            ]
                    }
              }
```

The projection is on field `a.c.e`. The condition is the following expression:

```
(a.b.[0].boolean == false && (a.c.d != 5 || a.b[1].decimal > 1 ||
a.b[1].decimal < 10))
```

$limit Syntax and Example

The `$limit` keyword sets the maximum number of documents to return. It only accepts positive integers. It throws an exception for negative or decimal values.

```
find <table path> --q {"$limit":<positive integer>}
```

The following example performs a `find` with a projection on the `a.c.e` and `m` fields and limits the result set to a max of 10 documents:

```
find /tbl --q {"$select":["a.c.e","m"],
              "$limit":10
             }
```

See OJAI Query Limit on page 2605 for more information about `$limit`.

$offset Syntax and Example

The `$offset` keyword skips the first n number of rows in the result. If n is greater than the total number of documents, no documents are returned. It only accepts positive integers.

```
find <table path> --q {"$offset":<positive integer>}
```

The following example performs a `find` operation with projection on the `a.c.e` and `m` fields and offsets the result set to skip first five documents:

```
find /tbl --q {"$select":["a.c.e","m"],
               "$offset":5
              }
```

See OJAI Query Offset on page 2605 for more information about `$offset`.

$orderby Syntax and Examples

The `$orderby` keyword sorts the result on the specified fields.

The following shows the syntax and example of sorting a single field in the default ascending order:

```
// Syntax for sorting a single field in the default ascending order
find <table path> --q {"$orderby":"<field path>"}

// Example sort on field path a.c.e in the default ascending order
find /tbl --q {"$orderby":"a.c.e"}
```

The following show the syntax and examples of sorting a single field in ascending or descending order where `<order>` is `ASC` for ascending and `DESC` for descending:

```
// Syntax for sorting a single field in ASC/DESC order
find <table path> --q {"$orderby":{"<field path>":"<order>"}}

// Example sort on field path a.c.e in ascending order
find /tbl --q {"$orderby":{"a.c.e":"asc"}}

// Example sort on field path a.c.e in descending order
find /tbl --q {"$orderby":{"a.c.e":"desc"}}
```

Note: The keywords `ASC` and `DESC` are case insensitive.

The following shows the syntax and an example of sorting multiple fields in ascending and descending order:

```
// Syntax for sorting multiple field paths in ascending/descending order
find <table path> --q {"$orderby":[{"<field path1>":"<order>"},
                                   {"<field path2>":"<order>"},
                                   {"<field path3>":"<order>"}
                                  ]
                      }

// Example sort on field path a.c.d (in the default ascending order)
// and field path a.c.e in descending order
find /tbl --q {"$orderby":["a.c.d",{"a.c.e":"desc"}]}
```

See OJAI Query Order By on page 2605 for more information about `$orderby`.

last-updated: Apr 11, 2022

$options Syntax and Example

The `$options` keyword enables you to influence a query's execution path. The general syntax is as follows:

```
find <table path> --q {"$options":{<option name>:<option value>}}
```

When specifying the `<option name>`, you must separate the components of the option name, replacing the dots with curly braces and colons and enclosing each component in quotes. The following example shows you how to do this for the `ojai.mapr.query.hint-using-index` option. The example forces the query to use a secondary index named `colIndex`:

```
find /apps/test --q {
    "$where":{"$eq":{"col":10}},
    "$options":{"ojai":{"mapr":{"query":{"hint-using-index":"colIndex"}}}}
    }
```

See OJAI Query Options on page 2588 for a complete list of available query options.

Query with --orderby

When querying JSON documents with the `find` command, you can use the `--orderby` option to order the data. You can specify either an ascending or descending sort using the keywords, `ASC` and `DESC`.

General Syntax

```
find <table path> --orderby <field path>:<sortorder>
```

The keywords `ASC` and `DESC` are case insensitive. Ascending is the default sort order.

Sample JSON Document

The following sample JSON document is used in examples in this section:

```
{
    "_id": "id1",
    "a": {
            "b": [{"boolean":false}, {"decimal": 123.456}],
            "c":{
                  "d":10,
                  "e":"Hello"
                 }
          },
    "m":"MapR wins"
}
```

Simple Sort

The following syntax and example are a simple sort on a single field path in the default ascending sort order:

```
// Syntax
find <table path> --orderby <field path>

// Example
find /tbl --orderby a.c.d
```

Specific Sort on Single Field

The following syntax and example are a sort with a specified ordering on a single field path:

```
// Syntax
find <table path> --orderby <field path>:<sort order>

// Example
find /tbl --orderby a.c.d:desc
```

Specific Sort on Multiple Fields

The following syntax and example specify a sort ordering on each field path:

```
// Syntax
find <table path> --orderby <field path>:<sort order>,<field path>:<sort
order>

// Example
find /tbl --orderby a.c.d:asc,a.c.e:desc
```

Mixed Mode Sort on Multiple Fields

The following syntax and example specify a sort ordering on one field path and use the default sort order (ascending) on another field path.

```
// Syntax
find <table path> --orderby <field path>:<sort order>,<field path>

// Example
find /tbl --orderby a.c.d:DESC,a.c.e
```

Query Examples with Other Options

This section contains examples of `findbyid` and `find` commands using options not used in examples in other sections.

Return all Documents

When you do not specify other options to `find` except the table path, the command returns all documents. The following example returns all documents that are in the `/data/movies` table:

```
find /data/movies
```

Return Limited Number of Documents with Specified Range of IDs

The following example returns at most 32 documents within a range of IDs that includes the specified starting ID and excludes the specified ending ID:

```
find /data/movies --fromid movie0000001 --toid movie0000100 --limit
32
```

Return all Documents with Specified ID

The following example returns the document that has the specified ID:

```
findbyid /data/movies --id movie0000002
```


last-updated: Apr 11, 2022

Return Documents Using Projection and Conditions

The following example performs a `find` operation with a projection and condition and limits the result to 10 documents:

```
find /tbl --c {
              "$and":[
                 {"$eq":{"a.b[0].boolean":false}},
                 {"$or":[
                    {"$ne":{"a.c.d":5}},
                    {"$gt":{"a.b[1].decimal":1}},
                    {"$lt":{"a.b[1].decimal":10}}
                 ]}
              ]}
           --fields m,a.c.e --limit 10
```

The projection is on fields on `m` and `a.c.e`. The condition is the following expression:

```
(a.b.[0].boolean == false && (a.c.d != 5 || a.b[1].decimal > 1 ||
a.b[1].decimal < 10))
```

Returns Documents in Specified Order

The following example is identical to the previous one, except it also includes an ordering on the result:

```
find /tbl --c {
              "$and":[
                 {"$eq":{"a.b[0].boolean":false}},
                 {"$or":[
                    {"$ne":{"a.c.d":5}},
                    {"$gt":{"a.b[1].decimal":1}},
                    {"$lt":{"a.b[1].decimal":10}}
                 ]}
               ]}
           --fields m,a.c.e
           --orderby m,a.c.e:desc
           --limit 10
```

dbshell indexscan

Description

The `indexscan` command scans secondary indexes and returns the document ID and the values of the indexed and included fields. This includes displaying information about errors encountered inserting into the index.

Syntax

```
maprdb root:> indexscan
        <table path>
        --indexname <index name>
        --limit
        --withtags
        --pretty
        --mode
        --where
        --fields
        --decodeindexedfields
```

Parameters

Parameters	Description
`*`, `--t`, `--table` (Required)	Path of the JSON table
`--indexname`, `--indexName` (Required)	Name of the secondary index
`--limit`	Maximum number of documents to return
`--withtags`, `--withTags`	Enables or disables printing with extended JSON type tags Values: `true`|`false`
`--pretty`	Enables or disables pretty printing of documents Values: `true`|`false`
`--mode`	Enables display of the error information for the index Value: `err` If you specify `--mode err`, the command scans only rows with errors and prints the `_id` and `$ERROR` fields. If you do not specify `--mode`, the command prints the `_id`, indexed, and included fields of rows that do not have errors. The following lists the types of errors: • `KEY_TOO_LONG` • `INVALID_CAST`
`--c`, `--where`	Condition, in JSON format, that filters the rows returned See OJAI Query Condition Syntax on page 2606 for a description of the syntax.
`--f`, `--fields`	Fields from the index to return See JSON Document Field Paths on page 515 for details about how to specify field paths.
`--decodeindexedfields`	Enables display of values for indexed fields that are nested documents or arrays Value: `true` **Note:** This parameter ignores all other values, including specifying no value.

Example: Simple Index

The following example uses a simple index where `index1` is on `table1`, field `a`.

```
// Insert one document
maprdb root:> insert /table1 --id 1 --value '{"a":7}'
Document with id: "1" inserted.

// Create index1 on table1 and index field a
```


last-updated: Apr 11, 2022

```
# maprcli table index add -path /table1 -index index1 -indexedfields a

// Perform a normal indexscan; the _id field and the indexed field for the
document is displayed
maprdb root:> indexscan /table1 --indexname index1
{"_id":"1","a":7}
1 document(s) found.

// Insert another document with _id value as 2 with field a as a map
maprdb root:> insert /table1 --id 2 --value '{"a":[1,2,3]}'

// Perform a normal indexscan; the document that does not have the error is
displayed
maprdb root:> indexscan /table1 --indexname index1
{"_id":"2","a":[1,2,3]}
{"_id":"1","a":7}
2 document(s) found.

// Perform an indexscan with error mode; no errors are displayed because
MapR-DB 6.1 allows
// you to create indexes on array fields
maprdb root:> indexscan /table1 --indexname index1 --mode err
0 document(s) found.
```

Example: Composite Index

The following example uses a composite index with included fields, in which `index2` is on table `table1`, with indexed fields a and b and included field c.

```
// Insert a document with fields 'a', 'b' and 'c'.
maprdb root:> insert /table1 --id 2 --value '{"a":7,"b":"mapr","c":"db"}'
Document with id: "2" inserted.

// Create index2 on table1 withindexed fields a and b, and included field c
# maprcli table index add -path /table1 -index index2 -indexedfields
a,b -includedfields c

// Perform an indexscan
maprdb root:> indexscan /table1 --indexname index2
{"_id":"2","c":"db","a":7,"b":"mapr"}
1 document(s) found.

// Insert a document that has field a as a map
maprdb root:> insert /table1 --id 1 --value '{"a":
{"m":4},"b":"mapr","c":"db"}'
Document with id: "1" inserted.

// Perform a normal indexscan
maprdb root:> indexscan /table1 --indexname index2
{"_id":"1","c":"db","a":{"m":4},"b":"mapr"}
{"_id":"2","c":"db","a":7,"b":"mapr"}
2 document(s) found.

// Perform an indexscan with error mode; no errors are displayed because
MapR-DB 6.1 allows
// you to create indexes on array fields
maprdb root:> indexscan /table1 --indexname index2 --mode err
0 document(s) found.
```


last-updated: Apr 11, 2022

Example: Index on Container Field Paths

Assume you have a table in the path `/apps/indexExample` with the following document:

```
{
   "_id": "10000",
   "FullName": {
      "LastName": "Smith",
      "FirstName": "John"
   },
   "Address": {
      "Street": "123 SE 22nd St.",
      "City": "Oakland",
      "State": "CA",
      "Zipcode": "94601-1001"
   },
   "Gender": "M",
   "AccountBalance": 999.99,
   "Email": "john.smith@company.com",
   "Phones": [
      {"Type": "Home", "Number": "555-555-1234"},
      {"Type": "Mobile", "Number": "555-555-5678"},
      {"Type": "Work", "Number": "555-555-9012"}
   ],
   "Hobbies": ["Baseball", "Cooking", "Reading"],
   "DateOfBirth": "10/1/1985"
}
```

The following example creates a composite index on the `Type` and `Number` subfields in the nested documents in the `Phones` array:

```
// Create idx3 on the table with indexed fields Phones[].Type and
Phones[].Number
# maprcli table index add -path /apps/indexExample -index idx3 \
    -indexedfields Phones[].Type,Phones[].Number

// Perform an indexscan WITHOUT the decodeindexedfields parameter.
// Three rows are returned, one for each element in the Phones[] array.
// The output contains no values for the indexed fields.
maprdb root:> indexscan /apps/indexExample --indexname idx3
{"_id":"10000"}
{"_id":"10000"}
{"_id":"10000"}
3 document(s) found

// Perform an indexscan WITH the decodeindexedfields parameter set to true.
// The output includes the values in the indexed fields.
maprdb mapr:> indexscan /apps/indexExample --indexname
idx3 --decodeindexedfields true
{"_id":"10000","$idx":["Home","555-555-1234"]}
{"_id":"10000","$idx":["Mobile","555-555-5678"]}
{"_id":"10000","$idx":["Work","555-555-9012"]}
3 document(s) found.
```

Troubleshooting Use Cases

Situations where you can use this command are as follows:

- List the contents of an index.
- Resolve encoding errors encountered inserting into an index.

last-updated: Apr 11, 2022

See Troubleshooting Secondary Indexes on page 1092 for more information on these use cases.

dbshell insert

Description

The dbshell `insert` command adds documents to JSON tables. Specify the ID of the document in one of two ways:

- As the value of the `_id` field in the document
- As the value of the `--id` parameter in the `insert` command

If a document with the specified ID already exists, the command replaces the document with the new one.

Parameters

insert Options	Description
`*`, `--t`, `--table` (Required)	Table path
`--id`	ID of the document to insert or replace
`--v`, `--value` (Required)	JSON document to insert or replace
`--c`, `--where`	OJAI condition, in JSON format The condition must qualify to perform the insert. See OJAI Query Condition Syntax on page 2606 for a description of the syntax.

Syntax

```
insert --table <table path> --value '{"_id": "<row-key", < table field >}'
```

```
insert --table <table path> --id <row-key> --value '{"_id": "<row-key", <
table field >}'
```

Example: Insert with `_id` Field

The following examples insert a document into a table using an `_id` field value:

```
insert /data/movies --value '{"_id":"movie0000002",
                              "title":"Developers on the Edge",
                              "studio":"Command Line Studios"}'
```

```
insert /tables/users --value '{"_id":"user001",
                               "first_name":"John",
                               "last_name":"Doe"}'
```


last-updated: Apr 11, 2022

Example: Insert with `--id` Parameter

The following examples insert a document into a table using the `--id` parameter in the insert command:

```
insert /data/movies --id movie0000003 --value '{"title":"The Golden
Master",
                                                "studio":"All-Nighter"}'
```

```
insert /tables/users --id user002 --value '{"first_name":"Jane",
                                             "last_name":"Dane"}'
```

dbshell jsonoptions

Description

The `jsonoptions` command sets the JSON output and displays the output appropriately.

Parameters

Table

jsonoptions Options	Description
`--pretty`	Enables or disables pretty printing mode Value: `true`|`false` Default: `true` Pretty print mode displays the content of the documents as an indented hierarchy of field/value pairs. For example, if the value is `true`, the DATE data type appears as a Map: `"dob" : {"$dateDay":"2012-10-20"}`
`--withtags`, `--withTags`	Enables or disables printing with extended JSON data type tags Value: `true`|`false` Default: `true` For example, if the value is `false`, the DATE data type appears as a simple value: `"dob" : "2012-10-20"`

Syntax

```
jsonoptions --pretty <true|false>
jsonoptions --withTags <true|false>
```

Example

```
jsonoptions --pretty true
jsonoptions --withTags false
```

dbshell list

last-updated: Apr 11, 2022

Description

Lists all JSON tables in a folder or that matches a specific file pattern where the table resides.

Parameters

Table

list Options	Description
*, --p, --patternOrPath	A path and/or a file pattern

Syntax

```
list <path>
```

Example and Output

```
maprdb root:>list /demo
/demo/user
/demo/checkin
/demo/review
/demo/business
/demo/tip
5 table(s) found.
```

dbshell replace

Description

The dbshell `replace` command replaces a document in a JSON table. You can specify a condition with the command.

Parameters

replace Options	Description
*, --t, --table (Required)	Table path
--id (Required)	ID of the document to replace If the specified ID does not exist, the command inserts a new document with the values provided in the command. **Note:** You can specify this parameter only once.
--v, --value (Required)	JSON document to insert or replace
--c, --where	OJAI condition, in JSON format The condition must qualify to perform the replace. See OJAI Query Condition Syntax on page 2606 for a description of the syntax.

Syntax

```
replace /tbl --id <id> --v {<document to replace>} [--c <condition>]
```


last-updated: Apr 11, 2022

Example

```
replace /tables/users --id user002 --value '{"first_name":"Jane",
"last_name":"Doe"}'
```

dbshell update

Description

The dbshell `update` command updates JSON documents using OJAI mutations. An OJAI mutation allows you to append, decrement, delete, increment, combine, replace, and update fields in a JSON document.

The following table lists the mutations OJAI supports. See Using OJAI Mutation Syntax on page 2561 for a detailed description of all operations. Each operation in the table links to examples in this topic.

Mutation Operation	Description
Append	Appends values to binary, string, and array fields
Decrement	Decrements field values
Delete	Deletes fields
Increment	Increments field values
Merge	Combines nested documents with existing documents
Put	Replaces field values or adds new fields
Set	Updates field values or adds new fields

Parameters

update Options	Description
`*`, `--t`, `--table` (Required)	Table path
`--id` (Required)	ID of the document to update **Note:** You can specify this parameter only once.
`--m`, `--mutation` (Required)	OJAI document mutation in JSON format See Using OJAI Mutation Syntax on page 2561 for a description of the syntax.
`--c`, `--where`	OJAI condition, in JSON format The condition must qualify to perform the update. See OJAI Query Condition Syntax on page 2606 for a description of the syntax.

Syntax

```
update <table path> --id <id> --m <mutation> [ --c <condition> ]
```

Note: If the mutation provided as a part of the `--m` parameter has spaces, then you must enclose it within single quotes.

last-updated: Apr 11, 2022

Sample JSON Document

The dbshell update examples in this topic use the following sample JSON document:

```
{
  "_id": "id1",
  "a": {
       "b": [{"boolean":false}, {"decimal": 123.456}],
       "c":{
             "d":10,
             "e":"Hello"
           }
       },
  "m":"MapR wins"
}
```

Append Operation

This example performs append operations on fields `a.b` and `a.c.e`:

```
update /tbl --id id1 --m {
      "$append":[{"a.b":{"appd":1}},{"a.c.e":" MapR"}]
      }
```

When you apply this update command to the sample JSON document, the following is the resulting document:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : false }, { "decimal" : 123.456 }, { "appd" : 1 }
],
    "c" : {
      "d" : 10,
      "e" : "Hello MapR"
    }
  },
  "m" : "MapR wins"
}
```

For more details about the `$append` operation, see OJAI Append Mutations on page 2562.

Decrement Operation

This example performs a decrement operation:

```
update /tbl --id id1 --m {
      "$decrement":{"a.c.d":5}
      }
```

When you apply this update command to the sample JSON document, the following is the resulting document:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : false }, { "decimal" : 123.456 } ],
    "c" : {
      "d" : 5,
      "e" : "Hello"
    }
```


last-updated: Apr 11, 2022

```
  },
  "m" : "MapR wins"
}
```

Note: To decrement multiple fields, use an an array to specify the fields.

For more details about the `$decrement` operation, see OJAI Decrement Mutations on page 2563.

Delete Operation

With the following example, the operation deletes multiple field paths in the document in a single command:

```
update /tbl --id id1 --m {
      "$delete": ["a.b[1]","a.c.e"]
      }
```

When you apply this update command to the sample JSON document, the following is the resulting document:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : false } ],
    "c" : {
      "d" : 10
    }
  },
  "m" : "MapR wins"
}
```

The following example shows that if you need to delete only a single field, do not use the array notation:

```
update /tbl --id id1 --m {
      "$delete":"a.b[1]"
      }
```

For more details about the `$delete` operation, see OJAI Delete Mutations on page 2563.

Increment Operation

This example performs an increment operation:

```
update /tbl --id id1 --m {
      "$increment":{"a.c.d":5}
      }
```

When you apply this update command to the sample JSON document, the following is the resulting document:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : false }, { "decimal" : 123.456 } ],
    "c" : {
      "d" : 15,
      "e" : "Hello"
    }
  },
  "m" : "MapR wins"
}
```


last-updated: Apr 11, 2022

Note: To increment multiple fields, use an an array to specify the fields.

For more details about the `$insert` operation, see OJAI Increment Mutations on page 2564.

Merge Operation

This example performs a merge operation:

```
update /tbl --id id1 --m {
      "$merge":{"a.c":{"d":11,"y":"yo"}}
      }
```

When you apply this update command to the sample JSON document, the following is the resulting document:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : false }, { "decimal" : 123.456 } ],
    "c" : {
      "d" : 11,
      "e" : "Hello",
      "y" : "yo"
    }
  },
  "m" : "MapR wins"
}
```

Note: `$merge` does not support the array format for merging two maps at two different field paths in the document.

For example, the following syntax is incorrect:

```
// WRONG Syntax
update /tbl --id id1 --m {"$merge":["a":{"b":1},{"a":{"d":"MapR"}}]}
```

The following syntax is correct:

```
// CORRECT Syntax
update /tbl --id id1 --m {"$merge":{"a":{"b":1,"d":"MapR"}}}
```

It results in the following document:

```
{
  "_id" : "id1",
  "a" : {
    "b" : 1,
    "c" : {
      "d" : 10,
      "e" : "Hello"
    },
    "d" : "MapR"
  },
  "m" : "MapR wins"
}
```

To merge multiple field paths that are non-overlapping, use the syntax described at either Multiple Mutation Operations on page 5310 or Updates Without Explicit Mutation Operation Names on page 5311.

For more details about the `$merge` operation, see OJAI Merge Mutations on page 2565.

last-updated: Apr 11, 2022

Put Operation

This example performs a put operation. Unlike the set operation, the put *replaces* field values. Like the set operation, you do not need an array representation for a single field.

```
update /tbl --id id1 --m {
      "$put":[{"a.b":{"boolean":true},{"a.c.d":"eureka"},{"a.x":1}]
      }
```

When you apply this update command to the sample JSON document, the following is the resulting document:

```
{
  "_id" : "id1",
  "a" : {
    "b" : { "boolean" : true },
    "c" : {
      "d" : "eureka",
      "e" : "Hello"
    },
    "x" : 1
  },
  "m" : "MapR wins"
}
```

For more details about the `$set` operation, see OJAI Put Mutations on page 2566.

Set Operation

With this example, the command updates the document fields `a.b[0].boolean`, `a.c.d`, and `a.x`. If the field does not exist, the update command creates and sets it. The update fails if the existing field type does not match the new value. If the field exists and is the same type, the value is updated.

```
update /tbl --id id1 --m {
       "$set":[{"a.b[0].boolean":true},{"a.c.d":11},{"a.x":1}]
       }
```

When you apply this update command to the sample JSON document, the following is the resulting document:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : true }, { "decimal" : 123.456 } ],
    "c" : {
      "d" : 11,
      "e" : "Hello"
    },
    "x" : 1
  },
  "m" : "MapR wins"
}
```

Note: If you need to set only a single field, the command looks like the following:

```
update /tbl --id id1 --m {
         "$set": {"a.b[0].boolean":true}
         }
```

For more details about the `$set` operation, see OJAI Set Mutations on page 2567.

last-updated: Apr 11, 2022

Multiple Mutation Operations

You can combine more than one mutation operation in a single OJAI mutation by specifying each operation separated by a comma.

The following is an example that combines multiple operations:

```
update /tbl --id id1 --m
        '{
        "$set":{"x":[1,2,3]},
        "$put":{"a.c.e":{"$binary":"AAAADg=="}},
        "$increment":"a.b[1].decimal",
        "$delete":"a.b[0]",
        "$merge":{"newDoc":{"k":"MapR DBShell rocks!!"}},
        "$append":{"m":"!!!"}
        }'
```

The following is the resulting output:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "decimal" : 124.456 } ],
    "c" : {
      "d" : 10,
      "e" : { "$binary" : "AAAADg==" }
    }
  },
  "m" : "MapR wins!!!",
  "newDoc" : { "k" : "MapR DBShell rocks!!" },
  "x" : [ 1, 2, 3 ]
}
```

The operations behave in the following manner:

- The `$set` operation adds a new array `[1,2,3]` with field path `x` into the document.
- The `$put` operation replaces the existing string `"Hello"` with a nested document `{"$binary":"AAAADg=="}`.
- The `$delete` operation deletes the field path `a.b[0]` from the document.
- The `$merge` operation merges a new nested document `{"newDoc":{"k":"MapR DBShell rocks!!"}}`.
- The `$append` operates appends the string `"!!!"` to the end of the string `"MapR wins"`.
- The `$increment` and `$delete` operate on different elements of the array `a.b`:
  - The `$increment` operation increments the value `123.456` in the second element of the array `a.b`.
  - The `$delete` operation deletes the field path `a.b[0]`, resulting in a single element array `a.b`.

Conflicting Operations

When you specify a mutation with field paths that are overlapping, MapR Database detects the conflict, discards the previous conflicting operation, and proceeds with the next operation.

For example, suppose you have the following document:

```
{"_id":"id1", "a":{"b":{"c":5}}}
```


last-updated: Apr 11, 2022

The following mutation has two operations with overlapping fields `a.b`:

```
{"$delete":"a.b","$set":{"a.b.d":10}}
```

You may have intended for the mtuation to first delete `a.b` and then to replace it with `a.b.d` as follows:

```
{"_id":"id1", "a":{"b":{"d":"10"}}}
```

But the *actual* result is the following:

```
{"_id":"id1", "a":{"b":{"c":5,"d":"10"}}}
```

In this case, the set operation on **`a.b.d`** causes the delete operation on **`a.b`** to be discarded.

Note: In the earlier example in this section, the `$increment` and `$delete` operations are not conflicting because one operates on `a.b[1]`, while the other operates on `a.b[0]`. On the other hand, the following are conflicting operations:

```
{"$increment":"a.b[1].decimal","$delete":"a.b"}
```

Updates Without Explicit Mutation Operation Names

As part of the update command, you can merge a nested document with a document without specifying a mutation operation name. When applying this type of update, the behavior is the same as the merge operation.

For example, suppose you run the following command:

```
update /tbl --id id1 --m {
      "k":"eureka",
      "a":{"c":{"d":1234}}
      }
```

If the document with key `"id1"` exists, the update command merges the nested document with the original document. If the document does not exist, the update creates a new document with the input provided.

Application of the command to the sample document results in the following:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : false }, { "decimal" : 123.456 } ],
    "c" : {
      "d" : 1234,
      "e" : "Hello"
    }
  },
  "k" : "eureka",
  "m" : "MapR wins"
}
```

For the following update command:

```
update /tbl --id id1 --m {
      "k":"eureka",
      "a":{"c":{"d":null}}
      }
```


last-updated: Apr 11, 2022

This is the resulting document:

```
{
  "_id" : "id1",
  "a" : {
    "b" : [ { "boolean" : false }, { "decimal" : 123.456 } ],
    "c" : {
      "d" : null,
      "e" : "Hello"
    }
  },
  "k" : "eureka",
  "m" : "MapR wins"
}
```

Note: In this example, field `a.c.d` remains in the document and is set to null.

Utilities for MapR Database JSON Tables

MapR Database JSON provides utilities to copy, export, and import data, compare table content, and verify the consistency of secondary indexes.

You can use the following utilities with MapR Database JSON tables:

MapR Database JSON CopyTable
Copies data from one MapR Database JSON table to another MapR Database JSON table.

If the destination table does not exist, `mapr copytable` creates the destination table with the same metadata (column families and access control expressions) as the source table, and then copies data.

If the destination table exists, `mapr copytable` copies data only.

Required Permissions

The user that runs `mapr copytable` must have the following permissions, which you can grant with access-control expressions:

- The permission `readAce` on the volume where the source table is located, and the permission `writeAce` on the volume where the destination table is or will be located.
- The permission `adminperm` on the source table.
- The permission for column-family and column reads (`readperm`) on the data in the source table that you want to copy.
- When bulkload = `false`, the permission for column writes (`writeperm`) on the destination table.
- When bulkload = `true` (default), the permission to load the destination table with bulk loads (`bulkloadperm`).
- If the destination table does not yet exist: `createrenamefamily` on the source table.

For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

For information about how to set permissions on tables, see Enabling Table and Stream Authorizations with ACEs on page 1461.

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

```
mapr copytable
-src <source table path>
-dst <destination table path>
[-fromID <start key>]
[-toID <end key>]
[-bulkload <true|false> (default: false)]
[-mapreduce <true|false> (default: true)]
[-cmpmeta <true|false> (default: true)]
[-numthreads <number of threads> (default: 16)]
[-maxsplits <integer> (default: 2000)]
```

Parameters

Parameter	Description
src	The path of the table that you want to copy from.
dst	The path of the table that you want to copy to.
fromID	The value of the `_id` field in the first document of the range of documents to copy. `startRow` is an alias for this parameter.
toID	The value of the `_id` field in the last document of the range of documents to copy. `stopRow` is an alias for this parameter.
bulkload	A Boolean value that specifies whether or not to perform a full bulk load of the table. The default is not to use bulk loading (`false`). To use bulk load, you must set the `-bulkload` parameter of the table to `true` by running the command `maprcli table edit -path <path to table> -bulkload true`.
mapreduce	A Boolean value that specifies whether or not to use a MapReduce program to perform the copying operation. The default, preferred method is to use a MapReduce program (true). When this parameter is set to false, a client process uses multiple threads to read rows of the source table and write rows to the destination table.
cmpmeta	A Boolean value that specifies whether or not to compare table metadata such as column families and ACEs. The default is to compare metadata (true). Such comparisons are done when the destination table exists before `mapr copytable` is run and checks that the user ID that runs `mapr copytable` has the proper permissions on the destination table. Set the value of this parameter to `false` before copying a table that contains a single column family to a table that contains two or more column families.
numthreads	When -mapreduce is false, this parameter specifies the number of threads allocated to perform the copying of data. The default is 16. If additional CPU resources are available, you might want to increase the number of threads to achieve better performance.

last-updated: Apr 11, 2022

Parameter	Description
maxsplits	Sets the maximum number of destination table presplit tablets. Default is 2000. If `copytable` fails with an Error NO ENTry message during table creation, the operation could not complete within the timeout (10 minutes). Reduce the value of `-maxsplits`. This functionality requires a patch. See Applying a Patch.

Example

The following example copies documents starting from ID `user000001` to ID `user009999`:

```
[user@hostname ~]$ mapr copytable -src /user1/tableA -dst
/mapr/clusterB/vol1/tableB -fromID user000001 -toID user009999
```

Monitoring `mapr copytable` Operations

Use one of the following methods to monitor the progress of the copying of table data:

- If the copy table operation runs as a MapReduce v2 application, monitor the application using the ResourceManager UI.
- If the copy table operation runs as a client process, go to the Tables view of the destination table in the MapR Control System. Then, on the Region tab, monitor the pace at which the number of rows increases.

MapR Database JSON DiffTables

Compares the row keys, column families, and field values in two JSON tables. Then, generates two directories that contain sequence files that you can use to merge the rows from the two JSON tables.

Sequence files are binary flat files. For more detail, see Sequence File. To convert a sequence file into a format that you can read, use the `mapr formatresult` utility.

This utility considers both the source table and the destination table to be a master table. Therefore, it generates two directories with sequence files. These sequence files contain the puts required to update each table so that it contains a superset of the rows defined in both tables at the time at which the utility was run.

This utility generates both of the following output directories in the output directory that you specify:

opsForDst	A directory containing sequence files that correspond to each put and delete required to make the destination table identical to the source table.
opsForSrc	A directory containing sequence files that correspond to each put and delete required to make the source table identical to the destination table.

A user with write permissions on a table can run the `mapr importtable` utility to implement the changes that are specified in the sequence files.

Required Permissions

The user that runs the `mapr difftables` utility must have the following permissions:

- The permission `readAce` on the volumes where the tables are located.
- The permission for column reads (`readperm`) on each table.

For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

last-updated: Apr 11, 2022

For information about how to set permissions on tables, see Enabling Table and Stream Authorizations with ACEs on page 1461.

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

```
mapr difftables
-src <source table path>
-dst <destination table path>
-outdir <output directory>
[-first_exit Exit when first difference is found. ]
[-columns comma-separated list of field paths ]
[-mapreduce] <true|false> (default: true)]
[-numthreads <numThreads> (default:16, valid only when -mapreduce is false)]
[-cmpmeta <true|false> (default: true)]
```

Parameters

Parameter	Description
src	The path of the first table to include in the comparison.
dst	The path of the second table to include in the comparison.
first_exit	By default, the utility compares all the table cells in the specified tables. Use this parameter if you want to exit after the first difference is identified between the tables. The parameter takes no value.
outdir	The path to a directory in which to place the generated sequence files. The utility creates the specified directory. If the specified directory already exists, the command fails.
columns	By default, the utility compares all fields in JSON tables. If you do not want to compare all fields, you can specify specific fields to include in the comparison. For example, suppose that want to compare a source table in table replication with a replica of that table. When you set up replication, you chose to replicate the default column family and two additional column families: `cf1` and `cf2`. For the -columns parameter, you would specify the value `",cf1,cf2"`, where the default column family is represented by the empty string.
mapreduce	A Boolean value that specifies whether or not to use a MapReduce program to perform the comparison. The default, preferred method is to use a MapReduce program (true). When this parameter is set to false, a client process uses multiple threads to perform the comparison.
numthreads	When -mapreduce is false, this parameter specifies the number of threads allocated to perform the comparison. The default is 16. If additional CPU resources are available, you might want to increase the number of thread to achieve better performance.

last-updated: Apr 11, 2022

Parameter	Description
cmpmeta	A Boolean value that specifies whether or not to compare table metadata such as column families and ACEs. The default is to compare metadata (true).

Example

The following example shows a comparison of two JSON tables

```
[user@hostname ~]$ mapr difftables -src /source_JSON_table -dst /
destination_JSON_table -outdir output/comparison1 -columns
"dateRange.endYear","contributors.date"
Header: hostName: maprdemo, Time Zone: Pacific Standard Time, processName:
null, processId: null
2015-10-01 14:46:22,537 INFO com.mapr.db.mapreduce.tools.DiffTables
parseArgs main: Comparing dateRange.endYear,contributors.date column
families from /source_JSON_table to /destination_JSON_table.
DiffTablesMeta completed. Metadata of the two tables is same.
2015-10-01 14:46:23,040 INFO com.mapr.db.mapreduce.tools.DiffTables
parseArgs main: Comparing dateRange.endYear,contributors.date column
families from /source_JSON_table to /destination_JSON_table.
2015-10-01 14:46:23,910 INFO org.mortbay.log info main:
Logging to org.slf4j.impl.Log4jLoggerAdapter(org.mortbay.log) via
org.mortbay.log.Slf4jLog
2015-10-01 14:46:24,100 INFO org.apache.hadoop.io.compress.zlib.ZlibFactory
<clinit> pool-4-thread-1: Successfully loaded & initialized native-zlib
library
2015-10-01 14:46:24,103 INFO org.apache.hadoop.io.compress.CodecPool
getCompressor pool-4-thread-1: Got brand-new compressor [.deflate]
2015-10-01 14:46:24,134 INFO org.apache.hadoop.io.compress.CodecPool
getCompressor pool-4-thread-1: Got brand-new compressor [.deflate]
tables '/source_JSON_table', and '/destination_JSON_table' didn't match
Number of rows processed in '/source_JSON_table' : 100
Number of rows processed in '/destination_JSON_table' : 100
Mismatch row count in '/source_JSON_table' : 1
Mismatch row count in '/destination_JSON_table' : 1
Rows with mismatch are stored in output/comparison1
```

MapR Database JSON DiffTablesWithCrc

This utility uses a cyclic redundancy check to detect differences between sets of rows in the specified MapR Database JSON tables. Then, for each set of non-identical rows, it performs a detailed comparison. Finally, it generates one or more directories of sequence files. You can use these files either to merge the rows from two MapR Database JSON tables.

Sequence files are binary flat files. You can learn more about them here. To convert a sequence file into a format that you can read, use the MapR Database JSON FormatResult on page 5318 utility.

This utility requires less network bandwidth than the `mapr diffTables` utility because it performs a detailed table comparison only on the sets of rows where the CRC algorithm detected a difference. Therefore, consider using this utility when the tables you compare are very similar and you are concerned about the data transfer rate.

This utility considers both the source table and the destination table to be a master table. Therefore, it generates two directories with sequence files. These sequence files contain the puts required to update each table so that each table can contain a superset of the rows in both tables at the time at which the utility was run.

This utility generates the following output directories:

- **opsForDst.** A directory containing sequence files that correspond to each put and delete required to make the destination table identical to the source table.

last-updated: Apr 11, 2022

- **opsForSrc.** A directory containing sequence files that correspond to each put and delete required to make the source table identical to the destination table.

Run the `mapr importtable` command to implement the puts and deletes specified in the sequence files.

A user with write permissions on a table can run the `mapr importtable` utility to implement the changes that are specified in the sequence files.

Requirements

- When the cluster runs YARN, it must use zero configuration failover for the ResourceManager.
- The user that runs the `mapr difftableswithcrc` utility must have the following permissions:
  - The permission `readAce` on the volumes where the tables are located.
  - The permission for column reads (`readperm`) on each table.

  For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

  For information about how to set permissions on tables, see Enabling Table and Stream Authorizations with ACEs on page 1461.

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

```
mapr difftableswithcrc
-src <source table path>
-dst <destination table path>
-outdir <output directory>
[-first_exit] Exit when first difference is found.
[-columns <comma separated list of field paths> ]
[-exclude_embedded_families <true|false>] (default: false)
  Don't include the  other column families with path embedded in specified
columns
[-cmpmeta <true|false> (default: true)]
```

Parameters

Parameter	Description
src	The path of the first table to include in the comparison.
dst	The path of the second table to include in the comparison.
outdir	The path to a directory in which to place the generated sequence files. The utility creates the specified directory. If the specified directory already exists, the command fails.
first_exit	By default, the utility compares all the table cells in the specified tables. Use this parameter if you want to exit after the first difference is identified between the tables. The parameter takes no value.

last-updated: Apr 11, 2022

Parameter	Description
columns	By default, the utility compares all fields in JSON tables. If you do not want to compare all fields, you can specify specific fields to include in the comparison. For example, suppose that want to compare a source table in table replication with a replica of that table. When you set up replication, you chose to replicate the default column family and two additional column families: `cf1` and `cf2`. For the -columns parameter, you would specify the value `",cf1,cf2"`, where the default column family is represented by the empty string.
cmpmeta	A Boolean value that specifies whether or not to compare table metadata such as column families and ACEs. The default is to compare metadata (true).

MapR Database JSON FormatResult

Parses a sequence file generated by the difftables utility for JSON tables and converts the results into a format that makes the results easier to understand.

Required Permissions

The user that runs the `FormatResult` utility must have the `readAce` and `writeAce` permissions on the volumes where the input and output paths are located.

For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

```
mapr formatresult
-indir <input file path>
-outdir <output file path>
[-mapreduce <true|false> (default: false)]
```

Parameters

Parameter	Description
indir	The path to a file or directory of files that contains the output of the `mapr difftables` utility.
outdir	The path to a file or a directory for the output. If the file or directory already exists, the utility fails. When a single sequence file is provided as input, the utility generates a single output file. When a directory of sequence files is provided as input, the utility generates a directory with output files.
mapreduce	A Boolean value that specifies whether or not to use a MapReduce program to perform the copying operation. The default, preferred method is to use a MapReduce program (true).

Example

This example shows the results of the following actions that followed a comparison by difftables of two JSON tables:

last-updated: Apr 11, 2022

1. Formatting the sequence file for the source JSON table.
2. Formatting the sequence file for the destination JSON table.
3. Viewing the content of the first sequence file.
4. Viewing the content of the second sequence file.

This is the command that was used for `maprdb difftables`:

```
mapr difftables -src /src_table -dst /dest_table -outdir
output/diffs -columns dateRange.endYear
```

Here is the command that was used for `mapr formatresult` and the resulting output:

```
[user@hostname ~]$ mapr formatresult -indir output/diffs/
OpsForSrcTable -outdir output/outputForSrcTable5
Header: hostName: maprdemo, Time Zone: Pacific Standard Time, processName:
null, processId: null
2015-10-01 14:46:48,887 INFO org.apache.hadoop.io.compress.zlib.ZlibFactory
<clinit> pool-1-thread-1: Successfully loaded & initialized native-zlib
library
2015-10-01 14:46:48,894 INFO org.apache.hadoop.io.compress.CodecPool
getDecompressor pool-1-thread-1: Got brand-new decompressor [.deflate]
2015-10-01 14:46:48,915 INFO org.apache.hadoop.io.compress.CodecPool
getDecompressor pool-1-thread-1: Got brand-new decompressor [.deflate]
2015-10-01 14:46:48,915 INFO org.apache.hadoop.io.compress.CodecPool
getDecompressor pool-1-thread-1: Got brand-new decompressor [.deflate]
2015-10-01 14:46:48,916 INFO org.apache.hadoop.io.compress.CodecPool
getDecompressor pool-1-thread-1: Got brand-new decompressor [.deflate]
Successfully created files in output/outputForSrcTable5
[user@hostname ~]$ mapr formatresult -indir output/diffs/
OpsForDstTable -outdir output/outputForDstTable5
Header: hostName: maprdemo, Time Zone: Pacific Standard Time, processName:
null, processId: null
2015-10-01 14:47:10,004 INFO org.apache.hadoop.io.compress.zlib.ZlibFactory
<clinit> pool-1-thread-1: Successfully loaded & initialized native-zlib
library
2015-10-01 14:47:10,012 INFO org.apache.hadoop.io.compress.CodecPool
getDecompressor pool-1-thread-1: Got brand-new decompressor [.deflate]
2015-10-01 14:47:10,030 INFO org.apache.hadoop.io.compress.CodecPool
getDecompressor pool-1-thread-1: Got brand-new decompressor [.deflate]
2015-10-01 14:47:10,031 INFO org.apache.hadoop.io.compress.CodecPool
getDecompressor pool-1-thread-1: Got brand-new decompressor [.deflate]
2015-10-01 14:47:10,031 INFO org.apache.hadoop.io.compress.CodecPool
getDecompressor pool-1-thread-1: Got brand-new decompressor [.deflate]
Successfully created files in output/outputForDstTable5
[user@hostname ~]$ hadoop fs -cat output/outputForSrcTable5/
opsforsrc_0.diff.txt
"row":{ "_id":"A1A4MDE5OQ==(P80199)", "value":{"_familypath":"","_value":
{"_timestamp":[0.0, 1443730581185.0, 1443730581185.0]}}}
[user@hostname ~]$ hadoop fs -cat output/outputForDstTable5/
opsfordst_0.diff.txt
"row":{ "_id":"A1A4MDE5OQ==(P80199)", "value":{"_familypath":"","_value":
{"_timestamp":[1443708157657.0, 1443708157657.0, 1443708157657.0],
"dateRange":{"_timestamp":[1443708157657.0, 1443708157657.0, 0.0],
"_value":{"endYear":{"_timestamp":[1443708157657.0, 1443708157657.0, 0.0],
"_value":1938.0}}}}}}
[user@hostname ~]$
```

MapR Database JSON ExportTable and ImportTable
Use these utilities together to export data from JSON tables into binary sequence files, and then import the data from the binary sequence files into other JSON tables. You can also use the `mapr importtable` utility to import changes that are specified in sequence files output by the `mapr difftables` utility.

- Syntax of `mapr exporttable`
- Parameters of `mapr exporttable`
- Syntax of `mapr importtable`
- Parameters of `mapr importtable`
- Example of using `mapr exporttable` and `mapr importtable` together

Required Permissions

- The `readAce` permission on the volume where the source table for `mapr exporttable` is located.
- The `writeAce` permission on the volume in which to save the output from `mapr exporttable`.
- The `readAce` permission on the volume where the files output by `mapr exporttable` is located.
- The `writeAce` permission on the volume in which the destination table is located.

For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run these utilities unless that user is given the relevant permission or permissions with access-control expressions.

Syntax of `mapr exporttable`

```
mapr exporttable
(option)
-src Name of table
-dst Directory path
[-columns Fields to include]
[-mapreduce : <true|false>, default is true]
```

Parameters of `mapr exporttable`

Parameter	Description
src	The path of the JSON table to export from.
dst	The directory within the MapR filesystem to export the files to.
columns	A comma-delimited list of fields to include in the exported files. **Example** `a,b,c` Do not use quotation marks and do not include spaces after commas.

last-updated: Apr 11, 2022

Parameter	Description
mapreduce	The cluster must have YARN installed and configured for this option to work. A Boolean value that specifies whether or not to use a MapReduce program to perform the operation. The default, preferred method is to use a MapReduce program (true). When this parameter is set to false, a client process uses multiple threads.

Syntax of `mapr importtable`

```
mapr importtable
(option)
-src Input binary file or directory path
-dst Destination table
[-bulkload <true|false>, default is false ]
[-mapreduce : <true|false>, default is true]
```

Parameters of `mapr importtable`

Parameter	Description

src	The path of the binary file or files to import.
	Examples
	`-src /temp/part0`
	`-src /temp/*`

dst	The JSON table to import the data into.
bulkload	A Boolean value that specifies whether or not to perform a full bulk load of the table. The default is not to use bulk loading (`false`). To use bulk load, you must set the `-bulkload` parameter of the table to `true` by running the command `maprcli table edit -path <path to table> -bulkload true`.
mapreduce	The cluster must have YARN installed and configured for this option to work. A Boolean value that specifies whether or not to use a MapReduce program to perform the operation. The default, preferred method is to use a MapReduce program (true). When this parameter is set to false, a client process uses multiple threads.

Example of using `mapr exporttable` and `mapr importtable` together

```
[user@hostname ~]$ mapr exporttable -columns contributors,creditLine -src /
collection/artworks -dst /tempExport
Header: hostName: hostname, Time Zone: Pacific Standard Time, processName:
null, processId: null
2015-10-01 23:02:38,044 INFO org.apache.hadoop.io.compress.zlib.ZlibFactory
<clinit> pool-2-thread-1: Successfully loaded & initialized native-zlib
library
```


last-updated: Apr 11, 2022

```
2015-10-01 23:02:38,059 INFO org.apache.hadoop.io.compress.CodecPool
getCompressor pool-2-thread-1: Got brand-new compressor [.deflate]
[user@hostname ~]$ hadoop mfs -ls /tempExport
Found 1 items
-rw-r--r-- Z U U   1 mapr mapr     108221 2015-10-01 23:02  268435456 /
tempExport/part0
           p 2049.184.918810  hostname:5660
           0 2180.39.131304  hostname:5660
[user@hostname ~]$ mapr importtable -src /tempExport/* -dst /new_collection/
artworks
Header: hostName: hostname, Time Zone: Pacific Standard Time, processName:
null, processId: null
2015-10-01 23:04:50,022 INFO org.apache.hadoop.io.compress.zlib.ZlibFactory
<clinit> pool-1-thread-1: Successfully loaded & initialized native-zlib
library
2015-10-01 23:04:50,029 INFO org.apache.hadoop.io.compress.CodecPool
getDecompressor pool-1-thread-1: Got brand-new decompressor [.deflate]
[user@hostname ~]$
```

MapR Database JSON ImportJSON
Imports one or more JSON documents into a MapR Database JSON table. The JSON documents must be flat text files.

Required Permissions

- The `readAce` permission on the volume where the JSON documents to import are located.
- The `writeAce` permission on the volume in which the destination table is located.

For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

```
mapr importJSON
[-idfield <Name of ID field in JSON Data>]
[-bulkload <true|false>, default is false]
[-mapreduce : <true|false>, default is true]
-src <text file or directory>
-dst <JSON table>
```

Parameters

-

Parameter	Description
idfield	The name of the field that contains the value to use for each document's `_id` field. An `_id` field is inserted into each document that is imported into a table, if the document does not already contain one. Documents that do not already contain an `_id` field must contain a field with a value that can be used for the inserted `_id` field. For example, each document might have a `product_ID` field with a value that would be suitable for the `_id` field. Use quotation marks around the name.
bulkload	A Boolean value that specifies whether or not to perform a full bulk load of the table. The default is not to use bulk loading (`false`). To use bulk load, you must set the `-bulkload` parameter of the table to `true` by running the command `maprcli table edit -path <path to table> -bulkload true`. This parameter cannot be set to `true` when the `-mapreduce` parameter is set to `false`.
mapreduce	A Boolean value that specifies whether or not to use a MapReduce program to perform the copying operation. The default, preferred method is to use a MapReduce program (true).
src	The path of a JSON document in text format or a directory of such documents. If you specify a directory and that directory contains only the JSON files to import, use an asterisk at the end of the path, as in this example: `/user/data/*` If you specify a directory and that directory contains both the JSON files to import and other files, use a more specific wildcard, such as `*.json` . The path must be in the MapR filesystem. To move files there from the Linux filesystem, use the command `hadoop fs -copyFromLocal`.
dst	The path of the destination MapR Database JSON table.

Example

Suppose you have the following three JSON documents in the `/tmp/users` directory in your MapR filesystem:

```
$ hadoop fs -cat /tmp/users/bcummings.json
{"_id":"bcummings","first_name":"Bettie","last_name":"Cummings"}

$ hadoop fs -cat /tmp/users/gjones.json
{"_id":"gjones","first_name":"Gilberto","last_name":"Jones"}

$ hadoop fs -cat /tmp/users/jdoe.json
{"_id":"jdoe","first_name":"John","last_name":"Doe"}
```


last-updated: Apr 11, 2022

The following command imports the three documents into the JSON table in the path `/apps/users`:

```
$ mapr importJSON -idField _id -src /tmp/users/* -dst /apps/users
```

You can run `mapr dbshell` to see the imported documents:

```
maprdb mapr:> find /apps/users
{"_id":"bcummings","first_name":"Bettie","last_name":"Cummings"}
{"_id":"gjones","first_name":"Gilberto","last_name":"Jones"}
{"_id":"jdoe","first_name":"John","last_name":"Doe"}
3 document(s) found.
```

MapR Database JSON verifyindex

Describes how to use the MapR Database JSON `verifyindex` command to verify that the data in a secondary index is consistent with its JSON table.

Syntax

```
mapr verifyindex
        -path < table path >
        -index < index name >
        -first_exit < true | false >
        -numthreads < thread number >
```

Parameters

Parameter	Description
path	(Required) Path to where the table exists.
index	(Required) Name of the secondary index on the table.
first_exit	(Optional) Exit when the first difference is found. Options: true or false. Default: false.
numthreads	(Optional) Number of parallel threads to use for the verification. Default: 16

Example

The following example creates a table, creates a secondary index on the table, inserts some documents, and then runs the `verifyindex` command to verify that there is data consistency between the JSON table and the secondary index. See Troubleshooting Secondary Indexes on page 1092 for an example where `verifyindex` detects data inconsistency.

```
// Create a table using dbshell add
# mapr dbshell

maprdb root:> create /t1
Table /t1 created.


// Create an index using maprcli table index add
# maprcli table index add -path /t1 -index i1 -indexedfields a -json
{
"timestamp":1499788406380,
"timeofday":"2017-07-11 08:53:26.380 GMT-0700",
"status":"OK",
"total":0,
"data":[ ]
```

```
}


// Insert documents into the table using dbshell insert
# mapr dbshell

maprdb root:> insert /t1 --v {"a":1,"b":2} --id 1
Document with id: "1" inserted.

maprdb root:> insert /t1 --v {"a":"mapr","b":3} --id 2
Document with id: "2" inserted.

maprdb root:> insert /t1 --v {"a":{"$numberLong":3},"b":4} --id 4
Document with id: "4" inserted.


// Run verifyindex to verify indexed data
# mapr verifyindex -path /t1 -index i1

Number of rows in table but not in index: 0
Number of rows in index but not in table: 0
Mismatch row count: 0
```

Troubleshooting Use Cases

Situations where you can use this command are as follows:

- Examine details on updates that have not yet propagated from a JSON table to one of its indexes.
- Detect if there are documents that are missing from an index.
- Detect other data consistency issues between an index and its parent JSON table.

See Troubleshooting Secondary Indexes on page 1092 for more information on these use cases.

MapR Database HBase Shell (Binary Tables)

You can manage MapR Database tables using HBase shell commands and additional HBase shell commands included in the MapR Data Platform distribution of Hadoop.

The HBase shell command is used on binary tables only. To run this command, execute the following:

```
hbase shell
```

Note: Before running the shell, ensure that your user ID has both the `readAce` and `writeAce` permissions on the volume. For information about these permissions, see Managing Whole Volume ACEs.

When you specify a table in HBase shell, use the following syntax:

- For a table on the local cluster, start the path at the volume mount point. For example, for a table named test under a volume with a mount point at /volume1, specify the following path as the table name: “/volume1/test”
- For a table on a remote cluster, you must also specify the cluster name in the path. For example, for a table named customer under volume1 in the sanfrancisco cluster, specify the following path as the table name: “/mapr/sanfrancisco/volume1/customer”

Note: You can access a table on a remote cluster when the remote cluster has an entry in the mapr-clusters.conf file on the node where the HBase shell is running.

The following table lists the supported HBase shell commands that you can use to manage MapR Database tables:

Command	Description
alter	Performs the following actions on MapR Database tables: • Adds a new table or column family • Modifies the following table-level attributes: • • BULKLOAD - A Boolean value that specifies whether to perform a full bulk load of the table. The default is false. For more information, see Bulk Loading and MapR Database Tables. • MAX_FILESIZE • AUTOSPLIT - A Boolean value that specifies whether to split the table into regions automatically as the table grows. The average size of each region is determined by the `regionsizemb` parameter. The default value is true. If you set the value to false, you can manually split tables into regions by using the maprcli table region split command. • Modifies the following attributes of a column family: • • TTL • VERSIONS (max or min) • COMPRESSION • IN_MEMORY Aside from the attributes listed above, no other attributes apply to MapR Database tables. Unlike HBase, you do not need to disable a table before altering a table.
alter_async	On MapR Database tables, this has the same behavior as the alter command.
count	Counts the number of rows in a specified table.
create	Creates a table in the specified path.
delete	Deletes a value in a specified table, row, and column. Optionally, you can also specify the timestamp associated with the value that you want to delete.
deleteall	Delete all values in a row based on the table name and row. Optionally, you can also specify the timestamp associated with the values that you want to delete.
describe	Describes a specified table.
disable	Marks a specified table as disabled. This state is recorded only in the client process (HBase shell) memory and does not actually disable any operations on a MapR Database table.
disable_all	Marks tables as disabled if they have names matching the specified regular expression. This state is recorded only in the client process (HBase shell) memory and does not actually disable any operations on the MapR Database tables.

last-updated: Apr 11, 2022

Command	Description
drop	Drops a specified table that is marked as disabled.
drop_all	Drops all tables that are marked as disabled.
enable	Marks a specified table as enabled. This state is recorded only in the client process (HBase shell) memory.
enable_all	Marks tables as enabled if they have a table name that matches the specified regular expression. This state is recorded only in the client process (HBase shell) memory.
exists	Returns boolean value true if the specified table exists.
exit	Exits the HBase shell.
get	Gets the contents of a row or cell.
get_counter	Returns the value of a counter at a specified table, row, and column.
incr	Increments a value at a specified table, row, and column.
is_disabled	Returns a value that indicates if a specified table is disabled. You can perform operations on MapR Database tables that are disabled.
is_enabled	Returns a value that indicates if a specified table is enabled. You can perform operations on MapR Database tables even if they are not enabled.
list	For HBase 1.1 or above, if the mapr.hbase.default.db property is set to maprdb, this command returns the MapR Database tables under the user's home directory.
list_perm	Lists all permissions set by Access Control Expressions for a specified table. This HBase shell command only operates on MapR Database tables. For more information, see list_perm.
put	Puts a value at a specified table, row, and column. Optionally, you can also specify the timestamp for that value.
scan	Scans a specified table. Optionally, you can also specify a dictionary of scanner specifications.
set_perm	Sets permissions with Access Control Expressions on a specified table, column family, or column qualifier. This Hbase shell command only operates on MapR Database tables. For more information, see set_perm.
show_filters	Shows all the filters supported by the Hbase or MapR Database tables. Provide the link to 4.1 supported filters doc
truncate	Disables, drops, and recreates a specified table.
version	Returns the HBase client version.
whoami	Returns the current user.

MapR Database does not support the following HBase shell commands:

- add_peer
- alter_status

last-updated: Apr 11, 2022

- assign
- balance_switch
- balancer
- close_region
- compact
- disable_peer
- enable_peer
- flush
- grant
- hlog_roll
- list_peer
- major_compact
- move
- remove_peer
- start_replication
- stop_replication
- status
- split
- revoke
- unassign
- user_permission
- zk_dump

For more information about the HBase shell commands, see the Apache HBase documentation.

list_perm

Lists all permissions set by Access Control Expressions for a specified MapR Database table

Syntax

```
list_perm "<table path>"
```

Example

```
hbase(main):006:0> list_perm "/table/"
        Scope Permission Access Control Expression
        defaultappendperm u:jon
        createrenamefamilyperm u:jon
        deletefamilyperm u:jon
```

```
        bulkloadperm u:jon
        defaultreadperm u:jon
        defaultwriteperm u:jon
        packperm u:jon
        replperm u:jon
        defaultmemoryperm u:jon
        adminaccessperm u:jon
        splitmergeperm u:jon
        defaultversionperm u:jon
        defaultcompressionperm u:jonr
        13 row(s) in 0.0070 seconds
```

set_perm
Set permissions with access control expressions on a MapR Database table, column family, or column qualifier.

Set permissions with *ACE* on a MapR Database table, column family, or column qualifier.

Syntax

To set the permission on a table:

```
set_perm "<table path>", "<permission>", "<ACE expression>"
```

To set the permission on a column family or column qualifier:

```
set_perm "<table path>", {COLUMN => "column family[:qualifier]", PERM =>
          "<permission>", EXPR => "<ACE expression>"}
```

Examples

Assigns user `jon` and user `mapr04` the `defaultreadperm` permission on table `/table`:

```
hbase(main):004:0> set_perm "/table/","defaultreadperm","u:jon|u:mapr04"
```

Assigns user `jon` and user `mapr05` the `compressionperm` permission on the `cf1` column family in table `/table`:

```
hbase(main):005:0> set_perm "/table/",{COLUMN => "cf1",PERM =>
"compressionperm", EXPR =>"u:jon|u:mapr05"}
```

Assigns user `jon` and user `mapr05` the `writeperm` permission on the `col1` column qualifier in `cf1` column family in table `/table`:

```
hbase(main):009:0> set_perm "/table/",{COLUMN => "cf1:col1",PERM =>
"writeperm", EXPR =>"u:jon|u:mapr05"}
```

Utilities for MapR Database Binary Tables

MapR Database provides utilities to copy and compare data in MapR Database binary tables.

You can use the following utilities with MapR Database binary tables:

MapR Database Binary CopyTable

Copies data from one MapR Database binary table to another MapR Database binary table.

The MapR Database `CopyTable` utility is different from Apache HBase's `CopyTable` utility. This utility has the following capability:

last-updated: Apr 11, 2022

- If the destination table does not exist, `CopyTable` creates the target table with the same metadata (column families and access control expressions) as the source table, and then copies data.
- If the destination table exists, `CopyTable` copies data only.
- If you manually set up replication to a MapR Database table, `CopyTable` can be used to perform an initial load of source data to the replica before table replication begins.

Note: When copying data to MapR Database tables, it is recommended that you use the MapR Database version of `CopyTable`.

Required Permissions

The user that runs the `CopyTable` utility must have the following permissions:

- The permission `readAce` on the volume where the source table is located, and the permission `writeAce` on the volume where the destination table is or will be located.
- The permission `adminperm` on the source table.
- The permission for column-family and column reads (`readperm`) on the data in the source table that you want to copy.
- When bulkload = `false`, the permission for column writes (`writeperm`) on the destination table.
- When bulkload = `true` (default), the permission to load the destination table with bulk loads (`bulkloadperm`).
- If the destination table does not yet exist: `createrenamefamily` on the source table.

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

If `CopyTable` is run between tables on different clusters, the user that runs the command must have the required permissions on each cluster.

For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

For information about how to set permissions on tables, see Enabling Table and Stream Authorizations with ACEs on page 1461.

Syntax

```
hbase com.mapr.fs.hbase.tools.mapreduce.CopyTable
    -src <source table path> -dst <destination table path>
    [-columns cf1[:col1],...] [-maxversions <max number of versions to
copy>]
    [-starttime <time>]
    [-endtime <time>]
    [-mapreduce <true|false> (default: true)]
    [-bulkload <true|false> (default: true)]
    [-numthreads <numThreads> (default:16, valid only when -mapreduce is
false)]
```


last-updated: Apr 11, 2022

Parameters

Parameters	Description
src	The path to the source table that you want to replicate. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `testsrc` under a volume with a mount point at /volume1, specify the following path: `/volume1/testsrc` • For a path on another cluster, you must also specify the cluster name in the path. For example, for a table named `customersrc` under `volume1` in the `sanfrancisco` cluster, specify the following path: `/mapr/sanfrancisco/volume1/customersrc`
dst	The path to the replica. • For a table on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under a volume with a mount point at /volume1, specify the following path: `/volume1/testdst` • For a table on another cluster, you must also specify the cluster name in the path. For example, for a table named `customerdst` under `volume1` in the `sanfrancisco` cluster, specify the following path:`/mapr/sanfrancisco/volume1/customerdst`
columns	By default, all columns in the source table are copied. If you do not want to copy all columns in the table, you can specify columns to copy. Provide a comma-separated list of column families or columns from a certain column family (column family:qualifier). For example, use the following syntax to copy only the purchases column family and the stars column in the reviews column family: `-columns purchases,reviews:stars`
maxversions	By default, all versions from the source table are copied. If you do not want to copy all versions, use this parameter to specify the number of versions to copy.
starttime	By default, all table values regardless of their associated timestamp are copied. You can specify a timestamp to indicate the table cell version at which to start the copy. The timestamp is a long integer value that is either user-defined or system-defined (epoch in milliseconds). Values with timestamps lower than the specified timestamp will not be copied to the destination.
endtime	By default, all table values regardless of their associated timestamp are copied. You can specify a timestamp to indicate the table cell version at which to end the copy. The timestamp is a long integer value that is either user-defined or system-defined (epoch in milliseconds). Values with timestamps greater than or equal to the specified timestamp will not be copied to the destination.

last-updated: Apr 11, 2022

Parameters	Description
mapreduce	A Boolean value that specifies whether or not to use a MapReduce program to perform the copying operation. The default, preferred method is to use a MapReduce program (true). When this parameter is set to false, a client process uses multiple threads to read rows of the source table and write rows to the destination table.
bulkload	A Boolean value that specifies whether or not to perform a full bulk load of the table. The default is to use bulk loading (`true`). When you use bulk loading, the utility automatically unsets the bulk load mode on the table to restore normal client operations at the end of the table copy operation. For more information, see Bulk Loading and MapR Database Tables.
numthreads	When -mapreduce is false, this parameter specifies the number of threads allocated to perform the copying of data. The default is 16. If additional CPU resources are available, you might want to increase the number of threads to achieve better performance.

Monitoring the CopyTable Operation

Use one of the following methods to monitor the progress of the copying of table data:

- If the copy table operation runs as a MapReduce v2 application, monitor the application using the ResourceManager UI.
- If the copy table operation runs as a client process, go to the Tables view of the destination table in the MapR Control System. Then, on the Region tab, monitor the pace at which the number of rows increases.

Example
Copies table data with timestamp greater than 1423226300000 (Fri, 06 Feb 2015 12:38:20 GMT) from one MapR Database table to another MapR Database table:

```
[user@hostname ~]$
hbase com.mapr.fs.hbase.tools.mapreduce.CopyTable -src /t1 -dst /
t1_copy7  -starttime 1423226300000
```

MapR Database Binary DiffTables

Compares the row key, column family, timestamp, and value of each table cell in each specified MapR Database table. Then, it generates one or two directories with sequence files that you can use to either make a MapR Database table identical to its master or merge the rows from two MapR Database tables.

Sequence files are binary flat files. To convert the sequence file into a format that is easier to understand, use the FormatResults utility.

By default, the DiffTables utility considers both the source table and the destination table to be a master table. Therefore, it generates two directories with sequence files. These sequence files contain the puts required to update each table so that it contains a superset of the rows defined in both tables at the time at which the utility was run.

When you specify a master table, the DiffTables utility generates one of the following output directories:

- **opsForDst.** A directory containing sequence files that correspond to each put and delete required to make the destination table identical to the source table.

- **opsForSrc.** A directory containing sequence files that correspond to each put and delete required to make the source table identical to the destination table.

A user with write permissions on a table can run the `hbase org.apache.hadoop.hbase.mapreduce.Import` command to implement the puts and deletes specified in the sequence files.

Required Permissions

The user that runs the DiffTables utility must have the following permissions:

- The permission `readAce` on the volumes where the tables are located.
- The permission for column reads (`readperm`) on each table.

For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

For information about how to set permissions on tables, see Enabling Table and Stream Authorizations with ACEs on page 1461.

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

```
hbase com.mapr.fs.hbase.tools.mapreduce.DiffTables
  -src <source table path>
    -dst <destination table path>
      -outdir <output directory>
    [-master <src|dst> ] The master table to use for the diff.
    [-first_exit] Exit when first difference is found.
    [-columns <comma separated list of family[:column]> ]
    [-starttime <start diff at timestamp>]
    [-endtime <end diff at timestamp>]
    [-maxversions] <max number of versions to diff>
    [-mapreduce] <true|false> (default: true)]
    [-numthreads <numThreads> (default:16, valid only when -mapreduce is
      false)]
      [-cmpmeta <true|false> (default: true)]
```

Parameters

Parameter	Description
src	The path to the source table. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `testsrc` under a volume with a mount point at /volume1, specify the following path: `/volume1/testsrc` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named customersrc under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customersrc`

last-updated: Apr 11, 2022

Parameter	Description
dst	The path to the destination table. • For a table on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under a volume with a mount point at /volume1, specify the following path: /volume1/testdst • For a table on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customerdst` under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customerdst`
master	The table that is considered to be the master table. The values are `src` and `dst`. By default, both the source table and the destination tables are considered to be a master.
first_exit	By default, the utility compares all the table cells in the specified tables. Use this parameter if you want to exit after the first difference is identified between the tables. The parameter takes no value.
outdir	The path to a directory in which to place the generated sequence files. The utility creates the specified directory. If the specified directory already exists, the command fails.
columns	By default, the utility compares all columns. If you do not want to compare all columns in the table, you can specify specific columns to include in the comparison. Provide a comma-separated list of column families or columns from a certain column family (column family:qualifier). For example, use the following syntax to include the column family purchases and the column stars in the reviews column family: `-columns purchases,reviews:stars`
starttime	By default, the utility compares all table values regardless of their associated timestamp. You can specify a timestamp to indicate the table cell version at which to start the comparison. The timestamp is a long integer value that is either user-defined or system-defined (epoch in milliseconds). Values with timestamps lower than the specified timestamp will not be included in the comparison.
endtime	By default, the utility compares all table values regardless of their associated timestamp. Values with timestamps greater than or equal to the specified timestamp will not be included in the comparison.
maxversions	By default, all versions from the master table are included in the comparison. If you do not want to compare all versions, use this parameter to specify the number of recent versions to include in the comparison.
mapreduce	A Boolean value that specifies whether or not to use a MapReduce program to perform the comparison. The default, preferred method is to use a MapReduce program (true). When this parameter is set to false, a client process uses multiple threads.

last-updated: Apr 11, 2022

Parameter	Description
numthreads	When -mapreduce is false, this parameter specifies the number of threads allocated to perform the comparison. The default is 16. If additional CPU resources are available, you might want to increase the number of thread to achieve better performance.
cmpmeta	A Boolean value that specifies whether or not to compare table metadata such as column families and ACEs. The default is to compare metadata (true).

Examples
The following example compares two MapR Database tables:

```
[user@hostname ~]$ hbase com.mapr.fs.hbase.tools.mapreduce.DiffTables -src /
customerTableA -dst /customerTableB -outdir /customerTableABCompare
2015-03-04 18:04:52,059 INFO  [main] Configuration.deprecation:
hadoop.native.lib is deprecated. Instead, use io.native.lib.available
DiffTablesMeta completed. Metadata of the two tables is same.
...
Mapreduce job completed. The tables mismatch.
NUM_ROWS_MISMATCH_IN_SRC:32; NUM_ROWS_MISMATCH_IN_DST:30. Please check diff
in /customerTableABCompare
```

MapR Database Binary DiffTablesWithCrc

This utility uses a cyclic redundancy check to detect differences between sets of rows in the specified MapR Database binary tables. Then, for each set of non-identical rows, it performs a detailed comparison. Finally, it generates one or more directories of sequence files. You can use these files either to make a MapR Database binary table identical to its master or merge the rows from two MapR Database binary tables.

Sequence files are binary flat files. You can learn more about them here. To convert a sequence file into a format that you can read, use the MapR Database Binary FormatResult on page 5338 utility.

This utility requires less network bandwidth than the `DiffTables` utility because it performs a detailed table comparison only on the sets of rows where the CRC algorithm detected a difference. Therefore, consider using this utility when the tables you compare are very similar and you are concerned about the data transfer rate.

Requirements

- When the cluster runs YARN, it must also use zero configuration failover for the ResourceManager.

By default, the utility considers both the source table and the destination table to be a master table. Therefore, it generates two directories with sequence files. These sequence files contain the puts required to update each table so that it can contain a superset of the rows defined in both tables at the time at which the utility was run.

When you specify a master table, the `mapr difftableswithcrc` utility generates one of the following output directories:

- **opsForDst.** A directory containing sequence files that correspond to each put and delete required to make the destination table identical to the source table.
- **opsForSrc.** A directory containing sequence files that correspond to each put and delete required to make the source table identical to the destination table.

last-updated: Apr 11, 2022

A user with write permissions on a table can run the `hbase org.apache.hadoop.hbase.mapreduce.Import` command to implement the puts and deletes specified in the sequence files.

Required Permissions

The user that runs the `mapr difftableswithcrc` utility must have the following permissions:

- The permission `readAce` on the volumes where the tables are located.
- The permission for column reads (`readperm`) on each table.

For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

For information about how to set permissions on tables, see Enabling Table and Stream Authorizations with ACEs on page 1461.

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

```
hbase com.mapr.fs.hbase.tools.mapreduce.DiffTablesWithCrc
-src <source table path>
-dst <destination table path>
-outdir <output directory>
[-master src|dst ] The master table to use for the diff.
[-first_exit] Exit when first difference is found.
[-cf <comma separated list of column families>]
[-starttime <start diff at timestamp>]
[-endtime <end diff at timestamp>]
[-maxVersions <max number of versions to copy>]
[-cmpmeta <true|false> (default: true)]
```

Parameters

Parameter	Description
src	The path to the source table. • For a path on the local cluster, start the path at the volume mount point. For example, for a table named `testsrc` under a volume with a mount point at /volume1, specify the following path:`/volume1/testsrc` • For a path on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customersrc` under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customersrc`

last-updated: Apr 11, 2022

Parameter	Description
dst	The path to the destination table. • For a table on the local cluster, start the path at the volume mount point. For example, for a table named `testdst` under a volume with a mount point at /volume1, specify the following path: `/volume1/testdst` • For a table on a remote cluster, you must also specify the cluster name in the path. For example, for a table named `customerdst` under volume1 in the sanfrancisco cluster, specify the following path:`/mapr/sanfrancisco/volume1/customerdst`
master	The table that is considered to be the master table. The values are `src` (the source table) and `dst` (the destination table). By default, both the source table and the destination table are considered to be the master.
first_exit	By default, the utility compares all the table cells in the specified tables. Set this parameter if you want to exit after the first difference is identified between the tables.
outdir	The path to a directory for the sequence files. The utility will create the specified directory. If the specified directory already exists, the command will fail.
cf	By default, the utility compares all columns from the master table. If you do not want to compare all columns in the table, you can specify specific columns to include in the comparison. Provide a comma-separated list of column families or columns from a certain column family (column family:qualifier). For example, use the following syntax to include the column family purchases and the column stars in the reviews column family: `-columns purchases,reviews:stars`
starttime	By default, the utility compares all table values regardless of their associated timestamp. You can specify a timestamp to indicate the table cell version at which to start the comparison. The timestamp is a long integer value that is either user-defined or system-defined (epoch in milliseconds). Values with timestamps lower than the specified timestamp will not be included in the comparison.
endtime	By default, the utility compares all table values regardless of their associated timestamp. You can specify a timestamp to indicate the table cell version at which to end the comparison. The timestamp is a long integer value that is either user-defined or system-defined (epoch in milliseconds). Values with timestamps greater than or equal to the specified timestamp will not be included in the comparison.
maxVersions	By default, the utility compares all versions from the master table. If you do not want to diff all versions, use this parameter to specify the number of recent versions to include in the comparison.
cmpmeta	A Boolean value that specifies whether or not to compare table metadata such as column families and ACEs. The default is to compare metadata (`true`).

last-updated: Apr 11, 2022

Example
Compares two MapR Database tables:

```
[user@hostname ~]$
hbase com.mapr.fs.hbase.tools.mapreduce.DiffTablesWithCrc -src /
customerTableA -dst /customerTableB -outdir /customerTableCompare
2015-03-04 17:52:40,912 INFO  [main] Configuration.deprecation:
hadoop.native.lib is deprecated. Instead, use io.native.lib.available
DiffTablesMeta completed. Metadata of the two tables is same.
....
Mapreduce job completed. The tables mismatch.
NUM_ROWS_MISMATCH_IN_SRC:32; NUM_ROWS_MISMATCH_IN_DST:30. Please check diff
in /customerTableCompare
```

MapR Database Binary FormatResult

Parses a sequence file generated by the `DiffTables` utility or the `DiffTablesWithCrc` utility and converts the results into a format that makes the results easier to understand.

Required Permissions
The user that runs the `FormatResult` utility must have the `readAce` and `writeAce` permissions on the volumes where the input and output paths are located.

For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

Note: The `mapr` user is not treated as a superuser. MapR Database does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

```
hbase com.mapr.fs.hbase.tools.mapreduce.FormatResult
-input <input file path>
-output <output file path>
[-mapreduce <true|false> (default: false)]
```

Parameters

Parameters	Description
input	The path to a file or directory of files that contains the output of either the DiffTables utility or the DiffTablesWithCrc utility.
output	The path to a file or a directory for the output. If the file or directory already exists, the utility fails. When a single sequence file is provided as input, the utility generates a single output file. When a directory of sequence files is provided as input, the utility generates a directory with output files.
mapreduce	A Boolean value that specifies whether or not to use a MapReduce program to perform the FormatResult operation. The default is not to use a MapReduce program (false).

Example
Formats a sequence file:

```
[user@hostname ~]$ hbase
com.mapr.fs.hbase.tools.mapreduce.FormatResult -input /dif1/tf4/opsForDst/
opsForDst-m-00001 -output /dif1/tf4/opsForDst_single/nomr -mapreduce false
2015-03-06 18:58:56,210 INFO  [main] Configuration.deprecation:
fs.default.name is deprecated. Instead, use fs.defaultFS
2015-03-06 18:58:57,492 INFO  [main] mapreduce.FormatResult: Translated
sequence file maprfs:///dif1/tf4/opsForDst/opsForDst-m-00001 to text file /
dif1/tf4/opsForDst_single/nomr
2015-03-06 18:58:57,527 INFO  [main] mapreduce.FormatResult: Total 1
text files created.
```

MapR Event Store For Apache Kafka Utilities

You can use the following utilities to with MapR Event Store For Apache Kafka streams:

`mapr copystream`
This utility copies data from one MapR Stream to another MapR Stream. You can use it, for example, if you want to set up replication manually from one stream to another.

If the destination stream does not exist, `mapr copystream` creates the destination stream with the same metadata as the source stream, and then copies data.

If the destination stream exists, `mapr copystream` copies data only.

Requried Permissions

To use this utility, you must have the following permissions:

- The permission `readAce` on the volume where the source stream is located, and the permission `writeAce` on the volume where the destination stream is located.
- On the source stream: either `consumeperm` or `copyperm`.
- On the destination stream: either `copyperm` or all three of the following permissions: `produceperm`, `consumeperm`, `topicperm`

For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

For information about how to set permissions on streams, see Enabling Table and Stream Authorizations with ACEs on page 1461.

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

```
mapr copystream
-src <srcStream>
-dst <dstStream>
[-mapreduce true/false default:false]
[-numthreads <nthreads> default:16]
```


last-updated: Apr 11, 2022

Parameters

Parameter	Description
src	The path and name of the stream to copy messages from.
dst	The path and name of the stream to copy messages to.
mapreduce	A Boolean value that specifies whether or not to use a MapReduce program to perform the copying operation. The default, preferred method is to use a MapReduce program (true). When this parameter is set to false, a client process uses multiple threads to read from the source stream and write to the destination stream. The MapReduce program runs as a MapReduce version 2 application based on the MapReduce mode that is configured on this node.
numthreads	When -mapreduce is false, this parameter specifies the number of threads allocated to perform the copying of data. The default is 16. If additional CPU resources are available, you might want to increase the number of threads to achieve better performance.

`mapr diffstreams`

This utility compares the message IDs, metadata, and data in two MapR Streams. Then, generates two directories that contain sequence files that you can use to merge the rows from the two MapR Streams.

Sequence files are binary flat files. You can learn more about them here. To convert a sequence file into a format that you can read, use the MapR Database JSON FormatResult on page 5318 utility.

This utility considers both the source stream and the destination stream to be a master stream. Therefore, it generates two directories with sequence files. These sequence files contain the puts required to update each stream so that it contains a superset of the rows defined in both tables at the time at which the utility was run.

This utility generates both of the following output directories in the output directory that you specify:

opsForDst	A directory containing sequence files that correspond to each put and delete required to make the destination stream identical to the source stream.
opsForSrc	A directory containing sequence files that correspond to each put and delete required to make the source stream identical to the destination stream.

Required Permissions

To use this utility, you must have the following permissions:

- The permission `readAce` on the volumes where the tables are located.
- On the source stream: either `consumeperm` or `copyperm`.
- On the destination stream: either `consumeperm` or `copyperm`.

For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

For information about how to set permissions on streams, see Enabling Table and Stream Authorizations with ACEs on page 1461.

last-updated: Apr 11, 2022

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

```
mapr diffstreams
-src <srcStream>
-dst <dstStream>
-outdir <output directory>
[-first_exit] Exit when first difference is found
[-mapreduce true/false default:false]
[-numthreads <nthreads> default:16]
```

Parameters

Parameter	Description
src	The path of the first stream to include in the comparison.
dst	The path of the second stream to include in the comparison.
outdir	The path to a directory in which to place the generated sequence files. The utility creates the specified directory. If the specified directory already exists, the command fails.
first_exit	By default, the utility compares all the data in the specified streams. Use this parameter if you want to exit after the first difference is identified between the streams. The parameter takes no value.
mapreduce	A Boolean value that specifies whether or not to use a MapReduce program to perform the comparison. The default, preferred method is to use a MapReduce program (true). When this parameter is set to false, a client process uses multiple threads. The MapReduce program runs as a MapReduce version 2 application based on the MapReduce mode that is configured on this node.
numthreads	When -mapreduce is false, this parameter specifies the number of threads allocated to perform the comparison. The default is 16. If additional CPU resources are available, you might want to increase the number of thread to achieve better performance.

`mapr diffstreamswithcrc`

This utility uses a cyclic redundancy check to detect differences between sets of messages in the specified MapR Streams. Then, for each set of non-identical messages, it performs a detailed comparison. Finally, it generates one or more directories of sequence files.

You can use these files either to make a MapR Stream identical to its master or merge the messages from two MapR Streams.

Sequence files are binary flat files. You can learn more about them here. To convert a sequence file into a format that you can read, use the MapR Database JSON FormatResult on page 5318 utility.

last-updated: Apr 11, 2022

This utility requires less network bandwidth than the `mapr diffstreams` utility because it performs a detailed table comparison only on the sets of messages where the CRC algorithm detected a difference. Therefore, consider using this utility when the streams you compare are very similar and you are concerned about the data transfer rate.

This utility considers both the source stream and the destination stream to be a master stream. Therefore, it generates two directories with sequence files. These sequence files contain the puts required to update each stream so that each stream can contain a superset of the messages in both streams at the time at which the utility was run.

These are the directories that the utility generates:

opsForDst	A directory containing sequence files that correspond to each put and delete required to make the destination stream identical to the source stream.
opsForSrc	A directory containing sequence files that correspond to each put and delete required to make the source stream identical to the destination stream.

Requirements

- When the cluster runs YARN, it must also use zero configuration failover for the ResourceManager.
- To use this utility, you must have the following permissions:
  - The permission `readAce` on the volumes where the tables are located.
  - On the source stream: either `consumeperm` or `copyperm`.
  - On the destination stream: either `consumeperm` or `copyperm`.

  For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

  For information about how to set permissions on streams, see Enabling Table and Stream Authorizations with ACEs on page 1461.

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

Run the `mapr importstream` command to implement the puts and deletes specified in the sequence files.

Syntax

```
mapr diffstreamswithcrc
-src <srcStream>
-dst <dstStream>
-outdir <output directory>
[-first_exit] Exit when first difference is found
```

Parameters

Parameter	Description
src	The path of the first stream to include in the comparison.
dst	The path of the second stream to include in the comparison.

last-updated: Apr 11, 2022

Parameter	Description
outdir	The path to a directory in which to place the generated sequence files. The utility creates the specified directory. If the specified directory already exists, the command fails.
first_exit	By default, the utility compares all the data in the specified streams. Use this parameter if you want to exit after the first difference is identified between the streams. The parameter takes no value.

`mapr exportstream` and `mapr importstream`

Use these utilities together to export data from MapR Streams into binary sequence files, and then import the data from the binary sequence files into other MapR Streams. You can also use the `mapr importstream` utility to import changes that are specified in sequence files output by the `mapr diffstreams` utility.

- Syntax of `mapr exportstream`
- Parameters of `mapr exportstream`
- Syntax of `mapr importstream`
- Parameters of `mapr importstream`

Required Permissions

To use the `mapr exportstream` utility, you must have the following permissions:

- The `readAce` permission on the volume where the source stream for `mapr exportstream` is located.
- The `writeAce` permission on the volume in which to save the output from `mapr exportstream`.
- On the source stream: either `consumeperm` or `copyperm`
- On the destination directory: write permission

To use the `mapr importstream` utility, you must have the following permissions:

- The `readAce` permission on the volume where the files output by `mapr exportstream` is located.
- The `writeAce` permission on the volume in which the destination stream is located.
- On the source directory: read permission on the directory and all of the files within it
- On the destination stream: either `copyperm` or all three of the following permissions: `produceperm`, `consumeperm`, `topicperm`

For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

For information about how to set permissions on streams, see Enabling Table and Stream Authorizations with ACEs on page 1461.

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run these utilities unless that user is given the relevant permission or permissions with access-control expressions.

last-updated: Apr 11, 2022

Syntax of `mapr exportstream`

```
mapr exportstream
-src <srcStream>
-dst <dstDir>
[-mapreduce true/false default:false]
```

Parameters of `mapr exportstream`

Description	Parameter
src	The stream to export data from.
dst	The directory within the MapR filesystem to export the files to. This directory must already exist before you run the utility.
mapreduce	A Boolean value that specifies whether or not to use a MapReduce program to perform the operation. The default, preferred method is to use a MapReduce program (true). When this parameter is set to false, a client process uses multiple threads.

Syntax of `mapr importstream`

```
mapr importstream
-src Input binary file or directory path
-dst Destination stream
[-mapreduce true/false default:false]
```

Parameters of `mapr importstream`

Description	Parameter

src	The path of the binary file or files to import.
	Examples
	`-src /temp/part0`
	`-src /temp/*`

dst	The stream to import data into.
mapreduce	A Boolean value that specifies whether or not to use a MapReduce program to perform the operation. The default, preferred method is to use a MapReduce program (true). When this parameter is set to false, a client process uses multiple threads.

`mapr perfconsumer`

This utility runs a consumer reading messages from topics in a MapR Stream. Use this utility to run consumers when you want to estimate the performance of consumers for your MapR Streams applications, given your network configuration.

This utility works in conjuction with the `mapr perfproducer` utility. When starting this utility, you can specify how many topics to read from, how many partitions to read from in each topic, and how many messages to read.

last-updated: Apr 11, 2022

The `mapr perfconsumer` utility uses the default values for all of the configuration parameters that apply to consumers. For a list of these parameters, see MapR Event Store For Apache Kafka Configuration Parameters.

The utility uses the default values for all of the configuration parameters that apply to consumers.

Each consumer runs as a single thread. You can run multiple instances of the utility at the same time. However, because consumers can be CPU-intensive, it is recommended to run at most 4 or 5 on a single cluster node.

When you run multiple instances of this utility, you can use the `-group` parameter to create consumer groups.

Monitor the performance of the running instances of the `mapr perfconsumer` utility by following the instructions that are given in Monitoring Consumers.

Prerequisites for running this utility

- Ensure that there is a MapR Stream that one or more instances of `mapr perfproducer` have already published messages to or are actively publishing messages to.
- Ensure that the user ID that runs the `mapr perfconsumer` utility has the `consumeperm` permission on the stream.
- Ensure that the user ID that runs the `mapr perfconsumer` utility has the `readAce` and `writeAce` permissions on the volume where the stream is located.

For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

For information about how to set permissions on streams, see Enabling Table and Stream Authorizations with ACEs on page 1461.

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

```
mapr perfconsumer
-path <stream-full-name>
[ -ntopics <num topics> (default: 2) ]
[ -npart <numpartitions per topic> (default: 4)
[ -nmsgs <num messages per topicfeed> (default: 100000) ]
[ -group <consumer group id> (default: null)
[ -topicsubscription <true/false> (default: false) ]
```

Parameters

Parameter	Description
path	The path to the stream.
ntopics	The number of topics for the consumer to subscribe to. The default is 2. If the number that you specify is greater than the number of topics that are in the stream, the utility hangs.

last-updated: Apr 11, 2022

Parameter	Description
npart	The number of partitions to read from in each topic that is subscribed. The default is 4. If the number that you specify is greater than the number of partitions that are in each topic, the utility hangs. If you specify a group ID with the `-group` parameter, the consumer's committed cursors are saved. If you do not specify a group ID, then the consumer's committed cursors are not saved. If you use the `-group` parameter to specify a group ID and you set the `-topicsubscription` parameter to `true`: • If the consumer fails, it's partitions can be redistributed among other consumer in the same group. If the consumer is the only consumer within a group, restarting the consumer with the same group ID causes the consumer to begin reading from the offsets of the saved committed cursors. • If the consumer fails and is then restarted, it starts at the oldest message in each partition.
nmsgs	The number of messages to read from each partition. The default is 100,000.
group	The identifier of a consumer group. When two or more consumers belong to a consumer group, they must read from the same number of topics. MapR Event Store For Apache Kafka distributes the partitions for those topics among the consumers in the group. The default is null.
topicsubscription	A vaue of `true` subscribes the consumer to topics. A value of `false` subscribes the consumer to topic partitions. The default is `false`.

`mapr perfproducer`

This utility runs a producer, generating messages and publishing them to a MapR Stream. Use this utility to run producers when you want to estimate the performance of producers for your MapR Streams applications, given your network configuration.

This utility starts a producer and generates data for the producer to publish in messages to a MapR Stream. When starting the utility, you can specify how many topics the producer publishes to, how many partitions to create for each topic, and how many messages to publish to each partition. You can also specify the method for distributing messages among the partitions in each topic.

For example, suppose you run the utility by issuing this command:

```
mapr perfproducer -path /myVolume/myDirectory/stream_a -ntopics
40 -npart -5
-nmsgs 100000 -rr true
```

The producer automatically creates 40 topics in the stream, creating each topic as it writes the first message to that topic. Each topic is created with 5 partitions. The producer writes 100,000 messages

last-updated: Apr 11, 2022

to each partition for a total of 20,000,000 messages. After publishing all of the messages, the utility terminates.

The `mapr perfproducer` utility uses the default values for all of the configuration parameters that apply to producers. For a list of these parameters, see MapR Event Store For Apache Kafka Configuration Parameters.

Each producer runs as a single thread. You can run multiple instances of the utility at the same time. However, because producers can be CPU-intensive, it is recommended to run at most 4 or 5 on a single cluster node.

When multiple instances of the `mapr perfproducer` utility publish to a single stream, the separate instances share topics. For example, if -ntopics is set to 40 for each instance that publishes to a single stream, together those instances create no more than 40 topics in the stream and they share those topics.

It is recommended that all producers that publish to a single cluster publish to the same number of topics and partitions within those topics. Therefore, use the same values for -ntopics and -npart for each instance of the `mapr perfproducer` utility that shares a stream with other instances.

Monitor the performance of the running instances of the `mapr perfproducer` utility by issuing the `maprcli` command `stream topic info` at intervals, as described in Monitoring Producers. The command `stream topic info` shows statistics for single topics. Because all of the topics that `mapr perfproducer` creates have the same number of partitions, and because `mapr perfproducer` writes the same number of messages to each partition, you can assume that the statistics that the command `stream topic info` displays for any one topic are close to the statistics for any other topic. The naming convention that `mapr perfproducer` uses when creating topics is simply `topic`*n*, which produces the names `topic0`, `topic1`, and so on. You can run the command `stream topic info` with any one of these names as the value of the `-topic` parameter.

To simulate consumers to estimate the performance of MapR Event Store For Apache Kafka in your network configuration, run one or more instances of the `mapr perfconsumer` utility against the stream.

Prerequisites for running this utility

- Create a MapR Stream in a MapR cluster for the `mapr perfproducer` utility to publish messages to. See stream create on page 1758.

  If you plan to replicate the stream for the purposes of the performance estimate, create the replica stream. Then, start replication from the first stream to the replica stream. For instructions on setting up replication between streams, see Managing Stream Replication on page 1130.

- Ensure that the user ID that runs the `mapr perfproducer` utility has the `readAce` and `writeAce` permissions on the volume where the stream is located.

- Ensure that the user ID that runs the `mapr perfproducer` utility has the `produceperm` and `topicperm` permissions on the stream.

For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

For information about how to set permissions on streams, see Enabling Table and Stream Authorizations with ACEs on page 1461.

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

```
mapr perfproducer
-path <stream-full-name>
```


last-updated: Apr 11, 2022

```
[ -ntopics <num topics> (default: 2) ]
[ -npart <numpartitions per topic> (default: 4) ]
[ -nmsgs <num messages per topicfeed> (default: 100000) ]
[ -msgsz <msg value size> (default: 200) ]
[ -rr <round robin true/false> (default: false) ]
[ -hashkey <true/false> (default: false) ]
```

Parameters

Parameter	Description
path	The path to the stream.
ntopics	The number of topics for the producer to publish to in the stream. The default is 2.
npart	The number of partitions to create for each topic. The default is 4.
nmsgs	The number of messages to publish to each partition. The default is 100,000.
msgz	The size of each message in bytes. The default is 200 bytes.
rr	Specifies to publish messages to partitions within a topic in round-robin fashion. See How Partitions are Chosen for Messages for detail about this method of distributing messages among topic partitions. **Note:** This parameter is incompatible with the `-hashkey` parameter. You must set one or the other to true, but not both. The default is `false`.
hashkey	Specifies to distribute messages among topic partitions according to the hash of each message key. See How Partitions are Chosen for Messages for detail about this method of distributing messages among topic partitions. **Note:** This parameter is incompatible with the `-rr` parameter. You must set one or the other to true, but not both. The default is `true`.

`mapr streamanalyzer`

This light-weight utility, which is a sample application for the `Streams` Java class for analytics on MapR Streams, lets you count the messages in a stream or a subset of the topics in a stream. The utility also lets you print either whole retrieved messages or a subset of the fields in each message.

You can download the source code for this utility here: `StreamAnalyzer.java`

For information about the `Streams` Java class and building applications that use it, see MapR Event Store For Apache Kafka Java API Library on page 2756. See Logical Schema of Messages on page 635 for information about how messages are structured.

last-updated: Apr 11, 2022

Ensure that the user ID that runs the utility has the `readAce` permission on the volume where the stream is located. For information about how to set permissions on volumes, see Setting Whole Volume ACEs on page 1459.

Note: The `mapr` user is not treated as a superuser. MapR Event Store For Apache Kafka does not allow the `mapr` user to run this utility unless that user is given the relevant permission or permissions with access-control expressions.

Syntax

```
mapr streamanalyzer -path <stream-full-name>
[ -topics <comma separated topic names> ]
[ -regex  <regular expression representing topic names> ]
[ -countMessages <true/false> (default: true) ]
[ -printMessages <true/false> (default: false) ]
[ -projectFields <comma separated field names> (default: all fields) ]
```

Parameters

Parameter	Description
`path`	The path and name of the stream.
`topics`	A comma-separated list of the names of topics to retrieve. If you do not specify this parameter or the `-regex` parameter, all of the topics in the stream are retrieved. Do not use this parameter if you use the `-regex` parameter.
`regex`	A regular expression that represents the names of the topics to retrieve. If you do not specify this parameter or the `-topics` parameter, all of the topics in the stream are retrieved. Do not use this parameter if you use the `-topics` parameter.
`countMessages`	Prints the number of retrieved messages to the standard output.
`printMessages`	Prints the contents of retrieved messages to the standard output.
`projectFields`	If the `-printMessages` parameter is set to `true`, this parameter causes only the specified fields to be printed to the standard output for each message. In the list of field names, separate the names with commas. Default: all fields. Valid field names: key, value, topic, offset, partition, and producer. If the `-printMessages` parameter is set to `false`, this parameter has no effect.

YARN Commands

This section describes the YARN commands.

last-updated: Apr 11, 2022

Commands

All YARN commands are invoked by the `/usr/bin/yarn` script.

```
Usage: yarn [--config confdir] [COMMAND] [COMMAND_OPTIONS]
```

COMMAND_OPTION	Description
`--config confdir`	Overrides the default Configuration directory. Default is `${HADOOP_HOME}/conf`.
`COMMAND`	Commands
`COMMAND_OPTIONS`	Command options

The following `yarn` commands may be run on the MapR Data Platform distribution of Apache Hadoop:

Command	Description
`application`	Lists applications, or prints the status or kills the specified application.
`classpath`	Prints the class path needed to get the Hadoop jar and the required libraries
`debugcontrol`	Saves additional DEBUG logs for scheduling to a separate file without restarting the RM
`daemonlog`	Gets and sets the log level for each daemon
`jar`	Runs jar file
`logs`	Dumps container logs
`node`	Prints node report(s)
`queue`	Prints queue information
`rmadmin`	Performs administrative tasks for Resource Manager
`version`	Print the version

The following yarn commands are not supported on the MapR Data Platform distribution of Apache Hadoop:

- `yarn applicationattempt`
- `yarn cluster`
- `yarn container`
- `yarn nodemanager`
- `yarn proxyserver`
- `yarn resourcemanager`
- `yarn sharedcachemanager`
- `yarn scmadmin`
- `yarn timelineserver`

last-updated: Apr 11, 2022

You can use the maprcli node services command or the Control System to start the services. For more information, see Managing Services on page 827.

yarn application
The `yarn application` lists applications, or prints the status or kills the specified application.

Syntax

```
yarn application
   [-list [<-appStates States>] [<-appTypes Types>] ]
   [-status ApplicationId]
   [-kill ApplicationId]
```

Parameters

The following commands parameters are supported for `yarn application`:

Parameter	Description
-list [<-appStates States>] [<-appTypes Types>]	Lists applications. Optionally, you can filter the applications based on type or state. • Use `-appTypes` to filer applications based on a comma-separated list of application types. • Use`-appStates` to filter applications based on a comma-separated list of the following valid application states:ALL, NEW, NEW_SAVING, SUBMITTED, ACCEPTED, RUNNING, FINISHED, FAILED, KILLED
-status ApplicationId	Prints the status of the application.
-kill ApplicationId	Kills the application.

yarn classpath

The `yarn classpath` command prints the class path needed to access the Hadoop jar and the required libraries.

Syntax

```
yarn classpath
```

Output

```
$ yarn classpath
/opt/mapr/hadoop/hadoop-<version>/etc/hadoop:/opt/mapr/hadoop/
hadoop-<version>/etc/hadoop:/opt/mapr/hadoop/hadoop-<version>/etc/
hadoop:/opt/mapr/hadoop/hadoop-<version>/share/hadoop/common/lib/*:
/opt/mapr/hadoop/hadoop-<version>/share/hadoop/common/*:
/opt/mapr/hadoop/hadoop-<version>/share/hadoop/hdfs:
/opt/mapr/hadoop/hadoop-<version>/share/hadoop/hdfs/lib/*:
/opt/mapr/hadoop/hadoop-<version>/share/hadoop/hdfs/*:
/opt/mapr/hadoop/hadoop-<version>/share/hadoop/yarn/lib/*:
/opt/mapr/hadoop/hadoop-<version>/share/hadoop/yarn/*:
/opt/mapr/hadoop/hadoop-<version>/share/hadoop/mapreduce/lib/*:
/opt/mapr/hadoop/hadoop-<version>/share/hadoop/mapreduce/*:
/contrib/capacity-scheduler/*.jar:
```


last-updated: Apr 11, 2022

```
/opt/mapr/hadoop/hadoop-<version>/share/hadoop/yarn/*:
/opt/mapr/hadoop/hadoop-<version>/share/hadoop/yarn/lib/*
```

yarn daemonlog

Gets or sets the log level for each daemon.

Syntax

```
yarn daemonlog
[-getlevel <host:port> <name>] | [-setlevel <host:port> <name> <level>]
```

Parameters

Parameter	Description
-getlevel <host:port> <name>	Prints the log level of the daemon running at <host:port>. This command internally connects to `http://<host:port>/logLevel?log=<name>`.
-setlevel <host:port> <name> <level>	Sets the log level of the daemon running at <host:port>. This command internally connects to `http://<host:port>/logLevel?log=<name>`.

yarn debugcontrol
used to save additional DEBUG logs for scheduling in YARN to a separate file without restarting the RM.

The logs are saved in /opt/mapr/Hadoop/Hadoop<version>/logs/yarn-mapr-scheduling-debug.log

Syntax

```
yarn debugcontrol
[-addapp <application_name>] | [-addqueue <queue_name> ]| [-removeapp
<application_name> ] | [-removequeue <queue_name> ] | [-getapps ] |
[-getqueues ]
```

Parameters

Parameter	Description
-addapp <application_name>	enables additional scheduling DEBUG on the application
-addqueue <queue_name>	enables additional scheduling DEBUG on the queue
-removeapp <application_name>	disable addition scheduling DEBUG on the application
-removequeue <queue_name>	disable addition scheduling DEBUG on the queue
-getapps	lists the applications with additional scheduling DEBUG
-getqueues	lists the queues with additional scheduling DEBUG

yarn jar

Runs a jar file that contains YARN code.

Syntax

```
yarn jar <jar> [<mainClass>] [<arguments>]
```

Parameters

The following commands parameters are supported for `yarn jar`:

Parameter	Description
`<jar>`	The JAR file.
<mainClass>	Sets the applications entry point.
`<arguments>`	Arguments to the program specified in the JAR file.

yarn logs
Dumps the YARN container logs.

Syntax

```
yarn logs -applicationId <application ID> [OPTIONS]

general options are:
 -appOwner <Application Owner>   AppOwner (assumed to be current user if
                                 not specified)
 -containerId <Container ID>     ContainerId (must be specified if node
                                 address is specified)
 -help                           Displays help for all commands.
 -nodeAddress <Node Address>     NodeAddress in the format nodename:port
                                 (must be specified if container id is
                                 specified)
```

Parameters

Parameter	Description
-applicationId	Specifies an application ID.
-appOwner <Application Owner>	Specifies the application owner. Defaults to the current user if this option is not specified.
-containerId <Container ID>	Specifies the container ID. Required when `-nodeAddress` is specified.
-nodeAddress <Node Address>	Specifies the node address in the following format: `nodename:port`. Required when `-containerId` is specified.

yarn node

Prints node report(s)

Syntax

```
yarn node
     [-list [-states <States>] | [-all]]
     [-status NodeId]
```

Parameters

Parameter	Description
-list [-states <states>] | [-all]]	Lists all running nodes. Optionally, filter nodes based on state or choose to list all the nodes. • Use `-states <states>` to filter nodes based on a comma-separate list of node states. • Use `-all` to list all nodes.
-status NodeId	Prints the status report of the node.

yarn queue

Prints queue information

Syntax

```
yarn queue -status <queue name>
```

Parameters

The following command parameter is supported for `yarn queue`:

Parameter	Description
`status`	The queue name.

yarn rmadmin

Runs the ResourceManager admin client.

Syntax

```
yarn rmadmin
    [-refreshQueues]
    [-refreshNodes]
    [-refreshUserToGroupsMapping]
    [-refreshSuperUserGroupsConfiguration]
    [-refreshAdminAcls]
    [-refreshServiceAcl]
    [-getGroups <username>]
    [-help <cmd>]
    [-transitionToActive <serviceId>]
    [-transitionToStandby <serviceId>]
    [-getServiceState <serviceId>]
    [-checkHealth <serviceId>]
```

Parameters

Parameter	Description
-refreshQueues	Reloads the queues' acls, states, and scheduler specific properties. The ResourceManager reloads the mapred-queues configuration file.
-refreshNodes	Refreshes the host information at the ResourceManager.
-refreshUserToGroupsMappings	Refreshes user-to-groups mappings.

last-updated: Apr 11, 2022

Parameter	Description
-refreshSuperUserGroupsConfiguration	Refreshes superuser proxy groups mappings.
-refreshAdminAcls	Refreshes acls for administration of ResourceManager.
-refreshServiceAcl	Reloads the service-level authorization policy file. The ResourceManager reloads the authorization policy file.
-getGroups <username>	Gets the groups that the user belongs to.
-help <cmd>	Displays help for the given parameter or all parameters if no parameter is specified.
-transitionToActive <serviceId>	Transitions the service into the Active state. You can use this parameter when the ResourceManager is configured to failover manually.
-transitionToStandby <serviceId>	Transitions the service into Standby state. You can use this parameter when the ResourceManager is configured to failover manually.
-getServiceState <serviceId>	Returns the service state. You can use this parameter when the ResourceManager is configured to failover manually or automatically but not with the zero configuration failover option.
-checkHealth <serviceId>	Requests a health check for the service. If the health check fails, the RMAdmin tool exits with a non-zero exit code. You can use this parameter when the ResourceManager is configured to failover manually or automatically but not with the zero configuration failover option.

yarn version

Prints the YARN version.

Syntax

```
yarn version
```

Output

```
$ yarn version
Hadoop 2.7.6.100-eep-800
Subversion git@github.com:mapr/private-hadoop-common -r
80dc89ae5df3a2cd01089f192c5d8a886e4788c9
Compiled by root on 2021-10-08T11:26Z
Compiled with protoc 3.11.1
From source with checksum 124ac1b54c81145154c71d2be2a66fc
This command was run using /opt/mapr/hadoop/hadoop-2.7.6/share/hadoop/
common/hadoop-common-2.7.6.100-eep-800.jar
```

Source Code for MapR Software

MapR releases source code to the open-source community for enhancements that HPE has made to the Apache Hadoop project and other ecosystem components.

HPE regularly releases updates to Apache Hadoop ecosystem projects as the projects are released by Apache, after HPE can verify that the changes do not impact product stability. Releases of ecosystem components are independent of the release cycle for the core MapR software, so that new updates can be released quickly and efficiently.

last-updated: Apr 11, 2022

Source code developed by HPE can be found on GitHub at http://github.com/mapr as of March 2013, coincident with version 2.1.2 of the MapR distribution. HPE may also release source code for other MapR projects at github.com/mapr. For each release that HPE includes in its distribution, HPE branches and tags the release on GitHub using the underlying project release number appended by `-mapr`.

Component Repositories on GitHub

The following repositories are available on GitHub for components that HPE has enhanced, patched, or created.

- oozie
- hcatalog
- pig
- hive
- mahout
- hbase
- flume
- whirr
- opentsdb

  Note: Select the highest MEP version supported by the MapR version that you are using.
- sqoop
- scribe

Finding Source Changes Prior to February 2013

GitHub is the single, central location for tracking changes that HPE applies to components in releases of the MapR distribution. Prior to February 2013, HPE included a list of patches in each component directory, as shown below. This information is no longer stored in the installation directory for recent releases, and instead is available at GitHub.

Example: Location of Information about Patches to HBase Prior to February 2012

```
$ ls /opt/mapr/hbase/hbase-0.92.1/
bin          hbase-0.92.1.jar        LICENSE.txt         pom.xml
CHANGES.txt  hbase-0.92.1-tests.jar  logs                README.txt
conf         hbase-webapps           mapr-hbase-patches  security
conf.new     lib                     NOTICE.txt          src

$ ls /opt/mapr/hbase/hbase-0.92.1/mapr-hbase-patches/
0000-hbase-with-mapr.patch                  0006-hbase-6285-fix.patch
0001-hbase-wait-for-fs+set-chunksize.patch  0007-hbase-6375-fix.patch
0002-hbase-source-env-vars.patch            0008-hbase-6455-fix.patch
0003-hbase-6158-fix.patch                   0009-bug-7745-fix.patch
0004-hbase-6018-fix.patch                   Readme.txt
0005-hbase-6236-fix.patch
```

Hadoop Commands

This section describes the Hadoop commands.

last-updated: Apr 11, 2022

All Hadoop commands are invoked by the `bin/hadoop` script.

Overview

This section contains the following:

Syntax Summary

The following syntax summary applies to all commands.

```
hadoop [--config confdir] [COMMAND] [GENERIC_OPTIONS] [COMMAND_OPTIONS]

hadoop2 [--config confdir] [COMMAND] [GENERIC_OPTIONS] [COMMAND_OPTIONS]
```

Hadoop has an option parsing framework that employs parsing generic options as well as running classes.

COMMAND_OPTION	Description
`-mode`	For both the hadoop and hadoop2 commands, setting this option is no longer valid. Both commands default to `yarn`, the only valid mode for the current MapR version.
`--config confdir`	Overwrites the default Configuration directory. Default is `${HADOOP_HOME}/conf`.
`COMMAND`	Various commands with their options are described in the following sections.
`GENERIC_OPTIONS`	The common set of options supported by multiple commands.
`COMMAND_OPTIONS`	Various command options are described in the following sections.

Note: Running the `hadoop` script without any arguments prints the help description for all commands.

Supported Commands for Hadoop 2.x

MapR supports the following `hadoop` commands for Hadoop 2.x:

Command	Description
`archive -archiveName NAME <src>* <dest>`	Creates a Hadoop archive, a file that contains other files. A Hadoop archive always has a `.har` extension.
CLASSNAME	The hadoop script can be used to invoke any class. `hadoop CLASSNAME` runs the class named CLASSNAME.
`classpath`	Prints the class path needed to access the Hadoop JAR and the required libraries.
`conf`	The `hadoop conf` command prints the configuration information for the current node.
`daemonlog`	The `hadoop daemonlog` command may be used to get or set the log level of Hadoop daemons.

last-updated: Apr 11, 2022

Command	Description
`distcp <source> <destination>`	The `hadoop distcp` command is a tool for large inter- and intra-cluster copying. It uses MapReduce to effect its distribution, error handling and recovery, and reporting. It expands a list of files and directories into input to map tasks, each of which will copy a partition of the files specified in the source list.
`fs`	The `hadoop fs` command runs a generic filesystem user client that interacts with the MapR filesystem.
`jar <jar>`	The `hadoop jar` command runs a JAR file. Users can bundle their MapReduce code in a JAR file and execute it using this command.
`mfs`	The `hadoop mfs` command performs operations on directories in the cluster. The main purposes of `hadoop mfs` are to display directory information and contents, to create symbolic links, and to set compression and chunk size on a directory.
`version`	The `hadoop version` command prints the Hadoop software version.

Warning: For Hadoop2, some `hadoop` commands are deprecated and replaced by the `mapred` command.

For example, if you run the `hadoop job` command, you see this message:

```
# hadoop job
DEPRECATED: Use of this script to execute mapred command is deprecated.
Instead, use the mapred command for it.
```

The syntax for the `mapred` command is:

```
mapred [--config confdir] [--loglevel loglevel] COMMAND
```

Commands used with `mapred` include:

Command	Description
`historyserver`	Runs job history servers as a standalone daemon
`hsadmin`	The job history server admin interface
`job`	Manipulates MapReduce applications
`pipes`	Runs a `pipes` job
`queue`	Gets information regarding `JobQueues`

Generic Options

Implement the Tool interface to make the following command-line options available for many of the Hadoop commands.

Generic Option	Description
`-conf <filename1 filename2 ...>`	Add the specified configuration files to the list of resources available in the configuration.

last-updated: Apr 11, 2022

Generic Option	Description
`-D <property=value>`	Set a value for the specified Hadoop configuration property.
`-fs <local|filesystem URI>`	Set the URI of the default filesystem.
`-jt <local|jobtracker:port>`	Specify a ResourceManager for a given host and port. This command option is a shortcut for `-D mapred.job.tracker=host:port`
`-files <file1,file2,...>`	Specify files to be copied to the map reduce cluster.
`-libjars <jar1,jar2,...>`	Specify JAR files to be included in the classpath of the mapper and reducer tasks.
`-archives <archive1,archive2,...>`	Specify archive files (JAR, tar, tar.gz, ZIP) to be copied and unarchived on the task node.

hadoop archive

The `hadoop archive` command creates a Hadoop archive, a file that contains other files. A Hadoop archive always has a `*.har` extension.

Syntax

```
hadoop [ Generic Options ] archive
    -archiveName <name>
    [-p <parent>]
    <source>
    <destination>
```

Parameters

Parameter	Description
`-archiveName <name>`	Name of the archive to be created.
`-p <parent_path>`	The parent argument is to specify the relative path to which the files should be archived to.
`<source>`	Filesystem pathnames, which work as usual with regular expressions.
`<destination>`	Destination directory, which would contain the archive.

Examples

Archive within a single directory

```
hadoop archive -archiveName myArchive.har -p /foo/bar /outputdir
```

The above command creates an archive of the directory `/foo/bar` in the directory `/outputdir`.

Archive to another directory

```
hadoop archive -archiveName myArchive.har -p /foo/bar a/b/c e/f/g
```


last-updated: Apr 11, 2022

The above command creates an archive of the directory `/foo/bar/a/b/c` in the directory `/foo/bar/e/f/g`.

hadoop classpath
The `hadoop classpath` command prints the class path needed to access the Hadoop jar and the required libraries.

Syntax

```
hadoop classpath
```

Output Example

```
$hadoop classpath
/opt/mapr/hadoop/hadoop-2.7.0/etc/hadoop:/opt/mapr/hadoop/hadoop-2.7.0/
share/hadoop/common/lib/
*:/opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/common/*:/opt/mapr/hadoop/
hadoop-2.7.0/share/hadoo
p/hdfs:/opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/hdfs/lib/*:/opt/mapr/
hadoop/hadoop-2.7.0/shar
e/hadoop/hdfs/*:/opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/yarn/lib/*:/opt/
mapr/hadoop/hadoop-2
.7.0/share/hadoop/yarn/*:/opt/mapr/hadoop/hadoop-2.7.0/share/hadoop/
mapreduce/lib/*:/opt/mapr/h
adoop/hadoop-2.7.0/share/hadoop/mapreduce/*:/contrib/capacity-scheduler/
*.jar:/opt/mapr/lib/kvs
tore*.jar:/opt/mapr/lib/libprotodefs*.jar:/opt/mapr/lib/baseutils*.jar:/opt/
mapr/lib/maprutil*.
jar:/opt/mapr/lib/json-20080701.jar:/opt/mapr/lib/flexjson-2.1.jar
```

hadoop daemonlog
The `hadoop daemonlog` command gets and sets the log level for each daemon.

Hadoop daemons all produce logfiles that you can use to learn about what is happening on the system. You can use the `hadoop daemonlog` command to temporarily change the log level of a component when debugging the system.

Syntax

```
hadoop daemonlog
    -getlevel | -setlevel
    <host>:<port>
    <name>
    [ <level> ]
```

Parameters

The following command options are supported for `hadoop daemonlog` command:

last-updated: Apr 11, 2022

Parameter	Description
`-getlevel <host:port><name>`	Prints the log level of the daemon running at the specified host and port, by querying `http://<host>:<port>/logLevel?log=<name>` • `<host>`: The host on which to get the log level. • `<port>`: The port by which to get the log level. • `<name>`: The daemon on which to get the log level. Usually the fully qualified classname of the daemon doing the logging. For example, `org.apache.hadoop.yarn.server.resourcemanager.resourcemanager` for the ResourceManager daemon.
`-setlevel <host:port> <name> <level>`	Sets the log level of the daemon running at the specified host and port, by querying `http://<host>:<port>/logLevel?log=<name>` * `<host>`: The host on which to set the log level. • `<port>`: The port by which to set the log level. • `<name>`: The daemon on which to set the log level. • `<level`: The log level to set the daemon.

Examples

Getting the log levels of a daemon

To get the log level for each daemon enter a command such as the following:

```
hadoop daemonlog -getlevel 10.250.1.15:50030
org.apache.hadoop.yarn.server.resourcemanager.resourcemanager
Connecting to http://10.250.1.15:50030/logLevel?
log=org.apache.hadoop.yarn.server.resourcemanager.resourcemanager
Submitted Log Name:
org.apache.hadoop.yarn.server.resourcemanager.resourcemanager
Log Class: org.apache.commons.logging.impl.Log4JLogger
Effective level: ALL
```

Setting the log level of a daemon

To temporarily set the log level for a daemon enter a command such as the following:

```
hadoop daemonlog -setlevel 10.250.1.15:50030
org.apache.hadoop.yarn.server.resourcemanager.resourcemanager DEBUG
Connecting to http://10.250.1.15:50030/logLevel?
log=org.apache.hadoop.yarn.server.resourcemanager.resourcemanager&level=DEBU
G
Submitted Log Name:
org.apache.hadoop.yarn.server.resourcemanager.resourcemanager
Log Class: org.apache.commons.logging.impl.Log4JLogger
Submitted Level: DEBUG
Setting Level to DEBUG ...
Effective level: DEBUG
```


last-updated: Apr 11, 2022

Using this method, the log level is automatically reset when the daemon is restarted.

To make the change to log level of a daemon persistent, enter a command such as the following:

```
hadoop daemonlog -setlevel 10.250.1.15:50030
log4j.logger.org.apache.hadoop.yarn.server.resourcemanager.resourcemanager
DEBUG
```

hadoop distcp
The `hadoop distcp` command is a tool used for large inter- and intra-cluster copying.

It uses MapReduce to effect its distribution, error handling and recovery, and reporting. It expands a list of files and directories into input to map tasks, each of which will copy a partition of the files specified in the source list.

Syntax

```
hadoop [ Generic Options ] distcp
    [-p[erbugp] ]
    [-i ]
    [-log ]
    [-m ]
    [-overwrite ]
    [-update  ]
    [-f <URI list> ]
    [-filelimit <n> ]
    [-sizelimit <n> ]
    [-delete ]
    <source>
    <destination>
```

Parameters

Command Options

The following command options are supported for the `hadoop distcp` command:

Parameter	Description
`<source>`	Specify the source URL.
`<destination>`	Specify the destination URL.
`-p[erbugp]`	Preserve e: ACE `r`: replication number b: block size u: user g: group p: permission `-p` alone is equivalent to `-perbugp`. Modification times are not preserved. When you specify `-update`, status updates are not synchronized unless the file sizes also differ.
`-i`	Ignore failures. As explained in the below, this option will keep more accurate statistics about the copy than the default case. It also preserves logs from failed copies, which can be valuable for debugging. Finally, a failing map will not cause the job to fail before all splits are attempted.
`-log <logdir>`	Write logs to `<logdir>`. The `hadoop distcp` command keeps logs of each file it attempts to copy as map output. If a map fails, the log output will not be retained if it is re-executed.
`-m <num_maps>`	Maximum number of simultaneous copies. Specify the number of maps to copy data. Note that more maps may not necessarily improve throughput. See *Map Sizing*.

Parameter	Description
`-overwrite`	Overwrite destination. If a map fails and `-i` is not specified, all the files in the split, not only those that failed, will be recopied. As discussed in the *Overwriting Files Between Clusters*, it also changes the semantics for generating destination paths, so users should use this carefully.
`-update`	Overwrite if `<source>` size is different from `<destination>` size. As noted in the preceding, this is not a "sync" operation. The only criterion examined is the source and destination file sizes; if they differ, the source file replaces the destination file. See *Updating Files Between Clusters*.
`-f <URI list>`	Use list at <URI list> as source list. This is equivalent to listing each source on the command line. The value of <URI list> must be a fully qualified URI.
`-filelimit <n>`	Limit the total number of files to be <= n. See *Symbolic Representations*.
`-sizelimit <n>`	Limit the total size to be <= n bytes. See *Symbolic Representations*.
`-delete`	Delete the files existing in the `<destination>` but not in `<source>`. The deletion is done by FS Shell.

Generic Options

The `hadoop distcp` command supports the following generic options: `-conf <configuration file>`, `-D <property=value>`, `-fs <local|file system URI>`, `-jt <local|jobtracker:port>`, `-files <file1,file2,file3,...>`, `-libjars <libjar1,libjar2,libjar3,...>`, and `-archives <archive1,archive2,archive3,...>`. For more information on generic options, see Generic Options.

Symbolic Representations

The parameter <n> in `-filelimit` and `-sizelimit` can be specified with symbolic representation. For example,

- 1230k = 1230 * 1024 = 1259520
- 891g = 891 * 1024^3 = 956703965184

Map Sizing

The `hadoop distcp` command attempts to size each map comparably so that each copies roughly the same number of bytes. Note that files are the finest level of granularity, so increasing the number of simultaneous copiers (i.e. maps) may not always increase the number of simultaneous copies nor the overall throughput.

If `-m` is not specified, `distcp` will attempt to schedule work for `min (total_bytes / bytes.per.map, 20 * num_task_trackers)` where `bytes.per.map` defaults to 256MB.

Tuning the number of maps to the size of the source and destination clusters, the size of the copy, and the available bandwidth is recommended for long-running and regularly run jobs.

Examples

For all of the below examples, the cluster name must be specified in the mapr-clusters.conf on page 2200 configuration file.

Basic inter-cluster copying

last-updated: Apr 11, 2022

The `hadoop distcp` command is most often used to copy files between clusters:

```
hadoop distcp maprfs://cluster1/foo \
maprfs://cluster2/bar
```

The command in the example expands the namespace under `/foo/bar` on cluster1 into a temporary file, partitions its contents among a set of map tasks, and starts a copy on each NodeManager node from cluster1 to cluster2. Note that the `hadoop distcp` command expects absolute paths.

Only those files that do not already exist in the destination are copied over from the source directory.

Updating files between clusters

Use the `hadoop distcp -update` command to synchronize changes between clusters.

```
$ hadoop distcp -update maprfs://cluster1/foo maprfs://cluster2/bar/foo
```

Files in the `/foo` subtree are copied from cluster1 to cluster2 only if the size of the source file is different from that of the size of the destination file. Otherwise, the files are skipped over.

Note that using the `-update` option changes distributed copy interprets the source and destination paths making it necessary to add the trailing `/foo` subdirectory in the second cluster.

Overwriting files between clusters

By default, distributed copy skips files that already exist in the destination directory, but you can overwrite those files using the `-overwrite` option. In this example, multiple source directories are specified:

```
$ hadoop distcp -overwrite maprfs://cluster1/foo/a \
maprfs://cluster1/foo/b \
maprfs://cluster2/bar
```

As with using the `-update` option, using the `-overwrite` changes the way that the source and destination paths are interpreted by distributed copy: the contents of the source directories are compared to the contents of the destination directory. The distributed copy aborts in case of a conflict.

Migrating Data from HDFS to MapR File System

The `hadoop distcp` command can be used to migrate data from an HDFS cluster to a MapR File System where the HDFS cluster uses the same version of the RPC protocol as that used by MapR. For a discussion, see Copying Data from Apache Hadoop.

```
$ hadoop distcp namenode1:50070/foo maprfs:///bar
```

You must specify the IP address and HTTP port (usually 50070) for the namenode on the HDFS cluster.

hadoop fs

The `hadoop fs` command runs a generic file system user client that interacts with the MapR File System.

Warning: On the Windows client, make sure that the `PATH` contains the following directories:

- `C:\Windows\system32`
- `C:\Windows`

If they are not present, the `hadoop fs` command might fail silently.

Syntax

```
hadoop [ Generic Options ] fs
        [-cat <src>]
        [-chgrp [-R] GROUP PATH...]
        [-chmod [-R] <MODE[,MODE]... | OCTALMODE> PATH...]
        [-chown [-R] [OWNER][:[GROUP]] PATH...]
        [-copyFromLocal <localsrc> ... <dst>]
        [-copyToLocal [-ignoreCrc] [-crc] <src> <localdst>]
        [-count[-q] <path>]
        [-cp <src> <dst> -p[e]]
        [-df <path>]
        [-du <path>]
        [-dus <path>]
        [-expunge]
        [-get [-ignoreCrc] [-crc] <src> <localdst>]
        [-getfattr [-R] {-n name | -d} [-e <encoding>] <path>]
        [-getmerge <src> <localdst> [addnl]]
        [-help [cmd]]
        [-ls <path>]
        [-lsr <path>]
        [-mkdir <path>]
        [-moveFromLocal <localsrc> ... <dst>]
        [-moveToLocal <src> <localdst>]
        [-mv <src> <dst>]
        [-put <localsrc> ... <dst>]
        [-rm [-skipTrash] <src>]
        [-rmr [-skipTrash] <src>]
        [-setfattr -n name [-v value] | -x name <path>]
        [-stat [format] <path>]
        [-tail [-f] <path>]
        [-test -[ezd] <path>]
        [-text <path>]
        [-touchz <path>]
```

Parameters

Command Options

The following command parameters are supported for `hadoop fs`:

Parameter	Description
`-cat <src>`	Fetch all files that match the file pattern defined by the `<src>` parameter and display their contents on *stdout*.
`-chmod [-R] <MODE[,MODE]... | OCTALMODE> PATH...`	Changes permissions of a file. This works similar to shell's `chmod` with a few exceptions. `-R` modifies the files recursively. This is the only option currently supported. `MODE` Mode is same as mode used for `chmod` shell command. Only letters recognized are `rwxXt`. That is, `+t,a+r,g-w,+rwx,o=r` `OCTALMODE` Mode specifed in 3 or 4 digits. If 4 digits, the first may be 1 or 0 to turn the sticky bit on or off, respectively. Unlike shell command, it is not possible to specify only part of the mode E.g. 754 is same as u=rwx,g=rx,o=r If none of 'augo' is specified, 'a' is assumed and unlike shell command, no umask is applied.

last-updated: Apr 11, 2022

Parameter	Description
`-chown [-R] [OWNER] [:[GROUP]] PATH...`	Changes owner and group of a file. This is similar to shell's `chown` with a few exceptions. `-R` modifies the files recursively. This is the only option currently supported. If only owner or group is specified then only owner or group is modified.The owner and group names may only consists of digits, alphabet, and any of `-.@/` *i.e.* `[-.@/a-zA-Z0-9]`. The names are case sensitive. **Warning:** Avoid using '.' to separate user name and group though Linux allows it. If user names have dots in them and you are using local file system, you might see surprising results since shell command `chown` is used for local files.
`-chgrp [-R] GROUP PATH...`	This is equivalent to `-chown ... :GROUP ...`
`-copyFromLocal <localsrc> ... <dst>`	Identical to the `-put` command.
`-copyToLocal [-ignoreCrc] [-crc] <src> <localdst>`	Identical to the `-get` command.
`-count[-q] <path>`	Count the number of directories, files and bytes under the paths that match the specified file pattern. The output columns are: `DIR_COUNT FILE_COUNT CONTENT_SIZE FILE_NAME` or `QUOTA REMAINING_QUATA SPACE_QUOTA REMAINING_SPACE_QUOTA DIR_COUNT FILE_COUNT CONTENT_SIZE FILE_NAME`
`-cp <src> <dst> [-p[e]]`	Copy files that match the file pattern `<src>` to a destination. When copying multiple files, the destination must be a directory. Specifying `-p` with the e option preserves an ACE applied to the file. Specifying `-p` without an argument preserves the ACE by default.
`-df [<path>]`	Shows the capacity, free and used space of the file system. If the file system has multiple partitions, and no path to a particular partition is specified, then the status of the root partitions will be shown.
`-du <path>`	Show the amount of space, in bytes, used by the files that match the specified file pattern. Equivalent to the Unix command `du -sb <path>/*` in case of a directory, and to `du -b <path>` in case of a file. The output is in the form name(full path) size (in bytes).
`-dus <path>`	Show the amount of space, in bytes, used by the files that match the specified file pattern. Equivalent to the Unix command `du -sb`. The output is in the form `name(full path)` size (in bytes).
`-fs [local | <filesystem URI>]`	Specify the file system to use. If not specified, the current configuration is used, taken from the following, in increasing precedence: `core-default.xml` inside the hadoop jar file `core-site.xml` in `$HADOOP_CONF_DIR`. The `local` option means use the local file system as your DFS. `<filesystem URI>` specifies a particular file system to contact. This argument is optional but if used must appear appear first on the command line. Exactly one additional argument must be specified.
`-get [-ignoreCrc] [-crc] <src> <localdst>`	Copy files that match the file pattern <src> to the local name. <src> is kept. When copying multiple files, the destination must be a directory.

last-updated: Apr 11, 2022

Parameter	Description
`getfattr [-R] -n <name> | -d [-e <encoding>] <path>`	Retrieve all the extended attribute values (if any) for a file or directory. Here: **`-R`** Recursively list the attributes for all files and directories. **`-n <name>`** The name of the extended attribute to retrieve. **`-d`** Retrieve all extended attributes associated with the pathname. Extended attributes to which the calling process does not have access may be omitted from the list. **`-e <encoding>`** Encode values after retrieving them. Valid encodings are text (enclosed in double quotes), hex (prefixed with 0x), and base64 (prefixed with 0s). **`<path>`** The file or directory.
`-getmerge <src> <localdst>`	Get all the files in the directories that match the source file pattern and merge and sort them to only one file on local fs. `<src>` is kept.
`-help [cmd]`	Displays help for given command or all commands if none is specified.
`-ls <path>`	List the contents that match the specified file pattern. If path is not specified, the contents of `/user/<currentUser>` will be listed. Directory entries are of the form `dirName (full path) <dir>` and file entries are of the form `fileName(full path) <r n>`. size where n is the number of replicas specified for the file and size is the size of the file, in bytes.
`-lsr <path>`	Recursively list the contents that match the specified file pattern. Behaves very similarly to `hadoop fs -ls`, except that the data is shown for all the entries in the subtree.
`-mkdir <path>`	Create a directory in specified location.
`-moveFromLocal <localsrc> ... <dst>`	Same as `-put`, except that the source is deleted after it's copied.
`-moveToLocal <src> <localdst>`	Not implemented yet
`-mv <src> <dst>`	Move files that match the specified file pattern `<src>` to a destination `<dst>`. When moving multiple files, the destination must be a directory.
`-put <localsrc> ... <dst>`	Copy files from the local file system into fs. To copy files, user must have write permission on the directory or the `addchild`, `deletechild`, and `writefile` access set using ACEs.
`-rm [-skipTrash] <src>`	Delete all files that match the specified file pattern. Equivalent to the Unix command `rm <src>`. The `-skipTrash` option bypasses trash, if enabled, and immediately deletes `<src>`
`-rmr [-skipTrash] <src>`	Remove all directories which match the specified file pattern. Equivalent to the Unix command `rm -rf <src>` The `-skipTrash` option bypasses trash, if enabled, and immediately deletes `<src>`

last-updated: Apr 11, 2022

Parameter	Description
`-setfattr -n <name> [-v <value>] | -x <name> <path>`	Set or remove an extended attribute name and value. Here: **`-n <name>`** The name of the extended attribute to set. **`-v <value>`** The value of the extended attribute to set. **`-x <name>`** The name of the extended attribute to remove. **`<path>`** The file or directory.
`-stat [format] <path>`	Print statistics about the file/directory at `<path>` in the specified format. Format accepts filesize in blocks (%b), filename (%n), block size (%o), replication (%r), modification date (%y, %Y)
`-tail [-f] <file>`	Show the last 1KB of the file. The `-f` option shows appended data as the file grows.
`-touchz <path>`	Write a timestamp in `yyyy-MM-dd HH:mm:ss` format in a file at `<path>`. An error is returned if the file exists with non-zero length.
`-test -[ezd] <path>`	If file { exists, has zero length, is a directory then return 0, else return 1.
`-text <src>`	Takes a source file and outputs the file in text format. The allowed formats are zip and TextRecordInputStream.

Generic Options

The following generic options are supported for the `hadoop fs` command: `-conf <configuration file>`, `-D <property=value>`, `-fs <local|file system URI>`, `-jt <local|jobtracker:port>`, `-files <file1,file2,file3,...>`, `-libjars <libjar1,libjar2,libjar3,...>`, and `-archives <archive1,archive2,archive3,...>`. For more information on generic options, see Generic Options.

Examples

Set an extended attribute on a file, file1.txt:

```
hadoop fs -setfattr -n user.key1 -v val1 /xattrs/m7user1/file1.txt
```

Remove an extended attribute specified by name:

```
hadoop fs -setfattr -x user.key1 /xattrs/m7user1/dir1
```

Retrieve an extended attribute for a file encoded in text:

```
[root@qa-node110 ~]# hadoop fs -getfattr -n user.key1 -e text /xattr/file1
# file: /xattr/file1
user.key1="value1"
```

Retrieve an extended attribute for a file encoded in hex:

```
[root@qa-node110 ~]# hadoop fs -getfattr -n user.key1 -e hex /xattr/file1
# file: /xattr/file1
user.key1=0x76616c756531
```


last-updated: Apr 11, 2022

Retrieve an extended attribute for a file encoded in base64:

```
[root@qa-node110 ~]# hadoop fs -getfattr -n user.key1 -e base64 /xattr/file1
# file: /xattr/file1
user.key1=0sdmFsdWUx
```

Retrieve an extended attribute specified by name:

```
[root@qa-node110 ~]# hadoop fs -getfattr -n user.key1 /xattr/file2
# file: /xattr/file2
user.key1="value1"
```

Retrieve all the extended attributes associated with the given file:

```
[root@qa-node110 ~]# hadoop fs -getfattr -d /xattr/file2
# file: /xattr/file2
user.key2="value2"
user.key1="value1"
```

Retrieve extended attributes recursively:

```
[root@qa-node110 ~]# hadoop fs -getfattr -R -d /xattr/
# file: /xattr
# file: /xattr/file1
user.key2="value2"
# file: /xattr/file2
user.key2="value2"
user.key1="value1"
```

Retrieve a specific extended attribute recursively on a directory:

```
[root@qa-node110 ~]# hadoop fs -getfattr -R -n user.key1 /xattr/
# file: /xattr
user.key1="value1"
# file: /xattr/file1
getfattr: No such attribute
# file: /xattr/file2
user.key1="value1"
```

Suppressing Warning Messages for the hadoop fs Command

After an upgrade to 4.0.x, the `hadoop fs` command returns the following warning message:

```
WARNING: org.apache.hadoop.metrics.jvm.EventCounter is deprecated. Please
use
org.apache.hadoop.log.metrics.EventCounter in all the log4j.properties
files.
```

This message does not cause any problems, but you can suppress it by modifying the following file:

```
/opt/mapr/hadoop/hadoop-0.20.2/conf/log4j.properties
```

In this file, replace the following line:

```
log4j.appender.EventCounter=org.apache.hadoop.log.EventCounter
```

with this line:

```
log4j.appender.EventCounter=org.apache.hadoop.log.metrics.EventCounter
```


last-updated: Apr 11, 2022

hadoop jar

The `hadoop jar` command runs a program contained in a JAR file. Users can bundle their MapReduce code in a JAR file and execute it using this command.

Syntax

```
hadoop jar <jar>
     [<arguments>]
```

Parameters

The following commands parameters are supported for `hadoop jar`:

Parameter	Description
`<jar>`	The JAR file.
`<arguments>`	Arguments to the program specified in the JAR file.

Examples

Streaming Application

Hadoop streaming applications are run using the `hadoop jar` command. The Hadoop streaming utility enables you to create and run MapReduce applications with any executable or script as the mapper and/or the reducer.

```
$ hadoop jar $HADOOP_HOME/hadoop-streaming.jar \
    -input myInputDirs \
    -output myOutputDir \
    -mapper org.apache.hadoop.mapred.lib.IdentityMapper \
    -reducer /bin/wc
```

The `-input`, `-output`, `-mapper`, and `-reducer` streaming command options are all required for streaming jobs. Either an executable or a Java class may be used for the mapper and the reducer. For more information about and examples of streaming applications, see *Hadoop Streaming* at the Apache project's page.

Running from a JAR file

The simple Word Count program is another example of a program that is run using the `hadoop jar` command. The `wordcount` functionality is built into the `hadoop-0.20.2-dev-examples.jar` program. You pass the file, along with the location, to Hadoop with the `hadoop jar` command and Hadoop reads the JAR file and executes the relevant instructions.

The Word Count program reads files from an input directory, counts the words, and writes the results of the application to files in an output directory.

```
$ hadoop jar /opt/mapr/hadoop/hadoop-0.20.2/hadoop-0.20.2-dev-examples.jar
wordcount /myvolume/in /myvolume/out
```

hadoop job

The `hadoop job` command enables you to manage MapReduce jobs.

Warning: This command is deprecated.

last-updated: Apr 11, 2022

Syntax

```
hadoop job [Generic Options]
        [-submit <job-file>]
        [-status <job-id>]
        [-counter <job-id> <group-name> <counter-name>]
        [-kill <job-id>]
        [-unblacklist <job-id> <hostname>]
        [-unblacklist-tracker <hostname>]
        [-set-priority <job-id> <priority>]
        [-events <job-id> <from-event-#> <#-of-events>]
        [-history <jobOutputDir>]
        [-list [all]]
        [-list-active-trackers]
        [-list-blacklisted-trackers]
        [-list-attempt-ids <job-id> <task-type> <task-state>]
        [-kill-task <task-id>]
        [-fail-task <task-id>]
        [-blacklist-tasktracker <hostname>]
        [-showlabels]
```

Parameters

Command Options

The following command options are supported for `hadoop job`:

Parameter	Description
`-submit <job-file>`	Submits the job.
`-status <job-id>`	Prints the map and reduce completion percentage and all job counters.
`-counter <job-id> <group-name> <counter-name>`	Prints the counter value.
`-kill <job-id>`	Kills the job.
`-unblacklist <job-id> <hostname>`	Removes a tasktracker job from the jobtracker's blacklist.
`-unblacklist-tracker <hostname>`	Admin only. Removes the TaskTracker at `<hostname>` from the JobTracker's global blacklist.
`-set-priority <job-id> <priority>`	Changes the priority of the job. Valid priority values are `VERY_HIGH`, `HIGH`, `NORMAL`, `LOW`, and `VERY_LOW`. The job scheduler uses this property to determine the order in which jobs are run.
`-events <job-id> <from-event-#> <#-of-events>`	Prints the events' details received by jobtracker for the given range.
`-history <jobOutputDir>`	Prints job details, failed and killed tip details.
`-list [all]`	The `-list all` option displays all jobs. The `-list` command without the `all` option displays only jobs which are yet to complete.
`-list-active-trackers`	Prints all active tasktrackers.

last-updated: Apr 11, 2022

Parameter	Description
`-list-blackisted-trackers`	Prints the TaskTracker nodes that JobTracker blacklisted with the reason for blacklisting.
`-list-attempt-ids <job-id><task-type>`	Lists the IDs of task attempts.
`-kill-task <task-id>`	Kills the task. Killed tasks are *not* counted against failed attempts.
`-fail-task <task-id>`	Fails the task. Failed tasks are counted against failed attempts.
`-blacklist-tasktracker <hostname>`	Pauses all current tasktracker jobs and prevent additional jobs from being scheduled on the tasktracker.
`-showlabels`	Dumps label information of all active nodes.

Generic Options

The following generic options are supported for the `hadoop job` command: `-conf <configuration file>`, `-D <property=value>`, `-fs <local|file system URI>`, `-jt <local|jobtracker:port>`, `-files <file1,file2,file3,...>`, `-libjars <libjar1,libjar2,libjar3,...>`, and `-archives <archive1,archive2,archive3,...>`. For more information on generic options, see Generic Options.

Examples

Submitting Jobs

The `hadoop job -submit` command enables you to submit a job to the specified jobtracker.

```
$ hadoop job -jt darwin:50020 -submit job.xml
```

Stopping Jobs Gracefully

Use the `hadoop kill` command to stop a running or queued job.

```
$ hadoop job -kill <job-id>
```

Viewing Job History Logs

Run the `hadoop job -history` command to view the history logs summary in specified directory.

```
$ hadoop job -history output-dir
```

This command will print job details, failed and killed tip details.

Additional details about the job such as successful tasks and task attempts made for each task can be viewed by adding the `-all` option:

```
$ hadoop job -history all output-dir
```

Blacklisting Tasktrackers

The `hadoop job` command when run as root or using `sudo` can be used to manually blacklist tasktrackers:

```
hadoop job -blacklist-tasktracker <hostname>
```


last-updated: Apr 11, 2022

Manually blacklisting a tasktracker pauses any running jobs and prevents additional jobs from being scheduled.

hadoop mfs
The `hadoop mfs` command displays directory information and contents, creates symbolic links and hard links, sets, gets, and removes Access Control Expressions (ACE) on files and directories, and sets compression and chunk size on a directory.

Syntax

```
hadoop mfs
  [ -count <path> ]
  [ -delace [-R] <path> ]
  [ -getace [-R] <path> ]
  [ -help <command> ]
  [ -ln <target> <symlink> ]
  [ -lnh <target> <hardlink> ]
  [ -ls <path> ]
  [ -lsd <path> ]

  [ -lso <path> ]
  [ -lsor <path> ]
  [ -lsr <path> ]
  [ -Lsr <path> ]
  [ -lsrv <path> ]
  [ -lss <path> ]
  [ -offload <file_path> [-v] ]
  [ -recall <file_path> [-v] ]
  [ -rmr <path> ]
  [ -setace [-R]
     [-readfile <ace>] [-writefile <ace>] [-executefile <ace>]
     [-addchild <ace>] [-deletechild <ace>] [-lookupdir <ace>] [-readdir
<ace>]
     [-aces "[rf:<ace>],[wf:<ace>],[ef:<ace>],[ac:<ace>],[dc:<ace>],
[rd:<ace>],[ld:<ace>]"]
     [-preservemodebits <true|false>] [-setinherit <true|false>] <path> ]
  [ -setaudit  on|off <dir|file|table> ]
  [ -setcompression  on|off|lzf|lz4|zlib <dir|table> ]
  [ -setchunksize <size> <dir> ]

  [ -setnetworkencryption on|off <target> ]
  [ -stat <path> ]
  [ -tierstatus <file_path> [-v] ]
```

Parameters

The normal command syntax is to specify a single option from the following table, along with its corresponding arguments. If you do not set compression and chunk size for a given directory, the values are inherited from the parent directory.

Parameter	Description
`-count <path>`	Counts and returns the number of directories and (regular, symbolic link, volume link, kvstores, and device) files in the specified path (recursively).

last-updated: Apr 11, 2022

Parameter	Description
`-delace [-R] <path>`	Deletes all ACEs associated with the specified file or directory and sets *ACE*s for the specified file or directory to the default value, which is the empty string. Here: • `[-R]` — Enables recursion allowing you to perform the operation in subdirectories as well. • `<path>` — Specifies the path to the file or directory. You cannot delete specific access types with this parameter. Instead, if necessary, reset the value for the specific access type to an empty string using the `-setace` parameter. If you use an empty string to deny a specific type of access, then that type of access is denied to all users. To deny specific types of access to specific users only, use the negation operator (`!`). The mode bits corresponding to the *ACE*s being deleted, do not change.
`-getace [-R] <path>`	Returns the permissions -- POSIX mode bits and ACEs -- for the given file or (recursively) for the directory. Recursion is enabled only if -R is specified; if -R is not specified, this parameter returns the permissions only for the given directory. Here: • `[-R]` — (Optional) Enables recursion allowing you to perform the operation in subdirectories as well. • `<path>` — (Required) Specifies the path to the file or directory. If one or more *ACE*s are available for the file or directory, a plus sign (+), which indicates that both *ACE*s and POSIX mode bits are set for the given file or directory, is returned. If the *ACE* on the file or directory is an empty string, the plus sign is not returned.
`-help <command>`	Displays help for the `hadoop mfs` command.
`-ln <target> <symlink>`	Creates a symbolic link `<symlink>` that points to the target path `<target>`, similar to the standard Linux `ln -s` command.
`-lnh <target> <hardlink>`	Creates a hardlink that associates a new name or a file path with an existing file. You must specify the following: **`<target>`** — File name, including the full path, of the file to link to. **`<hardlink>`** — New name, including the full path, of the file to link with.

last-updated: Apr 11, 2022

Parameter	Description
`-ls <path>`	Lists files in the directory specified by `<path>`. The `hadoop mfs -ls` command corresponds to the standard `hadoop fs -ls` command, but provides the following additional information: • Chunks used for each file • Server where each chunk resides • Whether compression is enabled for each file • Whether encryption is enabled for each file • Whether audit is enabled (A) or disabled (U) for each file
`-lsd <path>`	Lists files in the directory specified by `<path>`, and also provides information about the specified directory itself: • Whether compression is enabled for the directory (indicated by z) • The configured chunk size (in bytes) for the directory.
`-lso <path>`	Lists files in the directory specified by `<path>`. The `hadoop mfs -lso` command corresponds to the standard `hadoop fs -ls` command, but provides the following additional information: • Whether compression is enabled for each file • Whether encryption is enabled for each file • Whether audit is enabled (A) or disabled (U) for each file This command is faster than `hadoop fs -ls` as it uses an optimized printing method to dump data on screen.
`-lsor <path>`	Recursively lists files in the directory specified by `<path>`. This command is the recursive variant of the `hadoop mfs -lso` command.
`-lsr <path>`	Recursively lists files in the directory and subdirectories specified by `<path>`. The `hadoop mfs -lsr` command corresponds to the standard `hadoop fs -lsr` command, but provides the following additional information: • Chunks used for each file • Server where each chunk resides
`-Lsr <path>`	Equivalent to `lsr`, but additionally dereferences symbolic links
`-lsrv <path>`	Lists all paths recursively without crossing volume links.
`-lss <path>`	Lists files in the directory specified by `<path>`, with an additional column that displays the number of disk blocks per file. Disk blocks are 8192 bytes.

last-updated: Apr 11, 2022

Parameter	Description
`-offload <file_path> [-v]`	The file to offload to the storage tier. This is a blocking operation; the control is not returned until the operation is complete and the file has been offloaded. Use `-v` (for verbose) to view the status of the ongoing offload operation.
`-recall <file_path> [-v]`	The file to recall from the storage tier. This is a blocking operation; the control is not returned until the operation is complete and the file has been recalled. Use `-v` (for verbose) to view the status of the ongoing recall operation.
`-rmr <path>`	Recursively deletes files and directories in the specified path. This is a highly optimized version of the normal generic `hadoop fs rmr` command and is 10X faster for large directories. This option is useful when one or more directories in the specified path contains many (millions of) files.

last-updated: Apr 11, 2022

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td><pre>-setace [-R] [-readfile <ace>]
[-writefile <ace>] [-executefile
<ace>] [-addchild <ace>] [-deletechild
<ace>] [-lookupdir <ace>]
[-readdir <ace>] [-aces "[rf:<ace>],
[wf:<ace>],[ef:<ace>],[ac:<ace>],
[dc:<ace>],[rd:<ace>],[ld:<ace>]"]
[-preservemodebits <true|false>]
[-setinherit <true|false>] <path></pre></td><td>
Sets or modifies the read, write, and execute permissions for files or directories. This argument will:

Overwrite existing values with new values, if specified, for access types that were previously set.
Set new values for access types that have not yet been set.
Not set or modify access types that were not passed in with the command, whether they were previously set or unset.

Specify the ACEs immediately after the <code>-setace</code> parameter. Specify all the other parameters, after the ACE.

Here:
<table>
<tr><td><code>-R</code></td><td>Enables recursion allowing you to perform the operation in subdirectories as well. Recursion is enabled only if <code>-R</code> is specified; if <code>-R</code> is not specified, sets or modifies the permissions for the given directory only.</td></tr>
<tr><td><ace></td><td>ACE Syntax on page 1448 defined using boolean expressions and sub-expressions.</td></tr>
<tr><td><code>-readfile</code>
<code>-writefile</code>
<code>-executefile</code></td><td>Specifies permissions (defined using ACEs) for reading, writing, or executing the file.</td></tr>
<tr><td><code>-readdir</code>
<code>-lookupdir</code></td><td>Specifies permissions (defined using ACEs) for accessing the directory. To permit users to read files in the directory, grant the <code>lookupdir</code> access permission. To permit users to write to or execute files in the directory, grant the <code>readdir</code> and <code>lookupdir</code> access permissions.</td></tr>
<tr><td><code>-addchild</code>
<code>-deletechild</code></td><td>Specifies permissions (defined using ACEs) for adding or deleting subdirectories.</td></tr>
<tr><td><code>-aces</code></td><td>Specifies ACEs as a single string. Specify a comma-separated list of ACEs within quotes, up to 60 KB in length. You can set the following permissions.
<table>
<tr><td><code>rf</code></td><td>Refers to read file access.</td></tr>
<tr><td><code>wf</code></td><td>Refers to write file access</td></tr>
<tr><td><code>ef</code></td><td>Refers to execute file access</td></tr>
</table>
</td></tr>
</table>
</td></tr>
</table>

Parameter	Description
`-setaudit on|off <dir|file|table>`	Enables auditing of the specified directory, file, or MapR Database table. Enabling auditing of a directory does not enable auditing of files and subdirectories that exist in the directory. You must enable auditing on those existing files and subdirectories. However, any new files and subdirectories that you create will automatically be enabled for auditing. See How Does Auditing Work? on page 760. For operations on the object to be logged, auditing also needs to be enabled on the cluster and the volume in which the object is located. See Managing Auditing on page 757 for details. If auditing is enabled for a directory, new files and directories created within that directory are also enabled for auditing.
`-setchunksize <size> <dir>`	Sets the chunk size in bytes for the directory specified in `<dir>`. The `<size>` parameter must be a multiple of 65536.
`-setcompression on|off|lzf|lz4|zlib <dirtable>`	Turns compression on or off on the directory specified in `<dir>` or on the specified table, and sets the compression type to one of the following if compression is not turned off: • `on` — turns on compression using the default algorithm (LZ4) • `lzf` — turns on compression and sets the algorithm to LZF • `lz4` — turns on compression and sets the algorithm to LZ4 • `zlib` — turns on compression and sets the algorithm to ZLIB
`-setnetworkencryption on|off <target>`	Sets network encryption on or off for the filesystem object defined in `<target>`. The cluster encrypts network target to or from a file, directory, stream, or MapR table with network security enabled.
`-stat <path>`	Displays the statistics for the given file. Only the root user and the MAPR_USER user (user name under which MapR services run) have permissions to run this command. The path is required and specifies the path (to the file) on which to run the command. The output fields for this command are as follows.

last-updated: Apr 11, 2022

Parameter	Description

`tierstatus <file_path> [-v]`	The status of the offload or recall of the given file. If `-v` (for verbose) is also specified, for the given file, the command specifies whether data is local or offloaded as the final output. If the file:
	• Contains local data, returns the following final output:
	`File has local data`
	• Is completely offloaded, returns the following final output:
	`File does not have local data`
	See Output on page 5379 for more information.

Output

When used with the `-ls`, `-lsd`, `-lso`, `-lsor`, `-lsr`, or `-lss` options, `hadoop mfs` displays information about files and directories. For each file or directory `hadoop mfs` displays a line of basic information followed by lines listing the chunks that make up the file, in the following format:

```
{mode} {compression} {encryption} {audit} {diskFlush} {replication} {owner}
{group} {size} {date} {chunk size} {name} {chunk} {fid} {host} [{host}...]
{chunk} {fid} {host} [{host}...] ...
```

Volume links are displayed as follows:

```
{mode} {compression} {encryption} {audit} {diskFlush} {replication} {owner}
{group} {size} {date} {chunk size} {name} {chunk} {target volume name}
{writability} {fid} -> {fid} [{host}...]
```

The following table describes the values:

mode	A text string indicating the read, write, and execute permissions for the owner, group, and other permissions. See also Managing Permissions on page 755.
compression	• U: uncompressed • L: LZf • Z (Uppercase): LZ4 • z (Lowercase): ZLIB
encryption	U: unencrypted; E: encrypted
audit	U: disabled; A: enabled
disk flush	U:disabled; F:enabled
replication	The replication factor of the file (directories display a dash instead)
owner	The owner of the file or directory
group	The group of the file of directory

last-updated: Apr 11, 2022

size	The size of the file or directory
date	The date the file or directory was last modified
chunk size	The chunk size of the file or directory
name	The name of the file or directory
chunk	The chunk number. The first chunk is a primary chunk labeled "p", a 64K chunk containing the root of the file. Subsequent chunks are numbered in order.
fid	The chunk's file ID, which consists of three parts: • The ID of the container where the file is stored • The inode of the file within the container • An internal version number For volume links, the first `fid` is the chunk that stores the volume link itself; the `fid` after the arrow (`->`) is the first chunk in the target volume.
host	The host on which the chunk resides. When several hosts are listed, the first host is the first copy of the chunk, while subsequent hosts are replicas.
target volume name	The name of the volume pointed to by a volume link.
writability	Displays whether the volume is writable.

, `hadoop mfs` displays only the file ID (fid) and the file name of each file in the path.

When used with the `-stat <path>`option, `hadoop mfs` displays statistics for the given file. For each file, it displays the following:

Output field	Description
`uid`	The user ID of the owner.
`mtime`	The last modified time.
`nlink`	The number of hard links.
`type`	The type of the file. Value can be one of: • regular • directory • symlink • vollink • kvstore • device
`size`	The size of the file or directory. Depending on the type of file, it can be the actual size or the number of entries.
`mode`	The UNIX style permission mode bits for the file/directory.

last-updated: Apr 11, 2022

Output field	Description
`networkencryption`	The network encryption setting. Determines whether network encryption is enabled for this file.
`subtype`	The subtype for the specified type. The following subtypes are supported for some of the types: **regular** • FSTRegBucket • FSTRegCF • FSTRegKeyMap **kvstore** • FSTKvTable • FSTKvTabletMap • FSTKvSchema • FSTKvTablet • FSTKvSegMap • FSTKvSpillMap • FSTKvKeyMap • FSTKvXattr **device** • FSTDevPipe • FSTDevSocket • FSTDevBlock • FSTDevChar For all other types, subtypes are not valid.
`gid`	The group ID.
`compression`	The compression setting.

When used with `tierstatus`, the output varies based on whether or not data is local, was offloaded, or was recalled. The output looks similar to the following if:

- Data was completely offloaded:

```
File does not have local data
```

- Data could not be completely offloaded or data was recalled:

```
File has local data
```

Examples

View File Information

last-updated: Apr 11, 2022

The `hadoop mfs` command is used to view file contents. You can use this command to check if compression is turned off in a directory or mounted volume. For example,

```
# hadoop mfs -ls /
Found 121 items
vrwxr-xr-x  Z E U U   3 mapr mapr        121 2018-08-10 01:07
268435456 /.rw
           p mapr.cluster.root writeable 2049.50.131362 -> 2049.16.2
physical19.qa.lab:5660 physical20.qa.lab:5660 physical23.qa.lab:5660
vrwxr-xr-x  Z E U U   3 root root          1 2018-08-09 19:26  268435456 /
ATS-VOL1533867958
           p ATS-VOL1533867958 default 2049.138.131538 -> 2322.16.2
physical20.qa.lab:5660 physical19.qa.lab:5660 physical22.qa.lab:5660
vrwxr-xr-x  Z E U U   3 root root          1 2018-08-09 21:31  268435456 /
ATS-VOL1533875473
           p ATS-VOL1533875473 default 2049.190.131642 -> 2685.16.2
physical21.qa.lab:5660 physical27.qa.lab:5660 physical23.qa.lab:5660
drwxr-xr-x  Z E U U   - root root          1 2018-08-09 18:15  268435456 /
ATS-VOLUME-1533863729955
           p 2049.102.131466  physical19.qa.lab:5660 physical20.qa.lab:5660
physical23.qa.lab:5660
...
```

In the preceding example, the letter `Z` indicates LZ4 compression on the directory; the letter `U` indicates that the directory is uncompressed. In the following example, the listed item is both uncompressed (first `U`) and unencrypted (second `U`).

```
[root@node1-302 ~]# hadoop mfs -ls /hbase
Found 10 items
drwxr-xr-x  Z E U U   - root root          1 2018-08-09 19:26  268435456 /
ATS-VOL1533867958/data1533867963
           p 2322.32.131374  physical20.qa.lab:5660 physical19.qa.lab:5660
physical22.qa.lab:5660
...
```

Set ACEs

Example 1: The following command shows how to set separate read, write, and execute permissions (using *ACE*) on a file:

```
hadoop mfs -setace -readfile p -writefile 'g:group1&!u:user1' -executefile
p /file
```

When the command shown above runs, the POSIX mode bits for:

- Read access is set for owner, owning group, and others.
- Write access is set for none.
- Execute access is set for owner, owning group, others.

All other POSIX mode bits are set to 0.

Example 2: The following command shows how to set read, write, and execute permissions (using *ACE*) as a single string on the specified directory and force all files and subdirectories under the specified directory to inherit the parent *ACE*. *ACE*s that are not specified will be set to the empty string.

```
hadoop mfs -setace -aces "rf:u:root,wf:group1&!
user1,ef:p,rd:u:m7user1" -setinherit true /dir
```

When the command shown above runs, the POSIX mode bits for listing the contents (r) of the directory is set for owner/user and all other POSIX mode bits on the directory are set to 0; all new directories under this directory will inherit the parent POSIX mode bits. Also, new files in the directory will inherit the following POSIX mode bits:

- Read access is set to owner/user.
- Write access is set to none.
- Execute access set for others.

All other POSIX mode bits are set to 0.

Example 3: The following command shows how to set permissions (using *ACE*) as a single string on the specified directory and all the files and subdirectories recursively.

```
hadoop mfs -setace -R -aces "rf:p,wf:g:group1&!
u:user1,ef:p" -preservemodebits true /dir
```

When the command shown above runs, the POSIX mode bits are not modified to match the *ACE* setting.

Example 4: The following command shows how to deny a specific type of access, `writefile`, which was set in the first example above, without removing all other access types associated with the file. The empty string used in the following example will deny write access to all users, *role*s, and groups.

```
hadoop mfs -setace -writefile "" -preservemodebits false /file
```

When the command shown above runs, the POSIX mode bit for writing to the file is set to 0.

Get ACEs

The following command shows the *ACE*s and POSIX mode bits for the specified file only.

```
hadoop mfs -getace /m7user1/file1.txt
```

Output

```
Path: /m7user1/file1.txt
  readfile: !u:m7user1
  writefile: !u:m7user1
  executefile: !u:m7user1
  mode: ---------
```

Delete ACEs

The following command deletes all *ACE*s associated with the specified file and sets the ACE for the file to the empty string.

```
hadoop mfs -delace /file
```

The following command deletes all *ACE*s associated with the specified directory.

```
hadoop mfs -delace /dir
```

The following command deletes all *ACE*s associated with the specified directory and *ACE*s associated with the files and directories (recursively) below the specified directory.

```
hadoop mfs -delace -R /dir
```

Create a Hard Link to File

The following command shows how to create a hard link to the file, file1, using a new name, file2.

```
# hadoop mfs -lnh /madvol1/file1 /madvol1/file2
Creating Hardlink: /madvol1/file2 -> /madvol1/file1
```

Retrieve the Number of Hard Links

The following command shows how to retrieve the number of hard links (and other statistics) associated with a given file.

```
# hadoop mfs -stat /vol1/file1
Path: /vol1/file1
  fid: 23185.32.131232
  uid: root
  gid: root

  mtime: 2016-07-01 18:01:54
  nlink: 2
  type: FTRegular
  subtype: FSTInval
  size: 1024000000
  blocksize: 268435456
  mode: 644
  networkencryption: false
  compression: off
```

View the status of the offload or recall operation for the file named file2 in volume named vol1:

```
# hadoop mfs -tierstatus  /vol1/file2
File has local data.
```

View the status of file named test1 in volume named vol1:

```
# hadoop mfs -tierstatus /vol1/test1 -v
        FID          Has Local Data
2154.109.1049824        Yes
2172.143.524906         Yes
2172.153.524926         Yes
2172.166.524952         Yes
2172.167.524954         Yes
File has local data.
```

hadoop mradmin

The `hadoop mradmin` command runs Map-Reduce administrative commands.

Warning: This command is deprecated.

Syntax

```
hadoop [ Generic Options ] mradmin
      [-refreshServiceAcl]
      [-refreshQueues]
      [-refreshNodes]
      [-refreshUserToGroupsMappings]
      [-refreshSuperUserGroupsConfiguration]
      [-help [cmd]]
```

Parameters

The following command parameters are supported for `hadoop mradmin`:

last-updated: Apr 11, 2022

Parameter	Description
`-refreshServiceAcl`	Reload the service-level authorization policy file Job tracker will reload the authorization policy file.
`-refreshQueues`	Reload the queue acls and state JobTracker will reload the mapred-queues.xml file.
`-refreshUserToGroupsMappings`	Refresh user-to-groups mappings.
`-refreshSuperUserGroupsConfiguration`	Refresh superuser proxy groups mappings.
`-refreshNodes`	Refresh the hosts information at the job tracker.
`-help [cmd]`	Displays help for the given command or all commands if none is specified.

The following generic options are supported for `hadoop mradmin`:

Generic Option	Description
`-conf <configuration file>`	Specify an application configuration file.
`-D <property=value>`	Use value for given property.
`-fs <local|filesystem URI>`	Specify a filesystem.
`-jt <local|jobtracker:port>`	Specify a job tracker.
`-files <comma separated list of files>`	Specify comma separated files to be copied to the map reduce cluster.
`-libjars <comma seperated list of jars>`	Specify comma separated jar files to include in the classpath.
`-archives <comma separated list of archives>`	Specify comma separated archives to be unarchived on the computer machines.

hadoop pipes

The `hadoop pipes` command runs a pipes job.

Warning: This command is deprecated.

Hadoop Pipes is the C++ interface to Hadoop Reduce. Hadoop Pipes uses sockets to enable tasktrackers to communicate processes running the C++ map or reduce functions.

Syntax

```
hadoop [GENERIC OPTIONS ] pipes
    [-output <path>]
    [-jar <jar file>]
    [-inputformat <class>]
    [-map <class>]
    [-partitioner <class>]
    [-reduce <class>]
    [-writer <class>]
    [-program <executable>]
    [-reduces <num>]
```


last-updated: Apr 11, 2022

Parameters

Command Options

The following command parameters are supported for `hadoop pipes`:

Parameter	Description
`-output <path>`	Specify the output directory.
`-jar <jar file>`	Specify the jar filename.
`-inputformat <class>`	InputFormat class.
`-map <class>`	Specify the Java Map class.
`-partitioner <class>`	Specify the Java Partitioner.
`-reduce <class>`	Specify the Java Reduce class.
`-writer <class>`	Specify the Java RecordWriter.
`-program <executable>`	Specify the URI of the executable.
`-reduces <num>`	Specify the number of reduces.

Generic Options

The following generic options are supported for the `hadoop pipes` command: `-conf <configuration file>`, `-D <property=value>`, `-fs <local|filesystem URI>`, `-jt <local|jobtracker:port>`, `-files <file1,file2,file3,...>`, `-libjars <libjar1,libjar2,libjar3,...>`, and `-archives <archive1,archive2,archive3,...>`. For more information on generic options, see Generic Options.

hadoop queue

The `hadoop queue` command displays job queue information.

Warning: This command is deprecated.

Syntax

```
hadoop [ Generic Options ] queue
    [-list] | [-info <job-queue-name> [-showJobs]] | [-showacls]
```

Parameters

Command Options

The `hadoop queue` command supports the following command options:

Parameter	Description
`-list`	Gets list of job queues configured in the system. Along with scheduling information associated with the job queues.

last-updated: Apr 11, 2022

Parameter	Description
`-info <job-queue-name> [-showJobs]`	Displays the job queue information and associated scheduling information of particular job queue. If `-showJobs` option is present, a list of jobs submitted to the particular job queue is displayed.
`-showacls`	Displays the queue name and associated queue operations allowed for the current user. The list consists of only those queues to which the user has access.

Generic Options

The following generic options are supported for the `hadoop queue` command: `-conf <configuration file>`, `-D <property=value>`, `-fs <local|filesystem URI>`, `-jt <local|jobtracker:port>`, `-files <file1,file2,file3,...>`, `-libjars <libjar1,libjar2,libjar3,...>`, and `-archives <archive1,archive2,archive3,...>`. For more information on generic options, see Generic Options.

hadoop version

The `hadoop version` command prints the hadoop software version.

Syntax

```
hadoop  version
```

Output

```
$ hadoop version
Hadoop 2.7.6.100-eep-800
Subversion git@github.com:mapr/private-hadoop-common -r
80dc89ae5df3a2cd01089f192c5d8a886e4788c9
Compiled by root on 2021-10-08T11:26Z
Compiled with protoc 3.11.1
From source with checksum 124ac1b54c81145154c71d2be2a66fc
This command was run using /opt/mapr/hadoop/hadoop-2.7.6/share/hadoop/
common/hadoop-common-2.7.6.100-eep-800.jar
```

hadoop conf

The `hadoop conf` command outputs the configuration information for this node to standard output.

Syntax

```
hadoop [ generic options ] conf
```

Examples

Displaying the configured value of a specific parameter

```
[user@hostname ~]$ hadoop conf | grep mapreduce.map.memory.mb
<property><name>mapreduce.map.memory.mb</name><value>1024</
value><source>mapred-site.xml</source></property>
```

The above command returns 1024 as the configured value of the `mapreduce.map.memory.mb` parameter.

last-updated: Apr 11, 2022

Dumping a node's configuration to a text file

```
[user@hostname ~]$ hadoop conf | grep ... >nodeconfiguration.txt
```

The above command creates a text file named `nodeconfiguration.txt` that contains the node's configuration information.

The following information displays when you use the `tail` utility to examine the last few lines of the file:

```
[user@hostname ~]# tail nodeconfiguration.txt
            <property><name>yarn.app.mapreduce.am.resource.mb</
name><value>1536</value><source>mapred-default.xml</source></property>
            <property><name>mapreduce.framework.name</name><value>yarn</
value><source>mapred-default.xml</source></property>
            <property><name>mapreduce.job.reduce.slowstart.completedmaps</
name><value>1.00</value><source>mapred-default.xml</source></property>
            <property><name>yarn.resourcemanager.client.thread-count</
name><value>50</value><source>yarn-default.xml</source></property>

<property><name>mapreduce.cluster.temp.dir</name><value>${hadoop.tmp.dir}/
mapred/temp</value><source>mapred-default.xml</source></property>
            <property><name>yarn.resourcemanager.staging</name><value>/var/
mapr/cluster/yarn/rm/staging</value></property>
            <property><name>fs.mapr.working.dir</name><value>/user/
$USERNAME/</value></property>
            <property><name>mapreduce.jobhistory.intermediate-done-dir</
name><value>${yarn.app.mapreduce.am.staging-dir}/history/done_intermediate</
value><source>mapred-default.xml</source></property>
            <property><name>fs.s3a.attempts.maximum</name><value>10</
value><source>core-default.xml</source></property>
            </configuration>
```

API Documentation

MapR Data Platform supports public APIs for MapR File System, MapR Database, and MapR Event Store For Apache Kafka. These APIs are available for application-development purposes.

Important: Development using MapR Data Platform non-public (private or internal) APIs is not supported.

Table

Feature	Language	Link to Location	Description
MapR Event Store For Apache Kafka Administration	Java	For source Java APIs, see: • MapR Event Store For Apache Kafka Java API Library • Apache Kafka 1.1 APIs used with MapR Event Store For Apache Kafka See MapR Event Store For Apache Kafka Java Applications on page 2754 for streams and topics information for application development.	MapR Data Platform Streams Java API for performing administrative tasks on streams and topics, setting values for the attributes of streams, and accessing streams for analytics purposes. MapR Data Platform modified Kafka interfaces, classes, and packages used for MapR Data Platform streams API. This library is a MapR Data Platform modified version of the open source Kafka library.

last-updated: Apr 11, 2022

Table (Continued)

Feature	Language	Link to Location	Description
MapR Database JSON Client	Java	MapR Database JSON Client API Database Client API **Note:** Beginning with core version 6.0, the MapR Database `Table` interface in the MapR Database JSON Client API is deprecated and replaced by the `DocumentStore` interface in the OJAI API library. For details, see the next row.	Java API for administration and management of MapR Database JSON tables.
Java OJAI Client	Java	Java OJAI Client API	OJAI is a general-purpose JSON access layer that sits on databases, file systems, and message streams and enables access to structured, semi-structured and unstructured data using a common API. Used for working with MapR Database JSON.
Java OJAI Thin Client	Java	Java OJAI Client API	Allows you to write OJAI applications in Java to access MapR Database JSON. This is the thin client version of the Java OJAI Client.
OJAI REST API	REST	Using the MapR Database JSON REST API on page 2696	Provides an alternative to writing a Java OJAI application. Using HTTP calls, you can perform basic operations on MapR Database JSON tables.
Node.js OJAI Client	Node.js	Node.js OJAI Client API provides the Node.js API documentation.	Allows you to write OJAI applications in Node.js to access MapR Database JSON.
Python OJAI Client	Python	Python OJAI Client API provides the Python API documentation.	Allows you to write OJAI applications in Python to access MapR Database JSON.
C# OJAI Client	C#	C# OJAI Client API provides the C# documentation.	Allows you to write OJAI applications in C# to access MapR Database JSON.
Go OJAI Client	Go	Go OJAI Client API provides the Go documentation.	Allows you to write OJAI applications in Go to access MapR Database JSON.
MapR Database Binary tables	C	Creating C Apps - Binary Tables on page 2453	C API library (`libMapRClient`) for creating and accessing MapR Database binary tables. This library is a MapR Database-specific version of `libhbase`.
MapR Database JSON MapReduce	Java	MapR Database JSON MapReduce API Database JSON MapReduce API	Java API library that extends the Apache Hadoop MapReduce framework. Used to create MapReduce applications that write data from one JSON table to another.
File Access Control Expressions	Java	File ACE APIs	APIs to grant different permissions to multiple users, groups, and roles for files, directories, and whole volume data using boolean expressions and subexpressions.
	C	FileACE C APIs on page 1457	
MapR File System	C	Accessing the File System with C Applications on page 2378	MapR Database `libMapRClient` library supports access to the MapR Database filesystem. This library is a MapR Database modified version of `libhdfs`. Used to manage MapR File System files.

last-updated: Apr 11, 2022

Table (Continued)

Feature	Language	Link to Location	Description
MapR File System	Java	Accessing MapR XD Distributed File and Object Store in Java Applications on page 2434	MapR Database's native maprfs library for accessing the MapR Database filesystem.
Extended Attributes	Java	Extended Attributes API	APIs to associate additional metadata with a regular file or directory.

Other Docs

This section contains release-independent information, including: MapR Installer documentation, Ecosystem release notes, interoperability matrices, security vulnerabilities, and links to other MapR version documentation.

Products Covered in the MapR Data Platform Documentation

This section lists the products covered in the MapR Data Platform documentation portal and provides links to the related product documentation.

The MapR Data Platform documentation portal provides information and instructions for the following MapR Data Platform components and features:

MapR Data Platform File Store

File Store Documentation

Most of the file store conceptual and overview documentation is located in the MapR XD Distributed File and Object Store on page 451 section of the documentation portal. Additional file store documentation, including installation and upgrade, configuration, and administration is located in the 6.1 Installation on page 107, 6.1 Administration on page 752, and 6.1 Development on page 2358 sections of the documentation portal.

The following list provides some direct links to file store topics:

- Distributed datastore for files and persistent storage for containers
- POSIX Client
- Platinum POSIX Client
- Container Client including PACC, Flex Volume Plugin and CSI
- NFSv4 and NFSv3
- Multiple file server instances per node
- HDFS API and HttpFS
- Quotas
- Snapshots
- Data Topologies

last-updated: Apr 11, 2022

- Data Protection Replication
- Multi-site volume mirroring
- Data tiering (Cold) to 3rd party external stores
- Data tiering (Warm) to erasure coding
- Global Namespace
- Compression
- Unified Security including authentication, authorization, encryption (wire and data-at-rest) and auditing
-
- HPE Ezmeral Data Fabric Management using CLI, REST, and GUI
- Rolling Upgrades
- HPE Ezmeral Data Fabric Monitoring
- HPE Ezmeral Data Fabric Installer
- Resiliency and self-healing
- Auto-balancing
- Disaster recovery - See also Mirror Volumes on page 463.
- Multitenancy on File System on page 488

MapR Data Platform Document Database

Document Database Documentation

Most of the document database conceptual and overview documentation is located in the MapR Database on page 496 section of the documentation portal. Additional document database documentation, including installation and upgrade, configuration, and administration is located in the 6.1 Installation on page 107, 6.1 Administration on page 752, and 6.1 Development on page 2358 sections of the documentation portal.

The following list provides some direct links to document database topics:

- Column-Oriented Database using HBase API
- JSON document database using OJAI API
- Multi-master table replication
- Secondary indexes
- Multiple file server instances per node
- Change Data Capture (CDC)
- HPE Ezmeral Data Fabric DB Data Access Gateway
- Strong consistency - See also Mirroring and Replication and High Availability on page 502.
- Resiliency and self-healing - See also High Availability on page 502.

last-updated: Apr 11, 2022

- Disaster recovery - See also Mirroring and Replication and High Availability on page 502.
- Automatic compactions
- Multitenancy
- Unified Security including authentication, authorization, encryption (wire and data-at-rest) and auditing
- HPE Ezmeral Data Fabric Administration
- HPE Ezmeral Data Fabric Monitoring
- HPE Ezmeral Data Fabric Installer
- Rolling upgrades
- Apache HBase

MapR Data Platform Event Data Streams

Event Data Streams Documentation

Most of the event data streams conceptual and overview documentation is located in the MapR Event Store For Apache Kafka on page 627 section of the documentation portal. Additional event data streams documentation, including installation and upgrade, configuration, and administration is located in the 6.1 Installation on page 107, 6.1 Administration on page 752, and 6.1 Development on page 2358 sections of the documentation portal..

The following list provides some direct links to event data streams topics:

- Distributed publish-subscribe messaging infrastructure
- Support for Kafka API
- Kafka Connect
- Kafka REST Proxy
- KSQL
- Kafka Streams
- Kafka Schema Registry
- Multi-site Stream replication
- Automatic partition balancing
- Multi Tenancy
- Unified Security including authentication, authorization, encryption (wire and data-at-rest) and auditing
- HPE Ezmeral Data Fabric Administration
- HPE Ezmeral Data Fabric Monitoring
- HPE Ezmeral Data Fabric Installer
- Rolling upgrades

MapR Data Platform Analytics with Hadoop

Analytics with Hadoop Documentation

Most of the documentation related to analytics with Hadoop is located in the Ecosystem Components on page 3174 section of the documentation portal.

The following list provides some direct links to analytics with Hadoop topics:

- Apache YARN
- Apache MapReduce v2
- Apache Hive
- Apache Pig
- Apache Sqoop
- Apache Flume
- Apache Oozie
- Apache Hue
- HPE Ezmeral Data Fabric DB OJAI Connector for Apache Hive
- S3 Gateway on page 3959

MapR Data Platform Advanced Analytics with Spark

Advanced Analytics with Spark Documentation

Most of the documentation related to analytics with Spark is located in the Apache Spark on page 4022 section of the documentation portal.

The following list provides some direct links to analytics with Spark topics:

- Apache YARN
- Apache Spark
- Apache Spark SQL
- Apache Spark Streaming
- Apache Spark MLlib
- GraphX
- SparkR
- Support for Spark Standalone and Spark on YARN
- HPE Ezmeral Data Fabric DB OJAI Connector for Apache Spark
- HPE Ezmeral Data Fabric DB Binary Connector for Apache Spark
- HPE Ezmeral Data Fabric Streams Integration
- S3 Gateway on page 3959

last-updated: Apr 11, 2022

MapR Data Platform Interactive SQL Engine with Drill

Drill Interactive SQL Engine Documentation

Most of the documentation related to the Drill interactive SQL engine is located in the Apache Drill on page 3185 section of the documentation portal. You may also want to refer to the Apache Drill documentation at https://drill.apache.org/docs/.

The following list provides some direct links to Drill topics in the documentation portal and on the Apache Drill site:

- Schema-less ANSI-compliant distributed SQL query engine
- Queries on File
- Queries on HPE Ezmeral Data Fabric Document Database tables and secondary indexes
- Queries on Hive tables and views - See also CREATE VIEW.
- File formats (Text,JSON,Parquet)
- Multiple data type support
- Impersonation
- Support for Drill standalone and Drill-on-YARN
- Drill query and administration UI - See also Starting the Web UI.
- JDBC/ODBC drivers
- Drill Explorer
- SQLLine - See also Configuring the Drill Shell.
- REST API - See also REST API Introduction.
- Drill Monitoring
- S3 Gateway on page 3959- See also S3 Storage Plugin.

MapR Data Platform Platform Bundle

Platform Bundle Documentation

The following list provides some direct links to platform documentation topics:

- HPE Ezmeral Data Fabric File Store
- HPE Ezmeral Data Fabric Analytics with Hadoop
- HPE Ezmeral Data Fabric Advanced Analytics with Spark
- HPE Ezmeral Data Fabric Interactive SQL Engine with Drill
- HPE Ezmeral Data Fabric Document Database
- HPE Ezmeral Data Fabric Event Data Streams

MapR Installer

You must download and run the MapR Installer setup script before you can start the MapR Installer web interface or issue MapR Installer Stanza commands.

The MapR Installer web interface simplifies the installation of a MapR cluster. After taking you through the process of selecting services and configuring the cluster, the installer installs MapR software. You can use the MapR Installer to install:

- New-feature releases, such as 6.1.0 and 6.2.0
- Maintenance releases, such as 6.0.1 and 6.1.1
- Ecosystem components, such as Spark, Hive, and HBase – or other components contained in a MapR Ecosystem Pack (EEP)

Before you begin, review the MapR Installer Prerequisites and Guidelines on page 5396, which describe user, node, and security requirements for using the Installer. For cluster-planning information, see Planning the Cluster on page 107.

Watch a Video About Setting Up and Running the Installer

To view a demonstration of setting up and running the Installer, click here.

Steps for Setting Up and Running the MapR Installer

To set up and run the installer, complete the following steps:

1. **Select a node from which to run the MapR Installer.** The node from which you run the MapR Installer does not need to be one of the nodes on which you plan to install the cluster.
2. **Download the mapr-setup.sh script.** Choose *one* of the following options:
   - Download the setup script directly from package.mapr.hpe.com to the node that will run the MapR Installer:

     MapR 6.0.0 and later

     ```
     wget https://package.mapr.hpe.com/releases/installer/
     mapr-setup.sh -P /tmp
     ```

     MapR 5.2.2 and earlier (for more information, see Selecting an Installer Version to Use on page 5402)

     ```
     wget https://package.mapr.hpe.com/releases/installer-v1.11.0/
     mapr-setup.sh -P /tmp
     ```

   - Download to your local workstation, and copy to the node that will run the MapR Installer:
     a. On the Download Page, click **Download**, and save the setup script to a location on your workstation.
     b. Use a tool such as SCP to copy the file to the node that will run the MapR Installer:

        ```
        scp mapr-setup.sh user@server:/tmp
        ```


last-updated: Apr 11, 2022

3. **Change the file permissions so that you can run the file.**

```
chmod +x /tmp/mapr-setup.sh
```

4. **Run the `mapr-setup.sh` script to configure the node to run the MapR Installer**. Run the following command as the `root` user from the directory that contains the script. The script prompts you for some information. If you have not used this script before, consider reviewing Using mapr-setup.sh on page 5404 to be prepared.

```
/tmp/mapr-setup.sh
```

5. **Start the MapR Installer.** Open the MapR Installer URL:

```
https://<Installer Node hostname/IPaddress>:9443
```

   You are prompted to log in as the cluster administrator user that you configured while running the `mapr-setup.sh` script.

Other Tasks You Can Perform with the MapR Installer

Once the initial installation completes, you can use the same MapR Installer URL to upgrade the cluster, apply a patch, or add nodes and additional services:

Use this option	To
Extend Cluster	Add a host to an existing cluster.
Incremental Install	Add or upgrade services that are already installed on the cluster.
Maintenance Update	Update your cluster to a new patch version of MapR core or apply a patch.
Version Upgrade	Upgrade the cluster to a newer MapR release, apply a patch, and upgrade services that are already installed on the cluster.
Shutdown	Stop MapR services on the cluster.
Uninstall	Remove existing MapR software before proceeding with a new installation.

Note: The MapR Installer definitions are updated frequently. See Updating the MapR Installer on page 5409 to get the latest ecosystem components and MapR software.

HPE Privacy Statement

To learn how HPE uses, shares, transfers, and manages personal information, see the HPE Privacy Statement.

Getting Started with the Installer

This section describes things you need to do to start using the MapR Installer.

MapR Installer Prerequisites and Guidelines

The node on which you run the MapR installer and the nodes you plan to include in your MapR cluster must meet certain user, connectivity, and security requirements.

Installer Requirements

The node that runs the MapR Installer must meet the following requirements:

Installer Node

Beginning with MapR Installer 1.6, the node that runs the MapR Installer does not need to be one of the nodes you plan to install the cluster on. Ensure that the default umask for the root user is set to `0022` on all MapR nodes in the cluster. You can change the umask setting in the `/etc/profile` file, or in the `.cshrc` or `.login` file. The `root` user must have a `0022` umask because the cluster admin user requires access to all files and directories under the `/opt/mapr` directory, even those initially created by `root` services.

Package Dependencies

Depending on the operating system, the MapR Installer requires the following packages. If these packages are not found, the MapR Installer attempts to download them from Internet repositories:

Ubuntu Nodes	Red Hat / CentOS Nodes	SLES Nodes
• ca-certificates • curl* • debianutils • dnsutils • iputils-arping • libnss3 • libssl1.0.0 • libsysfs2 • netcat • nfs-common • ntp • ntpdate • openssl • python-dev • python-pycurl • sdparm • sudo • syslinux • sysstat • uuid-runtime • wget	• curl* • device-mapper • iputils • libsysfs • lvm2 • nc • nfs-utils • ntp • nss • openssl • python-devel • sdparm • sudo • syslinux • sysstat • wget • which • yum-utils • compat-openssl10 (required only when running MapR 6.1.x on RHEL version 8 and above)	• ca-certificates • curl* • device-mapper • iputils • libopenssl1_0_0 • sysfsutils • lvm2 • mozilla-nss • nfs-client • ntp • sdparm • sudo • syslinux • sysstat • util-linux • wget • libfreebl3

last-updated: Apr 11, 2022

*The curl version must be greater than 7.51.0.

Repository Connectivity

The MapR Installer requires connectivity to valid repositories for the:

- Operating system
- MapR Core
- MapR Ecosystem Pack (EEP)

The Installer can connect to an Internet repository or to a preinstalled local repository, as described in Using a Local, Shared Repository With the MapR Installer on page 5418. If the Installer dependencies and MapR packages are present, but there is no connectivity to an OS repository, the Installer fails with the following message:

```
ERROR: Unable to install dependencies
(installer). Ensure that a core
OS repo is enabled and retry
mapr-setup.sh
```

Java

The MapR Installer requires JDK 1.7 or higher. Even though MapR 6.0 requires JDK 1.8 or higher, the MapR Installer can function with JDK 1.7 because the installer can run outside the MapR cluster.

If JDK 1.7 or higher is unavailable, `mapr-setup.sh` will install OpenJDK.

- On RedHat/CentOS, `mapr-setup.sh` installs Open JDK Java 1.8.
- On Ubuntu, `mapr-setup.sh` installs Open JDK Java 1.7.

For more information about the supported Java JDK versions, see the JDK Support Matrix on page 5596 and Java on page 135.

SSH Access

The Installer must have SSH access to all nodes that you want to include in the cluster.

Port Availability

Port 9443 or the non-default port that you configure using `mapr-setup.sh` must be accessible on the MapR Installer node to all nodes that you want to include in the cluster.

Files Extracted into /tmp Require Execute Privileges

Do not mount `/tmp` with the `noexec` option. The HPE Ezmeral Data Fabric extracts certain files into `/tmp` and must run them from `/tmp`. Some processes can fail if `noexec` is set for `/tmp` because some files extracted into `/tmp` require execute privileges. In addition, if you use the `java.io.tmpdir` variable to change the location of the temporary directory used by Java processes, then the newly specified temporary directory must not be mounted with the `noexec` option.

Perform the following steps to change the location of the temporary directory used by Java processes using java.io.tmpdir variable:

last-updated: Apr 11, 2022

1. Create a custom tmp directory for mapr and set its permission similar to /tmp.

```
# mkdir /opt/mapr/tmp
# chmod 1777 /opt/mapr/tmp
```

2. Set the custom `tmp` directory as `java.io.tmpdir`.
   a. For Java version 8 and previous, append the following command to `/opt/mapr/conf/env_override.sh` location.

```
export
_JAVA_OPTIONS="-Djava.io.tmpdir
=/opt/mapr/tmp"
```

   b. For Java version 9 and later, run the following command:

```
export
JDK_JAVA_OPTIONS="-Djava.io.tmp
dir=/opt/mapr/tmp"
```

3. Restart `mapr-warden` service on the node.

Note: You cannot hide the `Picked up _JAVA_OPTIONS: <…>` message due to Java sources implementation.

Supported Web Browsers

Once the MapR Installer is installed and configured, you can use the following web browsers to access the MapR Installer web interface:

- Safari
- Firefox
- Chrome

User Requirements

The installation process requires a valid cluster admin user to be present on all nodes in the cluster. The MapR Installer can create a user (the `mapr` user) for you or use a user that you have created. If you choose to create a cluster admin user, make sure the following conditions are met:

- The user must have a home directory and a password.
- The user must be present on all nodes in the cluster.
- The numeric user and group IDs (MAPR_UID and MAPR_GUID) must be configured for the user, and these values must match on all nodes.
- The `mapr` user and `root` user must be configured to use bash. Other shells are not supported.

If the user is not a valid user, installation errors can result. For information about creating the user, see Managing Users and Groups on page 752.

last-updated: Apr 11, 2022

If you choose to have the Installer create the user, the Installer runs the following command to add a local user to serve as the cluster admin user:

```
useradd -m -u $MAPR_UID -g $MAPR_GID -G $(stat -c '%G' /etc/shadow)
$MAPR_USER
```

In this command:

- MAPR_USER defaults to `mapr`.
- MAPR_UID defaults to 5000.
- MAPR_GID defaults to 5000.
- The home directory is typically `/home/mapr`.

The installer also adds the following to the MAPR_USER `.bashrc` file:

```
[[ -f /opt/mapr/conf/env.sh ]] && . /opt/mapr/conf/env.sh
```

Node Requirements

Nodes that you want to include in the cluster must meet the following criteria:

Enabling Package Repositories for SLES 15

Before using the Installer for a new data-fabric installation on SLES 15 SP2, run the following command on all nodes to enable the Python 2 package repository. You must also run the command on the Installer node if the Installer node is not part of the cluster and is running SLES 15 SP2 (or a later supported service pack):

```
SUSEConnect -p
sle-module-python2/15.<version>/x86_64
```

If you are developing applications on the cluster, run the following command on all nodes:

```
SUSEConnect -p
sle-module-development-tools/
15.<version>/x86_64
```

To view the available SLES modules and learn how to enable or disable them, use the `SUSEConnect -l` command.

Fully Qualified Domain Names (FQDNs)

The nodes are expressed as fully-qualified domain names (FQDNs), as described in Connectivity on page 133. DO NOT specify hostnames as aliases or IP addresses.

OS and Security Updates

Nodes are configured to accept operating system and security updates. They must also be patched with the latest security fixes. See your operating-system vendor documentation for details.

Disk Space Requirements

Nodes meet the requirements listed in Preparing Each Node. The MapR Installer verifies the requirements prior to installation.

OS-partition, disk, and swap-space requirements are the same whether you install the cluster manually or by

using the Installer. See Minimum Disk Space on page 132.

For data disks, MapR Installer versions 1.12.0.0 and later require a minimum disk size that is equal to the physical memory on the node. If a data disk does not meet the minimum disk size requirement, a verification error is generated.

Access to the Installer Node

Nodes have HTTPS access to the MapR Installer node over port 9443.

Proxy Server Requirements

If nodes in the cluster use an HTTP proxy server, the nodes must also meet the following requirements:

- The no_proxy environment variable must be set.

  Nodes in the MapR cluster need to be able to communicate without the use of a proxy. If the `https_proxy` and `http_proxy` environment variable is set for nodes in the cluster, you must also set the `no_proxy` environment variable for the cluster admin user and the `root` user on each node. Configure the `no_proxy` environment variable to the IP range of the MapR nodes or to the sub-domain that contains the MapR nodes.

  In addition, you must follow this guideline from the Python documentation: "The `no_proxy` environment variable can be used to specify hosts which shouldn't be reached via proxy; if set, it should be a comma-separated list of hostname suffixes, optionally with `:port` appended, for example `cern.ch,ncsa.uiuc.edu,some.host:8080`."

  For cloud-based clusters (Amazon EC2, Google Compute Engine (GCE), and Microsoft Azure), you must include this entry in the no-proxy configuration:

  ```
  169.254.169.254
  ```

- The global proxy for package repositories must be set.

  The Installer creates repository files. However, the proxy setting is not configured for each repository. Therefore, configure global proxy settings on each node in the cluster.

  - On CentOS/RedHat, set global proxy settings in `/etc/yum.conf`.
  - On Ubuntu, set global proxy settings in `/etc/apt/apt.conf`.

Security Requirements

Before installing or upgrading MapR software using the MapR Installer, make sure that you have reviewed the list of known vulnerabilities in Security Vulnerabilities on page 6569. If a vulnerability applies to your release, contact your MapR support representative for a fix, and apply the fix immediately, if applicable.

last-updated: Apr 11, 2022

Cloud Requirements

When you run the MapR Installer on nodes in the cloud, you must:

- **Verify that port 9443 is open**.

  The MapR Installer requires that this port is available.

- **Ensure that the MapR Installer and service UI URLs should refer to an external URL and not an internal URL.**

  For example, when you open the MapR Installer URL, replace any internal hostname or IP address with its associated external address. For Amazon EC2 and Google Compute Engine (GCE) clusters, the MapR Installer automatically translates internal addresses to external addresses.

- **On the Configure Nodes page of the MapR Installer web interface, make sure that you do the following:**

  - Define each node using a fully-qualified domain name (FQDN) and internal, resolvable hostnames, as described in Connectivity on page 133.

  - For the remote authentication, use the same user ID and private key that you use to ssh into your cloud instances. This user must be `root` or a user with `sudo` permissions.

Related concepts

MapR Installer Updates on page 5481
MapR Installer updates provide new features or bug fixes.

Related tasks

Checking the MapR Installer Version on page 5412
Some MapR Installer features require you to use the latest version of the Installer. You can check the MapR Installer version easily from within the user interface.

Related reference

MapR Installer Support Matrix on page 5599
This matrix shows the operating systems that are supported by the MapR Installer.

Selecting an Installer Version to Use

Beginning with the EEP 6.2.0 release, several MapR Installer versions are available for use. The version that you use depends on your current MapR core version and whether or not you need to upgrade using the MapR Installer or Stanzas.

Choosing an Installer Version

In general, you should always use the latest available MapR Installer version. The latest MapR Installer version provides access to the latest MapR Ecosystem Packs (EEPs). To download the latest version, see Installer Downloads on page 5403 later on this page. For a list of recent Installer versions, see MapR Installer Updates on page 5481.

However, the newer MapR Installer versions do not support Release 5.x. You must use Installer 1.11.0.0 or 1.12.0.0 if your cluster is on a 5.x release. Installer 1.11.0.0 is the last version of the Installer to support installation for Release 5.x. Installer 1.12.0.0 is the last version of the installer to support upgrades from Release 5.2.x to Release 6.x.

If your cluster runs Release 5.2.x or a lower version of MapR core, you must use MapR Installer 1.11.0.0 to perform the following operations:

- New installs
- Incremental installs
- Patch installs

- Maintenance updates
- Extend-cluster operations

This table describes how to select the MapR Installer version based on your current MapR software and your upgrade plans:

MapR Core Version	Use This MapR Installer Version
5.1	Use Installer 1.11.0.0 for all operations, including upgrades to Release 5.2.x. Note that you must upgrade to MapR 5.2.x with EEP 3.0.1 or later if you eventually want to upgrade to Release 6.x. After upgrading to 5.2.x, if you need to upgrade to Release 6.x, update the MapR Installer to 1.12.0.0 before upgrading.
5.2.x	Use Installer 1.11.0.0 until you need to upgrade to Release 6.x. To upgrade to Release 6.x using the MapR Installer, your cluster must have Release 5.2.x with EEP 3.0.1 or later. Before upgrading, update the MapR Installer to 1.12.0.0.
6.x and later	Except for Ubuntu 16.04 clusters, you may use the latest MapR Installer version for all operations. See the Special Considerations for Ubuntu 16.04 Clusters on page 5403.

Special Considerations for Ubuntu 16.04 Clusters

As indicated in the MapR Installer Support Matrix on page 5599, Installer 1.17 and later are supported only on Ubuntu 18.04 and 20.04. If your cluster is installed on Ubuntu 16.04:

- The Installer version that you can use depends on whether or not the Installer node is part of the cluster where you plan to install data-fabric software:
  - If the Installer node is part of the cluster, and the cluster is running Ubuntu 16.04, you must use Installer 1.16.0.x to install the software. You cannot use Installer 1.17 or later unless you upgrade the cluster to Ubuntu 18.04 or 20.04.
  - If the Installer node is not part of the cluster, and the cluster is running Ubuntu 16.04, you can use the latest Installer version on a node that is running any supported version of Ubuntu, RHEL, or SLES to install core 6.1.0 or core 6.2.
- If you need to upgrade from core 6.1.x to core 6.2, you must first upgrade your operating system to Ubuntu 18.04 or 20.04 and then install Installer 1.17 or later to perform the core software upgrade.

Installer Downloads

This table summarizes download information for the MapR Installer:

Installer Version	Download Location	The `mapr-setup.sh` script provided in this location installs . . . .
Latest MapR Installer	`package.mapr.hpe.com/releases/installer/`	The most current version of the MapR Installer
Installer 1.16.0.x	`package.mapr.hpe.com/releases/installer-v1.16`	Installer 1.16.0.2
MapR Installer 1.12.0.0	`package.mapr.hpe.com/releases/installer-v1.12.0`	MapR Installer 1.12.0.0
MapR Installer 1.11.0.0	`package.mapr.hpe.com/releases/installer-v1.11.0`	MapR Installer 1.11.0.0

last-updated: Apr 11, 2022

Downloading Older Versions of the Installer

By default, the `mapr-setup.sh` file provided in the `installer-v<version>/` directories in `http://package.mapr.com/releases/` installs the most current version of the Installer – and *not* the version for which the directory is named.

However, you can download older Installer versions by using the latest `mapr-setup.sh`:

1. Download the 1.17 or later version of `mapr-setup.sh`. See MapR Installer on page 5395.
2. Add the following environmental variable to your current shell:

   ```
   export MAPR_INSTALLER_REPO_DIR=installer-v<version>
   ```

   The `export` command tells `mapr-setup.sh` to install the Installer package from the directory you specify. For example, the following command causes `mapr-setup.sh` to install the v1.11.0 Installer:

   ```
   export MAPR_INSTALLER_REPO_DIR=installer-v1.11.0
   ```

3. Run `mapr-setup.sh` as described in MapR Installer on page 5395 to install the Installer version you specified using the `export` command.

Related concepts

Updating the MapR Installer on page 5409
Update the MapR Installer to include the latest MapR ecosystem packages and installer fixes. Once you update the MapR Installer, you can install ecosystem components and software versions that were made available after you first configured the MapR Installer.

MapR Installer on page 5395
You must download and run the MapR Installer setup script before you can start the MapR Installer web interface or issue MapR Installer Stanza commands.

Planning Your MapR Core Upgrade on page 298
Describes how to develop a successful plan for your upgrade process.

MapR Installer Updates on page 5481
MapR Installer updates provide new features or bug fixes.

Related reference

MapR Installer Support Matrix on page 5599
This matrix shows the operating systems that are supported by the MapR Installer.

EEP Support and Lifecycle Status on page 5531
This page shows the EEPs that are supported for different core releases and the current lifecycle status for each EEP.

Using mapr-setup.sh

This topic describes how you can use and run the MapR Installer setup script.

Before you can run the MapR Installer, you must run the `mapr-setup.sh` script to set up the installation environment on a node that may or may not be part of the cluster. Then, you can run the MapR Installer to perform the installation.

To download and run the `mapr-setup.sh` script, see MapR Installer on page 5395.

mapr-setup.sh

The `mapr-setup.sh` script performs the following steps to prepare the node to run the MapR Installer:

1. Verifies and installs the operating system dependencies and Java requirements on the current node.

2. Checks for Internet connectivity to the remote repository.
   - If access to https://package.mapr.hpe.com/ is not available, the script prompts for the archive tarballs. Provide the full paths of these tarballs in a space-delimited list.
3. Asks for the hostname and port that cluster nodes can use to connect to the MapR Installer node.
4. Asks for the cluster admin user account and creates the account if it does not exist. This account must exist or be created on each node in the cluster.
5. Sets up a custom `yum` or `apt` repository.
   - If no archive file is provided, the script configures access to the https://package.mapr.hpe.com/ repository. For example, on RedHat / CentOS, the script creates the following remote repository: `/etc/yum.repos.d/mapr.installer.repo`
   - If archive files are provided, the script sets up a local repository.
6. Starts the MapR Installer.

Syntax

```
/opt/mapr/installer/bin/mapr-setup.sh
[docker client]
[docker installer]
[-a <full_path_to_archive_file(s)>]
[install]
[reload]
[remove]
[update]
[-h]
[-i <full_path_to_installer_package>]
[-n]
[-p <hostname:port>]
[-r <repository_URL>]
[-y]
```

Options

In general, you should run the `mapr-setup.sh` script without using any additional options. Consider using the following options only if you have a known Internet connectivity issue, your MapR Installer packages are not located in the default repository, or you need help with the installation process.

Option	Description
`docker client`	Use this option to create a MapR PACC image. For more information, see Creating a MapR PACC Image Using mapr-setup.sh on page 409.
`docker installer`	Use this option to create a MapR Installer container.
`install`	Use this option to install the MapR Installer and definition files. If you don't specify an option for `mapr-setup.sh`, the `install` option is invoked by default. **Example:** `./mapr-setup.sh install`

last-updated: Apr 11, 2022

Option	Description
`reload`	Use this option to reinstall the MapR Installer and definition files. This option is helpful in debugging. No prompt is returned when you use this option. **Example:** `./mapr-setup.sh reload`
`remove`	Use this option to remove the MapR Installer and definition files. This option does not remove the `mapr-setup.sh` script. **Example:** `./mapr-setup.sh remove`
`update`	Use this option to update the installer packages. The setup script checks https://package.mapr.hpe.com/ for new packages and installs the packages if they are available. **Example:** `./mapr-setup.sh update`

`-a|--archives`	Use this option to bypass the Internet connectivity check and directly create a local repository. Specify a space-delimited list of the full paths to the following archive files (the order of the files is important) as an argument to the option:
	• MapR Installer archive
	• MapR Ecosystem Pack (EEP) archive
	• MapR archive for the current release
	For more information, see Using a Local, Shared Repository With the MapR Installer on page 5418.
	Note: Use this option when the MapR Installer node does not have access to the Internet or is behind a restricted VPN or firewall.
	Example for Releases 5.2 and later:
	`./mapr-setup.sh -a mapr-installer-v1.5.201705041557.deb.tgz mapr-mep-v3.0.0.201704051422.deb.tgz mapr-v5.2.1GA.deb.tgz`
	Example for Releases 5.0 and 5.1:
	`./mapr-setup.sh -a mapr-5.0-5.1.201705082100.deb.tar.gz`
	For Releases 5.0 and 5.1, you only need to provide the core archive file.

`-h|--help`	Use this option to display the command-line help for the MapR Installer. **Note:** If you use this option with other options, the MapR Installer will ignore all options except for `-h`. **Example:** `./mapr-setup.sh -h`

Option	Description
`-i|--install`	Use this option to override the MapR Installer packages stored in the remote or local repository. This option take a space-delimited list of the two local packages needed to install the Installer (the order of the files is not important): • MapR Installer package • MapR Installer definitions package **Example:** `./mapr-setup.sh -i mapr-installer-definitions_1.5.201705021610_all.deb mapr-installer_1.5.201705021610_all.deb`
`-n|--noinet`	Use this option when you don't want `mapr-setup.sh` to fetch packages from the Internet. Instead of taking the files as an argument like `-a`, this option prompts you for a complete set of archive files (the order of the files is important): • MapR Installer archive • MapR Ecosystem Pack (EEP) archive • Core archive for the current release **Note:** Configure this option when the MapR Installer node does not have access to the Internet or is behind a restricted VPN or firewall. If you use this option together with `-a`, `mapr-setup.sh` will ignore `-n`. **Example:** `./mapr-setup.sh -n`
`-p|--port`	This option specifies the `hostname:port` to use for installation-related communication between the MapR Installer node and other nodes in the cluster. The MapR Installer also adds the hostname provided as a default entry for the list of cluster nodes on the *Configure Nodes* page. Both the `hostname` and the `port` are not required when configuring this option; you can choose to configure one or both values. **Note:** Configure this option when the MapR Installer node has multiple interfaces or hostnames and the result of `hostname` is not a value that other nodes in the cluster are able to communicate with. **Example:** `./mapr-setup.sh -p perfnode131.perf.lab:9441`
`-r|--repo`	When the MapR Installer packages are not located in the default repository, specify the top-level URL of the repository in which the MapR Installer packages are located. The default URL is https://package.mapr.hpe.com/. **Example:** `./mapr-setup.sh -r http://myrepo.download.pkgs/mapr/releases/`
`-R|--new_repo_url`	Specify a new repository URL for both ecosystem and MapR core components. Use this option only with the `reload` command: **Example:** `./mapr-setup.sh -R <new_url> reload`

last-updated: Apr 11, 2022

Option	Description
`-v|--verbose`	Use this option when you want additional information about the setup process. **Example:** `./mapr-setup.sh -v`
`-y|--yes`	Use this option to bypass the MapR Installer's usual prompts and immediately proceed with the default options. This option produces the same installation result as going through the `mapr-setup.sh` script prompts and choosing all of the default options, but with increased speed. **Example:** `./mapr-setup.sh -y`

Stopping mapr-setup.sh

If you need to stop `mapr-setup.sh` while it is running, press **Ctrl + C**. Depending on when you issue the **Ctrl + C** command, the script either stops or continues to execute until it is able to stop gracefully. You can run the script again or use the `remove` option, described earlier on this page, to remove the MapR Installer and definition files and then rerun the script.

Another way to exit the script is to answer NO when the script prompts for a YES or NO reply. For example, when the script asks if you want to upgrade dependent packages, if you reply NO, the script exits.

After Running mapr-setup.sh

To validate that the MapR Installer started correctly, do one of the following:

- Log in to the MapR Installer web interface using the cluster admin user name and password.
- Run the MapR Installer Stanza `export`command using the cluster admin user name and password.

If the MapR Installer does not start up correctly, check the logs. See Logs for the MapR Installer on page 5453.

If you want to change any parameter that you provided to `mapr-setup.sh` on a previous run (for example, the repository URL, the cluster admin user name, or another parameter), you can safely rerun `mapr-setup.sh` with the new parameters. Doing so updates the MapR Installer configuration to use the new parameters. However, do not rerun `mapr-setup.sh` while an installation or a `probe` command is in progress.

MapR Installer Web Interface

When you run the MapR Installer web interface, it performs the following tasks:

1. Displays the MapR services and ecosystem components that you can install based on the software version that you select.
2. Provides the option to install MapR Monitoring.
3. Guides you through node and cluster configuration.
4. Verifies that each node meets the node requirements.
5. Sets a default, configurable service layout across the nodes in the cluster based on the requirements of each service.

last-updated: Apr 11, 2022

6. Installs or upgrades the MapR software and associated operating-system dependencies.
7. If you chose to install a trial or community license, it will attempt to apply the license to your cluster.

MapR Installer Stanzas

Running `mapr-setup.sh` also installs MapR Installer Stanzas. Stanzas give you a script-based tool to perform all the installation tasks you can perform using the MapR Installer web interface. See MapR Installer Stanzas on page 5503. In addition, Stanza commands make it possible to probe a cluster that was installed without using the MapR Installer and use the `import` command to set up the installer database. See Using probe and import to Generate the Installer Database on page 5514.

MapR Installer Components

The MapR Installer uses the following components to set up the installation environment:

Name	Filename	Description
Configuration Script	`mapr-setup.sh`	Script that configures a node to run the MapR Installer. This includes setting up an Internet or local repository. The `mapr-setup.sh` script can also be used to create a MapR Persistent Application Client Container (PACC) or a MapR Installer Container. For more information, see • Creating a MapR PACC Image Using mapr-setup.sh on page 409
Installer Package	`mapr-installer-<version>`	Package that contains the MapR Installer.
Installer Definitions Package	`mapr-installer-definitions-<version>`	Package that contains the list of software versions, services, and ecosystem components that you can install with the MapR Installer.
Service Packages	various	If you use a remote repository, the MapR Installer accesses the installation packages from https://package.mapr.hpe.com/. If you use a local repository, the MapR Installer accesses the installation packages from the local repository. The `mapr-setup.sh` script creates the local repository with the packages available in the archive files that you provide to the `mapr-setup.sh` script. For more information, see Using a Local, Shared Repository With the MapR Installer on page 5418.

Updating the MapR Installer

Update the MapR Installer to include the latest MapR ecosystem packages and installer fixes. Once you update the MapR Installer, you can install ecosystem components and software versions that were made available after you first configured the MapR Installer.

To check the version of the MapR Installer, see Checking the MapR Installer Version on page 5412.

Update the MapR Installer Using an Internet Repository

If the node that runs the MapR Installer uses an Internet repository that points to https://package.mapr.hpe.com/, use **one** of the following options to gain access to the latest packages:

- **Option 1:**

last-updated: Apr 11, 2022

1. Rename the `mapr-setup.sh` script for the current version of the Installer.
2. Download the `mapr-setup.sh` script for the version of the Installer to which you want to update. For example, if your current version of the script is 1.11 and you want to update to 1.14, download version 1.14 of the script using the steps in MapR Installer on page 5395.
3. Run the following command:

```
bash /opt/mapr/installer/bin/mapr-setup.sh update
```

- **Option 2:** Use a package manager to update the packages:

On RHEL/CentOS	`yum update mapr-installer mapr-installer-definitions`
On Ubuntu	`apt-get install mapr-installer mapr-installer-definitions`
On SLES	`zypper update mapr-installer mapr-installer-definitions`

Once the update is complete, open the MapR Installer URL (`https://<hostname/IPaddress>:9443`) to update the cluster.

Update the MapR Installer Using a Local Repository

If the node that runs the MapR Installer uses a local repository, perform the following manual steps to get the latest packages for the MapR cluster version that you are updating or upgrading to:

1. Download the latest versions of the following archive files.

last-updated: Apr 11, 2022

For Releases 5.2 and later	• **The MapR Core archive file.** Download the core archive file `mapr-<version>GA.<dep|rpm>.tgz` from one of the following locations, based on the operating system of the node:
	  • https://package.mapr.hpe.com/releases/v.<version>/redhat/
	  • https://package.mapr.hpe.com/releases/v.<version>/ubuntu/
	  **Note:** The package `mapr-v<version>GA-upgrade.<rpm/deb>.tgz` is not for use with the MapR Installer.
	• **The MapR Installer archive file**. Based on the operating system of the node, download `mapr-installer-<version>.<yyyymmdd>.<dep|rpm>.tgz` from one of the following locations:
	  • https://package.mapr.hpe.com/releases/installer/redhat/
	  • https://package.mapr.hpe.com/releases/installer/ubuntu/
	• **The MapR Ecosystem Pack (EEP) archive file**. Based on the operating system of the node, download `mapr-mep-<version>.<yyyymmdd>.<dep|rpm>.tgz` from one of the following locations:
	  • https://package.mapr.hpe.com/releases/MEP/MEP-<version>/redhat
	  • https://package.mapr.hpe.com/releases/MEP/MEP-<version>/ubuntu

For Releases 5.0 and 5.1	**The Core archive file.** Download the `mapr-5.0-5.1.<yyyymmdd><dep|rpm>.tgz` archive file from one of the following locations, based on the operating system of the node: • https://package.mapr.hpe.com/releases/installer/ubuntu/ • https://package.mapr.hpe.com/releases/installer/redhat/

2. Run the following command:

```
bash /opt/mapr/installer/bin/mapr-setup.sh -a <full path to each archive file> update
```

   For more information about `mapr-setup.sh` options, see Using mapr-setup.sh on page 5404.

3. Once the update is complete, open the MapR Installer URL (`https://<hostname/IPaddress>:9443`) to update the cluster.

Related concepts

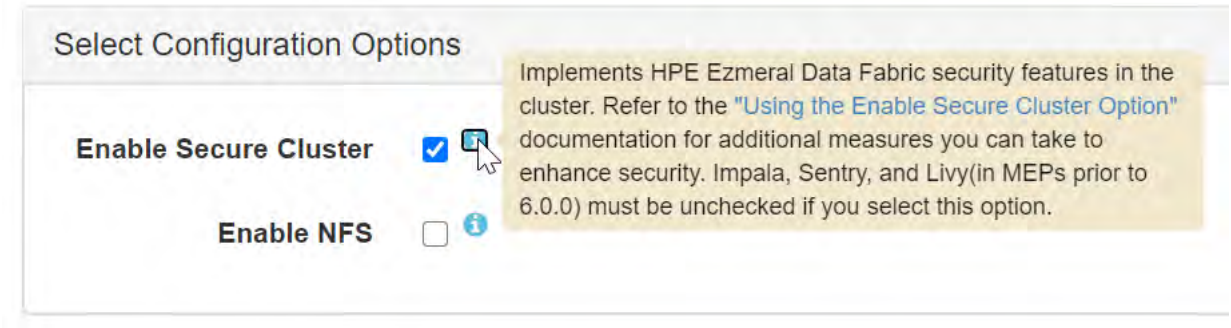
Using mapr-setup.sh on page 5404
This topic describes how you can use and run the MapR Installer setup script.

last-updated: Apr 11, 2022

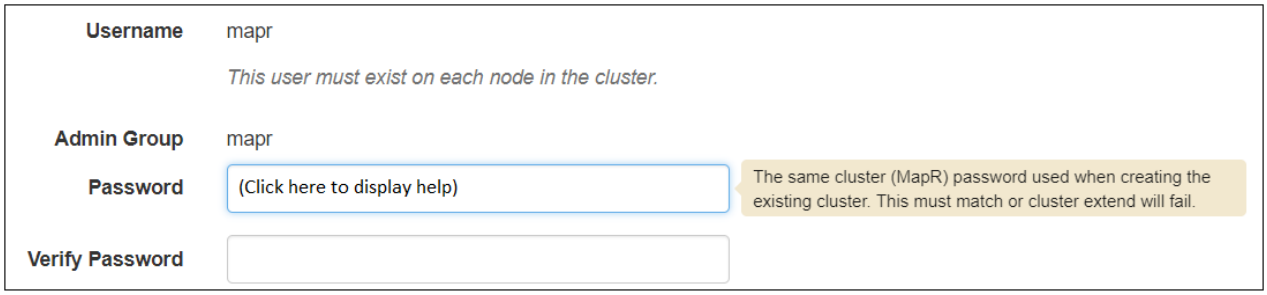
Online Help for MapR Installer Fields

This page describes how to use field-level help for the MapR Installer.

To get more out of using the MapR Installer, hold your cursor over the information icon ( ) next to each field or option. The help text can make it easier for you to use Installer options and fill in required information. For example:



If no information icon ( ) is visible near a field, place your cursor in the field to display the online help:

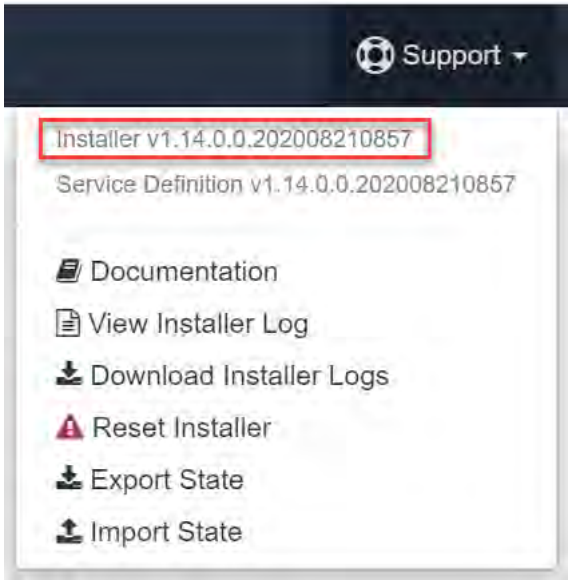


Checking the MapR Installer Version

Some MapR Installer features require you to use the latest version of the Installer. You can check the MapR Installer version easily from within the user interface.

To check the version of the currently installed MapR Installer:

1. From the MapR Installer web-based interface, click **Support**. The drop-down menu shows the MapR Installer version:



last-updated: Apr 11, 2022

2. **Optional:** Check your version against the recently released versions to see if a newer version is available. To update the MapR Installer, see Updating the MapR Installer.

Related concepts

Selecting an Installer Version to Use on page 5402
Beginning with the EEP 6.2.0 release, several MapR Installer versions are available for use. The version that you use depends on your current MapR core version and whether or not you need to upgrade using the MapR Installer or Stanzas.

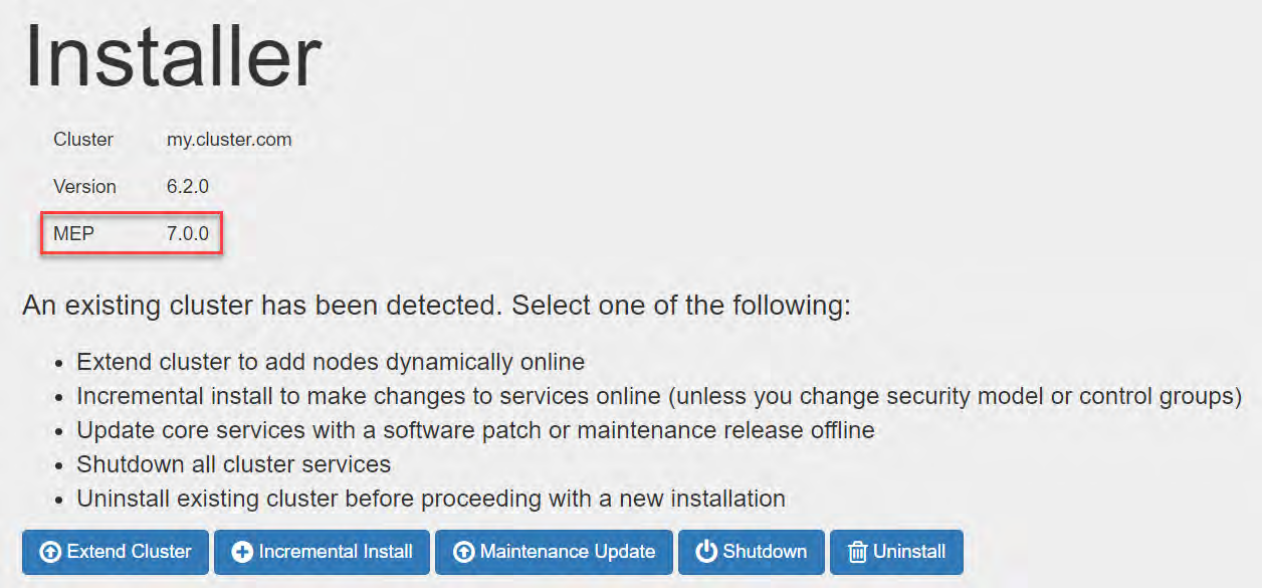
Checking the EEP Version on page 5413
Some MapR Installer operations require you to know the version of the currently installed MapR Ecosystem Pack (EEP). You can check the EEP version easily from within the MapR Installer user interface or derive the EEP version from your repository information.

Checking the EEP Version

Some MapR Installer operations require you to know the version of the currently installed MapR Ecosystem Pack (EEP). You can check the EEP version easily from within the MapR Installer user interface or derive the EEP version from your repository information.

Checking the EEP Version Using theMapR Installer

The Control System does not indicate the version of the currently installed EEP. However, if you used the MapR Installer to install the cluster, you can view the EEP version on the home page:



Checking the EEP Version in the Repository

If you installed the cluster manually, you can learn the EEP version by checking the `/etc/yum.repos.d/mapr_ecosystem.repo` file:

```
cd /etc/yum.repos.d/
ls -l
    total 52
    -rw-r--r-- 1 root root 1664 Apr 28  2018 CentOS-Base.repo
    -rw-r--r-- 1 root root 1309 Apr 28  2018 CentOS-CR.repo
    -rw-r--r-- 1 root root  649 Apr 28  2018 CentOS-Debuginfo.repo
    -rw-r--r-- 1 root root  314 Apr 28  2018 CentOS-fasttrack.repo
    -rw-r--r-- 1 root root  630 Apr 28  2018 CentOS-Media.repo
    -rw-r--r-- 1 root root 1331 Apr 28  2018 CentOS-Sources.repo
    -rw-r--r-- 1 root root 4768 Apr 28  2018 CentOS-Vault.repo
    -rw-r--r-- 1 root root  957 Dec 27  2016 epel.repo
```


last-updated: Apr 11, 2022

```
    -rw-r--r-- 1 root root 1056 Dec 27  2016 epel-testing.repo
    -rw-r--r-- 1 root root  169 May 14 10:04 mapr_core.repo
    -rw-r--r-- 1 root root  186 May 14 10:04 mapr_ecosystem.repo
   -rw-r--r-- 1 root root  171 May 13 11:25 mapr_installer.repo
more mapr_ecosystem.repo

    [MapR_Ecosystem]
    name = MapR Ecosystem Components
    baseurl = https://package.mapr.hpe.com/releases/MEP/MEP-6.2.0/redhat
    gpgcheck = 1
    enabled = 1
    protected = 1
```

Checking the EEP Version Using the Installed Packages

If the `/etc/yum.repos.d/mapr_ecosystem.repo` file is not available for any reason, another way to learn the currently installed EEP version is to check the versions of the installed packages. Use one of the following commands to display the MapR package versions:

OS	Command
On CentOS / RHEL	`yum list installed`
On SLES	`zypper packages --installed-only`
On Ubuntu	`apt list --installed`

For example:

```
yum list installed
    Loaded plugins: fastestmirror
    Loading mirror speeds from cached hostfile
     * base: mirror.sjc02.svwh.net
     * epel: mirror.sjc02.svwh.net
     * extras: mirror.sesp.northwestern.edu
     * updates: mirrors.syringanetworks.net
    Installed Packages
    ...
    mapr-apiserver.noarch            6.1.0.20180927000933-1
@MapR_Core
    mapr-asynchbase.noarch           1.7.0.201808282317-1
@MapR_Ecosystem
    mapr-cldb.x86_64                 6.1.0.20180926230239.GA-1
@MapR_Core
    mapr-collectd.x86_64             5.8.1.1.201905020932-1
@MapR_Ecosystem
    mapr-core.x86_64                 6.1.0.20180926230239.GA-1
@MapR_Core
    mapr-core-internal.x86_64        6.1.0.20180926230239.GA-1
@MapR_Core
    mapr-fileserver.x86_64           6.1.0.20180926230239.GA-1
@MapR_Core
    mapr-gateway.x86_64              6.1.0.20180926230239.GA-1
@MapR_Core
    mapr-hadoop-core.x86_64          2.7.0.20180926230239.GA-1
@MapR_Core
    mapr-hbase.noarch                1.1.8.201905020804-1
@MapR_Ecosystem
    mapr-hbase-rest.noarch           1.1.8.201905020804-1
@MapR_Ecosystem
    mapr-hbasethrift.noarch          1.1.8.201905020804-1
@MapR_Ecosystem
    mapr-installer.noarch            1.12.0.0.201905101627-1
```

```
@MapR_Installer
    mapr-installer-definitions.noarch 1.12.0.0.201905101627-1
@MapR_Installer
    mapr-kafka.noarch                 1.1.1.201901241010-1
@MapR_Ecosystem
    mapr-librdkafka.noarch            0.11.3.201901281115-1
@MapR_Ecosystem
    mapr-mapreduce2.x86_64            2.7.0.20180926230239.GA-1
@MapR_Core
    mapr-mastgateway.x86_64           6.1.0.20180926230239.GA-1
@MapR_Core
    mapr-opentsdb.noarch              2.4.0.201905082317-1
@MapR_Ecosystem
    mapr-webserver.noarch             6.1.0.20180927000933-1
@MapR_Core
    mapr-zk-internal.x86_64           6.1.0.20180926230239.GA-1
@MapR_Core
    mapr-zookeeper.x86_64             6.1.0.20180926230239.GA-1
@MapR_Core
    ...
```

Then compare the package versions against the component versions in the MapR repository. Some MapR package versions can be the same for multiple EEPs, so it is best to compare multiple packages to confirm the installed EEP.

Related concepts

MapR Ecosystem Packs on page 3174

MapR Installer on page 5395
You must download and run the MapR Installer setup script before you can start the MapR Installer web interface or issue MapR Installer Stanza commands.

Related tasks

Checking the MapR Installer Version on page 5412
Some MapR Installer features require you to use the latest version of the Installer. You can check the MapR Installer version easily from within the user interface.

Related reference

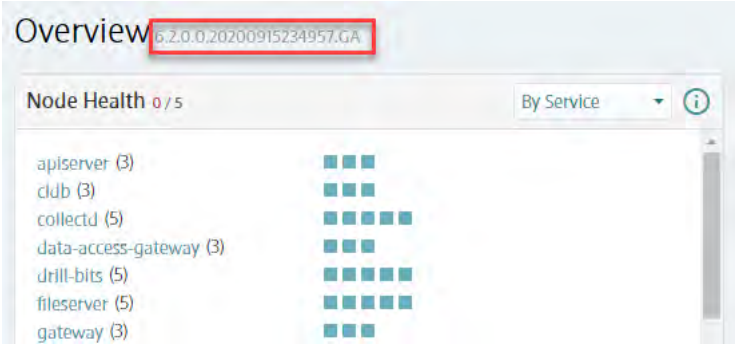
EEP Support and Lifecycle Status on page 5531
This page shows the EEPs that are supported for different core releases and the current lifecycle status for each EEP.

Checking the MapR Core Version

Some maintenance operations require you to know the version of the currently installed MapR release (sometimes referred to as the "MapR core version"). You can check the MapR core version easily from within the Control System or MapR Installer user interface or identify the version from your installed packages.

Checking the MapR Core Version Using the Control System

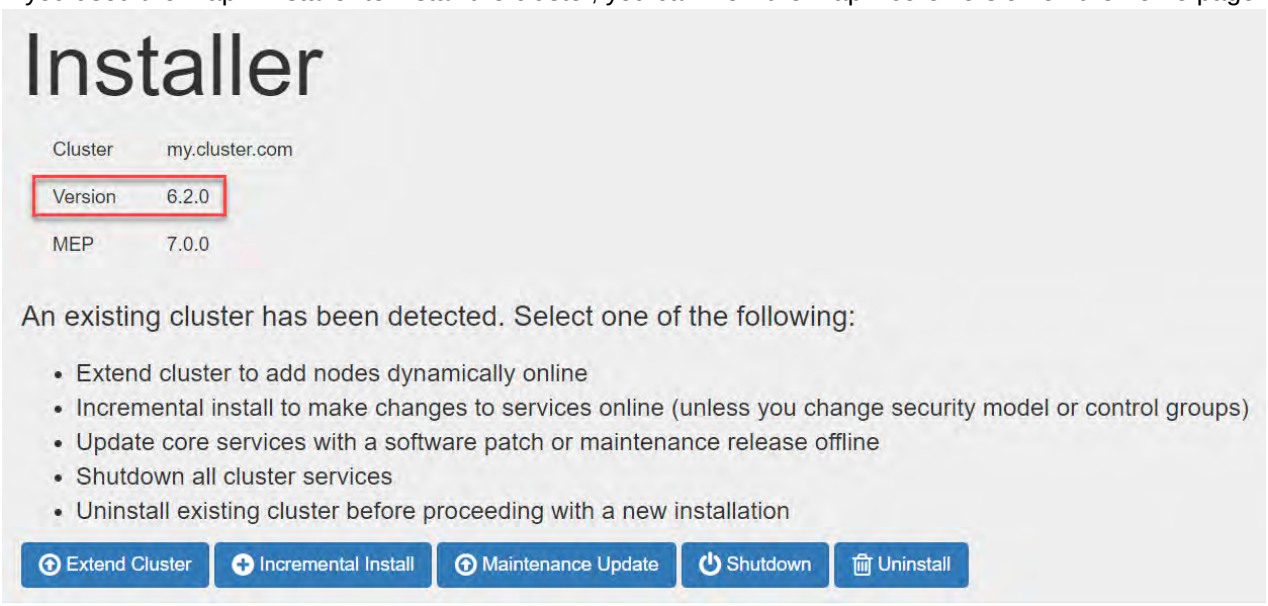
The Control System displays the MapR core version on the **Overview** page:



To connect to the Control System, see Setting Up the Control System on page 423.

Checking the MapR Core Version Using the MapR Installer

If you used the MapR Installer to install the cluster, you can view the MapR core version on the home page:



To set up the MapR Installer, see MapR Installer on page 5395.

Checking the MapR Core Version Using maprcli

Another way to view the currently installed MapR core version is to use the `maprcli dashboard info` command:

```
cd /opt/mapr/bin
maprcli dashboard info -json
    {
        "timestamp":1586880602331,
        "timeofday":"2020-04-14 09:10:02.331 GMT-0700 AM",
        "status":"OK",
        "total":1,
        "data":[
                {
                        "version":"6.1.0.20180926230239.GA",
                        "cluster":{
                                "name":"bridget.cluster.com",
                                "secure":true,
                                "dare":false,
                                "ip":"10.10.82.24",
                                "id":"6083111376716482211",
```


last-updated: Apr 11, 2022

```
                                "nodesUsed":1,
                                "totalNodesAllowed":-1
                        },
                        "volumes":{
                                "mounted":{
                                        "total":16,
                                        "size":1543
                                },
                                "unmounted":{
                                        "total":3,
                                        "size":1
                                }
                        },
                        "utilization":{
                                "cpu":{
                                        "util":38,
                                        "total":8,
                                        "active":3
                                },
    ...
```

Checking the MapR Core Version Using the Installed Packages

Another way to view the currently installed MapR core version is to check the versions of the installed `mapr-core.x86_64` or `mapr-core-internal.x86_64` packages. Use one of the following commands to display the MapR package versions:

On CentOS / Red Hat	`yum list installed`
On SLES	`zypper packages --installed-only`
On Ubuntu	`apt list --installed`

For example:

```
yum list installed
    Loaded plugins: fastestmirror
    Repodata is over 2 weeks old. Install yum-cron? Or run: yum makecache
fast
    Determining fastest mirrors
     * base: mirrors.cat.pdx.edu
     * epel: mirror.prgmr.com
     * extras: mirrors.usc.edu
     * updates: mirrors.sonic.net
    Installed Packages
    ...
    mapr-apiserver.noarch
6.1.0.20180927000933-1                  @MapR_Core
    mapr-asynchbase.noarch
1.7.0.201912020614-1                    @MapR_Ecosystem
    mapr-cldb.x86_64
6.1.0.20180926230239.GA-1               @MapR_Core
    mapr-collectd.x86_64
5.8.1.200.201911221655-1                @MapR_Ecosystem
    mapr-core.x86_64
6.1.0.20180926230239.GA-1               @MapR_Core
    mapr-core-internal.x86_64
6.1.0.20180926230239.GA-1               @MapR_Core
    mapr-drill.noarch
1.16.0.10.201912111313-1                @MapR_Ecosystem
    mapr-drill-internal.noarch
1.16.0.10.201912111313-1                @MapR_Ecosystem
```


last-updated: Apr 11, 2022

```
    mapr-fileserver.x86_64
6.1.0.20180926230239.GA-1              @MapR_Core
    mapr-gateway.x86_64
6.1.0.20180926230239.GA-1              @MapR_Core
    mapr-grafana.x86_64
6.0.2.100.201910091139-1               @MapR_Ecosystem
    mapr-hadoop-core.x86_64
2.7.0.20180926230239.GA-1              @MapR_Core
    mapr-hbase.noarch
1.1.13.0.201912080918-1                @MapR_Ecosystem
    mapr-hbase-rest.noarch
1.1.13.0.201912080918-1                @MapR_Ecosystem
    mapr-hbasethrift.noarch
1.1.13.0.201912080918-1                @MapR_Ecosystem
    mapr-historyserver.x86_64
2.7.0.20180926230239.GA-1              @MapR_Core
    mapr-installer.noarch
1.13.0.0.201912130933-1                @MapR_Installer
    mapr-installer-definitions.noarch
1.13.0.0.201912130933-1                @MapR_Installer
    mapr-kafka.noarch
1.1.1.201912120730-1                   @MapR_Ecosystem
    mapr-libhbase.x86_64
1.1.13.0.201912080918-1                @MapR_Ecosystem
    mapr-librdkafka.noarch
0.11.3.201901281115-1                  @MapR_Ecosystem
    mapr-mapreduce2.x86_64
2.7.0.20180926230239.GA-1              @MapR_Core
    mapr-mastgateway.x86_64
6.1.0.20180926230239.GA-1              @MapR_Core
    mapr-nodemanager.x86_64
2.7.0.20180926230239.GA-1              @MapR_Core
    mapr-opentsdb.noarch
2.4.0.201910311215-1                   @MapR_Ecosystem
    mapr-resourcemanager.x86_64
2.7.0.20180926230239.GA-1              @MapR_Core
    mapr-spark.noarch
2.4.4.0.201912121413-1                 @MapR_Ecosystem
    mapr-spark-historyserver.noarch
2.4.4.0.201912121413-1                 @MapR_Ecosystem
    mapr-spark-thriftserver.noarch
2.4.4.0.201912121413-1                 @MapR_Ecosystem
    mapr-webserver.noarch
6.1.0.20180927000933-1                 @MapR_Core
    mapr-zk-internal.x86_64
6.1.0.20180926230239.GA-1              @MapR_Core
    mapr-zookeeper.x86_64
6.1.0.20180926230239.GA-1              @MapR_Core
    ...
```

Related concepts

Checking the EEP Version on page 5413
Some MapR Installer operations require you to know the version of the currently installed MapR Ecosystem Pack (EEP). You can check the EEP version easily from within the MapR Installer user interface or derive the EEP version from your repository information.

Using a Local, Shared Repository With the MapR Installer

The MapR Installer can use a local repository instead of an Internet repository.

When you run the `mapr-setup.sh` script, it attempts to connect to https://package.mapr.hpe.com/ and configures an Internet repository. If there is no Internet connectivity, the script asks for archive files so that it can create a local repository.

Note: Passing `-a <full path each archive file>` to the `mapr-setup.sh` script bypasses the Internet connectivity check and automatically creates a local repository with the provided archive files.

To install with a local, shared repository, the node that runs `mapr-setup.sh` needs the following:

- **Any OS dependencies or Java Development Kit (JDK) packages that are required.**
- **A webserver.** The script will attempt to install a webserver on the node if a webserver is not available. The webserver is needed to provide the MapR package files to each node in the cluster. Note that this webserver is not configured to start automatically after a server restart. However, you can start the webserver manually by using the `systemctl start httpd` command.
- **Archive file(s).** To install release 5.2 and later, you need to download mutiple archive files. To install release 5.1 or 5.0, you need to download a single archive file.

last-updated: Apr 11, 2022

For release 5.2 and later	• **The MapR Core archive file.** Download the archive file `mapr-<version>GA.<dep|rpm>.tgz` from one of the following locations, based on the operating system of the node:
	  • https://package.mapr.hpe.com/releases/v.<version>/redhat/
	  • https://package.mapr.hpe.com/releases/v.<version>/ubuntu/
	  • https://package.mapr.hpe.com/releases/v.<version>/suse/
	  **Note:** The package `mapr-v<version>GA-upgrade.<rpm/deb>.tgz` is not for use with theMapR Installer.
	• **The MapR Installer archive file**. Based on the operating system of the node, download `mapr-installer-<version>.<yyyymmdd>.<dep|rpm>.tgz` from one of the following locations:
	  • https://package.mapr.hpe.com/releases/installer/redhat/
	  • https://package.mapr.hpe.com/releases/installer/ubuntu/
	• **The MapR Ecosystem Pack (EEP) archive file**. Based on the operating system of the node, download `mapr-mep-<version>.<yyyymmdd>.<dep|rpm>.tgz` from one of the following locations:
	  • https://package.mapr.hpe.com/releases/MEP/MEP-<version>/redhat
	  • https://package.mapr.hpe.com/releases/MEP/MEP-<version>/ubuntu
	  • https://package.mapr.hpe.com/releases/MEP/MEP-<version>/suse
	The MEP-<version> directory can be represented by a 2-digit number or a 3-digit number. The 3-digit directory contains a fixed MEP version, including patches, and is not continuously updated. The 2-digit EEP directory points to the latest MEP and patches and is continuously updated. See Understanding Two-Digit and Three-Digit EEPs on page 5450.

last-updated: Apr 11, 2022

For releases 5.0 and 5.1	• **The MapR archive file.** Download the `mapr-5.0-5.1.<yyyymmdd><dep|rpm>.tgz` archive file from one of the following locations, based on the operating system of the node:
	  • https://package.mapr.hpe.com/releases/installer/ubuntu/
	  • https://package.mapr.hpe.com/releases/installer/redhat/
	• **The MapR Core archive file.** Download the archive file `mapr-<version>GA.<dep|rpm>.tgz` from one of the following locations, based on the operating system of the node:
	  • https://package.mapr.hpe.com/releases/v.<version>/redhat/
	  • https://package.mapr.hpe.com/releases/v.<version>/ubuntu/
	  • https://package.mapr.hpe.com/releases/v.<version>/suse/
	  **Note:** The package `mapr-v<version>GA-upgrade.<rpm/deb>.tgz` is not for use with the MapR Installer.

- **The `mapr-setup.sh` script.**

 **Note:** Red Hat clusters must have access to a local EPEL repository, as described in Set Up the Internet Repository: RHEL/CentOS on page 310. The EPEL repository enables installation of these packages:

- clustershell
- pdsh
- pdsh-rcmd-ssh
- sshpass*

*Before installing a cluster on a SLES image, you must install `sshpass`, as described in Adding the MapR Repository on SUSE on page 144.

After downloading the `mapr-setup.sh` script and the archive files to the node that will run the MapR Installer, run the following command from the directory that contains the `mapr-setup.sh` script:

```
bash ./mapr-setup.sh -a <full path to the archive files>
```

MapR Installer FAQ

Review frequently asked questions about the MapR Installer.

General

What is the difference between the MapR Installer and the Quick Installer?	The MapR Installer is a robust, user-friendly replacement for the Quick Installer. You can use the MapR Installer to install a cluster with MapR

last-updated: Apr 11, 2022

	services and ecosystem components. You can also use the MapR Installer to update an existing cluster with additional nodes, MapR services, and ecosystem components. However, the MapR Installer does not install the MapR client.
Which versions of MapR software can I install?	You can use the MapR Installer to install releases 4.1, 5.0, 5.1, 5.2.x, and 6.x.
Can I use the MapR Installer to upgrade my cluster?	Yes. The MapR Installer can be used to upgrade a cluster that was installed using the MapR Installer or an Installer Stanza. See Upgrading MapR Core With the MapR Installer on page 308 for information about how to upgrade with the MapR Installer. If the cluster was manually installed, you can install the Installer and enable it to be used with subsequent installations or upgrades by following the steps in Using probe and import to Generate the Installer Database on page 5514.
Can I use the MapR Installer to install a patch?	Yes. See Applying a Patch on page 437.
Does the MapR Installer support adding a "compute-only" node?	A compute-only node is a node that is capable of performing computational tasks but is not expected to perform long-term data storage. The MapR Installer does not explicitly support adding a compute-only node to a MapR cluster. However, you can effectively work around the issue by adding a node that has the MapR File System and sufficient associated disk space. The disk space must be equal to or greater than the amount of physical memory on the node.

Preparing to Install

What are the MapR Installer requirements?	See the Prerequisites.
What information should I have before I start?	The `mapr-setup.sh` script requests the following information: • The fully-qualified domain name for each host and the port number that other nodes in the cluster can use to connect to the MapR Installer node. • A user for the cluster admin user account. If the user account doesn't exist, the `mapr-setup.sh` script prompts for the UID, GID, group name, and password so that it can create the account. The MapR Installer requests the following information: • The EEP that you want to install on a 5.2.x or later cluster • The MapR services that you want to install on the cluster • Hostnames of the nodes that you want to include in the cluster (specify fully-qualified domain names as described in Connectivity on page 133) • Credentials for the `root` user or a user with sudo privileges on each node in the cluster
What are the node requirements?	See the Prerequisites.

What are my options if I don't want to use an Internet repository?

See Using a Local, Shared Repository With the Installer.

Using the MapR Installer

Which license edition applies to my installation?

As of release 5.1, MapR licenses are categorized by new editions and modules that further define the features supported by an edition.

See the following table for descriptions of the license options. For more information about licensing, see HPE Ezmeral Data Fabric Software Licensing.

License Edition	Description
Community Edition	An unlimited, free, community-supported MapR edition with one free NFS Gateway. This edition includes Hadoop, MapR Database, and MapR Event Store For Apache Kafka. However, real-time global replication of MapR Database tables orMapR Event Store For Apache Kafka is not included.
Enterprise Edition	MapR Edition that enables enterprise-class features such as high availability, multi-tenancy, and disaster recovery. Each of the following modules for the Enterprise Edition unlocks a portion of the total platform capabilities: **Analytics** Enables enterprise-class features for analytic use cases, such as highly-available NFS and support for services like YARN and MapReduce. **Database** Enables enterprise-class features for operational NoSQL database, with MapR Database

last-updated: Apr 11, 2022

License Edition	Description
	JSON and binary tables, and real-time global database replication. **Streams** Enables enterprise-class features for publish/subscribe event streaming, with MapR Event Store For Apache Kafka and real-time global stream replication. For more information about MapR editions, see What's Included.

What happened to the M3, M5, M7, or Enterprise Edition licenses?

With the release of 5.1 and MapR Streams, the licensing model has been simplified, allowing more choice in which specific features are licensed on a cluster.

See the following table to understand how the new MapR licenses correspond to the legacy license editions that you are familiar with. For more information about licensing, see HPE Ezmeral Data Fabric Software Licensing. For more information about MapR editions, see What's Included.

Legacy Edition	New Edition & Module(s)
M3 or Community Edition	Community Edition. Starting in 5.1, the Converged Community Edition includes MapR Streams.
M5 or Enterprise Edition	Enterprise Edition with Hadoop Module
M7 or Enterprise Database Edition	Enterprise Edition with Hadoop and Database modules

What expressions can I use to specify multiple nodes?

You can enter the following types of expressions to specify nodes:

- `[0-99]` => Expands hostnames to 0, 1, 2, ...99. The second delimiter allows one or more digits.

last-updated: Apr 11, 2022

- `[00-99]` => Expands hostnames to 00,01,02,...99. Allows two or more digits of the same length
- `[a-z] or [A-Z]` => Expands hostnames to a,b,c,...z or A,B,C,...Z

To group hosts based on racks for performance or reliability, append ":" followed by the rack name to each expression.

Examples:

- `host1, host2, host3`
- `host[A-Z][0-99]` => hostA0, hostA1, hostA2 ,...hostZ99
- `host[000-333]` => host000, host001, host002, ... host333
- `host[0-3],otherhost[00-05]` => host0, host1, host2,...host3 and otherhost00,...otherhost05
- `host[0-5]:rack1,host[6-10]:rack2` => host1, host2, host3,...host5 on rack1. host6, host7, host8,...host10 on rack2

How do I change the service layout?

The MapR Installer uses groups to organize nodes and services. A group is a set of services that you can run on one or more nodes. A service can only be assigned to one group.

On the **Configure Service Layout** page, you can use the **Advanced Configuration** option to drag and drop services between existing groups to specify where the services are to be installed. You can also create new groups or change the list of nodes assigned to a group.

Note: Some services can only be assigned to one node.

Can I install a single-master service, such as Hive, on multiple nodes?

When you use the MapR Installer, single-master services are added to the default MASTER group. By design, only one node can be assigned to the MASTER group.

To install a single-master service on more than one node:

1. On the Configure Service Layout page, create a new group.
2. Assign multiple nodes to the new group.
3. Drag the single-master service to the new group.

Can I install a secure MapR cluster with the MapR Installer?

Yes. MapR Installer versions 1.10 and later support security by default.

Are there limitations to what you can do when you update an existing cluster?

See the restrictions documented in Using the Incremental Install Function on page 5443.

How do I uninstall the MapR Installer?

See Uninstalling Software Using the MapR Installer Uninstall Button on page 5452. If the installer node

last-updated: Apr 11, 2022

is part of the cluster, the MapR Installer packages can remain on the installer node after the cluster is uninstalled.

If you have uninstalled the cluster, you can also run one of the following commands to uninstall the MapR Installer packages from the installer node:

- On CentOS / Red Hat:

```
yum remove 'mapr-installer*'
```

- On Ubuntu:

```
apt-get remove 'mapr-installer*'
```

- On SLES

```
zypper remove 'mapr-installer*'
```

Troubleshooting

What can I do if I need to rerun mapr-setup.sh with a new repository URL and the script still points to the old URL?

The `/opt/mapr/installer/data/properties.json` file stores information such as the user ID of the cluster administrator, the user ID of the MapR Installer, the OS type, Internet access information, and the repository URLs for MapR Core and the ecosystem components. Once a repository URL has been stored in `properties.json`, the MapR Installer assumes that the URL will not change. Rerunning the setup script does not update the URL. Even upgrading the installer packages does not update the repository URL in `properties.json`. To pass a new repository value into `properties.json`, you have two options:

Option 1

You can remove the installer files and rerun the setup script:

```
mapr-setup.sh remove
```

Using the `remove` command removes `properties.json`, the installer database, and the installer packages, but not the setup script. After the files are removed, you can rerun the setup script to specify the new repository URL. For more information about options you can use with `mapr-setup.sh`, see Using mapr-setup.sh.

Option 2

If you need to retain the installer database and the cluster state information (for example, because you need to do an upgrade), you can:

1. Edit the `properties.json` file manually to change the `repo_core_url` and the `repo_eco_url` entries to the correct values.

last-updated: Apr 11, 2022

2. Restart the MapR Installer:

```
systemctl restart mapr-installer
```

What can I do if drop-down menus aren't working?	Try any or all of the following to correct the problem: • Refresh the browser page. • Clear the browser cache. • Close and restart the browser or browser tab.
Why does the Installer URL not work?	Check that the URL you are trying to access is external. For example, if you install on a cluster that is in the cloud, the URL that the MapR Installer lists may not work if it is an internal URL. Try accessing the external URL that is associated with the internal URL.
Why doesn't the MapR Installer list the ecosystem component that I want to install?	The MapR Installer Definitions package contains the MapR versions and services that you can install. Once you update the MapR Installer Definitions, you can install ecosystem components that were made available after you first configured the MapR Installer. See Updating the MapR Installer on page 5409.
On the Verify Nodes page, how can I get more information about a warning or error?	Hold your cursor over the warning or error in the right pane to see more information about the specific warning or error condition.
Why are the nodes listed on the Verify Nodes page different from those that I chose to install on?	If you abort an installation and then install on a different set of nodes, you must use the *Verify Nodes* page to manually remove nodes that were part of the aborted installation but are no longer part of the current installation.
I can't log in to Hue. What credentials should I use to log into Hue for the first time?	Log in with the cluster administrator username that you configured while running `mapr-setup.sh` and the password `mapr`.
What should I do if rerunning mapr-setup.sh generates errors because the properties.json file has incorrect information?	See Troubleshooting Repository URL Errors on page 5459.

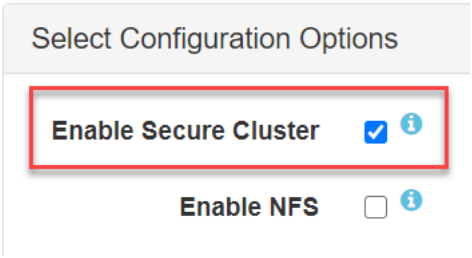
Installer Operations

This section describes operations you can perform using the MapR Installer.

Using the Enable MapR Secure Cluster Option

You use the Enable MapR Secure Cluster option to control whether or not the cluster is configured as a secure cluster.

This option appears on the **Version & Services** page of the web-based MapR Installer.



About the Enable MapR Secure Cluster Option

Using this option controls platform and ecosystem security in a MapR cluster. When you select the option, the MapR Installer runs the `configure.sh` script on the primary container location database (CLDB) to generate security keys and then distributes the keys to all the other CLDBs. The installer also distributes certificates to all the other nodes and activates security for the ecosystem components that support security.

Certain ecosystem components either do not support security or cannot be secured by the MapR Installer. If you enable security, you will not be allowed to select services such as Impala or Sentry.

Beginning with Release 6.1, data-on-wire encryption is enabled by default for newly created volumes when the **Enable** MapR **Secure Cluster** option is selected. Data-on-wire encryption encrypts data in a volume during transmission over the wire. In a secure cluster, you can enable or disable data-on-wire encryption for individual volumes using the Control System, the maprcli, or the REST API commands.

Using the Option With New and Already Installed Clusters

You can select or deselect the **Enable** MapR **Secure Cluster** option during a new installation or during an Incremental Install.

- For new installations:
  - The option is selected by default, meaning that new installations are configured with security unless you deselect the option.
  - Deselecting the option causes the cluster to be installed as a nonsecure cluster.
- For clusters that are already installed with EEP 4.0.0 or later:
  - You can select or deselect the option during an Incremental Install:
    - If security is not currently configured and you select the option, the cluster will be configured with security.
    - If security is already configured, you can remove security by deselecting the option.

      Note: If Drill is installed, be sure to review the limitations described in Securing Drill on page 3275 before removing security. Additional steps must be taken so that Drill in a nonsecured cluster can access all Drill znodes.

Using the Option During an Incremental Install

Normally, **Incremental Install** operations are conducted online. However, selecting or deselecting the **Enable** MapR **Secure Cluster** option during an **Incremental Install** requires the MapR Installer to stop the Warden and Zookeeper services, bringing the cluster offline temporarily.

In some instances, the **Enable** MapR**Secure Cluster** option is unavailable. For example, you cannot select this option during an upgrade of a nonsecured Release 5.x cluster to Release 6.0 or later. You must complete the upgrade to Release 6.0 or later first and then use the **Incremental Install** function to enable security.

Using the Enable MapR DARE Option

You use the Enable MapR DARE option to enable data-at-rest encryption for MapR volumes.

This option appears on the **Version & Services** page of the web-based MapR Installer when the **Enable** MapR **Secure Cluster** option is selected. The **Enable** MapR **DARE** option is *deselected* by default.

For more information about data-at-rest encryption, see Encryption in MapR on page 688.

last-updated: Apr 11, 2022

Considerations for Enabling MapR DARE Using the MapR Installer

Before using the **Enable** MapR **DARE** option, review these considerations:

Important: Once enabled, the **Enable** MapR **DARE** option cannot be disabled, since disk encryption cannot be turned off without reformatting the disks.

- The **Enable** MapR **DARE** option can only be used during the initial installation of a MapR cluster or during an incremental install.
- You cannot select the **Enable** MapR **DARE** option when:
  - The **Enable** MapR **Secure Cluster** option is deselected.
  - You perform an upgrade operation using the MapR Installer. Security changes are not allowed during an upgrade using the MapR Installer.
- If you select the **Enable** MapR **DARE** option, you can no longer deselect the **Enable** MapR **Secure Cluster** option during an **Incremental Install** operation.

Installing the S3 Gateway Using the Installer

Using the web-based Installer, you can install the S3 Gateway on the cluster.

You can install the S3 gateway during a new or incremental installation of Release 6.2 or later. The following option appears on the **Version & Services** page of the MapR Installer:

Object Store Gateway with S3-Compatible API (2.0.0)

Note these considerations for installing the S3 gateway using the Installer:

- Installer 1.15 installs S3 gateway 2.0.0 with EEP 7.0.0 and 7.0.1.
- If **Enable NFS** is specified, the Installer configures the S3 gateway to use NFS servers.
- If **Enable NFS** is not specified, by default the Installer installs Loopback NFS (`mapr-loopbacknfs-<version>`) on all nodes in the cluster. If you later perform an incremental install to **Enable NFS**, the Installer uninstalls Loopback NFS and uses NFS servers to interface with the S3 gateway.
- The S3 gateway:
  - Accesses the MapR file system through `/mapr`
  - Uses port 9000
  - Uses either the http or https protocol
- You cannot upgrade S3 gateway 1.0.x to S3 gateway 2.0.0 manually or by using the Installer. Upgrades are not supported.

To install the S3 gateway manually, see Installing S3 Gateway on page 204. To learn more about managing and using the S3 gateway, see S3 Gateway on page 3959.

Installing NFS Using the MapR Installer

Using the web-based MapR Installer, you can install version 3 or version 4 of the Network File System (NFS) on the cluster.

You can install NFS during a new or incremental installation of Release 6.1 or later. The **Enable** MapR **NFS** option appears on the **Version & Services** page of the MapR Installer.

Note these considerations for installing NFS using the MapR Installer:

last-updated: Apr 11, 2022

- Previous versions of the MapR Installer installed NFS (version 3) by default. MapR Installer 1.10 and later do NOT install NSF by default. You must select the **Enable** MapR **NFS** option to instruct the installer to install NFS. When you select the **Enable** MapR **NFS** option, the **NFS Version** option appears. NFSv3 is always selected by default. NFSv4 is available as an option for MapR 6.1.0 or later releases.

  CAUTION: The MapR Installer installs but does not secure NFS. Neither NSFv3 nor NSFv4 provides security by default. You can configure NFSv4 server to work with Active Directory and Kerberos servers, but you must first install Active Directory and Kerberos servers. For more information, see Configuring NFSv4 Server for Kerberos on page 1209. NFSv3 does not support security.

- With the MapR 1.10 installer, if you specify a pre-6.1.0 release, you must select the **Enable** MapR **NFS** option to install NFS. Selecting the option causes the installer to install NSFv3 by default. The **NFS Version** option is not available for pre-6.1.0 releases.

- When upgrading to Release 6.1 or later, the MapR Installer keeps the NFS version at version 3. After a new installation of Release 6.1 in which you specified NFSv3, or an upgrade to Release 6.1 using the MapR Installer, you can switch to NFSv4 by using the Incremental Install function of the installer.

- The software supports mixed-mode NFS configurations in which some nodes of a cluster use NFSv3 and other nodes use NFSv4, but mixed-mode configurations cannot be installed using theMapR Installer. The MapR Installer installs all nodes as NFSv3 or all nodes as NFSv4.

- NFSv3 and NFSv4 cannot be used on the same node concurrently.

- A new installation using a MapR Installer Stanza installs NFS only if you set `enable_NFS: true` in the config section of the stanza.

To install NFS manually, see Installing MapR NFS on page 386. To learn more about managing and using NFS, see Managing the MapR NFS Service on page 1176.

Using Custom Playbooks

Installer 1.12.0.0 and later enable you to use custom playbooks that can run a set of predefined tasks during or after operations using the MapR Installer or MapR Installer Stanzas.

Using custom playbooks, you can inject specific commands into the Installer 1.12.0.0 and later workflows to make configuration changes and ensure that those changes persist even after incremental installations or upgrades. For example, suppose you want to install a specific software package on all nodes before starting an installation. Custom playbooks enable you to check for and install the software as needed.

Or suppose you want to change a configuration setting before starting MapR Core. Custom playbooks enable you to change the setting as part of an installation or upgrade.

Restrictions to Using Custom Playbooks

Note these restrictions:

- You can use custom playbooks with any MapR cluster as long as the MapR Installer version is 1.12.0.0 or later. However, MapR Installer 1.12.0.0 and later have limited functionality when used with older MapR releases. See Selecting an Installer Version to Use on page 5402.

- Custom playbooks are not supported for manual installations. Custom playbooks are supported only for use with the MapR Installer and MapR Installer Stanzas on page 5503.

Prerequisite for Using Custom Playbooks

To create a custom playbook file, you must be familiar with Ansible. Ansible is a simple automation language that uses plain-text YAML files to describe the desired state of the cluster. You do not have to

install Ansible. Ansible 2.7 is installed whenever you load the MapR Installer using `mapr-setup.sh`. To begin learning Ansible, see these resources:

- Quick Start Video
- Getting Started with Ansible
- Introduction to Playbooks

Creating and Running a Custom Playbook

Follow these steps to use a custom playbook with the MapR Installer or MapR Installer Stanzas:

No.	Step	See for more information
1.	Prepare the roles structure and YAML files for the custom playbook.	• Predefined Roles for Custom Playbooks on page 5431 • Example Playbook Files on page 5432
2.	Test the roles using standalone Ansible.	• Testing the Roles on page 5433 • Playbook Debugger
3.	Convert the YAML files to a zipped archive file that has a `tgz` or `tar.gz` file extension.	• Creating the Zipped Archive on page 5433
4.	In the MapR Installer or a MapR Installer Stanza, specify the option to upload the zipped archive file.	• Installer Options for Custom Playbooks on page 5433 • MapR Installer Stanza Parameters for Custom Playbooks on page 5435
5.	Run the MapR Installer or MapR Installer Stanza to invoke the playbook.	• MapR Installer on page 5395 • MapR Installer Stanzas on page 5503
6.	If the operation returns a syntax error, fix the error and retry the operation.	• Troubleshooting Custom Playbook Errors on page 5435

Predefined Roles for Custom Playbooks

The role structure is an important part of your custom playbook. Your custom playbook must contain one or more of the following roles, and each role must be a directory in the zipped archive. The roles do not need to follow a specific order.

Role	Tasks in this role are run *after* . . .
preinstall	The MapR Installer has verified the nodes but before the installation workflow has begun
postecoreconfigure	`configure.sh` has run for MapR Core but before MapR Core has been started
postecoconfigure	Ecosystem components are configured and started
postinstall	The cluster is completely installed and running (at the end of the Installer Stage2 playbook)

last-updated: Apr 11, 2022

Here is an example role structure:

```
preinstall
   tasks
      main.yml
postcoreconfigure
   tasks
      main.yml
   vars
      main.yml
```

The uploaded tarball must contain relative path names with just the top-level role directories, and the tarball structure must adhere to the Ansible role-structure rules. If you need to configure your own roles in the tarball, you must include them as sub-roles to be run from within the pre-defined roles. For more information, see Ansible Roles.

Note: Because MapR Installer maintenance updates only update MapR Core and do not change any configuration settings on the cluster, the MapR Installer handles them differently from other operations. This means that whenever you perform a maintenance update, the Installer does *not* run the postecoconfigure and postinstall playbooks during the maintenance update.

Example Playbook Files

This example shows a preinstall task that installs the Midnight Commander application on all nodes in the cluster before the MapR Installer workflow is initiated. The example is contained in the `preinstall/tasks/main.yml` file:

```
---
- name: Install misc stuff - Midnight commander
  vars:
    packages_Suse: ['mc']
    packages_RedHat: ['mc', 'lsof']
    # syslinux-utils is for gethostip, libpython is required for collectd
    packages_Debian: ['mc']

  package: name={{ item }} state=present
  with_items: "{{ vars['packages_' + ansible_os_family] }}"
```

The following example shows the `postcoreconfigure/tasks/main.yml` file for the previous role structure. This example sets the number of RPC threads in `mfs.conf` from the current setting to 4 threads. The variable used to point to the file is defined in `postcoreconfigure/vars/main.yml`:

```
---
- debug: var=mapr_home

- name: Bump MFS RPC threads
  lineinfile:
    path: "{{ mapr_conf_dir }}/mfs.conf"
    regexp: '^(?P<threads>mfs.numrpcthreads=).*$'
    line: '\g<threads>4'
    backrefs: yes
```

This example shows the contents of the variables file (`postcoreconfigure/vars/main.yml`):

```
---
mapr_conf_dir: "{{ mapr_home }}/conf"
```


last-updated: Apr 11, 2022

Testing the Roles

Before uploading your zipped archive, develop and test your role files outside the installer using the standalone Ansible 2.7 debugger. You cannot use the Ansible debugger from within the MapR Installer. For more information, see Playbook Debugger.

Testing the roles by running them using the MapR Installer can be time-consuming. If a role contains logic errors, these errors won't be visible until the playbook is initiated. For example, a postinstall playbook that you run during a new installation does not generate an error until the installation is completed. A new installation can take 20 minutes or more, depending on the cluster size. Therefore, you can avoid having to re-run the Installer and playbook by testing the roles for logic errors before using them.

You can run the playbook debugger from the Installer node by using the `virt_runner` program in the`/opt/mapr/installer/bin/` directory. Use this command:

```
virt_runner ansible-playbook -i, -k <Playbook-file-name>
```

When you run the playbook in this fashion, you do not have access to the variables (for example, `mapr_home`) that the Installer defines. So you must set those up manually.

Creating the Zipped Archive

To create the zipped archive, you can use a tool such as 7-Zip or Linux commands. The following example uses the Linux `tar` command to create an archive of the directory structure and the `gzip` command to zip the archive.

Before using the `tar` command, change the directory to the directory in which you created the role structure. For example, if you created the role structure in the `/tmp` directory, you must issue the `tar` command from the `/tmp` directory:

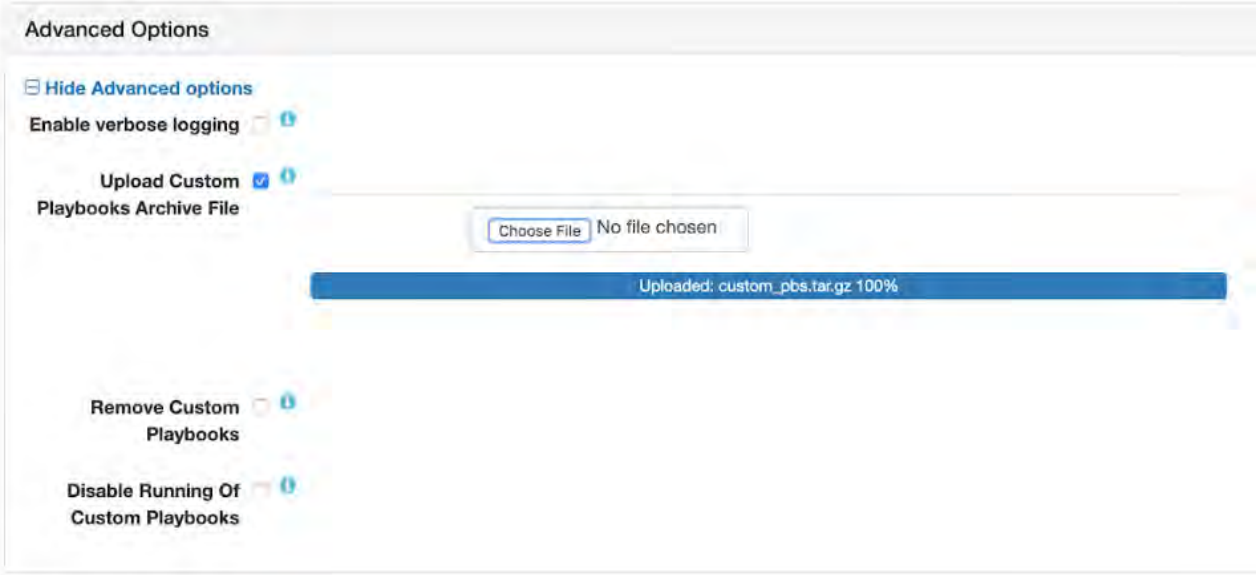
```
tar -cvf /tmp/custom_pbs.tar .
./
./preinstall/
./preinstall/tasks/
./preinstall/tasks/main.yml
./postcoreconfigure/
./postcoreconfigure/tasks/
./postcoreconfigure/tasks/main.yml
./postcoreconfigure/vars/
./postcoreconfigure/vars/main.yml
gzip /tmp/custom_pbs.tar
```

The `gzip` command creates a `custom_pbs.tar.gz` file in the `/tmp` directory. If you are using the MapR Installer, move the zipped file to the node hosting your browser so that you can upload it using the MapR Installer upload option.

Installer Options for Custom Playbooks

In the MapR Installer, the custom playbook options appear on the **Version & Services** page under **Advanced Options**. The following screen shows the options that are visible when a playbooks archive file has been uploaded:

last-updated: Apr 11, 2022



To use the options, you select an option and then advance through the Installer menus to complete the Installer task. You can access these options from the following MapR Installer tasks:

- Install
- Incremental Install
- Upgrade Version
- Maintenance Update

Whichever option you select, the Installer obeys the option *every time* you use the Installer for any operation. Therefore, if you select the option to upload a custom playbook, the Installer will run the playbook every time you use the Installer. If you select the option to disable playbooks, playbooks will be disabled until you select a different option. If a value has been set by a custom playbook and you run the same playbook again, the value is left unchanged.

Option	Description
Upload Custom Playbooks Archive File	Uploads a zipped TAR file for your custom playbook to `/opt/mapr/installer/data/tmp/custom_playbooks/`. The Installer displays the file name of the uploaded archive. An error is displayed if you upload a file that does not have a `tgz` or `tar.gz` file extension. The Installer allows you to upload one playbook at a time. If you upload a new playbook when another playbook is already loaded, the previously loaded playbook and its archive are removed, and the new playbook is loaded.
Remove Custom Playbooks	Removes the installed custom playbook and its archive from `/opt/mapr/installer/data/tmp/custom_playbooks/`. Both the playbook and the directory structure are removed.
Disable Running of Custom Playbooks	Enables you to run the MapR Installer or MapR Installer Stanzas on page 5503 *without* executing the uploaded custom playbook.

last-updated: Apr 11, 2022

MapR Installer Stanza Parameters for Custom Playbooks

To upload a custom playbook using a Stanza, specify the `cpbs_location` parameter followed by the path to the playbook in quotations. For example:

```
environment:
  mapr_core_version:6.1.0
  #DOC path to tar.gz archive of custom playbooks to be installed
  #DOC to remove the custom playbooks, provide an empty string
  #DOC for the filename, or remove cpbs_location all together
  cpbs_location: "/tmp/custom_plays.tar.gz"
```

To remove a custom playbook using a Stanza, specify an empty string for the file name:

```
environment:
  cpbs_location: ""
  ...
```

Or remove `cpbs_location` from the `environment` section.

To disable a custom playbook without removing it, set the `custom_pbs_disable` parameter to `true`

```
config:
  #DOC flag to indicate if you want to skip running of installed custom
playbooks
  custom_pbs_disable: true
```

Troubleshooting Custom Playbook Errors

The MapR Installer loads the custom roles and reports any syntax errors at execution time. The failure of a custom role causes a failure in the Installer operation. If a role has a syntax error, the MapR Installer can show an error like this:

> **Tip:** To view node details in the right pane of the **Installing MapR** screen, you need to select a red node icon on the left.

If the node detail indicates a custom playbook hook error, refer to the Installer Log for more information. Click **Support > View Installer Log**. For example:

last-updated: Apr 11, 2022

(Clicking the node log provides limited information for syntax errors.) For more information about the MapR Installer logs, see Logs for the MapR Installer on page 5453.

Using the Retry Button

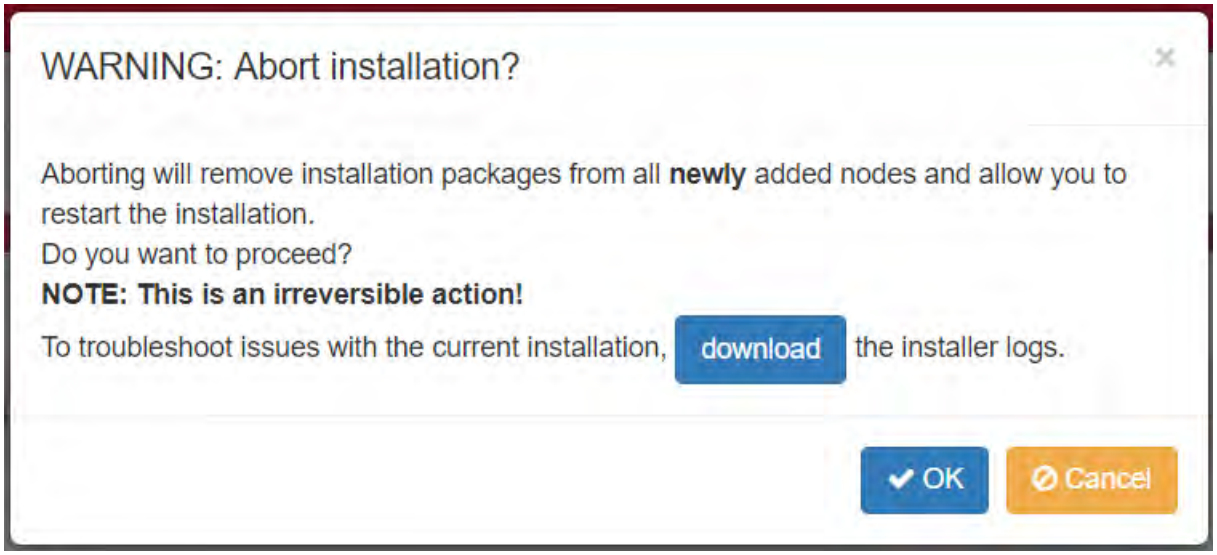
You have several options for fixing the issue. You can fix any problems by logging on to the Installer node and making changes directly in the playbook files located at `/opt/mapr/installer/data/custom_playbooks`. Then click the **Retry** button in the MapR Installer GUI interface. If you use this method, note that you must eventually propagate any fixes to your original tarball if you want to reuse the tarball.

Using the Abort Installation Button

Another option is to abort the installation and repeat the operation after fixing the error and uploading a corrected playbook. Before using this option, it is important to know that using **Abort Installation** does not remove changes made to the Installer database during the operation that you aborted. After an abort, the Installer can display the desired state rather than the actual state of the nodes.

For example, if you attempted to upgrade a EEP from EEP 6.1.0 to EEP 6.2.0 and then aborted the operation, the MapR Installer can display the currently installed EEP as EEP 6.2.0 even though EEP 6.2.0 has not been installed on the cluster nodes. This condition persists until you rerun the upgrade operation successfully or revert to the last known cluster state by using **Support > Import State**. See Importing or Exporting the Cluster State.

To abort the operation and start over, click **Abort Installation**. A confirmation dialog appears:



Click **OK** to return to the Installer home page. Fix the playbook error and retry the operation.

Extending a Cluster by Adding Nodes
This section describes how to add capacity to a MapR cluster by adding nodes.

You add nodes to a cluster by using the web-based MapR Installer (version 1.6 or later) or MapR Installer Stanzas. The nodes are added to a pre-existing group *online* without disturbing the running cluster. You can add nodes to on-premise clusters or to cloud-based clusters.

Important: After completing these steps, if you added a node to a group containing a CLDB or ZooKeeper, you must also perform the Post-Expansion Steps for Extending a Cluster on page 5442.

You can add multiple nodes to a group in the same operation, and you can add nodes to custom groups. You can also add the same node to multiple groups. The MapR Installer installs the new nodes with the same patch level as the existing nodes.

Note: Before adding nodes, use the MapR Installer to ensure that your cluster uses three-digit EEPs. If your cluster uses two-digit EEPs, adding a node can result in a version mismatch between the cluster and the newly added node. To change from two-digit to three-digit EEPs you can perform an **Incremental Install** using the MapR Installer. For more information, see Understanding Two-Digit and Three-Digit EEPs on page 5450.

Restrictions

Note these restrictions for using the **Extend Cluster** operation. When adding nodes:

- The cluster must already be installed before nodes can be added.
- You cannot add a node to a MASTER group, since these services can run only on one node.
- If you add a node to a CONTROL group that has a CLDB, you must do a manual, rolling restart of the entire cluster.
- If you add a node to a group that has OpenTSDB, you must add the same node to the group that contains AsyncHBase (currently, the MapR Installer does not check to ensure that this dependency is met).
- You cannot add services.
- You cannot change the EEP version or MapR Core version.
- You cannot add new service groups.

last-updated: Apr 11, 2022

- New nodes are added automatically to the DEFAULT group.
- A node added to a MapR secure cluster will be configured for security automatically. If the cluster is custom secure, you cannot use the MapRInstaller. See Customizing Security in MapR on page 1474.

Before Adding Nodes

1. Determine the group(s) to which new nodes will be added.
   You can add nodes to the following types of groups:
   - CLIENT
   - DATA
   - CONTROL
   - MULTI_MASTER
   - MONITORING_MASTER

   Note: If you add a node to a group containing a CLDB, you must restart all the nodes except for the new node. If you add a node to a group containing Zookeeper (but not CLDB), you must restart Zookeeper on all the Zookeeper nodes except for the new node. And you must restart the lead Zookeeper last.

   To gather information about groups, see Getting Information About Services and Groups on page 5512.
2. Ensure that the node(s) to be added meet the installation prerequisites described in the MapR Installer Prerequisites and Guidelines on page 5396.

Adding Nodes Using the MapR Installer Web Interface

1. Use a browser to connect to the cluster using the MapR Installer URL:

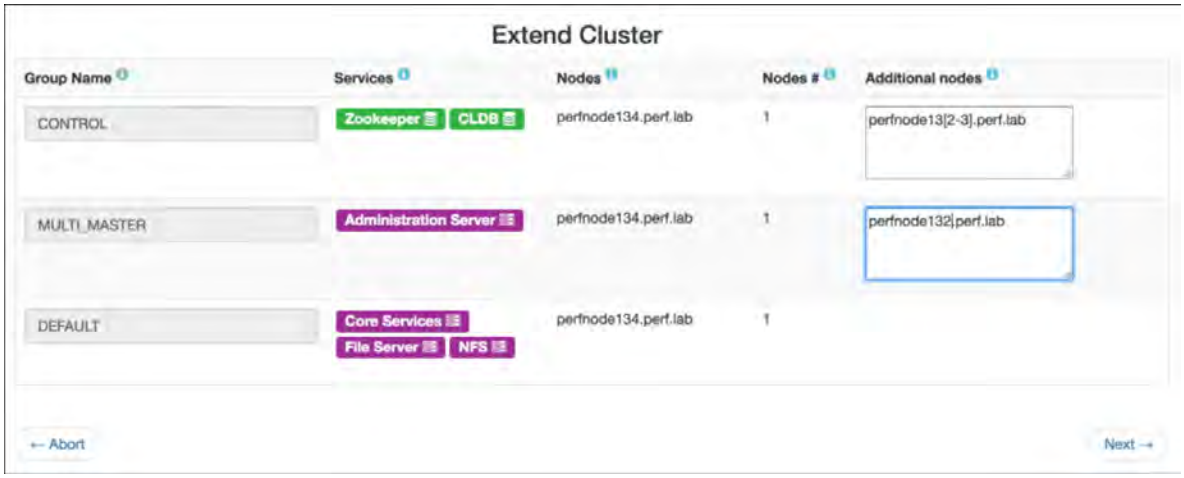
   ```
   https://<Installer Node hostname/IPaddress>:9443
   ```

2. On the status screen, click the **Extend Cluster** button. The MapR Installer displays the **Extend Cluster** screen showing the currently configured groups and services.

   Note: If the **Extend Cluster** button is not visible, you might need to clear your browser cache and refresh the browser page.
3. Specify the nodes to be added:
   - **On-Premise Cluster**

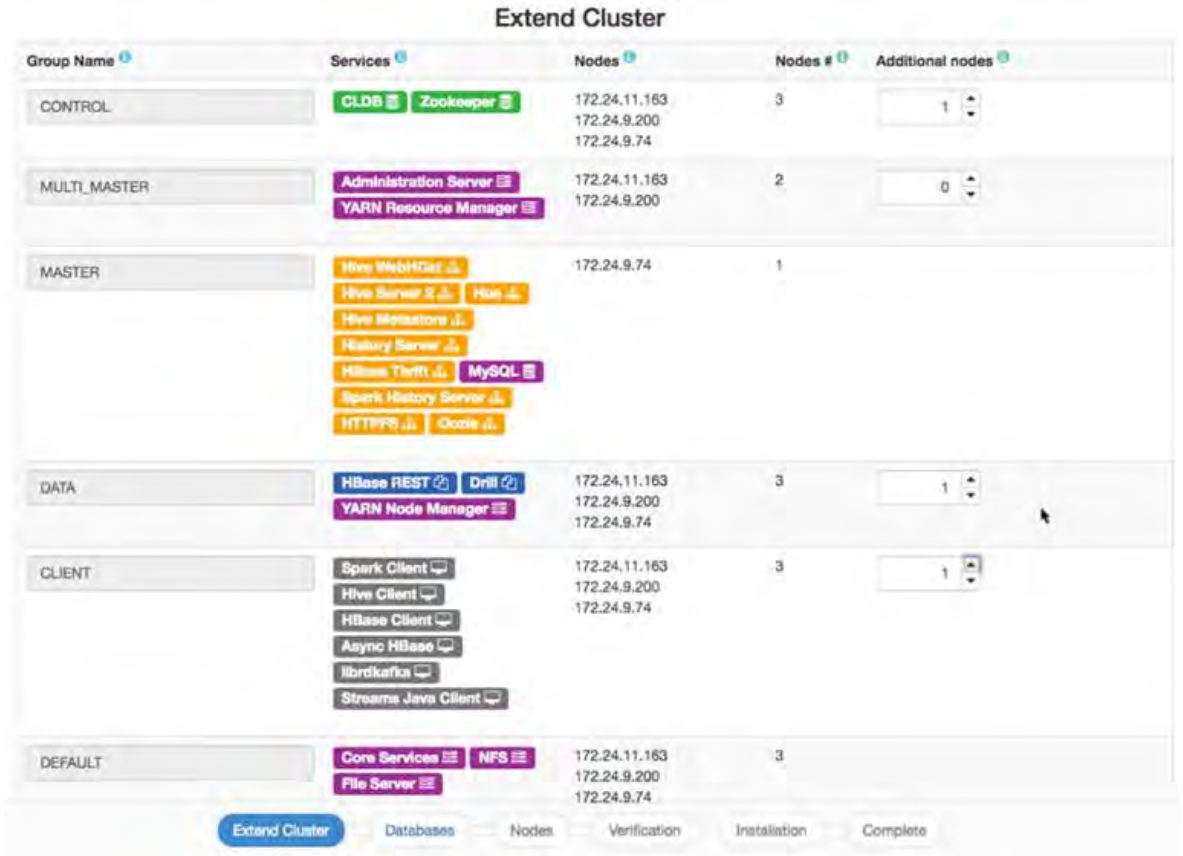
     In the **Additional Nodes** column, specify the host name(s) or IP address(es) of the nodes to be added. If you are adding multiple nodes, you can specify an array. The following example adds *perfnode132.perf.lab* and *perfnode133.perf.lab* to the CONTROL group and *perfnode132.perf.lab* to the MULTI_MASTER group:

last-updated: Apr 11, 2022



- **Cloud-Based Cluster**

  In the **Additional Nodes** column, specify the number of nodes to be added. You do not need to specify the host name or IP address. The following example adds a node to each of the CONTROL, DATA, and CLIENT groups:



4. Click **Next**. The MapR Installer checks the nodes to ensure that they are ready for installation and displays the **Authentication** screen.

last-updated: Apr 11, 2022

5. Enter your SSH password and other authentication information as needed, and click **Next**. The installer displays the **Verify Nodes** screen.

   Note: For a cloud-based cluster, the **Authentication** screen requests information that is specific to the type of cluster (AWS or Azure). Use the tooltips to learn more about the authentication information needed for your cluster.

6. Click a node icon to check the node status and see warnings or error information in the right pane. The node-icon color reflects the installation readiness for each node:
   - Green (ready to install)
   - Yellow (warning)
   - Red (cannot install)

   Note: If there are warnings or errors, hold your cursor over the warning or error in the right pane to see more information.

   Important: If node verification fails, try removing the node and retrying the operation. If node verification fails and you abort the installation, you must use the **Import State** command to reset the cluster to the last known state. Otherwise, you will not be able to perform subsequent MapR Installer operations. See Importing or Exporting the Cluster State on page 5447.

7. When you are satisfied that the nodes are ready to be installed, click **Next**. The MapR Installer adds the nodes. After the nodes are added, you can use the MapR Control System to view the nodes in the cluster.

8. Perform any post-expansion steps. Post-expansion steps are necessary only if you added a node to a group containing a CLDB or Zookeeper.

If you added a node to a group containing . . .	Do this
Zookeeper only	One node at a time, stop and restart the Zookeeper service on all Zookeeper nodes, restarting the master Zookeeper node last. You do not need to restart the Zookeeper node that was added. See Post-Expansion Steps for Extending a Cluster on page 5442.
Zookeeper and CLDB or CLDB only	One node at a time, restart all services on all nodes following the order prescribed in Manual Rolling Upgrade Description on page 319. You do not need to restart the node that was added. See Post-Expansion Steps for Extending a Cluster on page 5442.

Adding Nodes Using a MapR Installer Stanza

To add nodes using a MapR Installer Stanza, you add the `scaled_hosts2:` parameter (on-premise clusters) or the `scaled_count:` parameter (cloud-based clusters) to the Stanza file for the group that you want to scale. Then you run the Stanza using the `install` command. The services contained in the group are configured for the added node.

Note: If the group you are trying to scale does not contain the `mapr-core-5.2.x` service, the first group containing `mapr-core-5.2.x` will automatically get scaled.

1. In the Stanza file, add the `scaled_hosts2:` or `scaled_count:` parameter:

- **On-Premise Cluster**

  Add the `scaled_hosts2:` parameter to the group that you want to scale, specifying the host name(s) or IP address(s) of the nodes to be added. In the following example, the perfnode132.perf.lab node is added to the CLIENT group:

Stanza Before Scaling	Modified Stanza with scaled_hosts2 Parameter (On-Premise Cluster)

`groups:`
`- hosts:`
`  - perfnode131.perf.lab`
`  label: CLIENT`
`- services:`
`  - mapr-spark-client-2.0.1`
`  - mapr-hive-client-1.2`
`  - mapr-hbase-1.1`
`  - mapr-asynchbase-1.7.0`
`  - mapr-kafka-0.9.0`
	`groups:`
	`- hosts:`
	`  - perfnode131.perf.lab`
	`  label: CLIENT`
	`  scaled_hosts2:`
	`  - perfnode132.perf.lab`
	`- services:`
	`  - mapr-spark-client-2.0.1`
	`  - mapr-hive-client-1.2`
	`  - mapr-hbase-1.1`
	`  - mapr-asynchbase-1.7.0`
	`  - mapr-kafka-0.9.0`

- **Cloud-Based Cluster**

  Add the `scaled_count:` parameter to the group that you want to scale. Include a number after the parameter to indicate the number of additional nodes to be added to the group. You do not need to specify the host names or IP addresses of the nodes to be added. In the following example, one additional node is added to the CLIENT group:

Stanza Before Scaling	Modified Stanza with scaled_count Parameter (Cloud-Based Cluster)

`groups:`
`- hosts:`
`  - perfnode131.perf.lab`
`  label: CLIENT`
`- services:`
`  - mapr-spark-client-2.0.1`
`  - mapr-hive-client-1.2`
`  - mapr-hbase-1.1`
`  - mapr-asynchbase-1.7.0`
`  - mapr-kafka-0.9.0`
	`groups:`
	`- hosts:`
	`  - perfnode131.perf.lab`
	`  label: CLIENT`
	`  scaled_count: 1`
	`- services:`
	`  - mapr-spark-client-2.0.1`
	`  - mapr-hive-client-1.2`
	`  - mapr-hbase-1.1`
	`  - mapr-asynchbase-1.7.0`
	`  - mapr-kafka-0.9.0`

2. Run the Stanza file using the `install` command. See Installing or Upgrading MapR Core Using an Installer Stanza on page 5509. The MapR Installer SDK detects the new scaled_host(s) and gives you the option to proceed with the installation or cancel.

last-updated: Apr 11, 2022

3. Perform any post-expansion steps. Post-expansion steps are necessary only if you added a node to group containing a CLDB or Zookeeper.

If you added a node to a group containing . . .	Do this
Zookeeper only	One node at a time, stop and restart the Zookeeper service on all Zookeeper nodes, restarting the master Zookeeper node last. You do not need to restart the Zookeeper node that was added. See Post-Expansion Steps for Extending a Cluster on page 5442.
Zookeeper and CLDB or CLDB only	One node at a time, restart all services on all nodes following the order prescribed in Manual Rolling Upgrade Description on page 319. You do not need to restart the node that was added. See Post-Expansion Steps for Extending a Cluster on page 5442.

Post-Expansion Steps for Extending a Cluster
You must perform post-expansion steps if you added a node to a group containing the CLDB or Zookeeper.

For more information about extending a cluster, see Extending a Cluster by Adding Nodes on page 5437.
Restarting Zookeeper Nodes

If you used the MapR Installer **Extend Cluster** function to add a node to a group containing Zookeeper only, you must restart Zookeeper on all Zookeeper nodes except for the added node.

One node at a time, stop and restart the Zookeeper service on all Zookeeper nodes, restarting the primary Zookeeper node last. Use the following restart steps.

Restarting All Services for Zookeeper and CLDB

If you used the MapR Installer **Extend Cluster** function to add a node to a group containing Zookeeper and CLDB, or CLDB only, you must restart all services on all nodes.

One node at a time, restart all services on all nodes following the group upgrade order prescribed in Manual Rolling Upgrade Description on page 319. You do not need to restart the node that was added. Use the restart steps below.

Restart Steps

1. Change to the `root` user (or use `sudo` for the following commands).
2. Stop Warden.

```
sudo service mapr-warden stop
```

3. Stop Zookeeper.

```
service mapr-zookeeper stop
```

4. Start the ZooKeeper on nodes where it is installed.

```
service mapr-zookeeper start
```

5. On all nodes, start Warden. Example:

```
service mapr-warden start
```

6. Over a period of time (depending on the cluster size and other factors) the cluster comes up automatically. After the CLDB restarts, there is a 15-minute delay before replication resumes, in order to allow all nodes to register and heartbeat. This delay can be configured using the `config save` command to set the `cldb.replication.manager.start.mins` parameter.

Using the Incremental Install Function

Use the Incremental Install function of the web-based MapR Installer to control MapR security, add or upgrade services, upgrade MapR Ecosystem Packs (EEPs), and perform other maintenance functions.

Things You Can Do Using the Incremental Install

Using the Incremental Install function, you can:

- Enable or disable security by using the **Enable MapR Secure Cluster** option
- Add services that are supported for your current EEP
- Apply a patch
- Delete a service from a cluster by deselecting the service
- Upgrade the MapR Ecosystem Pack (EEP) to upgrade your services
- Change a 2-digit EEP to the equivalent 3-digit EEP (see Understanding Two-Digit and Three-Digit EEPs on page 5450)

Note: Before enabling security using the Incremental Install function, be sure to review the known issue (IN-1084) related to custom certificates. See MapR Installer Known Issues.

You cannot perform the following functions using an Incremental Install:

- Add a node to a cluster
- Delete a node from a cluster
- Upgrade the MapR Core version

Online Versus Offline Operations

Most **Incremental Install** operations are performed online. However, applying a patch or selecting or deselecting the **Enable MapR Secure Cluster** option are offline operations. See Using the Enable MapR Secure Cluster Option on page 5427. Making a change to security requires the MapR Installer to stop the Warden and Zookeeper services, bringing the cluster offline temporarily.

Using Incremental Install

1. Using a browser, log in to the MapR Installer:

   ```
   https://<Installer Node hostname/Ipaddress>:9443
   ```

2. Click the **Incremental Install** button. The **Version & Services** page appears.
3. Make the desired changes to add or remove security, add or delete services, apply a patch, or upgrade the EEP. Then click **Next**.
4. Advance through the MapR Installer screens, providing the admin password or other information as needed.
5. After the **Incremental Install** finishes, if you added services, use the Control System **Services** tab or the `maprcli service list -node` command to ensure that the services are running. If the services are not running, you might need to restart the nodes. For more information, see IN-1332 in MapR Installer Known Issues on page 5460.

last-updated: Apr 11, 2022

Enabling or Disabling Metrics Collection or Logging

You can use the MapR Installer to enable or disable metrics collection and logging during a new or incremental installation.

During installation using the MapR Installer, you can configure metrics and logging using settings on the **Monitoring** page of the MapR Installer user interface. Installing the metrics collection infrastructure is selected by default because the MapR Control System relies on these metrics to provide graphs and charts. Logging is deselected by default.

If you did not install metrics collection or logging during your initial installation, you can add it later by selecting the feature during an Incremental Install.

If you installed metrics collection or logging during your initial installation but you want to disable it, you can do so by deselecting the feature during an Incremental Install.

Note: If you do not install (or choose to uninstall) the metrics collection infrastructure, the MapR Control System cannot display graphs and charts.

Using the MapR Subnet and MapR External Advanced Options

Using the Installer advanced options available under Node Configuration, you can restrict the cluster to a subset of network interface cards (NICs) or specify public IP addresses that can be used with the cluster nodes.

Using these options in the Installer has the same effect as manually inserting the MAPR_SUBNETS and MAPR_EXTERNAL environment variables into the `env_override.sh` file on all nodes.

Attention: The MapR Installer does not validate the functionality of the subnets or IP addresses that you provide. If you provide incorrect values, it is possible for the installation to succeed initially and later develop connectivity issues. For this reason, it is critical that you supply accurate values for the Installer **MapR Subnet** and **MapR External** advanced options.

MapR Subnet

Allows you to set a subnet mask to restrict cluster services to certain interfaces. The values specified in this field are used to populate the MAPR_SUBNETS environment variable.

Specify one or more comma-separated subnet masks. For example:

```
10.10.15.0/24,10.10.16.0/24
```

The information on this page is specific to the web-based Installer. To configure the MAPR_SUBNETS environment variable for a manual installation, see Designating NICs for MapR on page 844.

MapR External

Allows you to specify external IP addresses for the CLDB, MapR File System, and MAST Gateway nodes.

Do NOT use DNS names. Specify a comma-separated list of tuples of host names and external IP addresses. For example:

```
node1:1.1.1.1,node2:1.1.1.2,node3:1.1.
1.3
```

The specified node names need to match the host name you specified for the host earlier in Node Configuration. The information on this page is specific to the web-based Installer. To configure the MAPR_SUBNETS environment variable for a manual

last-updated: Apr 11, 2022

installation, see Designating NICs for MapR on page 844.

Related reference

Environment Variables on page 2289
Describes the environment variables specific to the MapR Data Platform.

Related information

Designating NICs for MapR on page 844
Explains how to assign IP address blocks for MapR.

Online vs. Offline Operations

Most MapR Installer operations are offline operations, meaning that the cluster must be brought down in order to perform the operation. But there are some exceptions.

The following table shows which MapR Installer operations are offline and online:

Offline Operations	Online Operations
Upgrading MapR Core With the MapR Installer on page 308	Using the Incremental Install Function on page 5443*
Performing a Maintenance Update on page 5447	Extending a Cluster by Adding Nodes on page 5437**
Applying a Patch Using the MapR Installer on page 438	
Using the Enable MapR Secure Cluster Option on page 5427	

*Using the **Incremental Install** function to apply a patch or change security settings is an offline operation.

**Adding a node to a CONTROL group requires a manual, rolling restart of the entire cluster.

Starting Up a Cluster Using the MapR Installer Startup Button

You can use a single button to start MapR software on a cluster.

The **Startup** button is a feature of MapR Installer 1.8 or later. The **Startup** button appears on the status page of the MapR Installer web interface when the cluster is in the shutdown state. The **Startup** button starts Warden and Zookeeper, which in turn start other running services that are part of a MapR cluster.

Note: If the **Startup** button is not visible and the **Shutdown** button is present, the cluster is still running and is not in the shutdown state

The **Startup** button works differently depending on your cluster deployment:

Deployment	Startup Button Behavior
On premise	Starts the Zookeeper and Warden services on all nodes.
In the cloud	Starts the virtual machine nodes and then starts Zookeeper and Warden services on all nodes.

To use the **Startup** button:

1. For a cluster deployed in the cloud, use the AWS console or the Azure portal to ensure that the nodes are shut down before you try to start them.
2. Use a browser to connect to the cluster using the MapR Installer URL:

```
https://<Installer Node hostname/IPaddress>:9443
```

3. On the status screen, click the **Startup** button. The MapR Installer displays the **Authentication** screen.
4. Enter your authentication information if requested, and click **Startup**. The installer begins the startup process.
5. To shut down MapR software on the cluster, see Shutting Down a Cluster Using the MapR Installer Shutdown Button on page 5446.

Shutting Down a Cluster Using the MapR Installer Shutdown Button
You can use a single button to shut down MapR software on a cluster.

The **Shutdown** button appears on the status page of the MapR Installer web interface when you connect to an installed cluster using MapR Installer 1.6 or later. The **Shutdown** button shuts down Warden and Zookeeper, which in turn shut down other running services that are part of a MapR cluster. When you use the **Shutdown** button, the MapR Installer implements the same orderly shutdown used to perform software upgrades.

The **Shutdown** button works differently depending on where the cluster is deployed:

Deployment	Shutdown Button Behavior
On premise	Does not stop non-HPE software and does not power off the nodes.
In the cloud	Shuts down (but does not remove) all the nodes in the cluster: • If the installer node is part of the cluster, the installer node is not shut down, but Warden shuts down the services on the installer node. • To shut down the installer node, use AWS-console or Azure-portal commands to stop the instance. **CAUTION:** If a shutdown is initiated on an AWS cluster using an instance store, all data will be lost.

To use the **Shutdown** button:

1. Review Shutting Down a Cluster on page 795 for some pre-shutdown steps you may want to perform before shutting down Warden and Zookeeper.
2. Use a browser to connect to the cluster using the MapR Installer URL:

   ```
   https://<Installer Node hostname/IPaddress>:9443
   ```

3. On the status screen, click the **Shutdown** button. The MapR Installer asks you if you want to continue.
4. Click **OK**. The MapR Installer displays the **Authentication** screen.
5. Enter your authentication information if requested, and click **Shutdown**. The installer begins the shutdown process.
6. To restart MapR software on the cluster, see Starting Up a Cluster Using the MapR Installer Startup Button on page 5445.

   Note: Do not attempt to restart the cluster until you have confirmed that it is shut down. For an on-premise cluster, the presence of the **Startup** button on the MapR Installer status page indicates that the cluster is shut down and ready to be started. For a cluster deployed in the cloud, the **Startup** button must be present, *and* you must use the AWS or Azure console to verify that the servers are down before restarting.

Importing or Exporting the Cluster State

You can use the Import State and Export State commands to upload or download a YAML configuration file (a "Stanza") that describes the state of the cluster.

In the web-based MapR Installer, the **Import State** and **Export State** commands can be useful if you encounter a failure while using the installer and you want to revert to a previous cluster state. You can access these commands from the **Support** menu at the top of the MapR Installer user interface.

Import State	Opens the **Cluster State** dialog box, which enables you to reset the cluster to the last known state or to a desired state recorded in a YAML configuration file that you specify. You can use the **Import State** command at any time.
Export State	Downloads a YAML file capturing the current state of the cluster. You can use the **Export State** command at any time.

For more information about using MapR Installer Stanza files, see Installer Stanzas.

To import the cluster state, follow these steps:

1. Using a browser, log in to the MapR Installer:

   ```
   https://<Installer Node hostname/Ipaddress>:9443
   ```

   For more information about the MapR Installer, see MapR Installer on page 5395.

2. Click **Support > Import State**. The **Cluster State** dialog box appears.
3. Chose *one* of the following options:
   - Click **Reset** to revert the cluster to the last known state. (After a successful installation or Incremental Install using the MapR Installer or Stanzas, the last known state of the cluster is saved to `/opt/mapr/installer/data/last_known_state.yaml`.)
   - Click **Choose File**, select a YAML file, and then click **Reset** to load the YAML file.

To export the cluster state, follow these steps:

1. Using a browser, log in to the MapR Installer:

   ```
   https://<Installer Node hostname/Ipaddress>:9443
   ```

   For more information about the MapR Installer, see MapR Installer on page 5395.

2. Click **Support > Export State**.

   The cluster state is downloaded as `stanza.yaml`.

Performing a Maintenance Update

Perform a maintenance update when you want to upgrade to a new patch version of MapR core or apply a patch.

A maintenance update is an update to your installed MapR software that does not require configuration-file changes. Performing a maintenance update has no effect on the ecosystem packages (EEP components). You perform a maintenance update when you want to do either or both of the following:

- **Update to a new patch version of MapR core.** For example, you can perform a maintenance update to change your MapR core version from MapR 6.1.0 to MapR 6.1.1. You cannot use a maintenance update to change your MapR core version from a minor version, such as 6.1, to another minor version, such as 6.2. Use the **Version Upgrade** button for minor-version upgrades. The **Version Upgrade** button also permits an upgrade to a patch version of MapR core.

last-updated: Apr 11, 2022

- **Apply a patch.** The **Maintenance Update** page is one of several installer screens that offer the **Patch file** option. See Applying a Patch Using the MapR Installer on page 438.

You cannot perform a maintenance update if your current EEP version is incompatible with the selected MapR core version. For example, you cannot do a maintenance update from MapR 6.1.0 and EEP 6.3.0 to MapR 6.1.1 because EEP 6.3.0 is not compatible with MapR 6.1.1. For EEP and core compatibility information, see EEP Support and Lifecycle Status on page 5531.

Note: The maintenance update is an offline update (not a rolling update).

You perform a maintenance update using the MapR Installer. To perform a maintenance update:

1. Verify that your installed EEP is supported by the core version you plan to select for the maintenance update. To check your EEP version, see Checking the EEP Version on page 5413. For EEP and core compatibility information, see EEP Support and Lifecycle Status on page 5531.
2. Update the MapR Installer to the latest supported version. See Updating the MapR Installer on page 5409.
3. Prepare the cluster for a maintenance update by referring to one or both of these topics:
   - Preparing to Upgrade MapR Core on page 303
   - Verify Cluster Readiness for a Patch
4. Start the MapR Installer. For more information, see MapR Installer on page 5395.
5. Click the **Maintenance Update** button.
6. Change the MapR core version, or install a MapR core patch, or both.

   Important: During patch-file installation, do not refresh the browser page while the patch file is being uploaded. Doing so can interrupt the upload process.
7. Click **Next** to complete the update.

Related concepts

Checking the EEP Version on page 5413
Some MapR Installer operations require you to know the version of the currently installed MapR Ecosystem Pack (EEP). You can check the EEP version easily from within the MapR Installer user interface or derive the EEP version from your repository information.

MapR Installer Updates on page 5481
MapR Installer updates provide new features or bug fixes.

Related reference

EEP Support and Lifecycle Status on page 5531
This page shows the EEPs that are supported for different core releases and the current lifecycle status for each EEP.

Auto-Provisioning Templates

Describes the MapR Installer auto-provisioning templates.

Auto-provisioning templates let you select from different provisioning options to address a range of computing requirements. Each template provides a different mix of services and capabilities. You can select the template you need when installing using the MapR Installer web-based interface, MapR Installer Stanzas, or a cloud marketplace offering.

MapR Installer 1.14 provided the following auto-provisioning templates:

last-updated: Apr 11, 2022

Template Names in MapR Installer and Azure Templates	Template Names in AWS Templates	Template Name for Stanzas	Description
MapR Data Platform: Batch, interactive and real-time analytics	MapR-Data-Platform	`template-05-converged`	Deploys YARN, MapR Database, MapR-Streams, Drill, and Spark.
Data Lake: Common Hadoop Services	Data-Lake	`template-10-hadoop`	Provides the most common services deployed in an Apache Hadoop cluster for getting started with a Hadoop data lake. Includes YARN, MapReduce, Spark, and Hive on top of the MapR Data Platform.
Data Exploration: Interactive SQL with Apache Drill	Data-Exploration	`template-20-drill`	Provides services needed for users to perform schema-free interactive exploration of their data, including the Apache Drill SQL engine and the Hive Metastore.
MapR Database (MapR-DB) for Analytics	MapR Database-Analytics	`template-30-maprdb`	Deploys the MapR distributed NoSQL database, providing both JSON and binary data models to enable analytic applications to perform in-situ data processing.
MapR Database for Operational Applications	MapR Database-Operational-Analytics	`template-30-maprdb2`	Deploys the MapR distributed NoSQL database, providing both JSON and binary data models to enable operational applications to read and write data at high rates.
MapR Database and Distributed Query Service for Operational Applications	MapR Database-DQS-Operational-Analytics	`template-30-maprdb3`	Deploys the MapR distributed NoSQL database, providing both JSON and binary data models to enable operational applications to read and write data at high rates. It also deploys Drill as distributed query service performant distributed query execution.
Real-time Analytics: Apache Spark Streaming including SparkML and GraphX	Real-Time-Analytics-with-Streams	`template-40-maprstreams`	Deploys Spark Streaming and MapR Streams for real-time streaming applications.
MapR XD Distributed File and Object Store (MapR XD)	(Not Available)	`template-60-maprxd`	Provides common services for MapR-XD (MapR Core, MapR-FS, NFS).
Real-time and batch analytics with Apache Spark on MapR including SparkML and GraphX	Real-Time-Analytics-with-Spark	`template-60-spark`	Deploys real-time and batch analytics with Apache Spark on MapR, including SparkML and GraphX.
Custom Services	Custom-Configuration	N/A	Selecting this template allows you to customize the services that are installed. No services are selected by default. For cloud installations, see Customizing Your Deployment by Using the MapR Installer Web Interface.

MapR Installer 1.10 Updates to the Auto-Provisioning Templates

The following templates were renamed in MapR Installer 1.10:

last-updated: Apr 11, 2022

Old Name (MapR Installer 1.9)	New Name (MapR Installer 1.10)
MapR Converged Cluster: Batch, interactive and real-time analytics	MapR Data Platform: Batch, interactive and real-time analytics
Analytics with MapR Database	MapR Database (MapR-DB) for Analytics
Operational Applications with MapR Database	MapR Database for Operational Applications
Operational Applications with MapR Database and Distributed Query Service	MapR Database and Distributed Query Service for Operational Applications
MapR XD: Cloud Scale Data Platform	MapR File System and Object Store (MapR XD)

For MapR Installer 1.10, these features were added to the MapR File System and Object Store (MapR XD) template:

- MAST Gateway
- NSFv4

MapR Installer 1.9 Updates to the Auto-Provisioning Templates

For MapR Installer 1.9, the following changes to the auto-provisioning templates were implemented:

- A new auto-provisioning template was added. The **Operational Applications with MapR Database and Distributed Query Service** template includes the MapR DataBase and the OJAI Distributed Query services.
- Other templates were changed to enable the use of Drill as an optional selection with the OJAI Distributed Query Service. The following table compares the contents of the various MapR Database templates:

Template	MapR-DataBase	OJAI Distributed Query Service*	Drill
MapR Converged Cluster: Batch, interactive and real-time analytics	Y	Y	Y
Analytics with MapR Database	Y	Y	Y
Operational Applications with MapR Database	Y	N	N
Operational Applications with MapR Database and Distributed Query Service	Y	Y	N

*Prior to MapR Installer 1.9, this service was called the OJAI Query Service.

Understanding Two-Digit and Three-Digit EEPs

Understanding the differences between the EEP directories on https://package.mapr.hpe.com/releases/MEP/ can help you prevent versioning issues.

To install or update a MapR Ecosystem Pack (EEP), either manually or by using the MapR Installer, you must first choose a EEP version. The EEP version that you choose has a corresponding subdirectory on https://package.mapr.hpe.com/releases/MEP/ from which the ecosystem packages are retrieved.

For each released EEP, the https://package.mapr.hpe.com/releases/MEP/ directory includes both two-digit and three-digit subdirectories:

last-updated: Apr 11, 2022

Index of /releases/MEP

Name	Last modified	Size	Description
Parent Directory		-	
MEP-1.0.0/	02-Sep-2016 20:26	-	
MEP-1.0/	02-Sep-2016 20:26	-	
MEP-1.1.0/	29-Sep-2016 03:18	-	
MEP-1.1.1/	13-Dec-2016 19:24	-	
MEP-1.1.2/	05-Apr-2017 21:25	-	
MEP-6.0.2/	28-May-2019 16:26	-	
MEP-6.0/	28-May-2019 16:26	-	
MEP-6.1.0/	22-Feb-2019 06:38	-	
MEP-6.1.1/	28-May-2019 15:55	-	
MEP-6.1/	28-May-2019 15:55	-	
MEP-6.2.0/	14-Aug-2019 19:30	-	
MEP-6.2/	14-Aug-2019 19:30	-	
MEP-6.3.0/	13-Dec-2019 00:12	-	
MEP-6.3.1/	17-Sep-2020 08:44	-	
MEP-6.3.2/	02-Feb-2021 05:35	-	
MEP-6.3.3/	23-Mar-2021 20:49	-	
MEP-6.3.4/	01-Jun-2021 16:04	-	
MEP-6.3/	01-Jun-2021 16:04	-	
MEP-7.0.0/	17-Sep-2020 07:51	-	
MEP-7.0.1/	02-Feb-2021 05:37	-	
MEP-7.0/	02-Feb-2021 05:37	-	
MEP-7.1.0/	01-Jun-2021 16:08	-	
MEP-7.1/	01-Jun-2021 16:08	-	

The following table compares the two-digit and three-digit subdirectories and describes how they are used:

last-updated: Apr 11, 2022

EEP Subdirectory	Example	Continuously Updated?	Description
Two-digit	`MEP-3.0/`	Yes	Subdirectories using two digits (for example, MEP-3.0) contain the latest EEP and patches and are continuously updated. For example, if you select EEP 3.0 when using the MapR Installer, the installer installs or upgrades your cluster with the packages from the most current version of EEP 3.0.x. If the most current version is EEP 3.0.1, EEP 3.0.1 is installed. If the most current version is EEP 3.0.2, EEP 3.0.2 is installed, and so on. If you later decide to make changes to the cluster, the MapR Installer applies the packages from the most current version, which can be different from the version you installed originally. Two-digit EEP version numbers make new patches available automatically without the need for system reconfiguration.
Three-digit	`MEP-3.0.0/`	No	Subdirectories using three digits contain a fixed EEP version, including patches, and are not continuously updated. For example, if you select EEP 3.0.0 when using the MapR Installer, the installer uses the packages from the MEP-3.0.0 subdirectory and continues to use the MEP-3.0.0 subdirectory until you change the specified EEP version. Three-digit EEP version numbers ensure that your cluster uses only the specified EEP version. They are best for users who install MapR software manually and do not require automatic updates.

MapR Installer Use of Two-Digit and Three-Digit EEPs

MapR Installer version 1.5 automatically uses two-digit EEPs. MapR Installer versions 1.6 and later use three-digit EEP versions and do not allow you to select two-digit EEPs.

Upgrades from MapR Installer 1.5 to 1.6 or later

If you upgrade the MapR Installer from version 1.5 to a later version, you see both two-digit and three-digit EEPs in the EEP drop-down list for an Incremental Install. The two-digit EEPs continue to operate as they did previously, installing the latest three-digit EEP with the same first two digits. The three-digit EEPs operate as described earlier on this page. MapR Installer version 1.7 displays and supports only three-digit EEPs.

Note: If your cluster uses two-digit EEPs, MapR recommends that you upgrade to three-digit EEPs as soon as it is convenient to do so. Doing so enables you to use new features such as the "Extend Cluster" feature without introducing EEP version inconsistencies. You can upgrade by performing an Incremental Install and selecting a three-digit EEP.

Related concepts

MapR Installer Updates on page 5481
MapR Installer updates provide new features or bug fixes.

Related tasks

Checking the MapR Installer Version on page 5412
Some MapR Installer features require you to use the latest version of the Installer. You can check the MapR Installer version easily from within the user interface.

Uninstalling Software Using the MapR Installer Uninstall Button

You can use a single button to uninstall MapR software on all nodes in the cluster.

last-updated: Apr 11, 2022

The **Uninstall** button appears on the status page of the MapR Installer web interface when you connect to an installed cluster. The **Uninstall** button removes MapR software (but does not remove the MapR Installer) from all nodes in the cluster.

Note: You can also use MapR Installer Stanzas to uninstall software. See Uninstalling MapR Core Using an Installer Stanza on page 5511.

To use the **Uninstall** button:

1. Use a browser to connect to the cluster using the MapR Installer URL:

   ```
   https://<Installer Node hostname/IPaddress>:9443
   ```

2. On the status screen, click the **Uninstall** button. The MapR Installer displays the **Uninstall** screen.
3. Enter your SSH password and other authentication information as needed, and click **Next**. The installer begins the uninstall process.

Installer Troubleshooting

This section describes how to identify and solve problems when you use the MapR Installer.

Logs for the MapR Installer

This topic describes the logs generated by the MapR Installer and MapR Installer Stanzas.

MapR Installer logs are written to the following folder: `/opt/mapr/installer/logs`.

The following list describes each log:

`<nodename>.log[.x]`	Shows installation activities associated with a particular node. Every time you run the MapR Installer or a MapR Installer Stanza, a new copy of this log is created for each node in the cluster. When you encounter errors, you should check this log first, and then check the `mapr-installer.log`. The node log is created only if the software was able to run Ansible successfully on the node. If incorrect credentials were provided for the root user, or if there was an issue and Ansible quit before issuing the first logging callback, no log file is created for the node. If this happens, you must consult `mapr-installer.log`.
`installer.log[.x]`	Logs the REST calls sent to the installer and the database events that the installer runs. Every time you run the MapR Installer or a MapR Installer Stanza, a new copy of this log is created. If the log is not present, you might need to restart the installer (`service mapr-installer restart`).
`installer-cli.log[.x]`	Shows the progress of installation for a MapR Installer Stanza. Every time you run a MapR Installer Stanza, a new copy of this log is created.
`installer-process.log`	Serves as the top-level log file for the Ansible part of the installer. This log is created by the main Python script that runs the installer backend. This log typically shows the same information as `mapr-installer.log`.
`mapr-installer.log[.x]`	Shows the progress of Ansible scripts performed by the installer server. This log is useful if back-end issues prevent the creation of `<nodename>.log`.

last-updated: Apr 11, 2022

Every time you run the MapR Installer or a MapR Installer Stanza, a new copy of this log is created.

Creating an Archive of MapR Installer Logs
This topic describes how you can create a .zip archive of the logs generated by the MapR Installer and MapR Installer Stanzas. The .zip archive is a handy way to share log information with HPE support personnel.

To create a .zip archive of all the installer logs, use one of these commands:

- From the MapR Installer web-based interface, click **Support > Download Installer Logs**.
- From a browser, specify the following URL. When prompted, supply the user name and password for the MapR Installer:

```
https://<host_name>:9443/api/process/installer.zip
```

- From a terminal, specify the following syntax:

```
wget --no-check-certificate
https://mapr:mapr@<host_name>:9443/api/process/installer.zip
```

For information about MapR Installer logs, see Logs for the MapR Installer.

Using Service Verification
The service verification feature provides an easy way to verify that services on all nodes in the cluster are running and functional.

Important: Service verification is not currently implemented for all services. Support for additional services will be added in subsequent releases.

For secure or non-secure clusters, you can run the service verification feature from the Installer user interface. Service verification is useful after a new installation. For example, service verification can detect whether all services have successfully joined a cluster. You can run a service verification any time after the cluster is installed to check the general health of services on all nodes.

Before Using Service Verification

Before you can use the service verification feature:

- The cluster must be installed, and you must have installed it by using the Installer or Installer Stanzas.
- Ensure that the Installer is up to date. Service verification is supported only on Installer versions 1.15.0.0 and later.
- Service verification can only be performed from the Installer node. The Installer node is the node where you run the Installer.
- Running service verification requires `root`-user access (or sudoer access to `root`) for remote authentication. (When you perform service verification, the Installer node must ssh into each of the cluster nodes.)

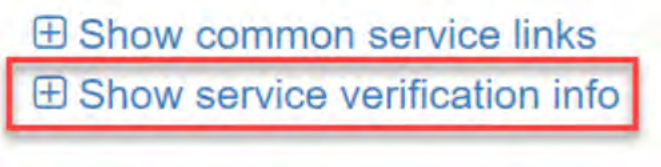
Performing Service Verification Using the Installer User Interface

To perform service verification from the Installer user interface:

1. On the Installer node, use a browser to navigate to the Installer home page, and log on as the cluster admin:

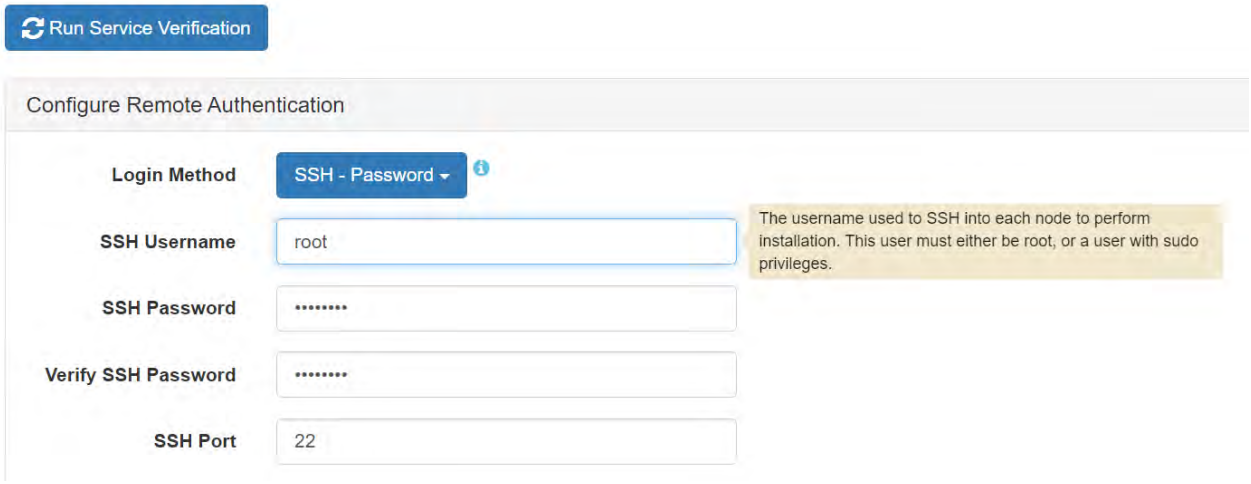
```
https://<Installer Node hostname/IPaddress>:9443
```

2. Scroll down until you see the following links:



3. Click the **Show service verification info** link to display the list of services and nodes.

4. Click **Run Service Verification** to start the verification. The Installer prompts for the `root` user password:



5. Enter the `root` user credentials, and click **Run Service Verification**. Verification can take anywhere from a few seconds to several minutes, depending on the size of the cluster, the network, and other attributes that affect performance. When the verification activity is complete, the Installer shows a list of

last-updated: Apr 11, 2022

services with the verification output for each service. For example:

Service Name			
Apiserver	NOT_IMPLEMENTED	NOT_IMPLEMENTED	NOT_INSTALLED
CLDB	NOT_IMPLEMENTED	NOT_IMPLEMENTED	NOT_IMPLEMENTED
Collectd	VERIFIED	VERIFIED	VERIFIED
Data Access Gateway	NOT_IMPLEMENTED	NOT_IMPLEMENTED	NOT_INSTALLED
File Server	NOT_IMPLEMENTED	NOT_IMPLEMENTED	NOT_IMPLEMENTED
Gateway	NOT_IMPLEMENTED	NOT_IMPLEMENTED	NOT_INSTALLED
Grafana	RUNNING_NOT_RESPONDING	RUNNING_NOT_RESPONDING	NOT_INSTALLED
HBase Client	RUNNING_NOT_RESPONDING	RUNNING_NOT_RESPONDING	RUNNING_NOT_RESPONDING
HBase Thrift	NOT_INSTALLED	NOT_INSTALLED	NOT_IMPLEMENTED
History Server	NOT_INSTALLED	NOT_INSTALLED	NOT_IMPLEMENTED
Apache Kafka Java Client	RUNNING_NOT_RESPONDING	RUNNING_NOT_RESPONDING	RUNNING_NOT_RESPONDING
Apache Kafka REST API	NOT_RUNNING	NOT_RUNNING	NOT_RUNNING
Mastgateway	NOT_IMPLEMENTED	NOT_IMPLEMENTED	NOT_IMPLEMENTED
YARN Node Manager	NOT_IMPLEMENTED	NOT_IMPLEMENTED	NOT_IMPLEMENTED
Oozie	NOT_INSTALLED	NOT_INSTALLED	NOT_RUNNING
OpenTSDB	VERIFIED	VERIFIED	VERIFIED
YARN Resource Manager	NOT_IMPLEMENTED	NOT_IMPLEMENTED	NOT_INSTALLED
Spark History Server	NOT_INSTALLED	NOT_INSTALLED	NOT_IMPLEMENTED
Spark Thrift Server	NOT_INSTALLED	NOT_INSTALLED	NOT_IMPLEMENTED
Zookeeper	NOT_IMPLEMENTED	NOT_IMPLEMENTED	NOT_IMPLEMENTED

Possible values are:

- FAILED_TO_EXECUTE
- NOT_IMPLEMENTED
- NOT_INSTALLED
- NOT_RUNNING
- NOT_STARTED
- RUNNING_NOT_RESPONDING
- VERIFIED

6. To rerun service verification – for example, after running it once and discovering some services are not responding – click **Run Service Verification** again.

Service Verification Logs

On each node where a service is installed, logged output from the service verification feature is saved to:

```
$MAPR_HOME/<service>/<service>-<version>/var/log/<service>/
verify_service.<date>
```

For example, service verification output for OpenTSDB might be found here:

```
# pwd
/opt/mapr/opentsdb/opentsdb-2.4.0/var/log/opentsdb
```

```
# ls
metrics_tmp                     ot_purgeData.log-20210414.gz
opentsdb_daemon.log             ot_purgeData.log-20210415.gz
opentsdb.err                    ot_purgeData.log-20210416.gz
opentsdb_install.log            ot_purgeData.log-20210417.gz
opentsdb.out                    ot_purgeData.log-20210418.gz
opentsdb_scandaemon.log         ot_purgeData.log-20210419.gz
opentsdb_scandaemon_query.log   ot_purgeData.log-20210420.gz
opentsdb_startup.log            ot_purgeData.log-20210421.gz
opentsdb_startup.log.1          ot_purgeData.log-20210422.gz
opentsdb_startup.log.2          ot_purgeData.log-20210423.gz
opentsdb_startup.log.3          ot_purgeData.log-20210424.gz
opentsdb_startup.log.4          ot_purgeData.log-20210425.gz
ot_purgeData.log                ot_purgeData.log-20210426.gz
ot_purgeData.log-20210404.gz    ot_purgeData.log-20210427.gz
ot_purgeData.log-20210405.gz    ot_purgeData.log-20210428.gz
ot_purgeData.log-20210406.gz    ot_purgeData.log-20210429.gz
ot_purgeData.log-20210407.gz    ot_purgeData.log-20210430.gz
ot_purgeData.log-20210408.gz    ot_purgeData.log-20210501.gz
ot_purgeData.log-20210409.gz    ot_purgeData.log-20210502.gz
ot_purgeData.log-20210410.gz    ot_purgeData.log-20210503
ot_purgeData.log-20210411.gz    queries.log
ot_purgeData.log-20210412.gz    verify_service.20210503_101756
ot_purgeData.log-20210413.gz
```

The following is an example of the log output for the Open TSDB service:

```
# more verify_service.20210503_101756
Starting verifier at Mon May  3 10:17:59 PDT 2021
checking to see if pid 664447 is alive
pid 664447 is alive
checking to see if opentsdb pid 664447 is responsive
opentsdb responded - rc=0, output = [{"metric":"cpu.percent","tags":
{"clustername":"markmapr62.mip.storage.hpecorp.net","clusterid":"69235743018
54689047"},"a
ggregateTags":["fqdn","cpu_class","cpu_core"],"dps":
{"1620062220":7999.999988058591,"1620062220":7999.999968097525,"1620062221":
7999.999950431058,"1620062223
":7999.999897014358,"1620062226":7999.999865572759,"1620062227":7999.9998389
31619,"1620062230":7999.999799699798,"1620062230":7999.999990135091,"1620062
231":
7999.999993930984,"1620062233":8000.000005408324,"1620062236":8000.000016566
8525,"1620062237":8000.00002602172,"1620062240":8000.000039944967,"162006224
0":80
00.000010000002,"1620062241":8000.000013042937,"1620062243":8000.00002224366
8,"1620062246":8000.000031188815,"1620062247":8000.000038768231,"1620062250"
:8000
.000049929692,"1620062250":7999.999869915985,"1620062251":7999.999874489055,
"1620062253":7999.9998883162625,"1620062256":7999.999901759388,"1620062257":
7999.
999913150034,"1620062260":7999.999929923924,"1620062260":8000.000010010003,"
1620062261":8000.000009999326,"1620062263":8000.000009967044,"1620062266":68
06.88
2651460229,"1620062267":5932.870612714869,"1620062270":4333.55159683651,"162
0062270":3833.0523301108274,"1620062271":2733.737487114993,"1620062273":1134
.2988
4221571,"1620062276":719.1767068273092}}]
```

Possible return codes for the log output are:

- 0 – running and responding to a simple interaction test
- 1 – not running

- 2 – running but not responding to a simple interaction test
- 3 – not started*

*From Warden's point of view, the service is enabled, but Warden has not started it yet.

Related concepts

Updating the MapR Installer on page 5409
Update the MapR Installer to include the latest MapR ecosystem packages and installer fixes. Once you update the MapR Installer, you can install ecosystem components and software versions that were made available after you first configured the MapR Installer.

MapR Installer Prerequisites and Guidelines on page 5396
The node on which you run the MapR installer and the nodes you plan to include in your MapR cluster must meet certain user, connectivity, and security requirements.

Starting and Stopping the Installer

Describes how and when you need to shut down and restart the Installer.

It is seldom necessary to shut down and restart the Installer, but you might need to do so in the following scenarios:

- You are troubleshooting an Installer issue, and stopping and restarting the Installer might resolve the issue.
- You need to upgrade the OS for a node.
- You want to run the Installer node in FIPS-compliant mode, and you cannot leave the Installer running because the Installer is not FIPS compliant.

Checking the Installer Status

When you are not sure if the Installer is running, use the following command to check the status:

```
systemctl status mapr-installer
```

Starting and Restarting the Installer

To start or restart the Installer (requires `root` authentication):

```
systemctl start mapr-installer
```

Stopping the Installer

Note that if you stop the Installer while an installation is in progress, any passwords that you provided to the Installer user interface are lost. The Installer retains password information in memory and not in the Installer database. If you then restart the Installer to continue the installation, the installation fails. In this scenario, you must restart the Installer and the Installer user interface and re-enter the password information to resume the installation.

To stop the Installer (requires `root` authentication):

```
systemctl stop mapr-installer
```

Related concepts

Resetting the Installer Database on page 5459
The `reset` command uninstalls the metadata from the Installer database. `reset` is for advanced users.

Related tasks

Starting Up a Cluster Using the MapR Installer Startup Button on page 5445
You can use a single button to start MapR software on a cluster.

Shutting Down a Cluster Using the MapR Installer Shutdown Button on page 5446
You can use a single button to shut down MapR software on a cluster.

Resetting the Installer Database

The `reset` command uninstalls the metadata from the Installer database. `reset` is for advanced users.

You can reset the Installer database by using the CLI `reset` command or by using the MapR Installer web interface (**Support > Reset Installer**).

The reset function can be useful for testing purposes, but use reset with caution. If you reset the installer database while packages are installed on the nodes, you will need to remove the packages manually.

Note: If you experience a failure while installing or uninstalling, the installer prompts you to retry the operation or uninstall and then reinstall from scratch. You should always retry or uninstall before considering using `reset`.

This example resets the Installer database. The `-nv` option specifies that certfificates will not be checked and the output mode is verbose:

```
./bin/mapr-installer-cli reset -nv
```

Troubleshooting Repository URL Errors

This page describes how to troubleshoot an issue in which an incorrect repository URL is stored in the MapR Installer `properties.json` file.

How Repository URL Errors Can Occur

The `properties.json` file stores information such as the user ID of the cluster administrator, the user ID of the MapR Installer, the OS type, Internet access information, and the repository URL. Once the repository URL has been stored in `properties.json`, the MapR Installer assumes that the URL will not change.

If you run `mapr-setup.sh -r <url>` and you make a mistake when typing the URL, the incorrect URL is added into the MapR Installer `properties.json` file. If you later run `mapr-setup.sh -r <url>` again but with the correct URL, the `properties.json` is not updated. Even upgrading the installer packages does not update the repository URL in `properties.json`.

Some versions of the MapR Installer generate a warning if you try to correct the URL by running `mapr-setup.sh -r <url>` again, but older versions of the MapR Installer do not generate a warning. Whether or not a warning is generated, you can correct the issue, but how you do so depends on the version of the MapR Installer that is installed.

Fix Using MapR Installer 1.10 or Later

MapR Installer 1.10 and later versions generate a warning if you run `mapr-setup.sh -r <url>` and provide a new URL. The warning describes two ways to change the URL currently stored in the `properties.json` file:

- You can use the `reload` or `remove` command, and then specify a new URL:

last-updated: Apr 11, 2022

1. Use *one* of the following commands:

   ```
   bash /tmp/mapr-setup.sh -R <new_url> reload
   ```

   or

   ```
   bash /tmp/mapr-setup.sh remove
   ```

   Using the `remove` command removes `properties.json`, the installer database, and the installer packages, but not the setup script.

2. Specify the new URL:

   ```
   bash /tmp/mapr-setup.sh -r <new_url>
   ```

- You can manually edit the `properties.json` file:

  1. Edit the `properties.json` file to specify the new URL:

     ```
     edit /opt/mapr/installer/data/properties.json
     ```

  2. Reload the MapR Installer:

     ```
     systemctl restart mapr-installer
     ```

Fix Using MapR Installer 1.9 or Earlier

MapR Installer 1.9 and earlier do NOT generate a warning if you run `mapr-setup.sh -r <url>` and provide a new URL. To pass a new repository value into `properties.json` for MapR Installer versions 1.9 or earlier, you must first remove the installer files:

```
mapr-setup.sh remove
```

Using the `remove` command removes `properties.json`, the installer database, and the installer packages, but not the setup script. After the files are removed, you can rerun `mapr-setup.sh` to specify the new repository URL:

```
bash /tmp/mapr-setup.sh -r <new_url>
```

To run `mapr-setup.sh`, see MapR Installer. For information about options you can use with `mapr-setup.sh`, see Using mapr-setup.sh.

MapR Installer Known Issues

This topic describes some MapR Installer known issues that you should be aware of while troubleshooting.

If you are viewing this page from within the Installer application, click here to display the information in a browser.

IN-3016	On Installer 1.16 and earlier, clicking **Abort** during the **Extend Cluster** operation returns you to the verification page and does not reset the Installer database back to its initial state. (Note that on Installer 1.17 and later, clicking **Abort** resets the Installer database automatically so that you can retry the operation.)

last-updated: Apr 11, 2022

Workaround: On Installer 1.16 and earlier, use these steps to reset the Installer database manually and retry the **Extend Cluster** operation:

1. Click **Support > Reset Installer**. This command uninstalls the metadata from the Installer database. For more information, see Resetting the Installer Database on page 5459.
2. Click **Support > Import State**. The **Cluster State** dialog box appears, enabling you to reset the cluster to the last known state. For more information, see Importing or Exporting the Cluster State on page 5447.
3. Click **Reset** to recover the Installer to the last known state and return to the Installer home page.
4. If necessary, retry the **Extend Cluster** operation.

IN-3007

During a multinode installation of core 6.2 on SLES 15 SP2, the Installer returns the following error:

```
 "msg": "user {{ ssh_id }} does
not have the ability to elevate
privileges - check for correct
sudoers config for example"
```

This issue can occur when Python is not installed on all cluster nodes.

Workaround: Check to ensure that Python is installed on all cluster nodes. Install Python, if it is not already installed.

IN-2924

Upon restart, cluster nodes running Loopback NFS do not remount `/mapr`. This issue can occur when using Installer 1.16.0.0 to perform a new or incremental installation. The issue is caused by a missing symlink.

Workaround: Manually create a symlink from `/usr/local/mapr-loopbacknfs/conf/mapr_fstab` to `/opt/mapr/conf/mapr_fstab`, and use the following commands to restart NFS and mount `/mapr`:

1. Restart the Loopback NFS service:

   ```
   maprcli node services -nodes <node
   names> -nfs restart
   ```

2. Run the `mount_local_fs.pl` script to mount `/mapr`:

   ```
   /opt/mapr/bin/mount_local_fs.pl
   ```

IN-2397

The Verify phase of the Installer can fail if the `authorized_keys` file contains a command such as the following:

```
no-port-forwarding,no-agent-forwarding
,no-X11-forwarding,command="echo
```


last-updated: Apr 11, 2022

```
'Please login as the user \"admin\"
rather than the user
\"root\".';echo;sleep 10" ssh-rsa ...
```

Any command in the `authorized_keys` file prevents the Installer from authenticating with remote nodes.

Workaround: Verify that the `authorized_keys` file does not contain commands that prevent the Installer from authenticating with remote nodes. In addition, if you are using keys for remote authentication, you must ensure that you can ssh into all nodes in the cluster using the user and password that you specified when you configured remote authentication.

IN-2500

After a new installation, the Installer home page displays two YARN ResourceManager links, but one of the links does not work.

Workaround: This is normal. Click the YARN ResourceManager links until you find the link that works. Even if the ResourceManager is installed on multiple nodes, the YARN ResourceManager only has one server running at a time, . If the running ResourceManager fails, a new ResourceManager is started on one of the other nodes.

IN-2705

If you used the Installer to install Sentry on an insecure cluster, you must manually add Sentry to the list of services that Warden monitors. To add Sentry to the list of services that Warden monitors, copy `/opt/mapr/sentry/sentry-<version>/conf.d/warden.sentry.conf` to `/opt/mapr/conf/conf.d`.

Neither Hive nor Impala is configured to use Sentry by default; however, you can configure Hive and Impala to use Sentry, as described in Configure Hive to use Sentry Authorization on page 3830 and Configure Impala to Use Sentry Authorization on page 3825

SPYG-1136

During a manual installation or upgrade, Collectd provided in core 6.1.0 won't start on RHEL / CentOS 8.2 because it expects the Python 2 libraries to be installed, and RHEL / CentOS 8.2 provides the Python 3 libraries instead. This issue does not affect installations or upgrades performed using the Installer.

Workaround: Before installing the monitoring components, check to see if Python 2 is installed. If the following error is generated, try installing Python 2 on RHEL / CentOS 8.2:

```
failed: libpython2.7.so.1.0: cannot
open shared object file
```

IN-2637

After a manual installation, Oozie and Hive services can fail to connect to a MySQL or MariaDB database because the server time-zone value is unrecognized or represents more than one time zone. The issue affects your installation if you applied the `mapr-patch` released on or after February 21, 2021 (including the latest `mapr-patch`). This issue affects manual installations but is fixed in Installer 1.14.0.0.

Workaround: For manual installations, you must configure either the server or JDBC driver (using the `serverTimezone` configuration property) to use a more specific time-zone value if you want to utilize time-zone support. After running `configure.sh` but before starting the Oozie or Hive services, update the `serverTimezone` parameter in the `hive-site.xml` or `oozie-site.xml`. For more information, see MySQL Bug #95036.

IN-2935

On RHEL or CentOS 8.3, new installations using the Installer can fail with the following error message:

```
mount.nfs: access denied by server
while mounting localhost:/mapr
```

This happens when the Installer cannot start the `mapr-loopbacknfs` service because the RHEL or CentOS NFS/NFS4 service is running.

Workaround: Edit the `/etc/systemd/system/mapr-loopbacknfs.service` file to add the following Conflicts directive to the `nfs-mountd.service`:

```
[Unit]
Description=MapR Technologies, Inc.
loopbacknfs service
After=rc-local.service
After=network.target syslog.target
Conflicts=nfs-mountd.service
```

The `Conflicts` command stops `nfs-mountd` before installation so it cannot interfere with starting `mapr-loopbacknfs`. After editing the `loopbacknfs.service` file, perform a daemon reload using the following command, and then retry the installation:

```
systemctl daemon-reload
```

IN-2947

For Installer 1.16.0 on Ubuntu 18.04, the **Extend Cluster** operation fails for clusters larger than three nodes. The operation fails on nodes where ZooKeeper is not installed. The failure occurs because the Installer attempts to update the ZooKeeper service file on a node that has no roles file for ZooKeeper.

Workaround: Make sure ZooKeeper is running on every node in the cluster, and retry the **Extend Cluster** operation.

ES-77, FLUD-55

During an upgrade from EEP 6.x to EEP 7.0.0 or EEP 7.0.1, some monitoring components do not get updated because of an error in the fourth digit of the package version. This issue can occur during manual upgrades or upgrades performed using the Installer. The affected components can include any or all of the following:

- Elasticsearch

last-updated: Apr 11, 2022

- Fluentd
- Grafana
- Kibana

Workaround: See Reinstalling Monitoring Components After an Upgrade in the data-fabric documentation.

IN-1976

During a version upgrade from core 6.0.1 and EEP 5.0.x to a later version of core, the upgrade succeeds, but the following error message is generated:

```
This version of Kibana requires
Elasticsearch v6.2.3 on all
nodes. I found the following
incompatible nodes in your
cluster: v5.4.1 @ 10.10.103.231:9200
(10.10.103.231), v5.4.1 @
10.10.102.21:9200 (10.10.102.21), v5.4.1
@ 10.10.103.230:9200 (10.10.103.230)
```

This happens because the Elasticsearch package script does not remove Elasticsearch completely and does not shut it down. Even though a new version of Elasticsearch is installed, the old version is still running and using the port needed by the new version.

Workaround:

1. Use the `jps` command to find the process for the old Elasticsearch.
2. Use `kill -9` to shut down the old Elasticsearch process. Warden will restart the newly installed Elasticsearch.

IN-2742

A `configure.sh` operation can hang because of a system control hang if you try to install on top of a "minimal" operating system installation and the RDMA RPM or service is not present. This issue can occur during manual installations or during installations using the Installer.

Workaround: Before running `configure.sh`, use one of the following workarounds:

Workaround #1 - Install the missing RDMA Dependencies

- **RHEL / CentOS**
  1. Install `libibverbs`:

     ```
     yum
     install
     libibverbs
     ```

2. Enable and start the RDMA service:

```
systemctl enable rdma && systemctl start rdma
```

3. Retry the MapR Data Platform installation.

- **Ubuntu 18**
  1. Install the `rdma-core` package:

```
apt-get install rdma-core
```

  2. Install `libibverbs1`:

```
apt-get install libibverbs1
```

  3. Enable and start the RDMA service:

```
systemctl enable iwpmd && systemctl start iwpmd
```

  4. Retry the installation.

- **Ubuntu 16**

  Release 6.2 does not provide RDMA support for Ubuntu 16 because Ubuntu 16 does not have the `rdma-core` package.

last-updated: Apr 11, 2022

Workaround #2 - Disable RDMA Support	**1.** Rename `/opt/mapr/lib/libibverbs.so`. For example: `mv /opt/mapr/lib/libibverbs.so /opt/mapr/libibverbs.so.sv` **2.** Restart the ZooKeeper and Warden nodes.
Workaround #3 - Install the Latest Core Patch	The latest patch contains the `export MAPR_RDMA_SUPPORT=false` environment variable, which removes RDMA support. For patch information, see "Downloading a Patch" in the data-fabric documentation.
IN-2784 & MFS-11853	Stopping a cluster by stopping ZooKeeper and Warden can cause clients that are accessing the file system through POSIX (for example, the S3 gateway) to hang if Loopback NFS is installed on a cluster node and is not stopped first. Note that beginning with Installer 1.15, the Installer installs Loopback NFS on all cluster nodes unless NFS is enabled. **Workaround:** If Loopback NFS is running and you need to stop the cluster, you must first unmount `/mapr` and stop Loopback NFS on all nodes. Then, you can stop ZooKeeper and Warden. For more information, see "Managing the mapr-loopbacknfs Service" in the data-fabric documentation.
IN-1343	In a new installation of six nodes or more using the MapR Installer, if only data nodes fail to install, retrying the installation can fail. **Workaround:** Use the MapR Installer uninstall feature, and retry the installation from scratch. See "Uninstalling Software Using the Installer Uninstall Button" in the data-fabric documentation.
IN-2132	On a SLES cluster, using MapR Installer version 1.10 or earlier of the `mapr-setup.sh` script can complete successfully even if sshpass is not installed. **Workaround:** Upgrade to the latest MapR Installer. You must use version 1.11 or later of the `mapr-setup.sh` script. If you cannot use MapR Installer version 1.11 or later of the `mapr-setup.sh` script, install sshpass before running the `mapr-setup.sh` script on a SLES cluster.

IN-2008

When you upgrade from a secure MapR 6.0.0 cluster to MapR 6.0.1 using MapR Installer 1.9, a security certificate for log monitoring is overwritten. As a result, Elasticsearch can fail to start after the upgrade. This issue is not present during a new installation of MapR 6.0.0 or MapR 6.0.1 or during an upgrade to MapR 6.1.0. This issue is fixed in MapR Installer 1.10 and later.

Workaround: To resolve the issue, you must remove the `.keystore_password` file, re-run the command to generate new Elasticsearch certificates, and then re-distribute the certificates. Use these steps:

1. Remove or rename the `.keystore_password` file. For example:

   ```
   rm /opt/mapr/elasticsearch/
   elasticsearch-x.x.x/etc/
   elasticsearch/.keystore_password
   ```

2. Perform steps 3 through 7 of "Step 9: Install Log Monitoring" in the data-fabric documentation. Completing steps 3 through 7 regenerates the Elasticsearch certificates and copies them to the other nodes.

IN-2443

An internal server error that includes a NullPointerException can be generated if you install a cluster on Ubuntu 16 using a MapR Installer Stanza. The error appears if Hive is installed but no password for Hive is included in the `.yaml` installation file.

Workaround: Add the Hive password to the `.yaml` installation file and re-run the Stanza.

IN-18

When using the `-v` or `--verbose` options with MapR Installer Stanzas, detailed error information is not provided on the command line. For example, if a `mapr` user or group is not present on a host during a new installation, the `mapr-installer-cli` reports "Verification Error" on the command line.

Workaround: To view more detailed error information when using the `-v` or `--verbose` options, check the `installer-cli.log[.x]` file after running the Stanza. For information about the MapR Installer logs, see "Logs for the Installer" in the data-fabric documentation.

IN-2200

Deploying a release 6.0.1 cluster on AWS fails when the following parameters are specified:

- `diskType: io1`
- `installerOnitsOwn: false`

Workaround: Try using a `diskType` of `gp2` (general-purpose SSD) instead of `io1` (provisioned IOPs SSD), or set `InstallerOnitsOwn` to `false` instead of `true`. Then retry the deployment.

IN-2152

During a MapR Installer upgrade from any release to MapR 6.0.1, core files can be generated for ecosystem components, which can cause alarms in

last-updated: Apr 11, 2022

	the MapR Control System following the upgrade. This happens because the upgrade sequence shuts down the cluster, then upgrades MapR Core packages, and then restarts MapR Core. Restarting MapR Core is necessary to upgrade some ecosystem components. When the old ecosystem components are started, version incompatibilities with the new version of MapR Core can cause core dumps. This is a temporary issue. Upgrading the ecosystem component, which happens later in the upgrade process, resolves the issue. The issue does not exist in MapR 6.1 and later releases, which have the ability to prevent services from restarting during an upgrade. **Workaround:** Ignore the Control System alarms, or upgrade to MapR 6.1 or later, which should not generate core alarms.
IN-1940	In MapR Installer versions 1.9 and earlier, the `probe` command can fail because of a runtime error if you have installed the Operational Applications with MapR Database template. The error is caused by the presence of the `mapr-drill-internal` package. Any node running the MapR Data Access Gateway requires the `mapr-drill-internal` package to be installed even though Drill is not installed as a service. The `mapr-drill-internal` package provides a set of client libraries used by the MapR Data Access Gateway. **Workaround:** Before using the `probe` command, update the Installer. The `probe` command is fixed in versions 1.10 and later.
IN-1635	In MapR Installer Stanza versions 1.9 and earlier, the `probe` command was hard coded with a cluster admin user of `mapr`. If you configured a cluster admin user other than `mapr`, the `probe`-generated YAML file could not be imported using the `import` command. **Workaround:** Before using the `probe` command, update the Installer to version 1.10 or later. Or, if you must use version 1.9 or earlier, edit the `probe`-generated YAML file to specify the correct cluster admin user.
IN-2123	In a secure cluster, the **Extend Cluster** operation fails if you try to extend the control group. The new control node cannot join the cluster because it inadvertently receives a new set of keys. This issue affects versions 1.7 through 1.10 of the MapR Installer and is fixed in MapR Installer 1.10.0.201812181130 and later versions. **Workaround:** You can resolve the issue by manually copying mapruserticket into the `/opt/mapr/conf` directory of the node to be added to the cluster.
IN-2141	The following issue applies to MapR Installer versions 1.7 through 1.10, but not all 1.10 versions. The issue is fixed in MapR Installer 1.10.0.201812181130 and later versions. An extend cluster (add node) operation can fail when you:

1. Install a MapR 6.x cluster manually with security enabled.
2. Run the MapR Installer Stanza `probe` command on the cluster or on a node to be added to the cluster.
3. Use the `import` command to import the `probe` .yaml file into the Installer.
4. Perform an extend cluster operation immediately after the `import` operation.

The extend cluster operation fails because keystore, truststore, and MapR server ticket (`maprserverticket`) files are not present on the installer node.

Workaround:

Before attempting the extend cluster operation, copy the keystore, truststore, and MapR server ticket (`maprserverticket`) files from any CLDB node to `/opt/mapr/installer/data/tmp` on the installer node. The files that need to be copied are:

- `cldb.key`
- `dare.master.key*`
- `maprserverticket`
- `ssl_keystore`
- `ssl_keystore.p12`
- `ssl_keystore.pem`
- `ssl_truststore`
- `ssl_truststore.p12`
- `ssl_truststore.pem`

*The DARE primary key is required only if DARE is enabled.

If metrics monitoring is configured on the cluster, you must also copy the tickets related to Fluentd, Kibana, and Elasticsearch to the same location.

IN-2217

During an upgrade to EEP 6.1.0 using the MapR Installer, the MapR Installer does not back up the Drill `conf`, `log`, and `jar` directories into `${MAPR_HOME}/drill/OLD_DRILL_VERSIONS`. This can happen when you upgrade Drill from an old version (for example, Drill 1.10 in EEP 3.0) to Drill 1.15.0.0 in EEP 6.1.0.

Recent packaging changes in Drill contribute to this issue. Drill 1.10 consists only of `mapr-drill-1.10` (role and binaries), whereas Drill 1.15.0.0 consists of `mapr-drill-1.15` (roles) and `mapr-drill-internal-1.15` (binaries). During the upgrade, the `mapr-drill-1.10` binaries are successfully uninstalled, but the

last-updated: Apr 11, 2022

OLD_DRILL_VERSIONS directory that is needed to back up Drill 1.10 is not created.

Workaround:

Before upgrading, perform the following steps:

1. Shut down the `mapr-drill-1.10` Drillbits.

```
maprcli node services -name
drill-bits -action stop -nodes
<node hostnames separated by a
space>
```

2. Create `${MAPR_HOME}/drill/OLD_DRILL_VERSIONS/drill-1.10`.
3. Copy the following directories of `mapr-drill-1.10` into the `OLD_DRILL_VERSIONS` directory:
   a. Copy the `conf` directory to `${MAPR_HOME}/drill/OLD_DRILL_VERSIONS/drill-1.10.0/conf`.
   b. Copy the `logs` directory to `${MAPR_HOME}/drill/OLD_DRILL_VERSIONS/drill-1.10.0/logs`.
   c. Copy the `jars/3rdparty` directory to `${MAPR_HOME}/drill/OLD_DRILL_VERSIONS/drill-1.10.0/jars`.
4. Proceed with the upgrade.
5. After successfully upgrading and starting `mapr-drill-1.15.0.0`, you may remove the `${MAPR_HOME}/drill/drill-1.10.0` directory.

IN-1915

During an upgrade using the MapR Installer, refreshing the browser page can cause the Installer to forget upgrade parameters that were specified before the refresh.

Workaround: Avoid refreshing the browser page during an upgrade operation. If you must refresh the page, go back to the first page of the upgrade operation and start over again to ensure that the Installer has the correct parameters before it begins the Verify phase of the upgrade.

IN-2035

During a version upgrade using the MapR Installer, if you select the **Advanced Configuration** button and then click **Previous** (one or more times) followed by **Abort**, the Installer can indicate that the upgrade completed even though the upgrade was aborted.

Workaround:

If this happens, you must reset the installer and reload the last known state. Follow these steps to reset the cluster state:

last-updated: Apr 11, 2022

1. Click **Support > Reset Installer**. A warning screen appears.
2. Click **OK**.
3. Click **Support > Import State**.
4. Click **Reset** to recover the cluster to the last known state. It is safe to retry the upgrade at this point.

For more information about the **Reset Installer** and **Import State** commands, see "Resetting the Installer Database" and "Importing or Exporting the Cluster State" in the data-fabric documentation.

IN-2065

`/mapr` sometimes does not get mounted after you enable NFS (v3 or v4) using the MapR Installer Incremental Install function. The Incremental Install function is an online operation. Enabling NFS using an Incremental Install can create a race condition between when the `mapr_fstab` file gets created and NFS is started by Warden. If NFS is started by Warden before the `mapr_fstab` file is created, `/mapr` does not get mounted.

Workaround:

If `/mapr` is not mounted, check the time stamp of the `/opt/mapr/conf/mapr_fstab` file to see if it is older than the time stamp in the `warden.log` file for starting NFS. For example:

```
[root@atsqa4-61 logs]# ls -ld /opt/
mapr/conf/fstab
rw-rr- 1 mapr mapr 39 Sep 26
11:31 /opt/mapr/conf/mapr_fstab

[root@atsqa4-61 logs]# fgrep starting
warden.log | fgrep nfs
2018-09-26 11:29:33,407 INFO
com.mapr.warden.service.baseservice.Se
rvice
[Thread-34]: -------------Service is
starting for: nfs4
```

If the time stamp of the `mapr_fstab` file is older than the Warden time stamp:

1. Restart the NFS service:

   ```
   maprcli node services -nodes <node
   names> -nfs4 start
   ```

2. Run the `mount_local_fs.pl` script to mount `/mapr`:

   ```
   /opt/mapr/bin/mount_local_fs.pl
   ```

INFO-420

The procedure for configuring storage using `disksetup` does not work for new installations of DARE-enabled MapR 6.1 clusters. With DARE

enabled, `disksetup` fails on any node that is not a CLDB node because there is no local copy of the `dare.master.key` file. When you use disksetup, non-CLDB nodes try to contact the CLDB, which must be running when the nodes attempt contact.

Workaround:

After running `configure.sh`, you must:

1. Format the disks on the CLDB nodes.
2. Start ZooKeeper on the ZooKeeper nodes.
3. Start Warden on the CLDB nodes.
4. Format the remaining node disks using `disksetup`.
5. Start Warden on the remaining nodes.

IN-2057

A fresh install of MapR 6.0.0 using the `sample_advanced.yaml` file for MapR Installer Stanzas (Installer version 1.9) can fail with the following error message:

```
ERROR: install command failed
Service mapr-data-access-gateway must
be a member of a template group.
Configured services require it:
['mapr-data-access-gateway']
```

The error is generated because the `.yaml` file is missing an entry for the `mapr-data-access-gateway` in the MASTER services section. The `mapr-data-access-gateway` service is needed for MapR Database installations.

Workaround:

In the MASTER services section of the `sample_advanced.yaml` file, add `mapr-data-access-gateway` to at least one of the host groups, and retry the installation.

IN-1272

During an upgrade to MapR 6.0 or later (Drill 1.11), `configure.sh` sometimes fails to disable the storage plugin for HBase. The HBase server is not supported in MapR Core 6.0 or later, so the HBase storage plugin should be disabled before a cluster is upgraded to MapR 6.0 or later. Otherwise, Drill queries against HBase will hang.

Workaround:

Before upgrading to Drill 1.11 or later, manually disable the HBase storage plug-in. To manually disable the plug-in, you can use the Drill Web Console or Drill REST API commands. You can disable the HBase storage plugin on the **Storage** page of the Drill Web

last-updated: Apr 11, 2022

Console at `http(s)://<drill-hostname>:8047`. For more information, see this page:

```
https://drill.apache.org/docs/
rest-api-introduction/#delete-/
storage/{name}.json
```

IN-1747

If you use the MapR Installer 1.10 **Uninstall** button to uninstall MapR software and a node is unreachable, you will not be able to uninstall the node later when the node is reachable.

Workaround:

Uninstall the MapR software on the node manually. See "Decommissioning a Node and Uninstalling Data Fabric Software from the Command-line" in the data-fabric documentation.

IN-2015

A fresh install of MapR 6.0.0 with EEP 4.1.1 using MapR Installer 1.9 can fail with the following error message:

```
file not found: /opt/mapr/
elasticsearch/elasticsearch-5.4.1/etc/
elasticsearch/sg/
admin-usr-clientCombo.pem
```

Workaround: Update the Installer to version 1.10 or later, and retry the operation.

IN-2018

Logging on to Kibana results in an authentication failure. This can happen on a CentOS cluster if you use MapR Installer 1.10 to install MapR 6.0.1 EEP 5.0.0, and then upgrade to MapR 6.1.0 and EEP 6.0.0.

Workaround: Try using MapR Installer 1.9 to install the MapR 6.0.1 cluster and MapR Installer 1.10 to upgrade the cluster. See "Updating the Installer" in the data-fabric documentation.

CORE-150

After using the **Incremental Install** function of the MapR Installer to apply security to an Azure-based MapR 6.1.0 cluster, the Hue and Spark-thrift server links are not accessible in the Installer interface. This issue can occur on an Azure-provisioned cluster whose internal DNS suffix starts with a number rather than a letter.

Workaround: Re-create the cluster in Azure so that the internal DNS suffix starts with a letter and not a number.

IN-2025

The **Extend Cluster** operation can fail during the **Verify Nodes** phase with an error indicating `Unscalable host groups found`. This error can occur when the MASTER group is missing or a single-instance service (for example, Grafana) has been moved out of the MASTER group. The `mapr-installer.log` reveals which cluster services are supposed to be in the MASTER group.

Workaround: Move any original MASTER services that caused the error back to the MASTER group. The `mapr-installer.log` indicates the services

last-updated: Apr 11, 2022

	that need to be moved along with the `Unscalable host groups found` error.
IN-2006	On a cluster with `mapr-drill` installed, the `probe` command can return the wrong database type value. **Workaround:** After using the `probe` command, check to see if the resulting YAML file has the correct `mapr_db` setting. Possible settings are: • `QS` • `DRILL` • `DRILLQS` If necessary, change the setting in the YAML file to match the value from the probed cluster.
IN-1955	If you install cluster software using the MapR Installer in a browser and then upgrade the MapR installer in the same browser tab and attempt an upgrade without starting a new browser, the stale browser cache can cause upgrade errors. **Workaround:** Clear your browser cache or open a new browser tab whenever you need to update the MapR Installer and perform a new installer operation.
IN-1983	After an upgrade from release 5.x to release 6.1 and EEP 6.0.0 using the MapR Installer, the `kafka-connect` service fails to start. This issue has been noticed on platforms that use `systemd`. **Workaround:** Stop the `kafka-connect` service manually, and restart the service.
IN-1972	During an upgrade from release 5.x to release 6.1, the MapR Installer prompts you for the MySQL user ID and password. If you enter a password that is different from the password you provided when you originally configured MySQL through the MapR Installer, the upgrade fails with this error: "Unable to connect to database.…" **Workaround:** When the MapR Installer prompts you for the MySQL user ID and password, enter the password that you specified when you first installed the cluster. If you did not specify a password for MySQL when you installed release 5.x, leave the password field blank.
IN-1904	If you initiate a system startup by clicking the **Startup** button on the Installer web interface, the **Authentication** screen is displayed. If you subsequently click the **Previous** button, the following buttons are shown as active even though they are not usable during system startup: • Extend Cluster • Incremental Install • Maintenance Update

last-updated: Apr 11, 2022

- Shutdown
- Uninstall

Workaround: Do not use the **Previous** button during startup.

IN-1657

After updating the Installer 1.7 or later, the Installer can lose awareness that a cluster was previously installed. For example, the MapR Installer might indicate the need for a fresh install.

Workaround: If this happens, do NOT proceed with installation or upgrade operations. Follow these steps to reset the cluster state:

1. Click **Support > Reset Installer**. A warning screen appears.
2. Click **OK**.
3. Click **Support > Import State**.
4. Click **Reset** to recover the cluster to the last known state. It is safe to use the Installer at this point.

IN-1804

For release 6.0 or later clusters, enabling security by using the Incremental Install function can overwrite custom certificates in the `ssl_truststore` and `ssl_keystore` files. When you turn on security, the MapR Installer runs the `configure.sh` script on the CLDB primary node to generate security keys and then distributes the keys to all the other CLDB nodes. The installer also distributes certificates to all the other nodes. This process can cause custom certificates to be overwritten. However, before enabling security, the MapR Installer makes a backup of the existing `ssl_keystore` and `ssl_truststore` files.

Workaround:

After enabling security, locate the backup of the `ssl_keystore` and `ssl_truststore` files. The backup uses this format:

```
/opt/mapr/conf/
ssl_keystore.sv.<timestamp>
```

Extract any custom certificates from the backup files, and manually merge or add them into the new `ssl_keystore` and `ssl_truststore` files.

To merge the files, you can use the `/opt/mapr/server/manageSSLKeys.sh` utility, as shown in "Configuring Secure Clusters for Running Commands Remotely" in the data-fabric documentation.

IN-997

When using MapR Installer 1.9 with Ubuntu distributions, an upgrade of the MapR Installer definitions requires a restart of the installer service. The restart is needed because the MapR Installer services version is not updated properly when you use either of the following commands:

last-updated: Apr 11, 2022

- `mapr-setup.sh reload`
- `apt-get install mapr-installer-definitions`

Workaround: After installing or reloading the MapR Installer definitions, issue one of these commands to fix the services version:

- `service mapr-installer restart`
- `systemctl restart mapr-installer`

IN-1671

For MapR Installer 1.8 and earlier, installation on Ubuntu 16.04 can fail if Python 2 is not available or if the default is set to Python 3. The installer requires `python` and `python-yaml` to be installed on all nodes.

Workaround: To install the Python packages manually:

```
sudo apt-get install python
python-yaml -y
```

IN-1336

The MapR Installer **Retry** function can be affected if the installer operation fails. Suppose you deselect a service during an **Incremental Install** operation. If the **Incremental Install** fails and you need to **Retry**, it's possible that the service will not be deselected (uninstalled).

Workaround: Manually remove (uninstall) the service by using one of these commands:

- **Red Hat / CentOS:** `yum remove`
- **Ubuntu:** `apt-get remove`
- **SLES:** `zypper remove`

IN-1392

During an **Extend Cluster** (add node) operation using the MapR Installer, if installation of the added node fails and you abort the operation, the installer can display the added node even though it did not get installed.

Workaround: When the MapR Installer indicates that a node did not get added correctly (typically during the Installation phase), select the node and click **Remove Node**. Then retry adding the node.

IN-1396

An installation using the MapR Installer fails with the following Ansible module error:

```
nValueError: need more than 1 value
to unpack
```

Workaround: Check for syntax errors in the `/etc/sysctl.conf` file, which can cause an Ansible error when the MapR Installer is attempting to set various kernel parameters.

last-updated: Apr 11, 2022

IN-1398

In the MapR Installer **Verify Nodes** page, if you click a host, the **Disks Selected for MapR** box in the right pane displays the disks that were specified for the host either manually or automatically. If you deselect a disk in the right pane and click **Retry**, the deselection is not always implemented.

Workaround: Click **Previous** to go back to the **Node Configuration** page, and re-specify the disks that you want. Then continue with the operation.

IN-1386

On a secure MapR cluster, YARN jobs can fail if you specify IP addresses rather than host names when you configure nodes using the MapR Installer.

Workaround:

Do not use an IP address for node configuration with the MapR Installer. If you already used an IP address, change the IP address in the yarn-site.xml file on all nodes. In the following example, the 10.10.10.7 IP address must be changed to a host name, such as `bld73.qa.lab`:

```
<property>
<name>yarn.timeline-service.hostname</
name>
<value>10.10.10.73</value>
</property>
```

IN-1333

On Ubuntu clusters, the `mapr-setup.sh` script fails to reload the MapR Installer definitions during an update of the MapR installer and definitions.

Workaround: After updating, restart the installer to load the definitions:

```
service mapr-installer restart
```

IN-907

The MapR Installer service fails if the `mapr` user or `root` user already exist and they are configured to use a shell other than bash. For more information about user requirements, see "Installer Prerequisites and Guidelines" in the data-fabric documentation.

Workaround: Configure the users to use bash. For more information about user requirements, see "Installer Prerequisites and Guidelines" in the data-fabric documentation.

IN-1079

Verification fails when the installed language pack is for a language other than English.

Workaround: Remove the non-English language pack and install the English language pack. In the following example, the non-English language pack is shown as German. Also, make sure your system locale is set to `en_us`, as described in "Infrastructure" in the data-fabric documentation.

```
sudo apt-get install language-pack-en
language-pack-en-base manpages
sudo apt-get remove language-pack-de
language-pack-de-base manpages-de
```


last-updated: Apr 11, 2022

IN-804	Using the Incremental Install operation to add a third node to a CONTROL group generates an error: `ERROR: configure_refresh.sh failed`. This issue applies to MapR Installer versions 1.6 and earlier. **Workaround:** Update the MapR Installer to version 1.7 or later, and retry the operation. See "Updating the Installer" in the data-fabric documentation.
IN-1314	When you use the MapR Installer to install ecosystem components that require a MySQL component such as Hive, Oozie, or Hue, the passwords you provide to install the MySQL database are displayed in the `mapr-installer.log` and `<nodename>.log` files. Beginning with MapR Installer 1.7, the permissions for the `mapr-installer.log` and `<nodename>.log` files are changed so that these passwords are not world readable. However, the passwords are still present in log files created with earlier versions of the MapR Installer. **Workaround:** For increased security, remove the earlier logs or change the user permissions for them.
IN-1042	Installation of the 5.2.x `mapr-metrics` package on SLES 12 SP2 fails because the `libmysqulclient16` package is not present. This can happen when `mapr-metrics` is installed manually or using the MapR Installer. This issue was detected during installations of release 5.2.x with EEP 3.0.0. **Workaround:** None.
IN-870	If your cluster uses 2-digit EEPs, and you use the MapR Installer **Extend Cluster** button to add a node, the node can be added with a patch version that is different from the patch version of other nodes in the cluster. See "Understanding Two-Digit and Three-Digit MEPs" in the data-fabric documentation. **Workaround:** A one-time change can prevent this issue. After updating the MapR Installer from version 1.5 to version 1.6 or later, but before performing any MapR Installer operations, use an Incremental Install function to change your 2-digit EEP version to the equivalent 3-digit EEP version. See "Updating the Installer" in the data-fabric documentation.
ES-27, IN-1387	On a new installation of a secure cluster using the MapR Installer, Elasticsearch fails to start, and logs indicate that Elasticsearch key generation failed. When this happens, Kibana and Fluentd also do not start. The MapR Installer allows the installation to complete. **Workaround:** Check the installer log for a message indicating that Elasticsearch could not be secured. Use the **Incremental Install** feature of the MapR Installer to retry installation of the MapR Monitoring logging components. Alternatively, you can configure security for the logging components manually. See "Step 9: Install Log Monitoring" in the data-fabric documentation.
IN-1332	On clusters with less than the recommended memory configuration (16 GB per node), services added during an Incremental Install operation might fail to

start because Warden allocated available memory to the MapR filesystem. The MapR Installer might not indicate a problem with the newly added services. If this issue occurs, the MapR filesystem cannot relinquish memory without restarting Warden.

Note: This issue can also occur on clusters with more than 16 GB of memory per node if the installed services require more memory than is currently installed.

Workaround:

Use the Control System or the `maprcli service list -node` command to determine if the added services are running. If not, perform a rolling restart of the nodes to which the new services were added. The rolling restart will rebalance memory across the filesystem and services. One node at a time, restart Warden on each node following the group upgrade order prescribed in "Manual Rolling Upgrade Description" in the data-fabric documentation. Use the following steps:

1. Change to the `root` user (or use `sudo` for the following commands).
2. Stop Warden.

   ```
   sudo service mapr-warden stop
   ```

3. Restart Warden.

   ```
   service mapr-warden start
   ```

IN-1339

Installation fails with the Installer reporting an **Unexpected failure during module execution**, and the following entry is present in the "Logs for the Installer" described in the data-fabric documentation:

```
os.write(self.sshpass_pipe[1],
to_bytes(self._play_context.password)
+ b'\n')
OSError: [Errno 9] Bad file descriptor
```

Workaround: Change the ssh settings as described in known issue IN-405 later on this page, and retry the installation.

IN-553

New installations on Ubuntu 14.04 using MapR Installer 1.6 or 1.7 can fail because of a JDK 1.8 issue.

Workaround:

If you are installing on Ubuntu 14.04, you must install Java JDK 1.8 before running the MapR Installer. For more information, see this website. If you are installing on RHEL/CentOS or SLES, the MapR Installer installs Java JDK 1.8 for you.

IN-405

MapR installation or cluster import using the probe command fails with the error message:

last-updated: Apr 11, 2022

	"Failed to resolve remote temporary directory from ansible-tmp- ...." **Workaround:** To proceed using the MapR Installer, disable SSH connection reuse by including this entry underneath the `[ssh_connection]` property of `/opt/mapr/installer/etc/ansible.cfg`: `ssh_args=-o ControlMaster=no -o ControlPath=none -o ControlPersist=no` This workaround can lead to longer install times. We recommend that you resolve any network connectivity issues in your environment.
IN-250	An upgrade to a new MapR core version and a new EEP using the MapR Installer can fail if the cluster being upgraded was initially installed with Hive Metastore but not with Hive. The Hive Metastore package has an installation dependency on Hive, but the Hive Metastore definitions do not enforce the dependency, resulting in inconsistencies in the installer database. This issue has been observed on Ubuntu platforms. **Workaround:** Before upgrading, if you have Hive Metastore installed by itself, use the **Incremental Install** feature of the MapR Installer to install Hive. Then proceed with the upgrade. Performing an upgrade without doing the **Incremental Install** of Hive will cause the upgrade to fail. In this scenario, you will have to reinstall or rebuild the database by using Stanza commands. You can use the `reset` command, followed by `probe`, and then edit the versions in the resulting YAML file. The last step is to import the edited YAML using the `import` command. See "Using probe and import to Generate the Installer Database" in the data-fabric documentation.
N/A	The MapR Installer Web Interface can inadvertently deselect services that you have selected, preventing them from being installed. For example, if you select an auto-provisioning template on the **Select Services** page, and you also select additional services (for example, Streams Tools), and go to the next page, when you return to the **Select Services** page, Streams Tools will be deselected, and you will need to reselect it to ensure that it is installed. **Workaround:** Reselect any services that are deselected.
MAPR-20606	The Configure Service Layout page may assign services to a group with the name "Unprovisioned Services." **Workaround:** In the MapR Installer Web Interface, click **Restore Default**.
N/A	You cannot use the MapR Installer after you upgrade the cluster using the command line.

After you use the command line to upgrade a cluster that you installed with the MapR Installer, the MapR Installer is not aware that the cluster runs the upgraded version. Therefore, MapR Installer does not install nodes and ecosystem components that apply to the upgraded version.

Workaround: Use the MapR Installer Stanzas `probe` and `import` commands to update the installer database. See "Using probe and import to Generate the Installer Database" in the data-fabric documentation.

Changing Timeout Values to Resolve MapR Installer Errors
Change timeout values to reduce errors when using the MapR Installer.

Sometimes the MapR Installer can return errors because the maprcli or filesystem commands that it uses time out because of network latency. For example, if logs indicate that the installer could not obtain the nodelist, or a filesystem operation timed out, changing a timeout value can enable the same MapR Installer operation to succeed.

You can change the following timeout parameters to improve the success of MapR Installer operations:

Timeout	Description	Default Value (minutes)
`standard`	The timeout used for maprcli commands and Hadoop filesystem operations.	2
`configure`	The timeout used for `configure.sh` operations.	10

You must specify timeout values as an integer greater than or equal to 0.

To change the `standard` or `configure` timeout values:

1. On the MapR Installer node, navigate to the following directory:

   ```
   /opt/mapr/installer/ansible/playbooks/group_vars/
   ```

2. Edit the `all` file to change one or both timeout values to a number greater than or equal to 0. In this example, the timeouts are set to 7 minutes and 15 minutes respectively:

   ```
   timeout:
     standard: 7
     configure: 15
   ```

3. If a MapR Installer error prompted you to change the timeout, retry the operation that failed.

Installer Release Notes

This release of the MapR Installer works with RedHat/CentOS, Ubuntu, and SLES.

The MapR Installer consists of the following packages along with the `mapr-setup.sh` script:

- **MapR Installer** - Package that contains the MapR Installer.
- **MapR Installer Definitions** - Package that contains the list of MapR versions, services, and ecosystem components that you can install with the MapR Installer.

The release notes include the following sections:

MapR Installer Updates
MapR Installer updates provide new features or bug fixes.

The following table shows the MapR Installer new-feature updates by version:

Version	Updates
1.17.0.1.202201201546-1	This version: • Provides Log4j fixes. For more information, see advisory 4916. • Fixes CVE-2021-42392 in the Installer.

1.17.0.0.202110261002-1	This version:
	• Adds support for EEP 8.0.0, EEP 7.1.1, EEP 6.3.5, and version updates to many ecosystem components. For a list, see What's New in EEP 8.0.0 on page 6509.
	• Is built on Ubuntu 18.04 and is not supported on Ubuntu 16.04. Note that core 6.2.0 is supported on Ubuntu 16.04. You can use Installer 1.17.0.0 to install core 6.2.0 on Ubuntu 16.04 as long as the Installer node is running an OS that Installer 1.17.0.0 supports. For details, see Selecting an Installer Version to Use on page 5402. For supported OS versions, see the MapR Installer Support Matrix on page 5599.
	• Includes some terminology updates. References to the *ecosystem pack (MEP)* have been changed to *Ezmeral Ecosystem Pack (EEP)*. For more information, see What's New in EEP 8.0.0 on page 6509.
	• Includes numerous fixes, the most significant of which are:
	  • IN-2264: Probe does not detect hive database settings like it does for hue and oozie
	  • IN-2490: Change ansible source module to mapr_ansible
	  • IN-2560: Releases installer-v1.11.0 and installer-v1.12.0 - mapr-setup.sh by default installs the most current version of the Installer
	  • IN-2848: correctly detect and report ansible syntax errors
	  • IN-2856: Installer Failing when doing incremental upgrade
	  • IN-2924: need to create symlink for mapr_fstab when using mapr-loopbacknfs
	  • IN-2934: Ubuntu16 - mapr-setup.sh gpgkeys: protocol `https` not supported
	  • IN-2935: Centos83 - fresh install failing with "mount.nfs: access denied by server while mounting localhost:/mapr"
	  • IN-2947: extend cluster on ubuntu fails on nodes that do not have zookeeper installed

last-updated: Apr 11, 2022

Version	Updates
1.16.0.3.202201072207-1	This version: • Provides Log4j fixes. For more information, see advisory 4916. • Fixes CVE-2021-42392 in the Installer.
1.16.0.2.202110191345-1	This version provides defect repair and enables installation for clusters running Ubuntu 16.04. See the special considerations for Ubuntu 16.04 clusters in Selecting an Installer Version to Use on page 5402. This version was released at the same time as Installer 1.17 and includes fixes for the following issues: • IN-2895: permission errors reading pid files due to systemd changes • IN-2984: SLES15 fails in verify - python3 incompatibility • IN-2989: Error during cluster installation - 'ascii' codec can't encode character • IN-2995: add check to the os prereq to check for mep-7.1.0 or above with SLES15 • IN-2996: java 11 jre is not being upgraded to jdk • IN-2997: installer does not install java11 on SLES15
1.16.0.1.202108210519-1	This version includes fixes for the following issues: • IN-2924: need to create symlink for mapr_fstab when using mapr-loopbacknfs • IN-2925: installer no longer correctly detects if zk are running • IN-2934: Ubuntu16 - mapr-setup.sh gpgkeys: protocol `https` not supported • IN-2935: Centos83 - fresh install failing with "mount.nfs: access denied by server while mounting localhost:/mapr" • IN-2947: extend cluster on ubuntu fails on nodes that do not have zookeeper installed • IN-2950: Bug in regex for mapr-setup.sh causes update command to fail

last-updated: Apr 11, 2022

Version	Updates

1.16.0.0.202105261033-1	This version:
	• Adds support for EEP 7.1.0, SLES 15 SP2, and the following ecosystem component version updates:
	  • Hadoop 2.7.5
	  • Object Store 2.1.0
	  • HTTP-FS 1.1
	  • Kafka 2.6.1
	• Includes fixes for the following issues:
	  • IN-2821: DB services fail with error - Could not connect to the database: java.sql.SQLException: The server time zone value 'PDT' is unrecognized or represents
	  • IN-2849: prereq check for services does not figure out that it needs to look for chronyd instead of ntpd for SLES 15
	  • IN-2862: Couldn't install cluster by mapr user with sudo
	  • IN-2879: FreshInstall - error while configuring mysql
	  • IN-2883: mapr-installer-cli probe fails on Ubuntu-based MapR 6.2 cluster
	  • IN-2892: mysql prereq check fails if no password is given - should only do so if not password was given on fresh install
	  • IN-2893: make sure cron is installed - both OT and ES depend on it
	  • IN-2895: permission errors reading pid files due to systemd changes

last-updated: Apr 11, 2022

Version	Updates

1.15.0.1.202103220200-1	This version:
	• Adds support for MapR Data Platform maintenance release 6.1.1, EEP 6.3.3, and SLES 12 SP5.
	• Includes fixes for the following issues:
	  • IN-2669: WebUI Node Configuration - usage of 'hostname -A' value
	  • IN-2727: Support for SLES 12 SP4 and SP5 for Installer versions that support MapR 6.1
	  • IN-2782: Tez UI not configured on security correctly via UI installer
	  • IN-2784: Can't install cluster on CentOS 8.1 EEP-7.0.1
	  • IN-2787: Check that sudoers file allow all commands for certain user if non-root user selected for installing
	  • IN-2793: add check to network prereq check to make sure nodes have a domain portion in FQDN hostname
	  • IN-2805: installer fails to parse mapr-patch file pattern for ubuntu files and 6.2.0 files

last-updated: Apr 11, 2022

Version	Updates

1.15.0.0.202101220818-1	This version:
	• Adds support for installing the S3 Gateway. For installation considerations, see Installing the S3 Gateway Using the Installer on page 5429.
	• Installs Loopback NFS (`mapr-loopbacknfs-<version>`) on all nodes in the cluster unless **Enable NFS** is specified.
	• Adds support for Oracle Enterprise Linux 8.2 on release 6.2.0.
	• Includes fixes for the following issues:
	  • IN-1289: Allow removal of installer node during verify
	  • IN-2004: Install loopbacknfs by default if customer does not install NFS
	  • IN-2232: We complain about firewall services being enabled on ubuntu
	  • IN-2608: mapr-setup.sh installing with local repository - error on mapr-installer.service start
	  • IN-2726: Installer should use HTTPS while connecting to https://package.mapr.hpe.com/
	  • IN-2728: Mapr-installer failed installation with "Module did not set no_log for update_password"
	  • IN-2735: Installer-master/ui1.14 - add mep506 632 701 support
	  • IN-2742: Ansible execute shell script with no TTY
	  • IN-2743: Certify Installer support for 6.2 on OEL (Oracle Enterprise Linux) 8.2
	  • IN-2744: Implement simple UI for service verification feature in the Installer UI
	  • IN-2754: Add loopbacknfs installation and configuration via MapR Installer
	  • IN-2770: Installer needs to reflect minimum cluster requirement for 4 node clusters, rather than 5 node
	  • IN-2778: need to change localhost login for root on shared mysql on ubuntu
	  • IN-2785: Installer logs plenty of 'loglevel_int' exception messages
	  • IN-2789: Centos 8 Installer with Use existing DB - database_existing.yml "No package MySQL-python available."

last-updated: Apr 11, 2022

Version	Updates
1.14.0.1.202010161154-1	This version adds support for Red Hat / CentOS 8.2 and Oracle Enterprise Linux 7.8, as indicated in MapR Installer Support Matrix on page 5599. This version also includes fixes for the following issues: • IN-2680: mapr-setup.sh should provide option to setup Proxy preload • IN-2714: Support MapR core (no EEP) for Oracle Enterprise Linux 7.8 for MapR 6.1 • IN-2715: Installer won't start on centos 8.2 when java 8 is installed • IN-2716: mapr-setup.sh - errors via Shell Script plugin validation • IN-2717: Installer Specification for versions supported and paths navigated on OS, java, and python environments • IN-2718: mapr-setaup.sh - ERROR: environment variable is set but do not contain http[s]: prefix - fix • IN-2719: proxy setting for mapr_core, mapr_installer.repo mep rep is incorrectly set to _none_always when installer on HPE network • IN-2720: install core 6.1.0 on centos 8 • IN-2721: centos8 610_631 fresh install error on Stanza - Running task: Calling do_configure.sh from 'configure.yml' MODULE FAILURE • IN-2723: Please try an upgrade from core 6.1.0 to core 6.2.0 via installer on centos 8.x • IN-2724: Probe is broken in 1.14 - python3 and 4 digit issues
1.14.0.0.202009160311-1	This version includes the following new functionality or characteristics: • MapR Installer 1.14.0.0 supports EEP 7.0.0. • MapR Installer 1.14.0.0 supports Red Hat / CentOS 8.1, Ubuntu 18.04, and Ubuntu 16.04, but does not support SLES, as indicated in Operating System Support Matrix on page 5522. • Unlike MapR Installer 1.13.0, MapR Installer 1.14.0.0 runs on JDK 11. • The Installer can install Kafka Schema Registry 5.1.2.0.

last-updated: Apr 11, 2022

Version	Updates
1.13.0.0.201912130933-1	This version includes the following new functionality or characteristics: • MapR Installer 1.13.0.0 supports EEP 6.3.0. • MapR Installer 1.13.0.0 supports Red Hat / CentOS 7.6 and 7.7 but does not support Ubuntu 18.04, as indicated in Operating System Support Matrix on page 5522. • MapR Installer 1.13.0.0 supports HBase 1.13.0. • Unlike MapR Installer 1.12.0, MapR Installer 1.13.0.0 requires Java 8 and cannot run on Java 7.
1.12.0.0.201905241518-1	This version includes the following new functionality or characteristics: • MapR Installer 1.12.0.0 and later support the upload of a zipped tarball containing custom Ansible roles. These instructions can be executed in the installer workflow to customize the installation. For more information, see Using Custom Playbooks on page 5430. • You cannot use MapR Installer 1.12.0.0 or later to perform basic installer operations with MapR 5.2.x releases. You must use MapR Installer 1.11.0.0 instead. See Selecting an Installer Version to Use on page 5402. • For data disks, MapR Installer versions 1.12.0.0 and later require a minimum disk size that is equal to the physical memory on the node. If a data disk does not meet the minimum disk size requirement, a verification error is generated. • If you click a link in a MapR Installer tooltip, the link is displayed in a new browser tab. Previously, clicking a tooltip link caused the browser to display the link in the same browser tab, and any changes made in the user interface were lost. This behavior has been corrected. • IN-2389 & INFO-1120: Sqoop2 cannot be installed using MapR Installer 1.12. • IN-2417: Extending a secure cluster no longer fails with a `maprusertickets` error when you add a CONTROL node and DATA node at the same time.

Version	Updates
1.11.0.0.201902141709-1	No new features. This version includes the following fixes: • IN-2254: Ran out of memory on 1.10 installer - bump memory limits. • IN-2171: Unable to install Installer from packages on SLES. • IN-2155: Restrict expanding cluster groups based on service type not on group name. • IN-2154: Installer verification phase fails with error "ValueError: zero length field name in format." • IN-2123: Unable to extend secure cluster, maprusertickt not copied into new node. • IN-2108: Adding node with Extend Cluster fails at "get list of the ECO warden config files." • IN-2094: Installer verification fails for LUKS encrypted disks. • IN-2078: AttributeError in prereq check about OS. • IN-2025: Installer Error "Unscalable Host Groups" during 'extend cluster' or 'incremental install'.
1.11.0.0.201901301400-1	This version includes the following new functionality or characteristics: • Support for EEP 6.1.0. For more information about EEP 6.1.0, see What's New in EEP 6.1.0 on page 6529. • The Installer provides a new advanced option that allows you to restrict the software to a subset of network interface cards (NICs) or to specify public IP addresses that can be used with the cluster nodes. See Using the MapR Subnet and MapR External Advanced Options on page 5444. • The Installer warns you if you specify a host name using an IP address rather than a fully qualified domain name (FQDN). See Connectivity on page 133. • The Installer warns you if you run `mapr-setup.sh -r <url>` and specify a URL that is different from the URL currently stored in the `properties.json` file. See Troubleshooting Repository URL Errors on page 5459. • The installer version (1.11.0.0) now uses four digits rather than three or two digits. For more information about 4-digit versions, see What's New in EEP 6.1.0 on page 6529.

last-updated: Apr 11, 2022

Version	Updates
1.10.0.201812181130-1	No new features. This version includes the following fixes: • IN-2171: Unable to install Installer from packages on SLES. • IN-2155: Restrict expanding cluster groups based on service type not on group name. • IN-2154: Installer verification phase fails with error "ValueError: zero length field name in format." • IN-2123: Unable to extend secure cluster, maprusertickt not copied into new node. • IN-2108: Adding node with Extend Cluster fails at "get list of the ECO warden config files." • IN-2094: Installer verification fails for LUKS encrypted disks. • IN-2078: AttributeError in prereq check about OS. • IN-2025: Installer Error "Unscalable Host Groups" during 'extend cluster' or 'incremental install.'
1.10.0.201809200839-1	This version includes the following new functionality: • The installer provides a new option for setting Data-at-Rest Encryption (DARE). See Using the Enable MapR DARE Option on page 5428. • The installer provides a new option for setting the NSF version. See Installing NFS Using the MapR Installer on page 5429. • Changes have been made to the auto-provisioning templates. See Auto-Provisioning Templates on page 5448. • With MapR Installer 1.10, metrics collection is enabled by default on MapR 6.1 or later and cannot be disabled. Metrics collection is required for metering. To support metrics collection, collectd is installed on all nodes in the cluster. Users can specify the full collection configuration for collectd or a minimum configuration to support only metering for billing purposes. If the minimum configuration is selected, MapR Database table metrics are disabled. • With MapR Installer 1.10, the installer no longer includes an option to support off-cluster Elasticsearch and OpenTDSDB when security is turned on. • When installing Oozie 4.3.0, MapR Installer 1.10 leverages the Hadoop credential provider API to encrypt the Oozie database user password.

<table>
<tr><th>Version</th><th>Updates</th></tr>
<tr><td>1.9.0.201803291415-1</td><td>This version includes the following new functionality:
In the MapR Installer, password verification is implemented for every password entered.
Passwords for the <code>admin</code> user are now requested during the configuration of Grafana and ElasticSearch if the security mode requires them. For more information, see theMapR Installer operational changes in Operational Changes (MapR 6.1.0).
Changes have been made to the auto-provisioning templates. See Auto-Provisioning Templates on page 5448.
The following services are renamed in MapR Installer 1.9:<table><tr><th>Old Name</th><th>New Name</th></tr><tr><td>OJAI Query Service</td><td>OJAI Distributed Query Service</td></tr><tr><td>HBase/MapR Database Common</td><td>MapR DataBase</td></tr></table></td></tr>
<tr><td>1.8.0.201801312110-1</td><td>This version includes support for MapR 6.0, EEP 4.1.0, and the following new features:
Support for Starting Up a Cluster Using the MapR Installer Startup Button on page 5445.</td></tr>
<tr><td>1.7.201801021321-1</td><td>No new features. This version includes the following fixes:
IN-1417: Cannot decode unicode character running Stanza
IN-1451: CFT_converged.yml when used with customized AMI errors out while creating stack.
IN-1445: URLs not accessible in AWS Marketplace offering.
IN-1422: Hue broken on fresh Install Azure and AWS Marketplace offers.
IN-1391: Use private hostname for Azure provisioning.</td></tr>
</table>

last-updated: Apr 11, 2022

Version	Updates
1.7.201711082221-1	This version includes the following new features: • Support for Version 6.1.0 Release Notes on page 34 and MapR Ecosystem Pack (EEP) Reference on page 6504 4.0.0, 3.0.2, 2.0.3, and 1.1.4. • A new option for enabling or disabling security for the cluster. See Using the Enable MapR Secure Cluster Option on page 5427. • **Import State** and **Export State** commands that allow you to recover more easily from MapR Installer failures. See Importing or Exporting the Cluster State on page 5447. • The ability to change timeout parameters to improve the success of MapR Installer operations. • The ability to uninstall a service by deselecting it during an Incremental Install operation. • Support for the `scaled_hosts2:` parameter for MapR Installer Stanzas. MapR Installer 1.7 (which includes MapR Installer Stanzas) no longer supports the `scaled_hosts:` parameter for adding nodes to an on-premise cluster. Instead, you need to use the `scaled_hosts2:` parameter. See Extending a Cluster by Adding Nodes.
1.6.201708241301-1	This version includes the following new features: • Support for Installing MapR in the cloud.
1.6.201708012220-1	This version includes the following new features: • Support for adding nodes. See Extending a Cluster by Adding Nodes on page 5437. • A new one-click **Shutdown** command. See Shutting Down a Cluster Using the MapR Installer Shutdown Button on page 5446. • Support for 3-digit EEPs. See Understanding Two-Digit and Three-Digit EEPs on page 5450. • Support for creating a MapR Installer PACC using the MapR Installer setup script. See Creating an Installer Container Using mapr-setup.sh on page 5498.

Version	Updates
1.5.201704051050-1	This version includes the following new features: • Support for EEPs 3.0, 2.0.1, and 1.1.2. See MapR Ecosystem Pack (EEP) Reference on page 6504. • Support for maintenance updates. See Performing a Maintenance Update on page 5447. • Enhancements to MapR Installer Stanzas. See Using probe and import to Generate the Installer Database on page 5514. • Enhancements to `mapr-setup.sh` in support of user-created MapR PACC containers. See Creating a MapR PACC Image Using mapr-setup.sh on page 409. • Support for applying patches. See Applying a Patch Using the MapR Installer on page 438. • Support for SLES 12 SP1.
1.4.201612081140-1	This version includes the following new features: • Support for EEP 2.0. • Support for MapR Installer Stanzas. • MapR Streams Tools. Note these restrictions: • MapR Installer 1.4 prevents you from downgrading a EEP. • MapR Installer 1.4 shows only EEP versions equal to or greater than the version currently installed on your cluster. This version includes the following updates: • 24891: Installer should not allow you to downgrade the EEP version. • 25213: Error in the process of upgrading Sentry by UI Installer 1.4. • 25277: v..swappiness should be set to "1" rather than "0" in RHEL/CentOS kernels > 2.6.32-303. • 23285: Installer leaves children behind after a stop. • 25421: mapr-setup fails to upgrade java.
1.3.201609291954-1	This version includes the following updates: • 23258: The installer now creates a new log for each node for each iteration of the installer. Previously, the installer overwrote the original log file. • 23868: The installer now updates the livy_server_host property on Hue nodes and restarts the live server correctly.

last-updated: Apr 11, 2022

Version	Updates
1.3.201608121412-1	This version includes the following new features: • When you install or upgrade to MapR 5.2, you select a EEP version before you select ecosystem components. • When you install or upgrade to MapR 5.2, you have the option to install MapR Monitoring. • For local MapR 5.2 installations, the `-a` parameter of `mapr-setup.sh` expects three archive files: MapR Installer archive, MapR Ecosystem Pack (EEP) archive, and the MapR 5.2 archive. This version includes the following updates: • 23853/22267: The installer validates advanced disk layouts such as /dev/mapper/luks-sdd or /dev/mapper/<lvm_logical_volume_name>. • 22904: The installer now allows you to update the cluster license during an incremental installation. • For local MapR 5.0 and 5.1 installations, the archive filename has changed to mapr-5.0-5.1.<yyyymmdd>.*.tgz. • 24295: The installer detects disks correctly on Red Hat 7 operating system versions. • 24340: The installer no longer points to an old package manager (RPM) for the EPEL repository.
1.2.201606140935-1	Version number change only.
1.2.201605032042-1	This version includes the following updates: • 23137: On each node in the cluster, MapR Installer automatically applies the latest patch available for the installed JDK version. • 22892: MapR Installer no longer fails on Red Hat operating system versions that do not support systemd.
1.2.201602260909-1	This version includes the following updates: • 21115: The installer performs a prerequisite check on the operating system version of Ubuntu and CentOS/Redhat nodes. • 21877: The `mapr-setup.sh` script provides the ability to specify a non-default hostname and port that nodes in the cluster can use to communucate with the MapR Installer node. You can set a non-default hostname or port using the new prompt that appears while mapr-setup.sh configures the MapR Installer node or pass the script the `-p <hostname:port>` option when you run the command to execute the mapr-setup.sh script.

last-updated: Apr 11, 2022

Version	Updates
1.1.201601080826-1	This version includes the following updates: • 21861: The installer sets the no_proxy environment variable for http connections to the MapR Installer node. • 21440: On Safari and Internet Explorer web browsers, the Advanced Configuration settings for the service layout are now accessible.
1.1.201510241120-1	This version includes the following updates: • 19850: The installer now configures the MySQL database to work with local connections. • 20115: The installer disables SELinux without requiring a reboot. • 19156: When you abort an installation, the installer provides a link to download the installation logs.
1.0.201508041436-1	This version includes the following update: • If Java is not installed on RedHat/CentOS, mapr-setup.sh installs Open JDK Java 1.8.
1.0.201507171311-1	This version includes the following updates: • 19541: The installer no longer supports SSL DHE cipher. It will also disable SSLv3. • 19465: When you use a custom service template, the installer now accurately displays service compatibility.
1.0.201507091731-1	This version includes the following updates: • 18888: You no longer need to restart the mapr-installer process after you use the package manager to upgrade the installer package.
1.0.201506081725-1	This version includes the following updates: • 18926: Installations no longer terminate due the restart of the sshd process. • Minor changes to text and layout on the Services page.
1.0.201506011415-1	Version number change only.
1.0.201505261302-1	This version includes the following fixes: • 18682: When you use package manager to update the installer definition file, you no longer need to re-run mapr-setup.sh. • 18748: The MapR Installer no longer fails to install Hive 0.13 on Ubuntu when the repository includes Hive 1.0.

Installer Help Links

Here are some topics that can help you use the MapR Installer Web Interface.

last-updated: Apr 11, 2022

Using the MapR Installer

To get started using the MapR Installer, see MapR Installer on page 5395.

Version-Specific Topics

Release 7.0 Links	• Upgrading MapR Core or EEP Components on page 284
	• For upgrades to core:
	  • Upgrade Workflows on page 285
	  • Preparing to Upgrade MapR Core on page 303
	  • Finishing the MapR Core Upgrade on page 322
	• For upgrades to ecosystem components:
	  • Preparing to Upgrade the MapR Ecosystem Pack on page 335
	  • Finishing the MapR Ecosystem Pack Upgrade on page 372
	• For cluster expansion using the Installer **Extend Cluster** function:
	  • Post-Expansion Steps for Extending a Cluster on page 5442

Release 6.1 Links	• Upgrading MapR or MapR Ecosystem Components
	• For upgrades to MapR Core:
	  • Upgrade Workflows on page 285
	  • Preparing to Upgrade MapR Core
	  • Finishing the MapR Core Upgrade
	• For upgrades to MapR Ecosystem components:
	  • Preparing to Upgrade the MapR Ecosystem Pack
	  • Finishing the MapR Ecosystem Pack Upgrade
	• For cluster expansion using the MapR Installer **Extend Cluster** function:
	  • Post-Expansion Steps for Extending a Cluster on page 5442

last-updated: Apr 11, 2022

Release 6.0 Links	• Upgrading MapR or EEP Components
	• For upgrades to MapR Core:
	  • Preparing to Upgrade MapR Core
	  • Finishing the MapR Core Upgrade
	• For upgrades to MapR Ecosystem components:
	  • Preparing to Upgrade the MapR Ecosystem Pack
	  • Finishing the MapR Ecosystem Pack Upgrade
	• For cluster expansion using the MapR Installer **Extend Cluster** function:
	  • Post-Expansion Steps for Extending a Cluster

Release 5.2 Links	• Upgrading MapR or MapR Ecosystem Components
	• For upgrades to MapR Core:
	  • Preparing to Upgrade MapR Core
	  • Finishing the MapR Core Upgrade
	• For upgrades to MapR Ecosystem components:
	  • Preparing to Upgrade the MapR Ecosystem Pack
	  • Finishing the MapR Ecosystem Pack Upgrade
	• For cluster expansion using the MapR Installer **Extend Cluster** function:
	  • Post-Expansion Steps for Extending a Cluster on page 5442

Release 5.1 Links	• MapR 5.1 Upgrade Guide • MapR 5.1 Pre-Upgrade Steps • MapR 5.1 Post-Upgrade Steps • Post-Expansion Steps for Extending a Cluster on page 5442

MapR Installer Containers

This section describes how you can obtain the MapR Installer as a Docker container.

A MapR Installer container is a Docker container that contains the MapR Installer. You can use a MapR Installer container to perform basic installer operations from a node that is not necessarily a part of a MapR cluster. For example, from a MapR Installer container, you can perform the following actions:

- Start and run the web-based MapR Installer to install a new cluster, apply a patch, or perform an update.
- Run Installer Stanza commands to probe an installed cluster.

You can create your own MapR Installer container by using the `mapr-setup.sh` script. Or you can download a pre-built container image for theMapR Installer.

last-updated: Apr 11, 2022

Creating an Installer Container Using mapr-setup.sh

This section describes how to create an Installer Container image by using the `mapr-setup.sh` script.

Creating the Image

To create an Installer image using `mapr-setup.sh`:

1. Download the `mapr-setup.sh` script to a Linux or Mac OS X platform where Docker 1.12.5 or later is installed and the Docker daemon is up and running. Choose *one* of the following download options:

   Note: Running `mapr-setup.sh` on Windows is not supported.

   - Download the setup script directly from package.mapr.hpe.com to the node that will run the MapR Installer:

     ```
     wget https://package.mapr.hpe.com/releases/installer/
     mapr-setup.sh -P /tmp
     ```

   - Download to your local workstation, and copy to the node that will run the MapR Installer:

     a. On the Download Page, click **Download**, and save the setup script to a location on your workstation.

     b. Use a tool such as SCP to copy the file to the node that will run the MapR Installer:

     ```
     scp mapr-setup.sh user@server /tmp
     ```

2. Run the `mapr-setup.sh` script with the `docker installer` command to create the Docker image:

   ```
   ./mapr-setup.sh docker installer
   ```

3. Respond to the command-line prompts to provide the information to configure the image. The following table describes each prompt. If you press **Enter** without specifying a value, `mapr-setup.sh` uses the default value shown in the square brackets ([]):

Prompt	Notes
`Build MapR UI Installer image? (y/n) [y]`	Type y to continue or n to exit the script.
`Image OS class (centos7, ubuntu16) [<local OS>]:`	Specify the base operating system on which to build the image. **Note:** SLES is not currently supported.
`Docker FROM base image name:tag [centos:centos7]:`	Specify the starting image used to create the new image. If necessary, you can enter your own tag and image name to choose a base image already created for your installation. If you do not enter a name, the script provides one for you.
`MapR installer image tag name [<name>]`	Accept the software-provided name for the container image, or provide your own name. This is the name you will use to run the image to create the MapR Installer container.

`Container memory: specify host XX[kmg] or 0 for no limit [0]:`	Specify the maximum amount of memory (in kilobytes, megabytes, or gigabytes) that Docker allows the container to access. For example: • 2g • 4096m • 0 Accepting the default (0), means there is no restriction on memory, and the container can use as much memory as the platform makes available.

4. After the last prompt, press **Enter**. The script:
   - Prepares the installer
   - Installs or verifies installer package dependencies
   - Installs installer packages
   - Cleans up unneeded files
   - Creates the `mapr-docker-installer.sh` sample-run file and displays the location of the file:

     ```
     Edit '/root/docker_images/installer/mapr-docker-installer.sh' to
     configure settings and then execute it to start the container
     ```

     `mapr-docker-installer.sh` contains environment variables for the image and makes it easy for you to start the container.
5. *(Optional)* Edit the `mapr-docker-installer.sh` script file if you want to change any environmental variables. For more information about the environmental variables, see Environmental Variables for the MapR Installer Container on page 5501.
6. Run the `mapr-docker-installer.sh` file to start mapr-installer services:

   ```
   ./docker_images/installer/mapr-docker-installer.sh
   ```

   After the installer service is started, you can issue Stanza commands or open the web-based MapR Installer in a browser:

   ```
   installer (380) started with log /opt/mapr/installer/logs/installer.log
   Started service mapr-installer

   ...Success

       To continue installing MapR software, open the following URL in a
   web browser

                If the address '172.17.0.2' is internal and not accessible
              from your browser, use the external address mapped to it
   instead

                                https://172.17.0.2:9443
   ```


last-updated: Apr 11, 2022

Note: The MapR Installer maintains the state of the cluster. When the installer is run from a container, the installer database is only as persistent as the container itself. If you need the installer data to be persistent, here are some options:

- If you shut down a MapR Installer container, use the `docker start` command (not the `docker run` command) to restart the same instance of the container. If you create a new instance of the container, the database will have no information.
- Mount the `/opt/mapr/` data directory outside the container to persistent storage to maintain the cluster state.
- Use the Stanza `export` command to export the state of the cluster before you shut down the container. See Exporting a Cluster Configuration.

Running the MapR Installer Container Using Stanza Commands

You can also use the sample-run file to execute a MapR Installer Stanza command. In this scenario, the command creates the installer container, runs a Stanza command, and then shuts down the container. For example, the following command runs the Stanza `probe` command on node 10.10.88.53:

```
./docker_images/installer/mapr-docker-installer.sh probe -o
config.ssh_id=root config.ssh_password=mapr
config.hosts='["10.10.88.53"]' -nv
```

For a list of the Stanza commands, see MapR Installer Stanza Commands on page 5515.

Using the Pre-Built MapR Installer Container Images

This section describes how to obtain and run the pre-built MapR Installer container images.

Pre-built MapR Installer container images are available on Docker hub. Images are available for:

- Ubuntu 16.04
- Ubuntu 14.04
- CentOS 7
- CentOS 6

The pre-built images are about 200 MB. A sample-run script that you can use to start the container is available on GitHub.

To use a pre-built image:

1. Download the pre-built MapR Installer container image and the sample-run script to a Linux or Mac OS X platform where Docker 1.12.5 or later is installed and the Docker daemon is up and running.
   - You can download the pre-built image from the data-fabric public repository.
   - You can download the sample-run script (`mapr-docker-installer.sh`) from this GitHub location.
2. *(Optional)* Edit the `mapr-docker-installer.sh` script file if you want to change any environmental variables. For more information about the environmental variables, see Environmental Variables for the MapR Installer Container on page 5501.
3. Run the `mapr-docker-installer.sh` file to start mapr-installer services:

   ```
   $ ./docker_images/installer/mapr-docker-installer.sh
   ```

After the installer service is started, you can issue Stanza commands or open the web-based MapR Installer in a browser:

```
installer (380) started with log /opt/mapr/installer/logs/installer.log
Started service mapr-installer

...Success

    To continue installing MapR software, open the following URL in a
web browser

              If the address '172.17.0.2' is internal and not accessible
          from your browser, use the external address mapped to it
instead

                              https://172.17.0.2:9443
```

The MapR Installer maintains the state of the cluster. When the installer is run from a container, the installer database is only as persistent as the container itself. If you need the installer data to be persistent, here are some options:

- If you shut down a MapR Installer container, use the `docker start` command (not the `docker run` command) to restart the same instance of the container. If you create a new instance of the container, the database will have no information.
- Mount the `/opt/mapr/` data directory outside the container to persistent storage to maintain the cluster state.
- Use the Stanza `export` command to export the state of the cluster before you shut down the container. See Exporting a Cluster Configuration.

Environmental Variables for the MapR Installer Container
This section describes environmental variables that you can modify to customize the sample-run script for the MapR Installer container.

About mapr-docker-installer.sh

`mapr-docker-installer.sh` is the sample-run script for the MapR Installer container. The script contains the `docker run` command that runs the container and environmental variables that can be passed into the container at run time. Modifying the variables is optional; the script can run without any changes to the environmental variables.

The following environmental variables can be changed in `mapr-docker-installer.sh`:

Environmental Variables in mapr-docker-installer.sh

Key	Variable	Description
`MAPR_CONTAINER_USER`	`<user-name>`	The user that the user application inside the Docker container will run as. This configuration is functionally equivalent to the Docker native `-u` or `--user`. Do not use Docker `-u` or `--user`, as the container needs to start as the `root` user to bring up FUSE before switching to the MAPR_CONTAINER_USER.

Key	Variable	Description
		The user specified here is the user that all storage operations on the MapR cluster will be performed as. Therefore, HPE recommends not using `root` or `mapr`. This user also owns the `/opt/mapr` directory tree.
`MAPR_CONTAINER_UID`	`<uid>`	The UID that the application inside the Docker container will run as. This is a companion to the MAPR_CONTAINER_USER option. If a UID is not provided, the default is UID 1000. Providing a UID is strongly recommended.
`MAPR_CONTAINER_GROUP`	`<group-name>`	The group that the application inside the Docker container will run as. This is a companion to the MAPR_CONTAINER_USER option. If a group name is not provided, the default is `users`. Providing a group name is strongly recommended.
`MAPR_CONTAINER_GID`	`<gid>`	The GID that the application inside the Docker container will run as. This is a companion to the MAPR_CONTAINER_USER option. If a GID is not provided, the default is GID 1000. Providing a GID is strongly recommended.
`MAPR_PKG_URL`	`<mapr-pkg-url>`	The URL of the repository that hosts the MapR packages (typically https://package.mapr.hpe.com/). If you change MAPR_PKG_URL, use the full URL to your MapR repository (for example, `<hostname>/releases`).
`MAPR_CONTAINER_PASSWORD`	`<mapr-password>`	The password for the `mapr` user. This password must be set if the container will remaining running. The password defaults to `mapr`.
`MAPR_STANZA_FILE`	`<stanza-file>`	The path to a MapR Installer Stanza file that needs to be mounted from the host. You must specify this field if you issue the Stanza `install` command as an argument to the script. For example: `mapr-docker-installer.sh install -nv`
`MAPR_MEMORY`	`<mapr-memory>`	Specify the maximum amount of memory (in kilobytes, megabytes, or gigabytes) that Docker allows the container to access. For example: • 2g

last-updated: Apr 11, 2022

Key	Variable	Description
		• 4096m • 0 Accepting the default (0), means there is no restriction on memory, and the container can use as much memory as the platform makes available.
MAPR_TZ	<time-zone>	The time zone inside the container. For a list of time-zone settings, see this website. The default is UTC.

MapR Installer Stanzas

This section describes how to prepare for and use MapR Installer Stanzas to install, upgrade, or uninstall MapR software.

MapR Installer Stanzas enable API-driven installation of MapR clusters. An extension of the Spyglass Initiative, MapR Installer Stanzas enable the creation of a YAML configuration file (a "Stanza") that describes a cluster. A command-line addition to the web-based MapR Installer allows you to execute the configuration file programmatically to automate new deployments.

You can use MapR Installer Stanzas when you need a script-based tool to install MapR software and you do not want to click through the menus and options provided by the MapR Installer. MapR Installer Stanzas provide less visual feedback than the MapR Installer, but they can be faster and more efficient at installing software on clusters with many nodes.

To use MapR Installer Stanzas:

1. **Review the** Installer Stanza Prerequisites.
2. **Use the steps in** Installer **to download and run the mapr-setup.sh script.** Performing these steps installs both the web-based MapR Installer and the MapR Installer Stanzas feature.
3. **Review or edit the Stanza file.** The Stanza file specifies the installation parameters for your cluster. See Working with MapR Installer Stanza Files on page 5504.
4. **Run the MapR Installer Stanza file.** See Running Installer Stanza Files.

MapR Installer Stanza Prerequisites

This topic describes some limitations and guidelines that you must understand before using MapR Installer Stanzas.

Most prerequisites that apply to the MapR Installer also apply to MapR Installer Stanzas. To review the MapR Installer prerequisites, see MapR Installer Prerequisites and Guidelines on page 5396.

Some additional prerequisites are unique to MapR Installer Stanzas:

- To install MapR Installer Stanza features, you must download version 1.4 or later of the MapR Installer. For more information, see MapR Installer Updates on page 5481 and Updating the MapR Installer on page 5409.
- You can use MapR Installer Stanzas to install, upgrade, or uninstall only Release 5.1 and later releases.
- You can use MapR Installer Stanzas to upgrade or uninstall only clusters that were previously installed using the MapR Installer or a MapR Installer Stanza. If your MapR software was installed manually, you cannot use MapR Installer Stanzas on the cluster because the cluster does not have the installer database.

last-updated: Apr 11, 2022

Working with MapR Installer Stanza Files

This topic describes how to use the sample Stanza files that are provided with the MapR Installer.

The MapR Installer Stanza file specifies how the cluster should be configured, including the configuration of nodes, disks, software versions, and services. You must configure a Stanza file before using MapR Installer Stanzas to install or upgrade a cluster. Sample Stanza files (basic and advanced) are located in this directory:

`/opt/mapr/installer/examples`

To create your Stanza file:

1. Make a copy of one of the sample files. For example, make a copy of the `sample_basic` or `sample_advanced` Stanza file, and rename the copy to a name of your choosing.
2. Using any text editor, edit the Stanza file to address the needs of your installation. Comment out any instructions that you don't need.

This table describes how to complete the fields in the Stanza file:

Section	Parameter	Required/Optional	Directions
environment	mapr_core_version	Required	Specify 5.1.0 or later. Be sure to include the third digit (for example, 5.1.0 or 5.2.0).
	patch_location	Optional	Specify the location of a MapR core patch file. The file name must use the format `mapr-patch-${mapr_core_version}`. For example: `mapr-patch-5.2.0.39122.GA-4198.x86_64.rpm` Use an absolute path to specify the patch file location.

config	hosts	Required	Specify the list of hosts on which you want to install packages. You can list the hosts on a single line as follows:
			`- examplenode[1-3].example.com`
			Or you can use multiple lines:
			`- examplenode1.example.com`
			`- examplenode2.example.com`
			`- examplenode3.example.com`
			Note:
			• The installer host must be one of the hosts in this list.
			• A comma-separated list of hosts is not supported.
			• You can override hosts and disks from the command line. The following example specifies installing only on hosts 01 through 04. To pass an array into an override, use single quotes, double quotes, and brackets as follows:
			`cli install -nv -t sample_adv.yaml -o config.hosts='["lab[01-04].yourlab.com"]'`
			Note: In a command with an override, the `key=value` pair or pairs that follow `-o` must not contain a blank space. For more information about `key=value` pairs, see MapR Installer Stanza Commands on page 5515.

last-updated: Apr 11, 2022

	ssh_id	Required	Specify the user ID that the installer will run under when it installs the packages. This user must have root access and must be present on every host defined in the hosts section.
	ssh_password	Optional	Specify the password (for use with the password-based login). This is the password that the installer uses to log into the node using ssh. Comment this line if you want to use a private-key-based login.
	ssh_key_file	Optional	Specify the path to a file that contains the private key (for use only with the private key-based login). Uncomment this line if you want to use a private-key-based login. If both the ssh_password and ssh_key_file lines are uncommented, the installer will default to the ssh_key_file.
	ssh_port	Optional	Uncomment this line if you need to specify a port other than the default, which is 22.
	license_type	Required	Specify M3 for the community edition or M5 for the enterprise edition.
	mep_version	Required for MapR 5.2.0 and later	Specify a currently supported EEP, such as 2.0. For EEP information, see EEP Components and OS Support on page 5536.
	disks	Required	List the disks on which the packages will be installed. The disk names should be the same on every node. This field is required, and you must use the following notation: `- /<diskname>` `- /<diskname>` `- /<diskname>` A comma-separated list of disks is not supported. Not all of the disks need to be present, but at least one disk on each node must be present.

	disk_stripe	Optional	Uncomment this line if you need to specify a disk stripe value other than the default, which is 3.
	elasticsearch_path	Optional	Uncomment this line if you need to specify a path where Elasticsearch data will be stored.

	services	Optional	Specify a predefined template of services. Services are the MapR core or ecosystem components (or tools) that run on each group. The installer configures default services automatically unless you specify a "groups" section in the Stanza file. To view the predefined templates, use the `list` command with the `--type TEMPLATE` argument. **Note:** • Metrics are provided by default. The metrics services apply only to MapR 5.2 and later. • Logging services, if specified, apply only to MapR 5.2 and later. • If you do not specify any services, the Installer will install only MapR core services. • For an incremental install or upgrade, the Installer discovers any services already installed. Additional services are installed only if you request specific services, a EEP upgrade, or a MapR core upgrade. • You can override services using a command such as the following: `-o config.services='{"mapr-oozie": {} , "mapr-hivemetastore": {"database": {"name": "hive", "user": "hive", "password":`

last-updated: Apr 11, 2022

groups (advanced layout only)		Optional	To provision the cluster manually, add services in groups. A group is a collection of hosts (nodes) that runs a specific set of services. The installer creates groups automatically unless you specify the groups manually (manual provisioning). To specify the groups manually, you must include a groups section and define the hosts, labels, and services that your cluster needs. For an example, see the sample_advanced Stanza file. The services you specify must not be of type GROUP. To view the type for a given service, use the `list services` command with the `--name <name-of-service>` argument. Also, when provisioning manually, you must ensure that the mapr-core service is present on every node.
	hosts	Required (if a groups section is specified)	Hosts are nodes that run a specific set of services.
	label	Optional	The label is a descriptor for each group. Use the label to describe the group function or some other aspect of the group that is meaningful to your installation. If a label is not specified, the installer will auto-generate a label based on the service names.
	services	Required (if a groups section is specified)	See the "Services" section earlier in this table.

Running MapR Installer Stanza Files
This section describes how to install, upgrade, and uninstall MapR software and check the progress of these operations by using MapR Installer Stanza commands.

Note: To run MapR Installer Stanza commands, you must navigate to the installer directory. This applies to all the examples in this section:

```
cd /opt/mapr/installer
```

Installing or Upgrading MapR Core Using an Installer Stanza
Use the Stanza `install` command to install Release 5.1 or later, install additional features, upgrade a cluster, perform a maintenance update, or apply a patch.

You can use the `install` command of the MapR Installer Stanza command suite to:

last-updated: Apr 11, 2022

- Perform a fresh install of Release 5.1 or later.
- Perform an incremental install (add or upgrade services that are already installed on the cluster).
- Upgrade a cluster to a newer data-fabric software version or a newer EEP version.
- Perform a maintenance update.
- Apply a patch (see Applying a Patch Using an Installer Stanza on page 444).
- Extend the cluster by adding nodes (see Extending a Cluster by Adding Nodes on page 5437).

For the `install` command syntax and options, see MapR Installer Stanza Commands on page 5515 later in this section.

This example installs data-fabric software using the parameters specified in the `sample_basic.yaml` Stanza file. To run this command, you should be logged in as the `mapr` user. The `-nv` option specifies that certificates will not be checked and the output mode is verbose. The `-t`option, which is required, specifies the use of a template file:

```
./bin/mapr-installer-cli install -nv -t ./examples/sample_basic.yaml
```

If you are using a MapR Installer Stanza to install data-fabric software on a cluster that has never had data-fabric software installed (a fresh installation), it is recommended to create the `mapr` user on all nodes. You can create the `mapr` user by using the `config.cluster_admin_password` override. For example:

```
./bin/mapr-installer-cli install -nv -t ./examples/sample_basic.yaml -o
config.cluster_admin_password=mapr
```

If you use the `install` command and an existing cluster is detected, the installer attempts an incremental install or upgrade using the parameters in the specified Stanza file:

- For incremental installs, the installer does not check or verify the configuration.
- You cannot add nodes or services during a version upgrade.

Note: If the password in the Stanza file or in the command contains a special character, such as an exclamation point (!), you might need to escape the character with a backslash (\). For example:

```
./bin/mapr-installer-cli install -u mapr:mapr\!@localhost -nv -t ../
examples/sample_basic.yaml
```

If you do not want to include a password in the Stanza file, you can specify a value contained in a secured file. This example uses a Stanza file (`sample_nopwd.yaml`) in which the `ssh_password` line has been removed. The secured file (`installer.cfg`) stores the value of `ssh_password` as `config.ssh_password=mapr`. The contents of `installer.cfg` are piped to the `install` command via an override (`-o -`). You must include the `-` after the `-o`; otherwise, the file contents are not read.

```
cat examples/installer.cfg | ./bin/mapr-installer-cli install -nv -t
examples/sample_nopwd.yaml -o -
```

To check the progress of the installation or upgrade, see Checking the Progress of Operations. For another example of using the `install` command, see How to Build Stanzas (blog).

New Installation of Release 6.0 Secure Cluster Using Stanzas

To install a Release 6.0 secure cluster using Stanzas, you must add two parameters to the Stanza:

- `config.security: "true"`

- `config.cluster_admin_password: "<mapr_user_password>"`

For example:

```
config:
  security: "true"
  cluster_admin_password: "mapr"
```

After the installation completes, `secure=true` should be set in the `/opt/mapr/conf/mapr-clusters.conf` file. This command should print the ticket details:

```
maprlogin print -ticketfile /opt/mapr/conf/maprusertickет
```

Note these considerations for installing a Release 6.0 secure cluster:

- If you use a Stanza to perform a secure install, you must log out and then log in one time for the bashrc to take effect.
- For non-bash environments, you must manually add the above epxort to your login profile.
- You can use the `probe` command to detect whether a cluster is secure or not.

Uninstalling MapR Core Using an Installer Stanza
Specifies how to uninstall MapR Core from the command line.

You can use the `uninstall` command to uninstall the current data-fabric software version. The `uninstall` command requires that you specify two overrides:

- `config.ssh_id`
- `config.ssh_password` or `config.ssh_key_file`

Note: Using the the `uninstall` command requires `root` privileges. You can provide the `root` ID and password or the `root` ID and `ssh_key_file`.

This example uninstalls the currently installed data-fabric software. The command uses an override to provide the `ssh_id` and `ssh_password` and includes `-nv` so that certificates will not be checked, and the output mode is verbose.

```
./bin/mapr-installer-cli uninstall -nv -o config.ssh_id=root -o
config.ssh_password=mapr
```

This example uninstalls the currently installed data-fabric software. The command uses an override to provide the `ssh_id` and `ssh_key_file` and includes `-nv` so that certificates will not be checked, and the output mode is verbose. The `ssh_key_file` for root normally resides in `/root/.ssh`. In this example, the file has been copied to `/home/mapr/root` so that the `mapr` user can access the key file.

```
./bin/mapr-installer-cli uninstall -nv -o config.ssh_id=root -o
config.ssh_key_file=/home/mapr/root_user_id_rsa
```

Exporting a Cluster Configuration

If a cluster was installed using the MapR Installer or a MapR Installer Stanza, you can use the `export` command to generate a Stanza that captures the state of the cluster. You can then modify the Stanza to perform incremental installs or upgrades.

last-updated: Apr 11, 2022

This example uses the `export` command to generate the `tt.yaml` Stanza file. The command includes `-n` so that certificates will not be checked:

```
./bin/mapr-installer-cli export -n --file /tmp/tt.yaml
```

Note: The `export` command does not export the `config.ssh_password` field. When using the exported YAML file, you need to provide the password manually, pass it as an override, or specify a value contained in a secured file, as described in Installing or Upgrading MapR Core Using an Installer Stanza on page 5509.

For another example of using the `export` command, see MapR Stanzas (blog).

Getting Information About Services and Groups

Use the Stanza `list` command to get additional information about your cluster.

You can use the `list` command and its arguments to get information about the configuration, services, groups, hosts, and services in the cluster. While the `list` command provides the state of the cluster, `list` output is not suitable for incremental installs and upgrades. You must use the `export` command if you want to generate a Stanza file that can be used for upgrading.

This example displays a listing of all the services, groups, and hosts that were installed:

```
./bin/mapr-installer-cli list installed -n
```

This example lists all the groups:

```
./bin/mapr-installer-cli list groups -n
```

This example lists all the hosts:

```
./bin/mapr-installer-cli list hosts -n
```

This example lists the installation status for all the hosts:

```
./bin/mapr-installer-cli list hosts_install_status -n -u https://
mapr:<password>@<installer_ip_addr>:9443
```

This example lists the services by template:

```
./bin/mapr-installer-cli list services -n --type TEMPLATE|more
```

You can use the `list services` command to learn about different kinds of services. For example:

Group Type	Example Command
CONTROL	`./bin/mapr-installer-cli list services -n --type CONTROL`
MULTI_MASTER	`./bin/mapr-installer-cli list services -n --type MULTI_MASTER` `./bin/mapr-installer-cli list services -n --type MONITORING_MASTER`

Group Type	Example Command

DATA	`./bin/mapr-installer-cli list services -n --type DATA`
	`./bin/mapr-installer-cli list services -n --type DEFAULT`
	`./bin/mapr-installer-cli list services -n --type CLIENT`

SINGLE MASTER	`./bin/mapr-installer-cli list services -n --type MASTER`

Verifying the Nodes

You can use the `check` command to verify that the nodes specified in the Stanza file are ready to be installed. The `check` command does not check the Stanza file and does not do any provisioning or installation. The `check` command runs the same checks that the MapR Installer runs during its Verification phase. For example, it checks that the nodes have the right OS version, that they have enough disk space, and that they have enough memory to support installation or other Installer operations.

This example verifies the nodes specified in the `sample_advanced_demo.yaml` file. The `-nv` option specifies that certificates will not be checked and the output mode is verbose. The `-t` option, which is required, specifies the use of a template file:

```
./bin/mapr-installer-cli check -nv -t sample_advanced_demo.yaml
```

Resetting the Installer Database

The `reset` command uninstalls the metadata from the Installer database. `reset` is for advanced users.

You can reset the Installer database by using the CLI `reset` command or by using the MapR Installer web interface (**Support > Reset Installer**).

The reset function can be useful for testing purposes, but use reset with caution. If you reset the installer database while packages are installed on the nodes, you will need to remove the packages manually.

Note: If you experience a failure while installing or uninstalling, the installer prompts you to retry the operation or uninstall and then reinstall from scratch. You should always retry or uninstall before considering using `reset`.

This example resets the Installer database. The `-nv` option specifies that certfificates will not be checked and the output mode is verbose:

```
./bin/mapr-installer-cli reset -nv
```

Adding a License Using Stanzas

Explains how to add a license using Installer Stanzas.

In addition to using `maprcli`, REST commands, or the Control System to add a license, you can add a license using MapR Installer Stanzas. After obtaining a valid license file from your MapR sales representative, copy the license file to a cluster node (for example, to `/tmp/license.txt`). You can then use the `license` command to apply the license to the cluster. For example:

```
mapr-installer-cli license -n -l <path-to-license-file>
```

Checking the Progress of Operations

This topic describes how to use the MapR Installer web interface and installer logs to check the status of MapR Installer Stanza operations.

last-updated: Apr 11, 2022

Using the MapR Installer to Check Status

The installer provides messages and progress bars to indicate the status of MapR Installer Stanza operations. However, you can also use the Installer interface to check the current status of your cluster. The MapR Installer provides a quick visual summary of the currently installed software versions and services.

To view the MapR Installer web interface:

```
https://<Installer Node hostname/IPaddress>:9443
```

Viewing the Installer Logs

To view or understand the logs generated by the MapR Installer or MapR Installer Stanza operations, see Installer Log Descriptions.

Shutting Down a Cluster Using an Installer Stanza Command

Use the Stanza `shutdown` command to shut down an on-premise or cloud-based cluster.

The Stanza `shutdown` command is available with MapR Installer 1.6 or later. The command shuts down Warden and Zookeeper, which in turn shut down other running services that are part of a data-fabric cluster. When you use the `shutdown` command, the MapR Installer implements the same orderly shutdown that it uses to perform software upgrades.

You must supply the ssh ID and password or the ssh ID and `ssh_key_file`.

The `shutdown` command works differently for cloud-based clusters. Note these considerations for the behavior of the `shutdown` command:

Cluster Location	Shutdown Command Behavior
On premise	Does not stop non-data-fabric software and does not power off the nodes.
In the cloud	Shuts down (but does not remove) all the nodes in the cluster. If the installer node is part of the cluster, the installer node is not shut down. To shut down the installer node, use AWS-console or Azure-portal commands to stop the instance.

This example shuts down a MapR cluster. The command uses an override to provide the `ssh_id` and `ssh_password` and includes `-nv` so that certificates will not be checked, and the output mode is verbose.

```
./bin/mapr-installer-cli shutdown -nv -o config.ssh-id=root -o
config.ssh_password=mapr
```

This example shuts down a MapR cluster. The command uses an override to provide the `ssh_id` and `ssh_key_file` and includes `-nv` so that certificates will not be checked, and the output mode is verbose. The `ssh_key_file` for root normally resides in `/root/.ssh`. In this example, the file has been copied to `/home/mapr/root` so that the `mapr` user can access the key file.

```
./bin/mapr-installer-cli shutdown -nv -o config.ssh_id=root -o
config.ssh_key_file=/home/mapr/root_user_id_rsa
```

Using probe and import to Generate the Installer Database

If a cluster has data-fabric software installed but has no installer database, you cannot install or upgrade the cluster using MapR Installer Stanza commands. And you cannot use the web-based MapR Installer on the cluster. However, you can generate an installer database on a cluster by using the `probe` and `import` commands. Once the installer database is generated, you can use the web-based installer and all MapR Installer Stanza commands on the cluster.

last-updated: Apr 11, 2022

Note: Using `probe` and `import` to generate the installer database is supported only on release 5.2 and later. If you need to use this feature on an earlier data-fabric software version, please contact your support representative.

Note: If you are not sure if your cluster has an installer database, you can use the `export` command to generate a YAML file that describes the cluster configuration. See Exporting a Cluster Configuration on page 5511.

Using probe

Before using the `probe` command, you must know the host names or IP addresses of the cluster nodes and the `root` user, which must be the same on all nodes. The probe command generates a template that will be used by the `import` command.

Note: Do not make changes to the `probe`-generated template file. After the installer database is created, you can use the `export` command to export a YAML for making changes.

In this example, the probing user, `mapr1`, probes an array of hosts (`config.hosts`) and generates a template file `/tmp/location.yaml`. The `-u` option provides login credentials for the MapR Installer. Note that the probing user must be able to do rpm and pkg queries and have permission to read certain files and directories within `/opt/mapr`.

```
mapr-installer-cli probe -n -o config.ssh_id=mapr1
config.ssh_password=xyz config.hosts='["hostname[1-3]","hostname7"]' -u
mapr1:xyz@<installer_hostname>:9443 > /tmp/location.yaml
```

Note: The `probe` command can use various methods to determine the EEP version of a node. One method checks the EEP repo URL defined on the node. If multiple EEP repos are defined on the same node, the `probe` command ignores all of them and tries to determine the EEP version based on theMapR packages that are present.

Using import

The `import` command prepares the installer database based on the probed template file. This example imports the probed YAML template file from the previous example. `-t` specifies the location of the template file:

```
mapr-installer-cli import -n -t /tmp/location.yaml
```

After you use the `import` command, the installer database should be operational. You can then use the web-based MapR Installer or Stanzas to perform additional operations on the cluster. See MapR Installer on page 5395.

MapR Installer Stanza Commands

This topic provides the syntax and options for MapR Installer Stanza commands.

Command Usage

To use MapR Installer Stanza commands, you must log in as the cluster administrator user that you configured while running the `mapr-setup.sh` script. For more information, see Managing Users and Groups.

`check`

last-updated: Apr 11, 2022

Use the `check` command to verify that the nodes specified in the Stanza file are ready to be installed. For more information, see Verifying the Stanza File.

```
check [-h]
     [--overrides [OVERRIDES [OVERRIDES …]]]
     [--no_check_certificate][--url URL]
     [--force][--verbose] --template TEMPLATE
```

`export`

You can use the `export` command to discover a currently installed cluster and generate a Stanza file that captures the configuration, the groups, and the hosts for the cluster. You can then modify the Stanza file to install other clusters. For more information, see Exporting a Cluster Configuration.

Note: The Stanza file generated by the `export` command does not contain the `ssh_password` or `ssh_key_file` values. You need to add those values manually or provide them on the command line by using an override.

```
export [-h]
     [--overrides [OVERRIDES [OVERRIDES ...]]]
     [--no_check_certificate] [--url URL]
     [--file FILE]
```

`import`

The `import` command prepares the installer database using the template generated by the `probe` command. You must specify a template (YAML file) to use the import command. For more information, see Using probe and import to Generate the Installer Database on page 5514.

```
import [-h][--no_check_certificate][--url URL][--verbose]
       --template TEMPLATE
```

`install`

You can use the `install` command to perform a fresh install, an incremental install, or an upgrade. For more information, see Installing or Upgrading MapR Core Using an Installer Stanza on page 5509.

```
install [-h]
     [--overrides [OVERRIDES [OVERRIDES …]]]
     [--no_check_certificate][--url URL]
     [--force][--verbose] --template TEMPLATE
```

`license`

Use the `license` command to add a license from a license file. For more information, see Adding a License Using Stanzas on page 5513.

```
license [-h]
      [--no_check_certificate][--url URL]
      --license LICENSE <license-file>
```

`list`

You can use the `list` command to display information about the configuration, services, groups, and hosts that are present in the MapR Installer database. For more information, see Getting Information About Services and Groups.

```
list [-h]
    {config,groups,hosts,hosts_install_status,installed,services} ...
```

`probe`

The `probe` command generates a template file that can be used to create an installer database on a cluster that doesn't have one. A cluster must have an installer database in order for you to use MapR Installer Stanzas on the cluster. If the installer database is not present, you can use the `probe` command followed by the `import` command to generate one. For more information, see Using probe and import to Generate the Installer Database on page 5514.

```
probe [-h]
      [--no_check_certificate][--url URL]
      [--force][--verbose] --template TEMPLATE
      [--overrides [OVERRIDES [OVERRIDES …]][--verbose]]
```

`reset`

The `reset` command uninstalls the metadata from the installer database. For more information, see Resetting the Installer Database.

```
reset [-h] [--overrides [OVERRIDES [OVERRIDES ...]]]
      [--no_check_certificate] [--url URL]
      [--force] [--verbose]
```

`shutdown`

The `shutdown` command shuts down the cluster. For more information, see Shutting Down a Cluster Using an Installer Stanza Command on page 5514.

```
shutdown [-h] [--overrides [OVERRIDES [OVERRIDES ...]]]
      [--no_check_certificate] [--url URL]
      [--force] [--verbose]
```

`uninstall`

The `uninstall` command removes MapR software. For more information, see Uninstalling Core Using an Installer Stanza.

```
uninstall [-h]
      [--overrides [OVERRIDES [OVERRIDES ...]]]
      [--no_check_certificate] [--url URL]
      [--force] [--verbose]
```

Command Options

All of the following command options are optional except for the `TEMPLATE` option:

Option	Description
`-h` or `--help`	Show this help message and exit.
`-o [OVERRIDES[OVERRIDES …]]` or `--overrides[OVERRIDES[OVERRIDES …]]x = y list of overrides or - for stdin`	Use the specification provided on the command line instead of the Stanza file instructions. Overrides are typically used when you want to specify a host name, user ID, password, or key file on the command line.

last-updated: Apr 11, 2022

	Note:
	The `key=value` pair or pairs specified in a Stanza override must not include blank spaces. For example, do not use this command:
	`./mapr-installer-cli probe -o config.hosts='["host1", "host2", "host3"]' .....`
	Use this command instead:
	`./mapr-installer-cli probe -o config.hosts='["host1","host2","host3"]' .....`
	In the second command, `config.hosts` is the `key`, and `'["host1","host2","host3"]'` is the `value`. An override can specify multiple `key=value` pairs with each pair separated by a space:
	`-o <key1=value1> <key2=value2> <key3=value3>`

`-n` or `--no_check_certificate`	Do not verify SSL certifications.
`-u URL` or `--url URL`	Installer URL (user:password@host:port)
`--file FILE`	The `export` file path.
`-f` or `--force`	Force action. This option eliminates user input and answers yes to all questions returned by the software.
`-v` or `--verbose`	Verbose output.
`-t TEMPLATE` or `--template TEMPLATE`	The specified Stanza template file (required for the `install` and `import` commands). The template file can be a local file on the installer node, or it can be a remotely accessible URL.

Command Line Help

You can access online help at the command line using any of these commands:

Command	This command displays...
`./bin/mapr-installer-cli -h`	Top-level help for the installer
`./bin/mapr-installer-cli <cmd> -h` where <cmd> is one of the following: • `check` • `export` • `import` • `install` • `license`	Help for the specified command

last-updated: Apr 11, 2022

• list • probe • reset • uninstall	

Interoperability Matrices

This section provides tables that show the operating system (OS), JDK, ecosystem, and MapR client support for the MapR Data Platform. Check these tables for information about software compatibility.

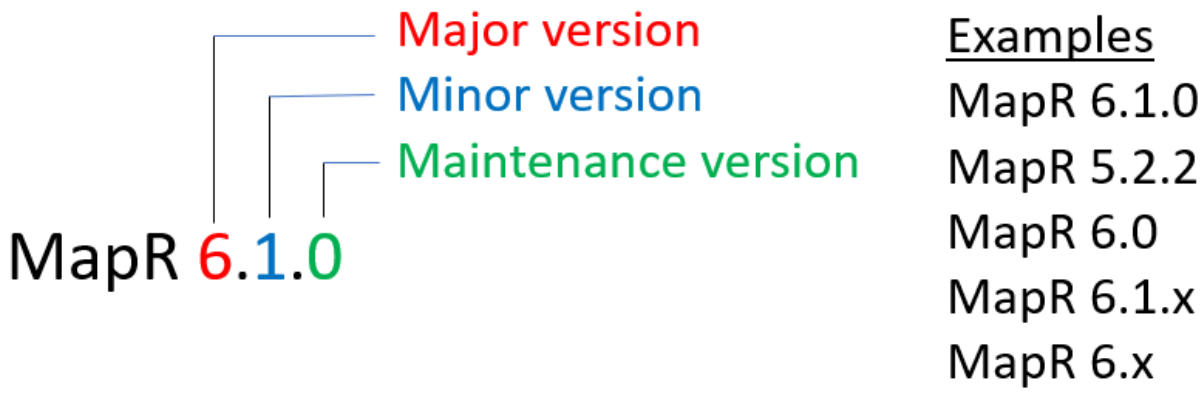
Understand MapR Versions

Understanding the version numbers used by MapR core, MapR Ecosystem Packs (EEPs), EEP components, and MapR patches can help you keep your software up to date and plan for upgrades.

MapR release versions generally follow the industry-standard **<major>.<minor>.<maintenance>** format, with a number representing each version. However, some MapR software products use different versioning formats. The following sections describe the characteristics of the various MapR versioning formats.

MapR Core Versions

In MapR interfaces and documentation, MapR core versions are expressed as a dot-separated string of numbers having two or three places. For example:



A Change to This Version	Typically Includes	For Example
MapR core major version	Significant new features and API changes that can introduce incompatibilities with the previous version.	MapR 6.x followed MapR 5.2.x and added built-in security features, database features (secondary indexes), new APIs (Apache OJAI and Change Data Capture), a new EEP series (EEP 4.x), filesystem enhancements, a new installer version, and a redesigned MapR Control System for monitoring the cluster.
MapR core minor version	New features, bug fixes, and API changes that are sometimes incompatible with the previous version.	MapR 6.1 followed MapR 6.0.1 and added security by default, a new ZooKeeper version, several new database features (Node.js and Python OJAI clients, complex type support, table metrics, support for the OJAI 3.0

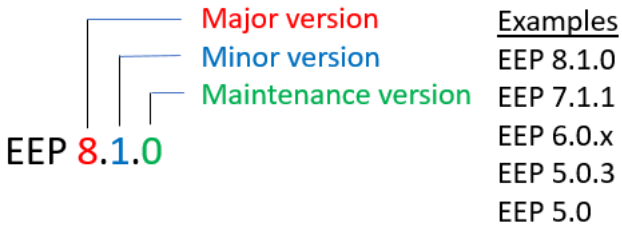
last-updated: Apr 11, 2022

A Change to This Version	Typically Includes	For Example
		API), a new installer version, a new EEP series (EEP 6.x), and storage tiers for the file system.
MapR core maintenance version	Bug fixes only. This version typically does *not* introduce incompatibilities with the previous version.[1]	MapR 5.2.2 followed maintenance releaseMapR 5.2.1 and introduced new EEPs, improvements to the disk space balancer tool, and a new MapR Installer version, but did not include new features or API changes.

[1]MapR 6.0.1 was an exception to this rule, adding REST API access to MapR Database JSON tables, event timestamp support for stream processing, streaming audit logs for the MapR XD Distributed File and Object Store, and a new EEP series (EEP 5.x).

EEP Versions

In MapR interfaces and documentation, MapR Ecosystem Pack (EEP) versions are expressed as a dot-separated string of numbers having two or three places. For example:



A Change to This EEP Version	Typically Includes	For Example
Major version	A new EEP series that supports a new major, minor, or maintenance version of MapR core.	EEP Support and Lifecycle Status on page 5531 shows how EEP major versions change when new versions of MapR core are introduced.
Minor version	A significant change in features or APIs but no change in support for the current MapR core version.	EEPs 6.0.1 and 6.1.0 both support MapR 6.1.0, but EEP 6.1.0 introduced new MapR Database features, such as the C# and Go OJAI clients, as well as new features for ecosystem components (Flume, Oozie, Tez, and Apache Kafka). EEP 6.2.0, which also supports MapR 6.1.0, introduced new ecosystem component versions (Hue, Impala, Spark) and updated some components (Oozie and Hive).
Maintenance version	Bug fixes only. EEP maintenance versions do not add new features or API changes and are backward compatible in the same EEP series.	EEPs 5.0.3, 5.0.2, and 5.0.1 added bug fixes to the EEP 5.0.x line while preserving compatibility with MapR 6.0.1.

About the MapR Patch Version

Beginning with EEP 6.1.0, version numbers for some ecosystem components and MapR tools use a dot-separated string having four places instead of two or three places. For example:

- Oozie 5.1.0.0
- MapR Installer 1.11.0.0

last-updated: Apr 11, 2022

The fourth numeral represents the MapR patch version. The MapR patch version adds a unique descriptor for patches that can occur between releases. In future releases, as new versions of components are released, more components and tools will transition to a version string that includes the MapR patch version.

How the MapR Patch Version Increments

Beginning with EEP 6.2.0, the MapR patch version can be a number in the range 0 through 999. For a newly released component, the patch version starts at 0 and increments by 1 when there is an EBF update. For example, if the EEP 6.1.0 general availability (GA) package version for Oozie is 5.1.0.0, the first bug-fix (EBF) package version will be Oozie 5.1.0.1. The next bug fix package version will be Oozie 5.1.0.2, and so on.

If the same version of a component is present in multiple EEPs, the patch version increments by 100 to provide a range of usable version numbers for future patches. This numbering scheme ensures that all patches have a unique version. The following table shows how the Oozie patch version increments for EEPs 6.1.0, 6.1.1, and 6.2.0:

EEP	Oozie GA Version	1st Oozie Patch Version	2nd Oozie Patch Version
6.1.0	5.1.0.0	5.1.0.1	5.1.0.2
6.1.1	5.1.0.100	5.1.0.101	5.1.0.102
6.2.0	5.1.0.200	5.1.0.201	5.1.0.202

To obtain EBF patches, you must contact HPE Support. HPE Support makes EBF patch versions available for specific known issues. See Patches for Known Issues.

Maven Version Format

Beginning with EEP 6.2, the MapR patch version also is present in Maven artifacts for components that use the four-place versioning. In addition, the Maven version format changes to include the associated EEP instead of the YYMM timestamp:

Old Maven format with YYMM timestamp: `maprdb-spark-2.2.1-mapr-1803.jar`

New Maven format with EEP number: `maprdb-spark-2.3.3.0-mapr-602.jar`

The Maven version increments in the same way that the MapR patch version increments. This table shows how the Spark package version and Maven versions increment when new EBF patch versions become available:

EEP	EEP 6.0.2	EEP 6.1.1	EEP 6.0.3*	EEP 6.1.2*
Spark Package Version	`mapr-spark-2.3.3.0.<timestamp>`	`mapr-spark-2.3.3.100.<timestamp>`	`mapr-spark-2.3.3.200.<timestamp>`	`mapr-spark-2.3.3.300.<timestamp>`
Spark 1st EBF Package Version	`mapr-spark-2.3.3.1.<timestamp>`	`mapr-spark-2.3.3.101.<timestamp>`	`mapr-spark-2.3.3.201.<timestamp>`	`mapr-spark-2.3.3.301.<timestamp>`
Spark Maven Version	`2.3.3.0-mapr-602`	`2.3.3.100-mapr-611`	`2.3.3.200-mapr-603`	`2.3.3.300-mapr-612`
Spark 1st EBF Maven Version	`2.3.3.1-mapr-602`	`2.3.3.101-mapr-611`	`2.3.3.201-mapr-603`	`2.3.3.301-mapr-612`

*This release is not currently available and is included only for illustration purposes.

A patch release can trigger periodic updates to the Maven repository. You may notice patches in the sftp.mapr.com repository that are not updated in the Maven repository. This is normal. The

last-updated: Apr 11, 2022

Maven repository will be updated eventually. The latest patch versions are always propagated first to sftp.mapr.com. If you have questions about patches, contact your HPE support representative.

Related concepts

Understanding Two-Digit and Three-Digit EEPs on page 5450
Understanding the differences between the EEP directories on https://package.mapr.hpe.com/releases/MEP/ can help you prevent versioning issues.

Maven Artifacts for MapR on page 4155
Maven artifacts can be used for dependency management when developing applications based on the MapR platform.

Checking the MapR Core Version on page 5415
Some maintenance operations require you to know the version of the currently installed MapR release (sometimes referred to as the "MapR core version"). You can check the MapR core version easily from within the Control System or MapR Installer user interface or identify the version from your installed packages.

Checking the EEP Version on page 5413
Some MapR Installer operations require you to know the version of the currently installed MapR Ecosystem Pack (EEP). You can check the EEP version easily from within the MapR Installer user interface or derive the EEP version from your repository information.

Related reference

Core Support and Lifecycle Status on page 5530
This page shows the support and lifecycle status for all versions of MapR Data Platform core software.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP Components and OS Support on page 5536
MapR Ecosystem Packs consist of ecosystem components and MapR-monitoring components that can run on a variety of operating systems.

EEP Support and Lifecycle Status on page 5531
This page shows the EEPs that are supported for different core releases and the current lifecycle status for each EEP.

Related information

Apache Hadoop Release Versioning
Change in Support for Older MEP Releases

Operating System Support Matrix

This matrix shows the operating system versions supported by release 6.0.0 and later releases.

For a list of the EEPs that you can use with each core release, see EEP Support and Lifecycle Status on page 5531. For release 5.2.x information, see Operating System Support Matrix (MapR 5.x) on page 5524.

OS Version / Release Version	Release 7.0.0	Release 6.2.0	Release 6.1.1	Release 6.1.0	Release 6.0.1	Release 6.0.0

Red Hat Enterprise Linux (64-bit)	8.4	Yes	Yes	Yes[4, 5, 7, 9]	Yes[1, 4, 5, 7, 9]	No	No
	8.3	Yes	Yes	Yes[4, 5, 7, 9]	Yes[1, 4, 5, 7, 9]	No	No
	8.2	Yes	Yes	No	Yes[1, 4, 5, 7, 9]	No	No
	8.1	Yes	Yes	No	No	No	No
	7.9	No	No	Yes	Yes[1]	Yes[6]	No
	7.8	No	No	No	Yes[1]	Yes[6]	No
	7.7	No	No	No	Yes[1]	No	No
	7.6	No	No	No	Yes	No	No
	7.5	No	No	No	Yes	Yes[2]	No
	7.4	No	No	No	Yes	Yes	Yes
	7.3	No	No	No	Yes	Yes	Yes
	6.10	No	No	No	No	No	No
CentOS (64-bit)[10]	8.3[10]	No	Yes	No	Yes[1, 4, 5, 7, 9]	No	No
	8.2[10]	No	Yes	No	Yes[1, 4, 5, 7, 9]	No	No
	8.1[10]	No	Yes	No	No	No	No
	7.9[10]	No	No	Yes	Yes	No	No
	7.8[10]	No	No	No	Yes[1]	Yes[6]	No
	7.7[10]	No	No	No	Yes[1]	No	No
	7.6[10]	No	No	No	Yes	No	No
	7.5[10]	No	No	No	Yes	Yes[2]	No
	7.4[10]	No	No	No	Yes	Yes	Yes
	7.3[10]	No	No	No	Yes	Yes	Yes
	6.10[10]	No	No	No	No	No	No
Ubuntu (64-bit)	20.04	Yes	No	No	No	No	No
	18.04	Yes	Yes	Yes	Yes	No	No
	16.04[8]	No	Yes	Yes	Yes	Yes	Yes
	14.04	No	No	No	Yes	Yes	Yes
SLES (64-bit)	15 SP3	Yes[11]	No	No	No	No	No
	15 SP2	Yes[11]	Yes[11, 12]	No	No	No	No
	12 SP5	No	No	Yes	Yes[1]	No	No
	12 SP4	No	No	Yes	Yes[1]	No	No
	12 SP3	No	No	No	Yes	Yes[2]	No
	12 SP2	No	No	No	Yes	Yes	Yes

last-updated: Apr 11, 2022

Oracle Enterprise Linux[3]	8.4	Yes	No	No	No	No	No
	8.3	Yes	No	No	No	No	No
	8.2	Yes	Yes	No	No	No	No
	7.8	No	No	No	Yes[1]	No	No
	7.4	No	No	No	Yes	Yes	Yes
	7.3	No	No	No	Yes	Yes	Yes

[1]Requires a patch. See this support advisory. To install patches, see Applying a Patch on page 437.

[2]Use Installer 1.10 or later with this release. Before upgrading your operating system to Red Hat Enterprise Linux 7.5, CentOS 7.5, or SLES12 SP3, be sure to update the Installer to version 1.10 or later. See Updating the MapR Installer on page 5409.

[3]Releases 7.0.0 and 6.2.0, including core and ecosystem components (EEP 7.x.x and later), can be used on the supported versions of Oracle Enterprise Linux. Releases 6.1 and 6.0 support Oracle Enterprise Linux for core installations only (ecosystem components are not supported).

[4]Supported only for EEPs 6.3.0 and later.

[5]NFSv4 is not supported for core 6.1.0 on Red Hat Enterprise Linux / CentOS 8.x.

[6]Requires a patch. Certified only for EEP 5.0.4. Other EEPs supported by release 6.0.x might work but have not been certified.

[7]Hue 4.3.0 in EEP 6.3.0 and 6.3.1 is not supported for Red Hat Enterprise Linux (RHEL) or CentOS 8.x. However, Hue 4.3.0.300 in EEP 6.3.2 can be used with RHEL or CentOS 8.x.

[8]RDMA is not supported with release 6.2 on Ubuntu 16.04.

[9]Installing release 6.1.0 on RHEL or CentOS 8.x requires you to enable the EPEL repository. Starting in RHEL 8.x, a `mapr-core-internal` package dependency (`sdparm`) is deprecated and moved to EPEL, and installation cannot complete without enabling it. For information about adding repositories, see Adding the MapR Repository on RHEL, CentOS, or Oracle Linux on page 143.

[10]CentOS Linux 8 has reached End of Life (EOL) status. For more information, see this page. See also the HPE Ezmeral Data Fabric Advisory in response to CentOS 8 Discontinuance.

[11]NFS v4 is not supported for releases 7.0.0 and 6.2.0 on SLES 15.

[12]Requires a patch (core 6.2.0.7 or later). The https://package.mapr.hpe.com/ repository automatically provides the right patch version.

Operating System Support Matrix (MapR 5.x)

This matrix shows the operating systems and versions supported by each MapR 5.x release.

OS Version / MapR Version	MapR 5.2.2	MapR 5.2.0, 5.2.1	MapR 5.1.0	MapR 5.0.0	MapR 4.0.2	MapR 4.0.1

Red Hat (64bit)[6]	7.6[8]	Yes	No	No	No	No	No
	7.5[1]	Yes[2]	No	No	No	No	No
	7.4[1]	Yes	No	No	No	No	No
	7.3[1]	Yes	Yes	Yes	No	No	No
	7.2[1]	Yes	Yes	Yes	No	No	No
	7.1[1]	No	No	Yes	No	No	No
	7.0[1]	No	No	Yes	Yes	Yes	Yes
	6.10	Yes	No	No	No	No	No
	6.9[8]	Yes	No	No	No	No	No
	6.7[7], 6.8	Yes	Yes	Yes	Yes	No	No
	6.5, 6.6[7]	Yes	Yes	Yes	Yes	Yes	Yes
	6.1, 6.2, 6.3, 6.4	No	No	Yes	Yes	Yes	Yes
	5.7, 5.8, 5.9	No	No	No	No	No	No
CentOS (64bit)[3,6]	7.6[8]	Yes	No	No	No	No	No
	7.5[1]	Yes[2]	No	No	No	No	No
	7.4[1]	Yes	No	No	No	No	No
	7.3[1]	Yes	Yes	Yes	No	No	No
	7.2[1]	Yes	Yes	Yes	No	No	No
	7.1[1]	No	No	Yes	No	No	No
	7.0[1]	No	No	Yes	Yes	Yes	Yes
	6.9[8]	Yes	No	No	No	No	No
	6.7, 6.8	Yes	Yes	Yes	Yes	No	No
	6.5, 6.6	Yes	Yes	Yes	Yes	Yes	Yes
	6.4	No	No	Yes	Yes	Yes	Yes
	6.1, 6.2, 6.3	No	No	Yes[3]	Yes[3]	Yes[3]	Yes[3]
	5.7, 5.8, 5.9	No	No	No	No	No	No
Ubuntu (64bit)	14.04	Yes	Yes	Yes	Yes	Yes	Yes
	12.04	Yes	Yes	Yes	Yes	Yes	Yes
	11.04	No	No	No	No	No	No
SLES (64bit)[1,4]	12, 12 SP1, 12 SP2	Yes	Yes	Yes	Yes	No	No
	11 SP3	No	No	No	Yes	Yes	Yes
	11 SP1, 11 SP2	No	No	No	No	No	No
Oracle Enterprise Linux[5]	7.2	Yes	Yes	Yes	No	No	No
	7.0, 7.1	No	No	Yes	No	No	No
	6.5, 6.6, 6.7, 6.8	Yes	Yes	Yes	No	No	No
	6.4	No	No	Yes	No	No	No

last-updated: Apr 11, 2022

[1]MapReduce (MRv1) metrics are supported only on Red Hat/CentOS 6. MapReduce (MRv1) metrics are not installable on SLES or RedHat / CentOS 7.

[2]MapR Installer 1.9 and earlier are not supported on Red Hat 7.5 or CentOS 7.5 with MapR 5.2.2. Before upgrading your operating system to Red Hat 7.5 or CentOS 7.5, be sure to update the MapR Installer to version 1.10 or later. See Updating the MapR Installer on page 5409.

[3]If you are installing MapR on CentOS Version 6.3 or earlier, see the Known Issues section in the Release Notes.

[4]Secure MapR clusters are not supported on SLES 11 platforms.

[5]MapR 5.2 and 5.1 support Oracle Enterprise Linux for Core installations only. Ecosystem components are *not* supported.

[6] MapR 5.0 supports RHEL/CentOS 7.0, but service management with `systemd` is not supported. Starting in RHEL/CentOS 7.0, `systemd` was included alongside the legacy process manager, `initd`, but was not enabled by default. In RHEL/CentOS 7.1, `systemd` became the default. Some RHEL/CentOS 7.0 ISOs may have a different default for `systemd` (on), but these will not work with MapR 5.0. Service management integration with `systemd` is only supported if you install MapR 5.1 or 5.2 on RHEL/CentOS 7.1. Customers who encounter issues with `systemd` support may choose to disable `systemd` on RHEL/CentOS 7.0 nodes, downgrade to RHEL/CentOS 6.x, or upgrade to MapR 5.1 or 5.2.

[7]MapR does not support RedHat when its kernel patch level is below kernel-2.6.32-504.16.2.el6. Patch levels below kernel-2.6.32-504.16.2.el6 include Linux kernel 3.14-3.17, which has a bug that may cause the following issues:

- Lost wake-ups between process threads.
- MapR processes hang.

For more information, see https://rhn.redhat.com/errata/RHSA-2015-0864.html.

> **Tip:** You can run the following command to determine the kernel patch level of your operating system:
> `uname -r`

[8]The MapR Installer and MapR Installer Stanzas are currently not supported for use with this OS version.

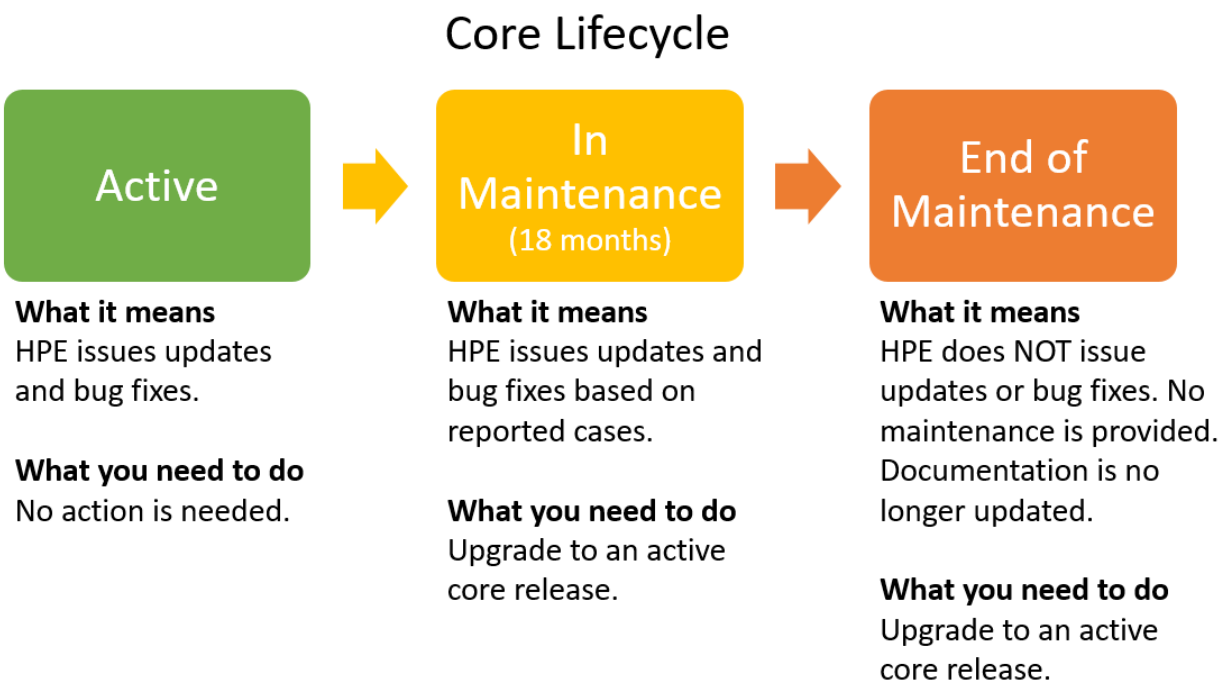
Understand the Core Lifecycle

Describes the MapR Data Platform core lifecycle and defines the lifecycle stages, which are Active, In Maintenance, and End of Maintenance.

Core Lifecycle Stages

Hewlett Packard Enterprise periodically releases new core software. Each core release is supported for an amount of time that can vary depending on the new releases that follow it. When new core versions are released, older core versions are deprecated or discontinued. Each core version therefore has its own lifecycle. As shown in the diagram, a core release can transition through three lifecycle stages:

- Active
- In Maintenance (18 months)
- End of Maintenance



Typically, within six months after a new release, Hewlett Packard Enterprise issues an advisory to indicate the end of maintenance for older versions of core. Eighteen months after the advisory is issued, the In-Maintenance core version reaches the End-of-Maintenance stage and is discontinued.

To view the current lifecycle status for every core release, see Core Support and Lifecycle Status on page 5530. The following table describes the lifecycle stages:

Support and the Lifecycle Stages

Support Activity	Notes	Lifecycle Stage		
		Active	In Maintenance	End of Maintenance
Proactive Maintenance (Minor, Maintenance, Patch)	Includes proactive fixes for security vulnerabilities, critical bugs, and other issues.	Yes	No	No
Reactive Maintenance (Escalation Support + Patch)	Requires the user to open cases resulting in tactical fixes for critical bugs, where backporting is feasible.	Yes	Yes[1]	No
Assisted Support (Usage / Debug Support)	Does not include patch fixes.	Yes	Yes	No

[1]Includes fixes for critical bugs and CVEs reported to Support. Does not include documentation updates.

Core Versions

In MapR Data Platform interfaces and documentation, core versions are expressed as a dot-separated string of numbers having two, three, or four places. Updates and bug

last-updated: Apr 11, 2022

fixes result in changes to the major, minor, maintenance, and patch versions of a core

release:

Notification of Changes in Support for Released Core Versions

To notify users about changes in EEP support, Hewlett Packard Enterprise issues periodic support advisories. When core releases are deprecated or discontinued, users of those releases are encouraged to upgrade to newer versions.

For service advisories, see:

- All Advisories
- Security Vulnerability Advisories
- Critical Issue or Bug Advisories
- Product Discontinuance (EOL) Advisories

Understand the EEP Lifecycle

This page describes the EEP lifecycle and defines the lifecycle stages, which are Active, Deprecated, and Discontinued.

EEP Lifecycle Stages

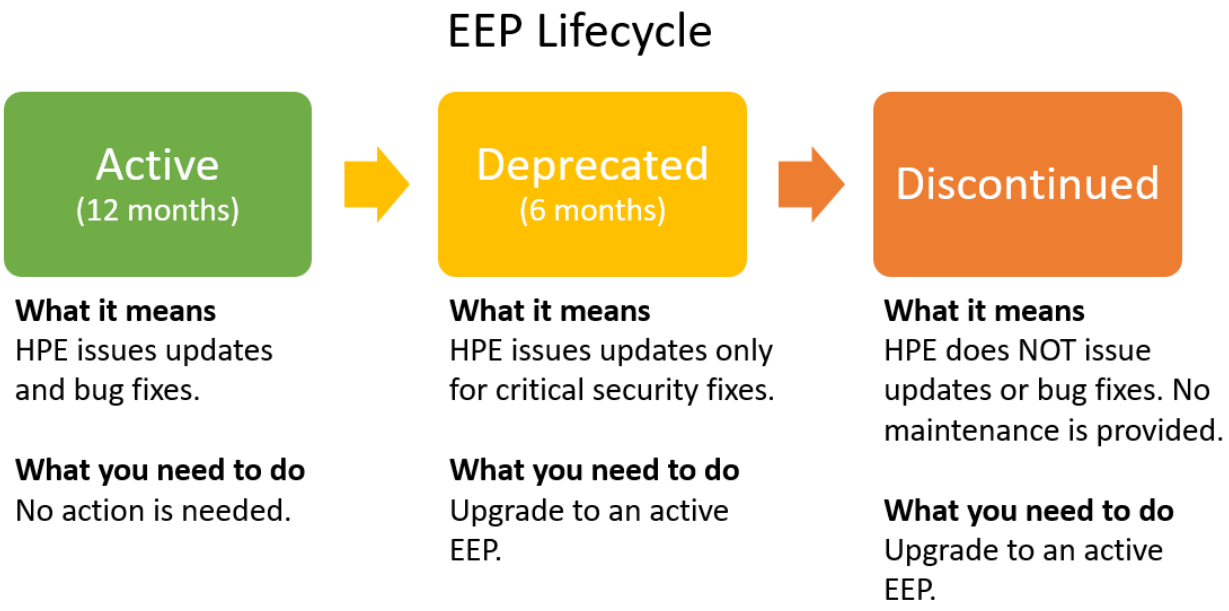
HPE periodically releases new ecosystem components as ecosystem packs (EEPs). Each EEP is supported for 18 months. When new EEPs are released, older EEPs are deprecated or discontinued. Each EEP therefore has its own lifecycle.

As shown in the diagram, an EEP can transition through three lifecycle stages:

- Active (12 months)
- Deprecated (6 months)
- Discontinued

The lifecycle stage determines if HPE issues updates and bug fixes for the EEP, which are reflected in changes to the major, minor, and maintenance versions of the EEP:

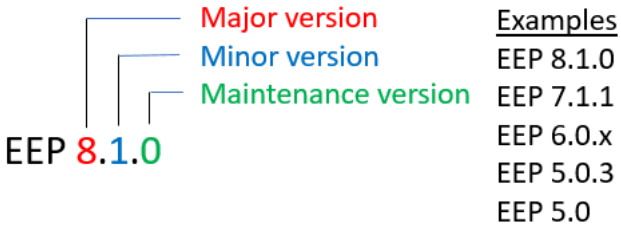
last-updated: Apr 11, 2022



To view the current lifecycle status for every EEP, see EEP Support and Lifecycle Status on page 5531.

EEP Versions

In MapR interfaces and documentation, MapR Ecosystem Pack (EEP) versions are expressed as a dot-separated string of numbers having two or three places. For example:



A Change to This EEP Version	Typically Includes	For Example
Major version	A new EEP series that supports a new major, minor, or maintenance version of MapR core.	EEP Support and Lifecycle Status on page 5531 shows how EEP major versions change when new versions of MapR core are introduced.
Minor version	A significant change in features or APIs but no change in support for the current MapR core version.	EEPs 6.0.1 and 6.1.0 both support MapR 6.1.0, but EEP 6.1.0 introduced new MapR Database features, such as the C# and Go OJAI clients, as well as new features for ecosystem components (Flume, Oozie, Tez, and Apache Kafka). EEP 6.2.0, which also supports MapR 6.1.0, introduced new ecosystem component versions (Hue, Impala, Spark) and updated some components (Oozie and Hive).
Maintenance version	Bug fixes only. EEP maintenance versions do not add new features or API changes and are backward compatible in the same EEP series.	EEPs 5.0.3, 5.0.2, and 5.0.1 added bug fixes to the EEP 5.0.x line while preserving compatibility with MapR 6.0.1.

last-updated: Apr 11, 2022

Notification of Changes in Support for Released EEPs

To notify users about changes in EEP support, HPE issues periodic support advisories. When EEPs are deprecated or discontinued, users of those EEPs are encouraged to upgrade to newer EEPs.

For service advisories, see:

- All Advisories
- Security Vulnerability Advisories
- Critical Issue or Bug Advisories
- Product Discontinuance (EOL) Advisories

Core Support and Lifecycle Status

This page shows the support and lifecycle status for all versions of MapR Data Platform core software.

Whenever possible, upgrade to the latest version of data-fabric core so that you can take advantage of new features, usability enhancements, and defect repair. If your installed version of core is "in maintenance," you have a limited amount of time to plan and execute a core upgrade.

For information about the core lifecycle, see Understand the Core Lifecycle on page 5526. For core compatibility with the leading Linux operating systems, see Operating System Support Matrix on page 5522.

Core Lifecycle and Maintenance Dates

Core Release	Release Date	Lifecycle Status	In Maintenance	End of Maintenance
7.0.0	March 7, 2022	Active	N/A	N/A
6.2.0	September 18, 2020	Active	January 2023	June 2024
6.1.1	March 29, 2021	Active	January 2023	June 2024
6.1.0	September 28, 2018	Active	January 2023	June 2024
6.0.1	April 6, 2018	In Maintenance	March 2022	September 2023
6.0.0	November 21, 2017	In Maintenance	March 2022	September 2023
5.2.2	August 2, 2017	End of Maintenance	December 13, 2017	April 30, 2019
5.2.1	April 6, 2017	End of Maintenance	N/A	April 30, 2019
5.2.0	August 19, 2016	End of Maintenance	N/A	April 30, 2019
5.1.0	February 29, 2016	End of Maintenance	N/A	April 30, 2019
5.0.0	July 20, 2015	End of Maintenance	N/A	April 30, 2019
4.1.0	April 30, 2015	End of Maintenance	N/A	January 20, 2017
4.0.2	January 30, 2015	End of Maintenance	N/A	January 20, 2017
4.0.1	September 16, 2014	End of Maintenance	N/A	January 20, 2017
4.0.0	June 23, 2014	End of Maintenance	N/A	January 20, 2017
3.1.1	June 13, 2014	End of Maintenance	N/A	February 29, 2016
3.1.0	March 11, 2014	End of Maintenance	N/A	February 29, 2016
3.0.3	May 5, 2014	End of Maintenance	N/A	February 29, 2016
3.0.2	October 28, 2013	End of Maintenance	N/A	February 29, 2016

last-updated: Apr 11, 2022

Core Release	Release Date	Lifecycle Status	In Maintenance	End of Maintenance
3.0.1	September 6, 2013	End of Maintenance	N/A	February 29, 2016
3.0.0	May 1, 2013	End of Maintenance	N/A	February 29, 2016

Related concepts

Understand MapR Versions on page 5519
Understanding the version numbers used by MapR core, MapR Ecosystem Packs (EEPs), EEP components, and MapR patches can help you keep your software up to date and plan for upgrades.

Checking the MapR Core Version on page 5415
Some maintenance operations require you to know the version of the currently installed MapR release (sometimes referred to as the "MapR core version"). You can check the MapR core version easily from within the Control System or MapR Installer user interface or identify the version from your installed packages.

Upgrading MapR Ecosystem Packs on page 335
Describes how to upgrade MapR Ecosystem Packs (EEPs), either as part of a MapR core upgrade or to take advantage of a new EEP for the current version of MapR core.

Related reference

Understand the Core Lifecycle on page 5526
Describes the MapR Data Platform core lifecycle and defines the lifecycle stages, which are Active, In Maintenance, and End of Maintenance.

Understand the EEP Lifecycle on page 5528
This page describes the EEP lifecycle and defines the lifecycle stages, which are Active, Deprecated, and Discontinued.

EEP Support and Lifecycle Status

This page shows the EEPs that are supported for different core releases and the current lifecycle status for each EEP.

Whenever possible, upgrade to the latest EEP supported by your core version so that you can take advantage of bug fixes and usability enhancements. If your installed EEP is deprecated or discontinued, see the corresponding upgrade recommendation and support advisory to understand your options.

To learn more about the EEP lifecycle, see Understand the EEP Lifecycle on page 5528. For core lifecycle information, see Core Support and Lifecycle Status on page 5530.

Core Release 7.0.0

EEP	EEP Lifecycle Status	EEP Is Deprecated	EEP Is Discontinued	EEP Upgrade Recommendation	Support Advisory
8.1.0	Active	March 6, 2023	September 30, 2023	N/A	N/A

Core Release 6.2.0

EEP	EEP Lifecycle Status	EEP Is Deprecated	EEP Is Discontinued	EEP Upgrade Recommendation	Support Advisory
8.1.0	Active	March 6, 2023	September 30, 2023	N/A	N/A
8.0.0	Replaced by EEP 8.1.0				
7.1.1	Active	October 31, 2022	April 30, 2023	N/A	
7.1.0	Active	June 30, 2022	December 31, 2022	N/A	
7.0.1	Deprecated	January 31, 2022	July 31, 2022	Upgrade to latest EEP 7.1.x	
7.0.0	Deprecated	September 18, 2021	March 31, 2022	Upgrade to latest EEP 7.1.x	

Core Release 6.1.1

EEP	EEP Lifecycle Status	EEP Is Deprecated	EEP Is Discontinued	EEP Upgrade Recommendation	Support Advisory
6.3.6*	Active	January 31, 2023	July 31, 2023	N/A	N/A
6.3.5*	Active	October 31, 2022	April 30, 2023	N/A	
6.3.4*	Active	June 30, 2022	December 31, 2022	N/A	
6.3.3	Active	March 31, 2022	September 30, 2022	N/A	

Core Release 6.1.0

EEP	EEP Lifecycle Status	EEP Is Deprecated	EEP Is Discontinued	EEP Upgrade Recommendation	Support Advisory
6.3.6*	Active	January 31, 2023	July 31, 2023	N/A	Advisory Issued
6.3.5*	Active	October 31, 2022	April 30, 2023	N/A	
6.3.4*	Active	June 30, 2022	December 31, 2022	N/A	
6.3.3*	Active	March 31, 2022	September 30, 2022	N/A	
6.3.2	Deprecated	January 31, 2022	July 31, 2022	Upgrade to latest EEP 6.3.x	
6.3.1	Deprecated	September 18, 2021	March 31, 2022	Upgrade to latest EEP 6.3.x	
6.3.0	Discontinued	December 16, 2020	December 31, 2021	Upgrade to latest EEP 6.3.x	
6.2.0	Discontinued	December 31, 2019	December 31, 2021	Upgrade to latest EEP 6.3.x	
6.1.1	Discontinued	December 31, 2019	December 31, 2021	Upgrade to latest EEP 6.3.x	
6.1.0	Discontinued	December 31, 2019	December 31, 2021	Upgrade to latest EEP 6.3.x	
6.0.2	Discontinued	December 31, 2019	December 31, 2021	Upgrade to latest EEP 6.3.x	
6.0.1	Discontinued	December 31, 2019	December 31, 2021	Upgrade to latest EEP 6.3.x	
6.0.0	Discontinued	December 31, 2019	December 31, 2021	Upgrade to latest EEP 6.3.x	

Core Release 6.0.1

EEP	EEP Lifecycle Status	EEP Is Deprecated	EEP Is Discontinued	EEP Upgrade Recommendation	Support Advisory
5.0.7	Active	June 30, 2022	June 30, 2023**	N/A	Advisory Issued
5.0.6	Discontinued	April 30, 2021	December 31, 2021	Upgrade to latest EEP 5.0.x	
5.0.5	Discontinued	April 30, 2021	December 31, 2021	Upgrade to latest EEP 5.0.x	
5.0.4	Discontinued	April 30, 2021	December 31, 2021	Upgrade to latest EEP 5.0.x	
5.0.3	Discontinued	December 31, 2020	December 31, 2021	Upgrade to latest EEP 5.0.x	
5.0.2	Discontinued	December 31, 2020	December 31, 2021	Upgrade to latest EEP 5.0.x	
5.0.1	Discontinued	December 31, 2020	December 31, 2021	Upgrade to latest EEP 5.0.x	
5.0.0	Discontinued	December 31, 2020	December 31, 2021	Upgrade to latest EEP 5.0.x	

last-updated: Apr 11, 2022

Core Release 6.0.0

EEP	EEP Lifecycle Status	EEP Is Deprecated	EEP Is Discontinued	EEP Upgrade Recommendation	Support Advisory
4.1.4	Discontinued	October 31, 2018	April 30, 2019	Upgrade to latest EEP 5.0.x	Advisory Issued
4.1.3	Discontinued	October 31, 2018	April 30, 2019	Upgrade to latest EEP 5.0.x	
4.1.2	Discontinued	October 31, 2018	April 30, 2019	Upgrade to latest EEP 5.0.x	
4.1.1	Discontinued	October 31, 2018	April 30, 2019	Upgrade to latest EEP 5.0.x	
4.1.0	Discontinued	October 31, 2018	April 30, 2019	Upgrade to latest EEP 5.0.x	
4.0.0	Discontinued	October 31, 2018	April 30, 2019	Upgrade to latest EEP 5.0.x	

last-updated: Apr 11, 2022

Core Release 5.2.x

EEP	EEP Lifecycle Status	EEP Is Deprecated	EEP Is Discontinued	EEP Upgrade Recommendation	Support Advisory
3.0.5	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	Prepare for MapR 5.x End of Maintenance by April 2019
3.0.4	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	
3.0.3	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	
3.0.2	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	
3.0.1	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	
3.0	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	
2.0.3	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	
2.0.2	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	
2.0.1	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	
2.0	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	
1.1.4	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	
1.1.3	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	
1.1.2	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	
1.1.1	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	
1.1.0	Discontinued	October 31, 2018	April 30, 2019	Upgrade to EEP 6.3.x or later (requires a core upgrade)	

*Before using this EEP with the specified core version, you must apply the latest core patch. See Downloading a Patch on page 437.

last-updated: Apr 11, 2022

**EEP 5.0.7 is supported on core 6.0.1 until core 6.0.1 is discontinued or a newer EEP 5.0.x (with x > 7) becomes available.

Related concepts

Understand MapR Versions on page 5519
Understanding the version numbers used by MapR core, MapR Ecosystem Packs (EEPs), EEP components, and MapR patches can help you keep your software up to date and plan for upgrades.

Checking the EEP Version on page 5413
Some MapR Installer operations require you to know the version of the currently installed MapR Ecosystem Pack (EEP). You can check the EEP version easily from within the MapR Installer user interface or derive the EEP version from your repository information.

Upgrading MapR Ecosystem Packs on page 335
Describes how to upgrade MapR Ecosystem Packs (EEPs), either as part of a MapR core upgrade or to take advantage of a new EEP for the current version of MapR core.

MapR Ecosystem Pack (EEP) Reference on page 6504
This section contains links to information that is specific to a given EEP.

Related reference

Understand the EEP Lifecycle on page 5528
This page describes the EEP lifecycle and defines the lifecycle stages, which are Active, Deprecated, and Discontinued.

EEP Components and OS Support

MapR Ecosystem Packs consist of ecosystem components and MapR-monitoring components that can run on a variety of operating systems.

MapR 6.0 and later support multiple MapR Ecosystem Packs (EEPs). For information about the EEPs supported by various core versions, see EEP Support and Lifecycle Status on page 5531. For more information about the components in each EEP, see the EEP Release Notes on page 5658.

EEP 8.1.0 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 8.1.0 and shows the operating system support for each component.

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SLES		RHEL / CentOS				Ubuntu	
	15 SP2/SP3	12 SPx	8.4	8.3	8.2	8.1	18.04/20.04	16.04
Airflow 2.2.1.0[1]	Yes	No	Yes	Yes	Yes	Yes	Yes	No
AsyncHBase 1.8.2.0	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Data Access Gateway 4.0.0.0	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Drill 1.16.1.400	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Hadoop 2.7.6.200	Yes	No	Yes	Yes	Yes	Yes	Yes	No
HBase 1.4.13.200	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Hive 2.3.9	Yes	No	Yes	Yes	Yes	Yes	Yes	No
HttpFS 1.1.0.200	Yes	No	Yes	Yes	Yes	Yes	Yes	No

Hue 4.6.0.300[2]	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Kafka 2.6.1.100	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Kafka Connect HDFS 10.0.0.100	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Kafka Connect JDBC 10.0.1.100	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Kafka REST 6.0.0.100	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Kafka Schema Registry 6.0.0.100	Yes	No	Yes	Yes	Yes	Yes	Yes	No
KSQL 6.0.0.100	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Livy 0.7.0.100	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Oozie 5.2.1.200	Yes	No	Yes	Yes	Yes	Yes	Yes	No
S3 Gateway 2.2.0.0	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Spark 3.2.0.0	Yes	No	Yes	Yes	Yes	Yes	Yes	No
MapR Monitoring Components[3]								
Collectd 5.12.0.400	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Elasticsearch 6.8.8.500	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Fluentd 1.10.3.400	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Grafana 7.5.10.400	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Kibana 6.8.8.500	Yes	No	Yes	Yes	Yes	Yes	Yes	No
OpenTSDB 2.4.1.400	Yes	No	Yes	Yes	Yes	Yes	Yes	No

[1]Airflow is not supported on SLES 15 SP2.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 8.0.0 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 8.0.0 and shows the operating system support for each component.

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	**SLES**		**RHEL / CentOS**				**Ubuntu**	
	15 SP2	**12 SPx**	**8.4**	**8.3**	**8.2**	**8.1**	**18.04**	**16.04**
AsyncHBase 1.8.2.0	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Data Access Gateway 3.0.0.0	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Drill 1.16.1.300	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Flume 1.9.0.200	Yes	No	Yes	Yes	Yes	Yes	Yes	No

last-updated: Apr 11, 2022

Hadoop 2.7.6.100	Yes	No	Yes	Yes	Yes	Yes	Yes	No
HBase 1.4.13.100	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Hive 2.3.9	Yes	No	Yes	Yes	Yes	Yes	Yes	No
HttpFS 1.1.0.100	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Hue 4.6.0.200[1]	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Kafka 2.6.1.0	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Kafka Connect HDFS 10.0.0.0	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Kafka Connect JDBC 10.0.1.0	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Kafka REST 6.0.0.0	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Kafka Schema Registry 6.0.0.0	Yes	No	Yes	Yes	Yes	Yes	Yes	No
KSQL 6.0.0.0	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Livy 0.7.0.100	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Oozie 5.2.1.100	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Pig 0.17.0.100	Yes	No	Yes	Yes	Yes	Yes	Yes	No
S3 Gateway 2.2.0.0	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Spark 3.1.2.0	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Sqoop 1.4.7	Yes	No	Yes	Yes	Yes	Yes	Yes	No
MapR Monitoring Components[2]								
Collectd 5.12.0.300	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Elasticsearch 6.8.8.400	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Fluentd 1.10.3.300	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Grafana 7.5.10.300	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Kibana 6.8.8.400	Yes	No	Yes	Yes	Yes	Yes	Yes	No
OpenTSDB 2.4.1.300	Yes	No	Yes	Yes	Yes	Yes	Yes	No

[1]The Spark Notebook UI in Hue is a beta feature.

[2]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 7.1.1 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 7.1.1 and shows the operating system support for each component.

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

last-updated: Apr 11, 2022

Ecosystem Components	SLES		RHEL / CentOS	Ubuntu		OEL
	15 SP2	12 SPx	8.4, 8.3, 8.2, and 8.1	18.04	16.04	8.2
AsyncHBase 1.8.2.0	Yes	No	Yes	Yes	Yes	Yes
Data Access Gateway 3.0.0.0	Yes	No	Yes	Yes	Yes	Yes
Drill 1.16.1.200	Yes	No	Yes	Yes	Yes	Yes
Flume 1.9.0.100	Yes	No	Yes	Yes	Yes	Yes
HBase 1.4.13.0	Yes	No	Yes	Yes	Yes	Yes
Hadoop 2.7.5.0	Yes	No	Yes	Yes	Yes	Yes
Hive 2.3.8	Yes	No	Yes	Yes	Yes	Yes
HttpFS 1.1.0.0	Yes	No	Yes	Yes	Yes	Yes
Hue 4.6.0.0[1]	Yes	No	Yes	Yes	Yes	Yes
Impala 2.12.0.500	Yes	No	Yes	No	No	No
KSQL 5.1.2.200	Yes	No	Yes	Yes	Yes	Yes
Livy 0.7.0.0	Yes	No	Yes	Yes	Yes	Yes
MapR Object Store 2.1.0.0	Yes	No	Yes	Yes	Yes	Yes
Oozie 5.2.1.0	Yes	No	Yes	Yes	Yes	Yes
Pig 0.17.0.0	Yes	No	Yes	Yes	Yes	Yes
Sentry 1.7.0[2]	Yes	No	Yes	Yes	Yes	Yes
Spark 2.4.7.100	Yes	No	Yes	Yes	Yes	Yes
Sqoop 1.4.7	Yes	No	Yes	Yes	Yes	Yes
MapR Monitoring Components[3]						
Collectd 5.10.0.0	Yes	No	Yes	Yes	Yes	Yes
Elasticsearch 6.8.8.300	Yes	No	Yes	Yes	Yes	Yes
Fluentd 1.10.3.200	Yes	No	Yes	Yes	Yes	Yes
Grafana 7.5.2.200	Yes	No	Yes	Yes	Yes	Yes
Kibana 6.8.8.300	Yes	No	Yes	Yes	Yes	Yes
OpenTSDB 2.4.0	Yes	No	Yes	Yes	Yes	Yes

[1]The Spark Notebook UI in Hue is a beta feature.

[2]MapR support for Sentry is limited to Impala users.

[3]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 7.1.0 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 7.1.0 and shows the operating system support for each component.

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

last-updated: Apr 11, 2022

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SLES		RHEL / CentOS	Ubuntu		OEL
	15 SP2	12 SPx	8.4, 8.3, 8.2, and 8.1	18.04	16.04	8.2
AsyncHBase 1.8.2.0	Yes	No	Yes	Yes	Yes	Yes
Data Access Gateway 3.0.0.0	Yes	No	Yes	Yes	Yes	Yes
Drill 1.16.1.200	Yes	No	Yes	Yes	Yes	Yes
Flume 1.9.0.100	Yes	No	Yes	Yes	Yes	Yes
HBase 1.4.13.0	Yes	No	Yes	Yes	Yes	Yes
Hadoop 2.7.5.0	Yes	No	Yes	Yes	Yes	Yes
Hive 2.3.8	Yes	No	Yes	Yes	Yes	Yes
HttpFS 1.1.0.0	Yes	No	Yes	Yes	Yes	Yes
Hue 4.6.0.0[1]	Yes	No	Yes	Yes	Yes	Yes
Impala 2.12.0.500	Yes	No	Yes	No	No	No
KSQL 5.1.2.200	Yes	No	Yes	Yes	Yes	Yes
Livy 0.7.0.0	Yes	No	Yes	Yes	Yes	Yes
MapR Object Store 2.1.0.0	Yes	No	Yes	Yes	Yes	Yes
Oozie 5.2.1.0	Yes	No	Yes	Yes	Yes	Yes
Pig 0.17.0.0	Yes	No	Yes	Yes	Yes	Yes
Sentry 1.7.0[2]	Yes	No	Yes	Yes	Yes	Yes
Spark 2.4.7.100	Yes	No	Yes	Yes	Yes	Yes
Sqoop 1.4.7	Yes	No	Yes	Yes	Yes	Yes
MapR Monitoring Components[3]						
Collectd 5.10.0.0	Yes	No	Yes	Yes	Yes	Yes
Elasticsearch 6.8.8.300	Yes	No	Yes	Yes	Yes	Yes
Fluentd 1.10.3.0	Yes	No	Yes	Yes	Yes	Yes
Grafana 7.5.2.200	Yes	No	Yes	Yes	Yes	Yes
Kibana 6.8.8.300	Yes	No	Yes	Yes	Yes	Yes
OpenTSDB 2.4.0	Yes	No	Yes	Yes	Yes	Yes

[1]The Spark Notebook UI in Hue is a beta feature.

[2]MapR support for Sentry is limited to Impala users.

[3]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 7.0.1 Components and OS Support
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 7.0.1 and shows the operating system support for each component.

last-updated: Apr 11, 2022

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SLES	RHEL / CentOS	Ubuntu	OEL
AsyncHBase 1.8.2.0	No	Yes	Yes	Yes
Drill 1.16.1.100	No	Yes	Yes	Yes
Flume 1.9.0.100	No	Yes	Yes	Yes
Hadoop 2.7.4.100	No	Yes	Yes	Yes
HBase 1.4.12.100	No	Yes	Yes	Yes
Hive 2.3.7	No	Yes	Yes	Yes
HttpFS 1.0	No	Yes	Yes	Yes
Hue 4.6.0.0[1]	No	Yes	Yes	Yes
Impala 2.12.0.400	No	Yes	Yes	Yes
KSQL 5.1.2.100	No	Yes	Yes	Yes
MapR Data Access Gateway 3.0.0.0	No	Yes	Yes	Yes
MapR Object Store 2.0.0.0	No	Yes	Yes	Yes
Oozie 5.2.0.100	No	Yes	Yes	Yes
Pig 0.17.0.0	No	Yes	Yes	Yes
Sentry 1.7.0[2]	No	Yes	Yes	Yes
Spark 2.4.7.0	No	Yes	Yes	Yes
Sqoop 1.4.7	No	Yes	Yes	Yes
MapR Monitoring Components[3]				
Collectd 5.10.0.0	No	Yes	Yes	Yes
Elasticsearch 6.8.8.0	No	Yes	Yes	Yes
Fluentd 1.10.3.0	No	Yes	Yes	Yes
Grafana 6.7.4.0	No	Yes	Yes	Yes
Kibana 6.8.8.0	No	Yes	Yes	Yes
OpenTSDB 2.4.0	No	Yes	Yes	Yes

[1]The Spark Notebook UI in Hue is a beta feature.

[2]MapR support for Sentry is limited to Impala users.

[3]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 7.0.0 Components and OS Support
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 7.0.0 and shows the operating system support for each component.

last-updated: Apr 11, 2022

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SLES	RHEL / CentOS	Ubuntu	OEL
AsyncHBase 1.8.2.0	No	Yes	Yes	Yes
Drill 1.16.1.0	No	Yes	Yes	Yes
Flume 1.9.0.0	No	Yes	Yes	Yes
HBase 1.4.12.0	No	Yes	Yes	Yes
Hive 2.3.7	No	Yes	Yes	Yes
HttpFS 1.0	No	Yes	Yes	Yes
Hue 4.6.0.0[1]	No	Yes	Yes	Yes
Impala 2.12.0.200	No	Yes	No	No
KSQL 5.1.2.0	No	Yes	Yes	Yes
MapR Data Access Gateway 3.0.0.0	No	Yes	Yes	Yes
MapR Object Store 2.0.0.0	No	Yes	Yes	Yes
Oozie 5.2.0.0	No	Yes	Yes	Yes
Pig 0.17.0.0	No	Yes	Yes	Yes
Sentry 1.7.0[2]	No	Yes	Yes	Yes
Spark 2.4.5.0	No	Yes	Yes	Yes
Sqoop 1.4.7	No	Yes	Yes	Yes
YARN 2.7.4.0	No	Yes	Yes	Yes
MapR Monitoring Components[3]				
Collectd 5.10.0.0	No	Yes	Yes	Yes
Elasticsearch 6.8.8.0	No	Yes	Yes	Yes
Fluentd 1.10.3.0	No	Yes	Yes	Yes
Grafana 6.7.2.0	No	Yes	Yes	Yes
Kibana 6.8.8.0	No	Yes	Yes	Yes
OpenTSDB 2.4.0.x	No	Yes	Yes	Yes

[1]The Spark Notebook UI in Hue is a beta feature.

[2]MapR support for Sentry is limited to Impala users.

[3]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 6.3.6 Components and OS Support
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.3.6 and shows the operating system support for each component.

last-updated: Apr 11, 2022

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS						Ubuntu	
	12 SP5	8.4	8.3	8.2	8.1	7.9	7.8	18.04	16.04
AsyncHBase 1.7.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Data Access Gateway 2.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Drill 1.16.0.400	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Flume 1.8.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
HBase 1.1.13.500	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Hive 2.3.6	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
HttpFS 1.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Hue 4.3.0.500[1]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Impala 2.12.0.650	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
Kafka Streams 1.1.1	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Kafka Connect HDFS 4.1.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Kafka Connect JDBC 4.1.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Kafka REST 4.1.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
KSQL 4.1.1	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Livy 0.5.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Oozie 5.1.0.800	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Pig 0.16	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
S3 Gateway 1.0.1	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Sentry 1.7.0[2]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Spark 2.4.4.500	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Sqoop 1.4.7	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
MapR Monitoring Components[3]									
Collectd 5.8.1.210	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Elasticsearch 6.8.8.110	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Fluentd 1.10.3.110	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Grafana 7.5.2.110	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Kibana 6.8.8.110	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
OpenTSDB 2.4.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

[1]The Spark Notebook UI in Hue is a beta feature.

last-updated: Apr 11, 2022

[2]MapR support for Sentry is limited to Impala users.

[3]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 6.3.5 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.3.5 and shows the operating system support for each component.

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS						Ubuntu	
	12 SP5	8.4	8.3	8.2	8.1	7.9	7.8	18.04	16.04
AsyncHBase 1.7.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Data Access Gateway 2.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Drill 1.16.0.300	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Flume 1.8.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
HBase 1.1.13.400	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Hive 2.3.6	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
HttpFS 1.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Hue 4.3.0.400[1]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Impala 2.12.0.600	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
Kafka 1.1.1	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Kafka Connect HDFS 4.1.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Kafka Connect JDBC 4.1.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Kafka REST 4.1.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
KSQL 4.1.1	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Livy 0.5.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Oozie 5.1.0.700	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Pig 0.16	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
S3 Gateway 1.0.1	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Sentry 1.7.0[2]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Spark 2.4.4.400	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Sqoop 1.4.7	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Sqoop2 2.0.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
MapR Monitoring Components[3]									
Collectd 5.8.1.201	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Elasticsearch 6.8.8.100	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Fluentd 1.10.3.100	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Grafana 7.5.2.100	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Kibana 6.8.8.100	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
OpenTSDB 2.4.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

[1]The Spark Notebook UI in Hue is a beta feature.

[2]MapR support for Sentry is limited to Impala users.

[3]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 6.3.4 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.3.4 and shows the operating system support for each component.

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7.0	Y	Y	Y
Data Access Gateway 2.0	Y	Y	Y
Drill 1.16.0.200	Y	Y	Y
Flume 1.8.0	Y	Y	Y
HBase 1.1.13.300	Y	Y	Y
Hive 2.3.6	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 4.3.0.400[1]	Y	Y	Y
Impala 2.12.0.300	Y (12 SP1)	Y	N
KSQL 4.1.1	Y	Y	Y
Livy 0.5	Y	Y	Y
MapR Object Store 1.0.1	Y	Y	Y
Myriad 0.2	Y	Y	Y
Oozie 5.1.0.600	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[2]	Y	Y	Y
Spark 2.4.4.300	Y	Y	Y
Sqoop 1.4.7	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y

last-updated: Apr 11, 2022

MapR Monitoring Components[3]			
Collectd 5.8.1.201	Y	Y	Y
Elasticsearch 6.8.8.100	Y	Y	Y
Fluentd 1.10.3.100	Y	Y	Y
Grafana 7.5.2.100	Y	Y	Y
Kibana 6.8.8.100	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]The Spark Notebook UI in Hue is a beta feature.

[2]MapR support for Sentry is limited to Impala users.

[3]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 6.3.3 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.3.3 and shows the operating system support for each component.

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7.0	Y	Y	Y
Drill 1.16.0.100	Y	Y	Y
Flume 1.8.0	Y	Y	Y
HBase 1.1.13.200	Y	Y	Y
Hive 2.3.6	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 4.3.0.300[1]	Y	Y	Y
Impala 2.12.0.300	Y (12 SP1)	Y	N
KSQL 4.1.1	Y	Y	Y
MapR Data Access Gateway 2.0	Y	Y	Y
MapR Object Store 1.0.1	Y	Y	Y
Myriad 0.2	Y	Y	Y
Oozie 5.1.0.500	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[2]	Y	Y	Y
Spark 2.4.4.200	Y	Y	Y
Sqoop 1.4.7	Y	Y	Y

last-updated: Apr 11, 2022

Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[3]			
Collectd 5.8.1.201	Y	Y	Y
Elasticsearch 6.8.8.100	Y	Y	Y
Fluentd 1.10.3.100	Y	Y	Y
Grafana 6.7.4.100	Y	Y	Y
Kibana 6.8.8.100	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]The Spark Notebook UI in Hue is a beta feature.

[2]MapR support for Sentry is limited to Impala users.

[3]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 6.3.2 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.3.2 and shows the operating system support for each component.

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7.0	Y	Y	Y
Drill 1.16.0.100	Y	Y	Y
Flume 1.8.0	Y	Y	Y
HBase 1.1.13.200	Y	Y	Y
Hive 2.3.6	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 4.3.0.300[1]	Y	Y	Y
Impala 2.12.0.300	Y (12 SP1)	Y	N
KSQL 4.1.1	Y	Y	Y
MapR Data Access Gateway 2.0	Y	Y	Y
MapR Object Store 1.0.1	Y	Y	Y
Myriad 0.2	Y	Y	Y
Oozie 5.1.0.500	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[2]	Y	Y	Y
Spark 2.4.4.200	Y	Y	Y

last-updated: Apr 11, 2022

Sqoop 1.4.7	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[3]			
Collectd 5.8.1.201	Y	Y	Y
Elasticsearch 6.8.8.100	Y	Y	Y
Fluentd 1.10.3.100	Y	Y	Y
Grafana 6.7.4.100	Y	Y	Y
Kibana 6.8.8.100	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]The Spark Notebook UI in Hue is a beta feature.

[2]MapR support for Sentry is limited to Impala users.

[3]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 6.3.1 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.3.1 and shows the operating system support for each component.

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7.0	Y	Y	Y
Drill 1.16.0.x	Y	Y	Y
Flume 1.8.0	Y	Y	Y
HBase 1.1.13.x	Y	Y	Y
Hive 2.3.6	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 4.3.0.200[1]	Y	Y	Y
Impala 2.12.0.100	Y (12 SP1)	Y	N
KSQL 4.1.1	Y	Y	Y
MapR Data Access Gateway 2.0	Y	Y	Y
MapR Object Store 1.0.1	Y	Y	Y
Myriad 0.2	Y	Y	Y
Oozie 5.1.0.400	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[2]	Y	Y	Y

last-updated: Apr 11, 2022

Spark 2.4.4.100	Y	Y	Y
Sqoop 1.4.7	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[3]			
Collectd 5.8.1.300	Y	Y	Y
Elasticsearch 6.8.8.100	Y	Y	Y
Fluentd 1.10.3.100	Y	Y	Y
Grafana 6.0.2.200	Y	Y	Y
Kibana 6.8.8.200	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]The Spark Notebook UI in Hue is a beta feature.

[2]MapR support for Sentry is limited to Impala users.

[3]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 6.3.0 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.3.0 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.16.0.10	Y	Y	Y
Flume 1.8	Y	Y	Y
HBase 1.1.13.0	Y	Y	Y
Hive 2.3.6	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 4.3.0.100[1]	Y	Y	Y
Impala 2.12.0.100	Y (12 SP1)	Y	N
KSQL 4.1.1	Y	Y	Y
MapR Data Access Gateway 2.0	Y	Y	Y
MapR Object Store 1.0.1	Y	Y	Y
Myriad 0.2	Y	Y	Y
Oozie 5.1.0.300	Y	Y	Y
Pig 0.16	Y	Y	Y

last-updated: Apr 11, 2022

Sentry 1.7.0[2]	Y	Y	Y
Spark 2.4.4.0	Y	Y	Y
Sqoop 1.4.7	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[3]			
Collectd 5.8.1.200	Y	Y	Y
Elasticsearch 6.5.3.200	Y	Y	Y
Fluentd 1.4.0.100	Y	Y	Y
Grafana 6.0.2.100	Y	Y	Y
Kibana 6.5.3.200	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]The Spark Notebook UI in Hue is a beta feature.

[2]MapR support for Sentry is limited to Impala users.

[3]MapR Monitoring Components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 6.2.0 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.2.0 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.16.0.0	Y	Y	Y
Flume 1.8	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.3.3	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 4.3.0.0[2]	Y	Y	Y
Impala 2.12.0.0	Y (12 SP1)	Y	N
KSQL 4.1.1	Y	Y	Y
MapR Data Access Gateway 2.0	Y	Y	Y
MapR Object Store 1.0.1	Y	Y	Y
Myriad 0.2	Y	Y	Y
Oozie 5.1.0.200	Y	Y	Y

last-updated: Apr 11, 2022

Pig 0.16	Y	Y	Y
Sentry 1.7.0[3]	Y	Y	Y
Spark 2.4.0.0	Y	Y	Y
Sqoop 1.4.7	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[4]			
Collectd 5.8.1.100	Y	Y	Y
Elasticsearch 6.5.3.100	Y	Y	Y
Fluentd 1.4.0.0	Y	Y	Y
Grafana 6.0.2.0	Y	Y	Y
Kibana 6.5.3.100	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

[4]MapR Monitoring Components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 6.1.1 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.1.1 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.15.0.7	Y	Y	Y
Flume 1.8	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.3.3	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 4.2[2]	Y	Y	Y
Impala 2.10	Y (12 SP1)	Y	N
KSQL 4.1.1	Y	Y	Y
MapR Data Access Gateway 2.0	Y	Y	Y
MapR Object Store 1.0.1	Y	Y	Y

last-updated: Apr 11, 2022

Myriad 0.2	Y	Y	Y
Oozie 5.1.0.100	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[3]	Y	Y	Y
Spark 2.3.3.100	Y	Y	Y
Sqoop 1.4.7	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[4]			
Collectd 5.8.1.1	Y	Y	Y
Elasticsearch 6.5.3.1	Y	Y	Y
Fluentd 1.3.2.1	Y	Y	Y
Grafana 5.4.2.1	Y	Y	Y
Kibana 6.5.3.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

[4]Supported for monitoring use cases only.

EEP 6.1.0 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.1.0 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.15.0.0	Y	Y	Y
Flume 1.8	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.3.3[2]	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 4.2[3]	Y	Y	Y
Impala 2.10	Y (12 SP1)	Y	N
KSQL 4.1.1	Y	Y	Y
MapR Data Access Gateway 2.0	Y	Y	Y

last-updated: Apr 11, 2022

MapR Object Store 1.0.1	Y	Y	Y
Myriad 0.2	Y	Y	Y
Oozie 5.1.0.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.3.2.0	Y	Y	Y
Sqoop 1.4.7	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.8.1.0	Y	Y	Y
Elasticsearch 6.5.3.0	Y	Y	Y
Fluentd 1.3.2.0	Y	Y	Y
Grafana 5.4.2.0	Y	Y	Y
Kibana 6.5.3.0	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Equivalent to Apache Hive 2.3.4.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]Supported for monitoring use cases only.

EEP 6.0.2 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.0.2 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.14	Y	Y	Y
Flume 1.8	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.3.3	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 4.2[2]	Y	Y	Y
Impala 2.10	Y (12 SP1)	Y	N
KSQL 4.1.1	Y	Y	Y

last-updated: Apr 11, 2022

MapR Data Access Gateway 2.0	Y	Y	Y
MapR Object Store 1.0.1	Y	Y	Y
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[3]	Y	Y	Y
Spark 2.3.3.0	Y	Y	Y
Sqoop 1.4.7	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[4]			
Collectd 5.8.0	Y	Y	Y
Elasticsearch 6.2.3	Y	Y	Y
Fluentd 1.1.2	Y	Y	Y
Grafana 4.6.5	Y	Y	Y
Kibana 6.2.3	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

[4]Supported for monitoring use cases only.

EEP 6.0.1 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.0.1 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.14	Y	Y	Y
Flume 1.8	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.3[2]	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 4.2[3]	Y	Y	Y
Impala 2.10	Y (12 SP1)	Y	N

last-updated: Apr 11, 2022

KSQL 4.1.1	Y	Y	Y
MapR Data Access Gateway 2.0	Y	Y	Y
MapR Object Store 1.0.0	Y	Y	Y
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.3.2.0	Y	Y	Y
Sqoop 1.4.7	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.8.0	Y	Y	Y
Elasticsearch 6.2.3	Y	Y	Y
Fluentd 1.1.2	Y	Y	Y
Grafana 4.6.1	Y	Y	Y
Kibana 6.2.3	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Equivalent to Apache Hive 2.3.3.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]Supported for monitoring use cases only.

EEP 6.0.0 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.0.0 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.14	Y	Y	Y
Flume 1.8	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.3[2]	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 4.2[3]	Y	Y	Y

last-updated: Apr 11, 2022

Impala 2.10	Y (12 SP1)	Y	N
KSQL 4.1.1	Y	Y	Y
MapR Data Access Gateway 2.0	Y	Y	Y
MapR Object Store 1.0.0	Y	Y	Y
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.3.1	Y	Y	Y
Sqoop 1.4.7	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.8.0	Y	Y	Y
Elasticsearch 6.2.3	Y	Y	Y
Fluentd 1.1.2	Y	Y	Y
Grafana 4.6.1	Y	Y	Y
Kibana 6.2.3	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Equivalent to Apache Hive 2.3.3.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]Supported for monitoring use cases only.

EEP 5.0.7 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.7 and shows the operating system support for each component.

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7.0[1]	Y	Y	Y
Drill 1.13	Y[2]	Y	Y
Flume 1.8.0	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.1.1	Y	Y	Y

last-updated: Apr 11, 2022

HttpFS 1.0	Y	Y	Y
Hue 3.12[3]	Y	Y	Y
Impala 2.10.0	Y (12 SP1)	Y	N
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.7.2	Y	Y	Y
Elasticsearch 5.4.1	Y	Y	Y
Fluentd 0.14.20	Y	Y	Y
Grafana 7.5.2.0	Y	Y	Y
Kibana 5.4.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Supported with Open JDK 1.8 and Oracle JDK 1.8.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 5.0.6 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.6 and shows the operating system support for each component.

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7.0[1]	Y	Y	Y
Drill 1.13	Y[2]	Y	Y
Flume 1.8.0	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y

last-updated: Apr 11, 2022

Hue 3.12[3]	Y	Y	Y
Impala 2.10.0	Y (12 SP1)	Y	N
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.7.2	Y	Y	Y
Elasticsearch 5.4.1	Y	Y	Y
Fluentd 0.14.20	Y	Y	Y
Grafana 4.6.5	Y	Y	Y
Kibana 5.4.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Supported with Open JDK 1.8 and Oracle JDK 1.8.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 5.0.5 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.5 and shows the operating system support for each component.

To understand which MapR core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7.0[1]	Y	Y	Y
Drill 1.13	Y[2]	Y	Y
Flume 1.8.0	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[3]	Y	Y	Y

Impala 2.10.0	Y (12 SP1)	Y	N
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.7.2	Y	Y	Y
Elasticsearch 5.4.1	Y	Y	Y
Fluentd 0.14.20	Y	Y	Y
Grafana 4.6.5	Y	Y	Y
Kibana 5.4.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Supported with Open JDK 1.8 and Oracle JDK 1.8.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]MapR-monitoring components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 5.0.4 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.4 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.13	Y[2]	Y	Y
Flume 1.8	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[3]	Y	Y	Y

last-updated: Apr 11, 2022

Impala 2.10	Y (12 SP1)	Y	N
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.7.2	Y	Y	Y
Elasticsearch 5.4.1	Y	Y	Y
Fluentd 0.14.20	Y	Y	Y
Grafana 4.6.5	Y	Y	Y
Kibana 5.4.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Supported with Open JDK 1.8 and Oracle JDK 1.8.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]MapR Monitoring Components are not supported as standalone products. They are only supported for MapR monitoring use cases.

EEP 5.0.3 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.3 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.13	Y[2]	Y	Y
Flume 1.8	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[3]	Y	Y	Y

last-updated: Apr 11, 2022

Impala 2.10	Y (12 SP1)	Y	N
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.7.2	Y	Y	Y
Elasticsearch 5.4.1	Y	Y	Y
Fluentd 0.14.20	Y	Y	Y
Grafana 4.6.5	Y	Y	Y
Kibana 5.4.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Supported with Open JDK 1.8 and Oracle JDK 1.8.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]Supported for monitoring use cases only.

EEP 5.0.2 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.2 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.13	Y[2]	Y	Y
Flume 1.8	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[3]	Y	Y	Y
Impala 2.10	Y (12 SP1)	Y	N

last-updated: Apr 11, 2022

Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.7.2	Y	Y	Y
Elasticsearch 5.4.1	Y	Y	Y
Fluentd 0.14.20	Y	Y	Y
Grafana 4.6.1	Y	Y	Y
Kibana 5.4.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Supported with Open JDK 1.8 and Oracle JDK 1.8.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]Supported for monitoring use cases only.

EEP 5.0.1 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.1 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.13	Y[2]	Y	Y
Flume 1.8	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[3]	Y	Y	Y
Impala 2.10	Y (12 SP1)	Y	N
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y

Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.7.2	Y	Y	Y
Elasticsearch 5.4.1	Y	Y	Y
Fluentd 0.14.20	Y	Y	Y
Grafana 4.6.1	Y	Y	Y
Kibana 5.4.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Supported with Open JDK 1.8 and Oracle JDK 1.8.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]Supported for monitoring use cases only.

EEP 5.0.0 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.0 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.13	Y[2]	Y	Y
Flume 1.8	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[3]	Y	Y	Y
Impala 2.10	Y (12 SP1)	Y	N
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y

last-updated: Apr 11, 2022

Spark 2.2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.7.2	Y	Y	Y
Elasticsearch 5.4.1	Y	Y	Y
Fluentd 0.14.20	Y	Y	Y
Grafana 4.6.1	Y	Y	Y
Kibana 5.4.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Supported with Open JDK 1.8 and Oracle JDK 1.8.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]Supported for monitoring use cases only.

EEP 4.1.4 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 4.1.4 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.12	Y[2]	Y	Y
Flume 1.7	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[3]	Y	Y	Y
Impala 2.7.0	Y (12 SP1)	Y	N
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.1.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y

last-updated: Apr 11, 2022

Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.7.2	Y	Y	Y
Elasticsearch 5.4.1	Y	Y	Y
Fluentd 0.14.20	Y	Y	Y
Grafana 4.6.5	Y	Y	Y
Kibana 5.4.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Supported with Open JDK 1.8 and Oracle JDK 1.8.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]Supported for monitoring use cases only.

EEP 4.1.3 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 4.1.3 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.12	Y[2]	Y	Y
Flume 1.7	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[3]	Y	Y	Y
Impala 2.7.0	Y (12 SP1)	Y	N
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			

last-updated: Apr 11, 2022

Collectd 5.7.2	Y	Y	Y
Elasticsearch 5.4.1	Y	Y	Y
Fluentd 0.14.20	Y	Y	Y
Grafana 4.4.2	Y	Y	Y
Kibana 5.4.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Supported with Open JDK 1.8 and Oracle JDK 1.8.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]Supported for monitoring use cases only.

EEP 4.1.2 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 4.1.2 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.12	Y[2]	Y	Y
Flume 1.7	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[3]	Y	Y	Y
Impala 2.7.0	Y (12 SP1)	Y	N
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.7.2	Y	Y	Y
Elasticsearch 5.4.1	Y	Y	Y

Fluentd 0.14.20	Y	Y	Y
Grafana 4.4.2	Y	Y	Y
Kibana 5.4.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Supported with Open JDK 1.8 and Oracle JDK 1.8.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]Supported for monitoring use cases only.

EEP 4.1.1 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 4.1.1 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.12	Y[2]	Y	Y
Flume 1.7	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[3]	Y	Y	Y
Impala 2.7.0	Y (12 SP1)	Y	N
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.7.2	Y	Y	Y
Elasticsearch 5.4.1	Y	Y	Y
Fluentd 0.14.20	Y	Y	Y
Grafana 4.4.2	Y	Y	Y

last-updated: Apr 11, 2022

Kibana 5.4.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Supported with Open JDK 1.8 and Oracle JDK 1.8.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]Supported for monitoring use cases only.

EEP 4.1.0 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 4.1.0 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.12	Y[2]	Y	Y
Flume 1.7	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[3]	Y	Y	Y
Impala 2.7.0	Y (12 SP1)	Y	N
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.7.2	Y	Y	Y
Elasticsearch 5.4.1	Y	Y	Y
Fluentd 0.14.20	Y	Y	Y
Grafana 4.4.2	Y	Y	Y
Kibana 5.4.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Supported with Open JDK 1.8 and Oracle JDK 1.8.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]Supported for monitoring use cases only.

EEP 4.0.0 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 4.0.0 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix on page 5522.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7[1]	Y	Y	Y
Drill 1.11	Y[2]	Y	Y
Flume 1.7	Y	Y	Y
HBase-compatible API[1]	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[3]	Y	Y	Y
Impala 2.7.0	Y (12 SP1)	Y	N
Myriad 0.2	Y	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[4]	Y	Y	Y
Spark 2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
MapR Monitoring Components[5]			
Collectd 5.7.2	Y	Y	Y
Elasticsearch 5.4.1	Y	Y	Y
Fluentd 0.14.20	Y	Y	Y
Grafana 4.4.2	Y	Y	Y
Kibana 5.4.1	Y	Y	Y
OpenTSDB 2.4.0	Y	Y	Y

[1]Supported for use only with MapR Database binary tables.

[2]Supported with Open JDK 1.8 and Oracle JDK 1.8.

last-updated: Apr 11, 2022

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

[5]Supported for monitoring use cases only.

EEP 3.0.5 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 3.0.5 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix (MapR 5.x) on page 5524.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.10	Y[1]	Y	Y
Flume 1.7	Y	Y	Y
HBase 1.1.8	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[2]	Y	Y	Y
Impala 2.7.0	Y (12 SP1)	Y (6.5, 6.6, 6.8, 7.0, 7.1, and 7.2, 7.3)	N
Mahout 0.12.0	Y	Y	Y
Myriad 0.1	N	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[3]	Y	Y	Y
Spark 2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
Storm 0.10	Y	Y	Y
MapR Monitoring Components[4]			
Collectd 5.7.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.00	Y	Y	Y
Grafana 4.1.2	Y	Y	Y
Kibana 4.5.4	Y	Y	Y
OpenTSDB 2.3.0	Y	Y	Y

[1]Supported with Open JDK 1.8 and Oracle JDK 1.7 or 1.8.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

[4]Supported for monitoring use cases only.

EEP 3.0.4 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 3.0.4 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix (MapR 5.x) on page 5524.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.10	Y[1]	Y	Y
Flume 1.7	Y	Y	Y
HBase 1.1.8	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[2]	Y	Y	Y
Impala 2.7.0	Y (12 SP1)	Y (6.5, 6.6, 6.8, 7.0, 7.1, and 7.2, 7.3)	N
Mahout 0.12.0	Y	Y	Y
Myriad 0.1	N	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[3]	Y	Y	Y
Spark 2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
Storm 0.10	Y	Y	Y
MapR Monitoring Components[4]			
Collectd 5.7.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.00	Y	Y	Y
Grafana 4.1.2	Y	Y	Y
Kibana 4.5.4	Y	Y	Y
OpenTSDB 2.3.0	Y	Y	Y

[1]Supported with Open JDK 1.8 and Oracle JDK 1.7 or 1.8.

last-updated: Apr 11, 2022

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

[4]Supported for monitoring use cases only.

EEP 3.0.3 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 3.0.3 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix (MapR 5.x) on page 5524.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.10	Y[1]	Y	Y
Flume 1.7	Y	Y	Y
HBase 1.1.8	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[2]	Y	Y	Y
Impala 2.7.0	Y (12 SP1)	Y (6.5, 6.6, 6.8, 7.0, 7.1, and 7.2, 7.3)	N
Mahout 0.12.0	Y	Y	Y
Myriad 0.1	N	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[3]	Y	Y	Y
Spark 2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
Storm 0.10	Y	Y	Y
MapR Monitoring Components[4]			
Collectd 5.7.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.00	Y	Y	Y
Grafana 4.1.2	Y	Y	Y
Kibana 4.5.4	Y	Y	Y
OpenTSDB 2.3.0	Y	Y	Y

[1]Supported with Open JDK 1.8 and Oracle JDK 1.7 or 1.8.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

[4]Supported for monitoring use cases only.

EEP 3.0.2 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 3.0.2 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix (MapR 5.x) on page 5524.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.10	Y[1]	Y	Y
Flume 1.7	Y	Y	Y
HBase 1.1.8	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[2]	Y	Y	Y
Impala 2.7.0	Y (12 SP1)	Y (6.5, 6.6, 6.8, 7.0, 7.1, and 7.2, 7.3)	N
Mahout 0.12.0	Y	Y	Y
Myriad 0.1	N	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[3]	Y	Y	Y
Spark 2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
Storm 0.10	Y	Y	Y
MapR Monitoring Components[4]			
Collectd 5.7.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.00	Y	Y	Y
Grafana 4.1.2	Y	Y	Y
Kibana 4.5.4	Y	Y	Y
OpenTSDB 2.3.0	Y	Y	Y

[1]Supported with Open JDK 1.8 and Oracle JDK 1.7 or 1.8.

last-updated: Apr 11, 2022

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

[4]Supported for monitoring use cases only.

EEP 3.0.1 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 3.0.1 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix (MapR 5.x) on page 5524.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.10	Y[1]	Y	Y
Flume1.7	Y	Y	Y
HBase 1.1.8	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[2]	Y	Y	Y
Impala 2.7.0	Y (12 SP1)	Y (6.5, 6.6, 6.8,7.0, 7.1, and 7.2, 7.3)	N
Mahout 0.12.0	Y	Y	Y
Myriad 0.1	N	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[3]	Y	Y	Y
Spark 2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
Storm 0.10	Y	Y	Y
MapR Monitoring Components[4]			
Collectd 5.7.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.00	Y	Y	Y
Grafana 4.1.2	Y	Y	Y
Kibana 4.5.4	Y	Y	Y
OpenTSDB 2.3.0	Y	Y	Y

[1]Supported with Open JDK 1.8 and Oracle JDK 1.7 or 1.8.

last-updated: Apr 11, 2022

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

[4]Supported for monitoring use cases only.

EEP 3.0 Components and OS Support
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 3.0 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

For the Linux operating-system versions supported by MapR software, see Operating System Support Matrix (MapR 5.x) on page 5524.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.10	Y[1]	Y	Y
Flume1.7	Y	Y	Y
HBase 1.1.8	Y	Y	Y
Hive 2.1.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.12[2]	N	Y	Y
Impala 2.7.0	Y (12 SP1)	Y (6.5, 6.6, 6.7, 6.8, 7.0, 7.1, and 7.2, 7.3)	N
Mahout 0.12.0	Y	Y	Y
Myriad 0.1	N	Y	Y
Oozie 4.3.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.7.0[3]	Y	Y	Y
Spark 2.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
Storm 0.10	Y	Y	Y
MapR Monitoring Components[4]			
Collectd 5.7.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.00	Y	Y	Y
Grafana 4.1.2	Y	Y	Y
Kibana 4.5.4	Y	Y	Y
OpenTSDB 2.3.0	Y	Y	Y

[1]Supported with Open JDK 1.8 and Oracle JDK 1.7 or 1.8.

last-updated: Apr 11, 2022

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

[4]Supported for monitoring use cases only.

EEP 2.0.3 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 2.0.3 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.9	Y[3]	Y	Y
Flume 1.6	Y	Y	Y
HBase 1.1.8	Y	Y	Y
Hive 1.2.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.10[1]	Y	Y	Y
Impala 2.5.0	N	Y (6.5, 6.6, 6.8, 7.0, 7.1, and 7.2)	N
Mahout 0.12.0	Y	Y	Y
Myriad 0.1	N	Y	Y
Oozie 4.2.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.6.0	Y	Y	Y
Spark 2.0.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
Storm 0.10.0	Y	Y	Y
MapR Monitoring Components[2]			
Collectd 5.7.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.00	Y	Y	Y
Grafana 4.1.2	Y	Y	Y
Kibana 4.5.4	Y	Y	Y
OpenTSDB 2.3.0	Y	Y	Y

[1]The Spark Notebook UI in Hue is a beta feature.

[2]Supported for monitoring use cases only.

[3]Supported with Open JDK 1.8 and Oracle JDK 1.7 or 1.8.

last-updated: Apr 11, 2022

EEP 2.0.2 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 2.0.2 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.9	Y[3]	Y	Y
Flume 1.6	Y	Y	Y
HBase 1.1.8	Y	Y	Y
Hive 1.2.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.10[1]	Y	Y	Y
Impala 2.5.0	N	Y (6.5, 6.6, 6.8, 7.0, 7.1, and 7.2)	N
Mahout 0.12.0	Y	Y	Y
Myriad 0.1	N	Y	Y
Oozie 4.2.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.6.0	Y	Y	Y
Spark 2.0.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
Storm 0.10.0	Y	Y	Y
MapR Monitoring Components[2]			
Collectd 5.5.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.00	Y	Y	Y
Grafana 4.1.2	Y	Y	Y
Kibana 4.5.4	Y	Y	Y
OpenTSDB 2.3.0	Y	Y	Y

[1]The Spark Notebook UI in Hue is a beta feature.

[2]Supported for monitoring use cases only.

[3]Supported with Open JDK 1.8 and Oracle JDK 1.7 or 1.8.

EEP 2.0.1 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 2.0.1 and shows the operating system support for each component.

last-updated: Apr 11, 2022

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.9	Y[3]	Y	Y
Flume 1.6	Y	Y	Y
HBase 1.1.8	Y	Y	Y
Hive 1.2.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.10[1]	Y	Y	Y
Impala 2.5.0	N	Y (6.5, 6.6, 6.8, 7.0, 7.1, and 7.2)	N
Mahout 0.12.0	Y	Y	Y
Myriad 0.1	N	Y	Y
Oozie 4.2.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.6.0	Y	Y	Y
Spark 2.0.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
Storm 0.10.0	Y	Y	Y
MapR Monitoring Components[2]			
Collectd 5.5.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.00	Y	Y	Y
Grafana 4.1.2	Y	Y	Y
Kibana 4.5.4	Y	Y	Y
OpenTSDB 2.3.0	Y	Y	Y

[1]The Spark Notebook UI in Hue is a beta feature.

[2]Supported for monitoring use cases only.

[3]Supported with Open JDK 1.8 and Oracle JDK 1.7 or 1.8.

EEP 2.0 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 2.0 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu

last-updated: Apr 11, 2022

AsyncHBase 1.7	Y	Y	Y
Drill 1.9	N	Y	Y
Flume 1.6	Y	Y	Y
HBase 1.1.1	Y	Y	Y
Hive 1.2.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.10[1]	Y	Y	Y
Impala 2.5	N	Y (6.5, 6.6, 6.8, 7.0, 7.1, and 7.2)	N
Mahout 0.12.2	Y	Y	Y
Myriad 0.1	N	Y	Y
Oozie 4.2.0	Y	Y	Y
Pig 0.16	Y	Y	Y
Sentry 1.6.0	Y	Y	Y
Spark 2.0.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.7	Y	Y	Y
Storm 0.10.2	Y	Y	Y
MapR Monitoring Components[2]			
Collectd 5.5.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.6	Y	Y	Y
Grafana 3.1.1	Y	Y	Y
Kibana 4.5.1	Y	Y	Y
OpenTSDB 2.2.1	Y	Y	Y

[1]The Spark Notebook UI in Hue is a beta feature.

[2]Supported for monitoring use cases only.

EEP 1.1.4 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 1.1.4 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.8	N	Y	Y
Flume 1.6	Y	Y	Y
HBase 1.1.8	Y	Y	Y

last-updated: Apr 11, 2022

Hive 1.2.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.9[1]	Y	Y	Y
Impala 2.5.0	N	Y (6.5, 6.6, 6.8, 7.0, 7.1, and 7.2)	N
Mahout 0.12.0	Y	Y	Y
Myriad 0.1	N	Y	Y
Oozie 4.2.0	Y	Y	Y
Pig 0.15	Y	Y	Y
Sentry 1.6.0	Y	Y	Y
Spark 1.6.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.6	Y	Y	Y
Storm 0.10.0	Y	Y	Y
MapR Monitoring Components[2]			
Collectd 5.7.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.00	Y	Y	Y
Grafana 4.1.2	Y	Y	Y
Kibana 4.5.4	Y	Y	Y
OpenTSDB 2.3.0	Y	Y	Y

[1]The Spark Notebook UI in Hue is a beta feature.

[2]Supported for monitoring use cases only.

EEP 1.1.3 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 1.1.3 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.8	N	Y	Y
Flume 1.6	Y	Y	Y
HBase 1.1.8	Y	Y	Y
Hive 1.2.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.9[1]	Y	Y	Y

last-updated: Apr 11, 2022

Impala 2.5.0	N	Y (6.5, 6.6, 6.8, 7.0, 7.1, and 7.2)	N
Mahout 0.12.0	Y	Y	Y
Myriad 0.1	N	Y	Y
Oozie 4.2.0	Y	Y	Y
Pig 0.15	Y	Y	Y
Sentry 1.6.0	Y	Y	Y
Spark 1.6.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.6	Y	Y	Y
Storm 0.10.0	Y	Y	Y
MapR Monitoring Components[2]			
Collectd 5.5.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.00	Y	Y	Y
Grafana 4.1.2	Y	Y	Y
Kibana 4.5.4	Y	Y	Y
OpenTSDB 2.3.0	Y	Y	Y

[1]The Spark Notebook UI in Hue is a beta feature.

[2]Supported for monitoring use cases only.

EEP 1.1.2 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 1.1.2 and shows the operating system support for each component.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.8	N	Y	Y
Flume 1.6	Y	Y	Y
HBase 1.1.8	Y	Y	Y
Hive 1.2.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.9[1]	Y	Y	Y
Impala 2.5.0	N	Y (6.5, 6.6, 6.8, 7.0, 7.1, and 7.2)	N
Mahout 0.12.0	Y	Y	Y

last-updated: Apr 11, 2022

Myriad 0.1	N	Y	Y
Oozie 4.2.0	Y	Y	Y
Pig 0.15	Y	Y	Y
Sentry 1.6.0	Y	Y	Y
Spark 1.6.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.6	Y	Y	Y
Storm 0.10.0	Y	Y	Y
MapR Monitoring Components[2]			
Collectd 5.5.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.00	Y	Y	Y
Grafana 4.1.2	Y	Y	Y
Kibana 4.5.4	Y	Y	Y
OpenTSDB 2.3.0	Y	Y	Y

[1]The Spark Notebook UI in Hue is a beta feature.

[2]Supported for monitoring use cases only.

EEP 1.1.1 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 1.1.1 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.8	N	Y	Y
Flume 1.6	Y	Y	Y
HBase 1.1	Y	Y	Y
Hive 1.2	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.9	Y	Y	Y
Impala 2.5.0	N	Y (6.5, 6.6, 6.8, 7.0, 7.1, and 7.2)	N
Mahout 0.12.0	Y	Y	Y
Myriad 0.1	N	Y	Y
Oozie 4.2.0	Y	Y	Y
Pig 0.15	Y	Y	Y
Sentry 1.6.0	Y	Y	Y

last-updated: Apr 11, 2022

Spark 1.6.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.6	Y	Y	Y
Storm 0.10.0	Y	Y	Y
MapR Monitoring Components[1]			
Collectd 5.5.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.00	Y	Y	Y
Grafana 3.1.1	Y	Y	Y
Kibana 4.5.1	Y	Y	Y
OpenTSDB 2.2.1	Y	Y	Y

[1]Supported for monitoring use cases only.

EEP 1.1 Components and OS Support

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 1.1 and shows the operating system support for each component.

To understand which MapR Core versions can use this EEP, see EEP Support and Lifecycle Status on page 5531.

Ecosystem Components	SUSE	RHEL / CentOS	Ubuntu
AsyncHBase 1.7	Y	Y	Y
Drill 1.8	N	Y	Y
Flume 1.6	Y	Y	Y
HBase 1.1.1	Y	Y	Y
Hive 1.2.1	Y	Y	Y
HttpFS 1.0	Y	Y	Y
Hue 3.9[1]	Y	Y	Y
Impala 2.5	N	Y (6.5, 6.6, 6.8, 7.0, 7.1, and 7.2)	N
Mahout 0.12.2	Y	Y	Y
Myriad 0.1	N	Y	Y
Oozie 4.2.0	Y	Y	Y
Pig 0.15	Y	Y	Y
Sentry 1.6.0	Y	Y	Y
Spark 1.6.1	Y	Y	Y
Sqoop 1.4.6	Y	Y	Y
Sqoop2 1.99.6	Y	Y	Y
Storm 0.10.2	Y	Y	Y
MapR Monitoring Components[2]			

last-updated: Apr 11, 2022

Collectd 5.5.1	Y	Y	Y
Elasticsearch 2.3.3	Y	Y	Y
Fluentd 0.14.00	Y	Y	Y
Grafana 3.1.1	Y	Y	Y
Kibana 4.5.1	Y	Y	Y
OpenTSDB 2.2.0	Y	Y	Y

[1]The Spark Notebook UI in Hue is a beta feature.

[2]Supported for monitoring use cases only.

Discontinued Ecosystem Components

Provides information about discontinued ecosystem components.

Ecosystem components can be discontinued when either of the following is true:

- Newer components are available that serve the same function but provide better features and performance.
- The Open Source community decides to retire a product.

To understand what *discontinued* and *deprecated* mean when these terms are applied to a component or EEP, see Understand the EEP Lifecycle on page 5528.

HPE uses the support advisory process to notify users in advance that a component will be discontinued. The last EEP version to support a deprecated component is the EEP version that is released at the time of the deprecation announcement. The following table lists the components that are either already discontinued or scheduled to be discontinued:

Component	Deprecated	Discontinued	Last EEP Version Supporting Component*	Suggested Replacement	Announcement
S3 Gateway on page 3959	July 31, 2022	July 31, 2023	• EEP 8.1.x • EEP 8.0.x • EEP 7.x • EEP 6.x	HPE Ezmeral Data Fabric Object Store	February 2022
Oozie on page 3989	June 30, 2022	June 30, 2023	• EEP 8.1.x • EEP 8.0.x • EEP 7.x • EEP 6.x • EEP 5.x	Apache Airflow	January 2022
Pig on page 4009	October 31, 2021	October 31, 2022	• EEP 8.0.0 • EEP 6.3.5 • EEP 5.0.7	None	October 2021

Component	Deprecated	Discontinued	Last EEP Version Supporting Component*	Suggested Replacement	Announcement
Flume on page 3367	October 31, 2021	October 31, 2022	• EEP 8.0.0 • EEP 6.3.5 • EEP 5.0.7	None	October 2021
Sqoop on page 4136	October 31, 2021	October 31, 2022	• EEP 8.0.0 • EEP 6.3.5 • EEP 5.0.7	None	October 2021
Impala	June 1, 2021	December 31, 2021	• EEP 7.1.0 • EEP 6.3.4 • EEP 5.0.7	Spark or equivalent partner product. See the HPE Technology Partner website.	June 2021
Sentry	June 1, 2021	December 31, 2021	• EEP 7.1.0 • EEP 6.3.4 • EEP 5.0.7	None	June 2021
Data Science Refinery	September 18, 2020	May 31, 2022	N/A	HPE Ezmeral ML Ops	N/A
Myriad	September 2020	See note**	• EEP 5.0.x • EEP 6.3.x	Kubernetes	MapR Marketing Newsletter
Sqoop2	September 2020	See note**	• EEP 5.0.x • EEP 6.3.x	Equivalent partner product. See the HPE Technology Partner website.	MapR Marketing Newsletter
Mahout	October 2017	April 2019	• EEP 3.0.5	None	MapR Marketing Newsletter
Mezos	October 2017	April 2019	• EEP 3.0.2	Kubernetes	MapR Marketing Newsletter
Storm	October 2017	April 2019	• EEP 3.0.5	None	MapR Marketing Newsletter
Hue-Livy	October 2017	April 2019	• EEP 3.0.5	Livy	MapR Marketing Newsletter

*Later EEPs in the same series can include a discontinued component. For example, if the last EEP version supporting a discontinued component is x.y.0, the component might be present in x.y.1, x.y.2, x.y.3, and so on. This is to ensure that upgrades complete successfully. But HPE does not support using the component in later EEPs (x.y.1, x.y.2, x.y.3, and so on), and the last supported EEP version remains x.y.0.

**Discontinued for EEP 7.0.0 and later in September 2020. To be discontinued for EEP 5.0.x and EEP 6.3.x when those EEPs reach end of life.

Related reference

EEP Support and Lifecycle Status on page 5531

last-updated: Apr 11, 2022

This page shows the EEPs that are supported for different core releases and the current lifecycle status for each EEP.

Understand the EEP Lifecycle on page 5528
This page describes the EEP lifecycle and defines the lifecycle stages, which are Active, Deprecated, and Discontinued.

Component Versions for Released EEPs

The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Components Supported by EEPs 1.x - 8.x

The following tables show the latest supported versions of each component in a EEP series. The tables reflect the component versions at release time and do not include patch versions issued between releases.

Core 7.0.0 with EEP 8.x

	EEP Series Versions
Ecosystem Component	**EEP-8.1.0**
AsynchHBase	1.8.2
Drill	1.16.1.400
Data Access Gateway	4.0.0.0
Flume	—
Hadoop	2.7.6.200
HBase	1.4.13.200
Hive	2.3
HttpFS	1.1.0.200
Hue	4.6.0.300
Kafka Connect	10.0.0.100
Kafka REST	6.0.0.100
Kafka Schema Registry*	6.0.0.100
KSQL	6.0.0.100
Kafka Streams	2.6.1.100
Livy**	0.7.0.100
Oozie	5.2.1.200
Pig	—
S3 Gateway	2.2.0.0
Spark	3.2.0.0
Sqoop	—
Tez***	0.9
Monitoring Components	
Collectd	5.12.0.400

Ecosystem Component	EEP Series Versions
	EEP-8.1.0
Elasticsearch	6.8.8.500
Fluentd	1.10.3.400
Grafana	7.5.10.400
Kibana	6.8.8.500
Open TSDB	2.4.1.400

Core 6.2.0 with EEP 7.x and EEP 8.x

Ecosystem Component	EEP Series Versions				
	EEP-8.0.0	EEP-7.1.1	EEP-7.1.0	EEP-7.0.1	EEP-7.0.0
AsynchHBase	1.8.2	1.8.2	1.8.2	1.8.2	1.8.2
Drill	1.16.1.300	1.16.1.200	1.16.1.200	1.16.1.100	1.16.1
Data Access Gateway	3.0.0.0	3.0.0.0	3.0.0.0	3.0.0.0	3.0.0.0
Flume	1.9.0.200	1.9.0.100	1.9.0.100	1.9.0.100	1.9.0.0
Hadoop	2.7.6.100	2.7.5.0	2.7.5.0	2.7.4.100	2.7.4.0
HBase	1.4.13.100	1.4.13.0	1.4.13.0	1.4.12.100	1.4.12.0
Hive	2.3	2.3	2.3	2.3	2.3
HttpFS	1.1.0.100	1.1.0.0	1.1.0.0	1.0	1.0
Hue	4.6.0.200	4.6.0.0	4.6.0.0	4.6.0.0	4.6.0.0
Impala	—	2.12.0.500	2.12.0.500	2.12.0.400	2.12.0.200
Kafka Connect	10.0.0.0	5.1.2.200	5.1.2.200	5.1.2.100	5.1.2.0
Kafka REST	6.0.0.0	5.1.2.200	5.1.2.200	5.1.2.100	5.1.2.0
Kafka Schema Registry*	6.0.0.0	5.1.2.200	5.1.2.200	5.1.2.100	5.1.2.0
KSQL	6.0.0.0	5.1.2.200	5.1.2.200	5.1.2.100	5.1.2.0
Kafka Streams	2.6.1.0	2.1.1.200	2.1.1.200	2.1.1.100	2.1.1.0
Livy**	0.7.0.100	0.7.0.0	0.7.0.0	0.5.0	0.5.0
Oozie	5.2.1.100	5.2.1.0	5.2.1.0	5.2.0.100	5.2.0.0
Pig	0.17.0.100	0.17.0.0	0.17.0.0	0.17.0.0	0.17.0.0
S3 Gateway	2.2.0.0	2.1.0.0	2.1.0.0	2.0.0.0	2.0.0.0
Sentry	—	1.7.0	1.7.0	1.7.0	1.7.0
Spark	3.1.2.0	2.4.7.100	2.4.7.100	2.4.7.0	2.4.5.0
Sqoop	1.4.7	1.4.7	1.4.7	1.4.7	1.4.7
Tez***	0.9	0.9	0.9	0.9	0.9
Monitoring Components					
Collectd	5.12.0.300	5.10.0.0	5.10.0.0	5.10.0.0	5.10.0.0

last-updated: Apr 11, 2022

Ecosystem Component	EEP Series Versions				
	EEP-8.0.0	EEP-7.1.1	EEP-7.1.0	EEP-7.0.1	EEP-7.0.0
Elasticsearch	6.8.8.400	6.8.8.300	6.8.8.300	6.8.8.0	6.8.8.0
Fluentd	1.10.3.300	1.10.3.200	1.10.3.0	1.10.3.0	1.10.3.0
Grafana	7.5.10.300	7.5.2.200	7.5.2.200	6.7.4.0	6.7.4.0
Kibana	6.8.8.400	6.8.8.300	6.8.8.300	6.8.8.0	6.8.8.0
Open TSDB	2.4.1.300	2.4.0	2.4.0	2.4.0	2.4.0

Core 6.1.1 with EEP 6.x

Ecosystem Component	EEP Series Versions			
	EEP-6.3.6	EEP-6.3.5	EEP-6.3.4	EEP-6.3.3
AsynchHBase	1.7.0	1.7.0	1.7.0	1.7.0
Drill	1.16.0.400	1.16.0.300	1.16.0.200	1.16.0.100
Data Access Gateway	2.0	2.0	2.0	2.0
Flume	1.8.0	1.8.0	1.8.0	1.8.0
HBase	1.1.13.500	1.1.13.400	1.1.13.300	1.1.13.200
Hive	2.3	2.3	2.3	2.3
HttpFS	1.0	1.0	1.0	1.0
Hue	4.3.0.500	4.3.0.400	4.3.0.400	4.3.0.300
Impala	2.12.0.650	2.12.0.600	2.12.0.300	2.12.0.300
Kafka Connect	4.1.0	4.1.0	4.1.0	4.1.0
Kafka REST	4.1.0	4.1.0	4.1.0	4.1.0
KSQL	4.1.1	4.1.1	4.1.1	4.1.1
Kafka Streams	1.1.1	1.1.1	1.1.1	1.1.1
Livy**	0.5.0	0.5.0	0.5.0	0.5.0
Myriad	0.2	0.2	0.2	0.2
Oozie	5.1.0.800	5.1.0.700	5.1.0.600	5.1.0.500
Pig	0.16	0.16	0.16	0.16
S3 Gateway	1.0.1	1.0.1	1.0.1	1.0.1
Sentry	1.7.0	1.7.0	1.7.0	1.7.0
Spark	2.4.4.500	2.4.4.400	2.4.4.300	2.4.4.200
Sqoop	1.4.7	1.4.7	1.4.7	1.4.7
Sqoop2	—	2.0.0	2.0.0	2.0.0
Tez***	0.9	0.9	0.9	0.9
Monitoring Components				
Collectd	5.8.1.210	5.8.1.201	5.8.1.201	5.8.1.201
Elasticsearch	6.8.8.110	6.8.8.100	6.8.8.100	6.8.8.100

last-updated: Apr 11, 2022

Ecosystem Component	EEP Series Versions			
	EEP-6.3.6	EEP-6.3.5	EEP-6.3.4	EEP-6.3.3
Fluentd	1.10.3.110	1.10.3.100	1.10.3.100	1.10.3.100
Grafana	7.5.2.110	7.5.2.100	7.5.2.100	6.7.4.100
Kibana	6.8.8.110	6.8.8.100	6.8.8.100	6.8.8.100
Open TSDB	2.4.0	2.4.0	2.4.0	2.4.0

Core 6.1.0 with EEP 6.3.x to Current

Ecosystem Component	EEP Series Versions						
	EEP-6.3.6	EEP-6.3.5	EEP-6.3.4	MEP-6.3.3****	EEP-6.3.2	EEP-6.3.1	EEP-6.3.0
AsynchHBase	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0
Drill	1.16.0.400	1.16.0.300	1.16.0.200	1.16.0.100	1.16.0.100	1.16.0.22	1.16.0.10
Data Access Gateway	2.0	2.0	2.0	2.0	2.0	2.0	2.0
Flume	1.8.0	1.8.0	1.8.0	1.8.0	1.8.0	1.8.0	1.8.0
HBase	1.1.13.500	1.1.13.400	1.1.13.300	1.1.13.200	1.1.13.200	1.1.13.100	1.1.13.0
Hive	2.3	2.3	2.3	2.3	2.3	2.3	2.3
HttpFS	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Hue	4.3.0.500	4.3.0.400	4.3.0.400	4.3.0.300	4.3.0.300	4.3.0.200	4.3.0.100
Impala	2.12.0.650	2.12.0.600	2.12.0.300	2.12.0.300	2.12.0.300	2.12.0.100	2.12.0.100
Kafka Connect	4.1.0	4.1.0	4.1.0	4.1.0	4.1.0	4.1.0	4.1.0
Kafka REST	4.1.0	4.1.0	4.1.0	4.1.0	4.1.0	4.1.0	4.1.0
Kafka Schema Registry*	—	—	—	—	—	—	4.1.1
KSQL	4.1.1	4.1.1	4.1.1	4.1.1	4.1.1	4.1.1	4.1.1
Kafka Streams	1.1.1	1.1.1	1.1.1	1.1.1	1.1.1	1.1.1	1.1.1
Livy**	0.5.0	0.5.0	0.5.0	0.5.0	0.5.0	0.5.0	0.5.0
Myriad	0.2	0.2	0.2	0.2	0.2	0.2	0.2
Oozie	5.1.0.800	5.1.0.700	5.1.0.600	5.1.0.500	5.1.0.500	5.1.0.400	5.1.0.300
Pig	0.16	0.16	0.16	0.16	0.16	0.16	0.16
S3 Gateway	1.0.1	1.0.1	1.0.1	1.0.1	1.0.1	1.0.1	1.0.1
Sentry	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0
Spark	2.4.4.500	2.4.4.400	2.4.4.300	2.4.4.200	2.4.4.200	2.4.4.100	2.4.4.0
Sqoop	1.4.7	1.4.7	1.4.7	1.4.7	1.4.7	1.4.7	1.4.7
Sqoop2	—	2.0.0	2.0.0	2.0.0	2.0.0	2.0.0	1.99.7
Tez***	0.9	0.9	0.9	0.9	0.9	0.9	0.9

last-updated: Apr 11, 2022

Ecosystem Component	EEP Series Versions						
	EEP-6.3.6	EEP-6.3.5	EEP-6.3.4	MEP-6.3.3***	EEP-6.3.2	EEP-6.3.1	EEP-6.3.0
Monitoring Components							
Collectd	5.8.1.210	5.8.1.201	5.8.1.201	5.8.1.201	5.8.1.201	5.8.1.201	5.8.1.200
Elasticsearch	6.8.8.110	6.8.8.100	6.8.8.100	6.8.8.100	6.8.8.100	6.8.8.100	6.5.3.200
Fluentd	1.10.3.110	1.10.3.100	1.10.3.100	1.10.3.100	1.10.3.100	1.10.3.100	1.4.0.100
Grafana	7.5.2.110	7.5.2.100	7.5.2.100	6.7.4.100	6.7.4.100	6.7.4.100	6.0.2.100
Kibana	6.8.8.110	6.8.8.100	6.8.8.100	6.8.8.100	6.8.8.100	6.8.8.100	6.5.3.200
Open TSDB	2.4.0	2.4.0	2.4.0	2.4.0	2.4.0	2.4.0	2.4.0

Core 6.1.0 with EEP 6.0.x through EEP 6.2.x

Ecosystem Component	EEP Series Versions					
	EEP-6.2.0	EEP-6.1.1	EEP-6.1.0	EEP-6.0.2	EEP-6.0.1	EEP-6.0.0
AsynchHBase	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0
Drill	1.16.0.0	1.15.0.7	1.15.0.0	1.14.0	1.14.0	1.14.0
Data Access Gateway	2.0	2.0	2.0	2.0	2.0	2.0
Flume	1.8.0	1.8.0	1.8.0	1.8.0	1.8.0	1.8.0
HBase	1.1.8	1.1.8	1.1.8	1.1.8	1.1.8	1.1.8
Hive	2.3	2.3	2.3	2.3	2.3	2.3
HttpFS	1.0	1.0	1.0	1.0	1.0	1.0
Hue	4.3.0.0	4.2.0	4.2.0	4.2.0	4.2.0	4.2.0
Impala	2.12.0.0	2.10.0	2.10.0	2.10.0	2.10.0	2.10.0
Kafka Connect	4.1.0	4.1.0	4.1.0	4.1.0	4.1.0	4.1.0
Kafka REST	4.1.0	4.1.0	4.1.0	4.1.0	4.1.0	4.1.0
Kafka Schema Registry*	4.1.1	4.1.1	4.1.1	4.1.1	4.1.1	4.1.1
KSQL	4.1.1	4.1.1	4.1.1	4.1.1	4.1.1	4.1.1
Kafka Streams	1.1.1	1.1.1	1.1.1	1.1.1	1.1.1	1.1.1
Livy**	0.5.0	0.5.0	0.5.0	0.5.0	0.5.0	0.5.0
Myriad	0.2	0.2	0.2	0.2	0.2	0.2
Oozie	5.1.0.200	5.1.0.100	5.1.0.0	4.3.0	4.3.0	4.3.0
Pig	0.16	0.16	0.16	0.16	0.16	0.16
S3 Gateway	1.0.1	1.0.1	1.0.1	1.0.1	1.0.1	1.0.0
Sentry	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0
Spark	2.4.0.0	2.3.3.100	2.3.2.0	2.3.3.0	2.3.2.0	2.3.1
Sqoop	1.4.7	1.4.7	1.4.7	1.4.7	1.4.7	1.4.7

last-updated: Apr 11, 2022

Ecosystem Component	EEP Series Versions					
	EEP-6.2.0	EEP-6.1.1	EEP-6.1.0	EEP-6.0.2	EEP-6.0.1	EEP-6.0.0
Sqoop2	1.99.7	1.99.7	1.99.7	1.99.7	1.99.7	1.99.7
Tez***	0.9	0.9	0.9	0.9	0.9	0.9
Monitoring Components						
Collectd	5.8.1.100	5.8.1.1	5.8.1.0	5.8.0	5.8.0	5.8.0
Elasticsearch	6.5.3.100	6.5.3.1	6.5.3.0	6.2.3	6.2.3	6.2.3
Fluentd	1.4.0.0	1.3.2.1	1.3.2.0	1.1.2	1.1.2	1.1.2
Grafana	6.0.2.0	5.4.2.1	5.4.2.0	4.6.5	4.6.1	4.6.1
Kibana	6.5.3.100	6.5.3.1	6.5.3.0	6.2.3	6.2.3	6.2.3
Open TSDB	2.4.0	2.4.0	2.4.0	2.4.0	2.4.0	2.4.0

Core 6.0.1 with EEP 5.x

Ecosystem Component	EEP Series Versions							
	EEP-5.0.7	EEP-5.0.6	EEP-5.0.5	EEP-5.0.4	EEP-5.0.3	EEP-5.0.2	EEP-5.0.1	EEP-5.0.0
AsynchHBase	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0
Drill	1.13.0	1.13.0	1.13.0	1.13.0	1.13.0	1.13.0	1.13.0	1.13.0
Data Access Gateway	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Flume	1.8.0	1.8.0	1.8.0	1.8.0	1.8.0	1.8.0	1.8.0	1.8.0
HBase	1.1.8	1.1.8	1.1.8	1.1.8	1.1.8	1.1.8	1.1.8	1.1.8
Hive	2.1	2.1	2.1	2.1	2.1	2.1	2.1	2.1
HttpFS	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Hue	3.12.0	3.12.0	3.12.0	3.12.0	3.12.0	3.12.0	3.12.0	3.12.0
Impala	2.10.0	2.10.0	2.10.0	2.10.0	2.10.0	2.10.0	2.10.0	2.10.0
Kafka Connect	4.0.0	4.0.0	4.0.0	4.0.0	4.0.0	4.0.0	4.0.0	4.0.0
Kafka REST	4.0.0	4.0.0	4.0.0	4.0.0	4.0.0	4.0.0	4.0.0	4.0.0
Kafka Streams	1.0.1	1.0.1	1.0.1	1.0.1	1.0.1	1.0.1	1.0.1	1.0.1
Livy**	0.3.0	0.3.0	0.3.0	0.3.0	0.3.0	0.3.0	0.3.0	0.3.0
Myriad	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
Oozie	4.3.0	4.3.0	4.3.0	4.3.0	4.3.0	4.3.0	4.3.0	4.3.0
Pig	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16
Sentry	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0
Spark	2.2.1	2.2.1	2.2.1	2.2.1	2.2.1	2.2.1	2.2.1	2.2.1
Sqoop	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6

last-updated: Apr 11, 2022

Ecosystem Component	EEP Series Versions							
	EEP-5.0.7	EEP-5.0.6	EEP-5.0.5	EEP-5.0.4	EEP-5.0.3	EEP-5.0.2	EEP-5.0.1	EEP-5.0.0
Sqoop2	2.0.0	2.0.0	2.0.0	1.99.7	1.99.7	1.99.7	1.99.7	1.99.7
Tez***	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8
Monitoring Components								
Collectd	5.7.2	5.7.2	5.7.2	5.7.2	5.7.2	5.7.2	5.7.2	5.7.2
Elasticsearch	5.4.1	5.4.1	5.4.1	5.4.1	5.4.1	5.4.1	5.4.1	5.4.1
Fluentd	0.14.20	0.14.20	0.14.20	0.14.20	0.14.20	0.14.20	0.14.20	0.14.20
Grafana	7.5.2.0	6.7.4.200	6.7.4.200	4.6.5	4.6.5	4.6.1	4.6.1	4.6.1
Kibana	5.4.1	5.4.1	5.4.1	5.4.1	5.4.1	5.4.1	5.4.1	5.4.1
Open TSDB	2.4.0	2.4.0	2.4.0	2.4.0	2.4.0	2.4.0	2.4.0	2.4.0

Core 6.0.0 with EEP 4.x

Ecosystem Component	EEP Series Versions				
	EEP-4.1.4	EEP-4.1.3	EEP-4.1.2	EEP-4.1.1	EEP-4.1.0
AsynchHBase	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0
Drill	1.12.0	1.12.0	1.12.0	1.12.0	1.12.0
Flume	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0
HBase	1.1.8	1.1.8	1.1.8	1.1.8	1.1.8
Hive	2.1	2.1	2.1	2.1	2.1
HttpFS	1.0	1.0	1.0	1.0	1.0
Hue	3.12.0	3.12.0	3.12.0	3.12.0	3.12.0
Impala	2.7.0	2.7.0	2.7.0	2.7.0	2.7.0
Kafka Connect	2.0.1	2.0.1	2.0.1	2.0.1	2.0.1
Kafka REST	2.0.1	2.0.1	2.0.1	2.0.1	2.0.1
Kafka Streams	0.9.0	0.9.0	0.9.0	0.9.0	0.9.0
Livy**	0.3.0	0.3.0	0.3.0	0.3.0	0.3.0
Myriad	0.2	0.2	0.2	0.2	0.2
Oozie	4.3.0	4.3.0	4.3.0	4.3.0	4.3.0
Pig	0.16	0.16	0.16	0.16	0.16
Sentry	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0
Spark	2.1.0	2.1.0	2.1.0	2.1.0	2.1.0
Sqoop	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6
Sqoop2	1.99.7	1.99.7	1.99.7	1.99.7	1.99.7
Tez***	0.8	0.8	0.8	0.8	0.8
Monitoring Components					

Ecosystem Component	EEP Series Versions				
	EEP-4.1.4	EEP-4.1.3	EEP-4.1.2	EEP-4.1.1	EEP-4.1.0
Collectd	5.7.2	5.7.2	5.7.2	5.7.2	5.7.2
Elasticsearch	5.4.1	5.4.1	5.4.1	5.4.1	5.4.1
Fluentd	0.14.20	0.14.20	0.14.20	0.14.20	0.14.20
Grafana	4.6.5	4.4.2	4.4.2	4.4.2	4.4.2
Kibana	5.4.1	5.4.1	5.4.1	5.4.1	5.4.1
Open TSDB	2.4.0	2.4.0	2.4.0	2.4.0	2.4.0

Core 5.2.x with EEP 3.x

Ecosystem Component	EEP Series Versions					
	EEP-3.0.5	EEP-3.0.4	EEP-3.0.3	EEP-3.0.2	EEP-3.0.1	EEP-3.0.0
AsynchHBase	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0
Drill	1.10.0	1.10.0	1.10.0	1.10.0	1.10.0	1.10.0
Flume	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0
HBase	1.1.8	1.1.8	1.1.8	1.1.8	1.1.8	1.1.8
Hive	2.1	2.1	2.1	2.1	2.1	2.1
HttpFS	1.0	1.0	1.0	1.0	1.0	1.0
Hue	3.12.0	3.12.0	3.12.0	3.12.0	3.12.0	3.12.0
Impala	2.7.0	2.7.0	2.7.0	2.7.0	2.7.0	2.7.0
Kafka Connect	2.0.1	2.0.1	2.0.1	2.0.1	2.0.1	2.0.1
Kafka REST	2.0.1	2.0.1	2.0.1	2.0.1	2.0.1	2.0.1
Kafka Streams	0.9.0	0.9.0	0.9.0	0.9.0	0.9.0	0.9.0
Mahout	0.12.0	0.12.0	0.12.0	0.12.0	0.12.0	0.12.0
Myriad	0.1	0.1	0.1	0.1	0.1	0.1
Oozie	4.3.0	4.3.0	4.3.0	4.3.0	4.3.0	4.3.0
Pig	0.16	0.16	0.16	0.16	0.16	0.16
Sentry	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0
Spark	2.1.0	2.1.0	2.1.0	2.1.0	2.1.0	2.1.0
Sqoop	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6
Sqoop2	1.99.7	1.99.7	1.99.7	1.99.7	1.99.7	1.99.7
Storm	0.10.0	0.10.0	0.10.0	0.10.0	0.10.0	0.10.0
Tez***	0.8	0.8	0.8	0.8	0.8	0.8
Monitoring Components						
Collectd	5.7.1	5.7.1	5.7.1	5.7.1	5.7.1	5.7.1
Elasticsearch	2.3.3	2.3.3	2.3.3	2.3.3	2.3.3	2.3.3
Fluentd	0.14.00	0.14.00	0.14.00	0.14.00	0.14.00	0.14.00

last-updated: Apr 11, 2022

Ecosystem Component	EEP Series Versions					
	EEP-3.0.5	EEP-3.0.4	EEP-3.0.3	EEP-3.0.2	EEP-3.0.1	EEP-3.0.0
Grafana	4.1.2	4.1.2	4.1.2	4.1.2	4.1.2	4.1.2
Kibana	4.5.4	4.5.4	4.5.4	4.5.4	4.5.4	4.5.4
Open TSDB	2.3.0	2.3.0	2.3.0	2.3.0	2.3.0	2.3.0

Core 5.2.x with EEP 2.x

Ecosystem Component	EEP Series Versions			
	EEP-2.0.3	EEP-2.0.2	EEP-2.0.1	EEP-2.0.0
AsynchHBase	1.7.0	1.7.0	1.7.0	1.7.0
Drill	1.9.0	1.9.0	1.9.0	1.9.0
Flume	1.6.0	1.6.0	1.6.0	1.6.0
HBase	1.1.8	1.1.8	1.1.8	1.1.1
Hive	1.2	1.2	1.2	1.2
HttpFS	1.0	1.0	1.0	1.0
Hue	3.10.0	3.10.0	3.10.0	3.10.0
Impala	2.5.0	2.5.0	2.5.0	2.5.0
Kafka Connect	2.0.1	2.0.1	2.0.1	2.0.1
Kafka REST	2.0.1	2.0.1	2.0.1	2.0.1
Kafka Streams	0.9.0	0.9.0	0.9.0	0.9.0
Mahout	0.12.0	0.12.0	0.12.0	0.12.0
Myriad	0.1	0.1	0.1	0.1
Oozie	4.2.0	4.2.0	4.2.0	4.2.0
Pig	0.16	0.16	0.16	0.16
Sentry	1.6.0	1.6.0	1.6.0	1.6.0
Spark	2.0.1	2.0.1	2.0.1	2.0.1
Sqoop	1.4.6	1.4.6	1.4.6	1.4.6
Sqoop2	1.99.7	1.99.7	1.99.7	1.99.7
Storm	0.10.0	0.10.0	0.10.0	0.10.0
Monitoring Components				
Collectd	5.7.1	5.7.1	5.5.1	5.5.1
Elasticsearch	2.3.3	2.3.3	2.3.3	2.3.3
Fluentd	0.14.00	0.14.00	0.14.00	0.14.00
Grafana	4.1.2	4.1.2	4.1.2	3.1.1
Kibana	4.5.4	4.5.4	4.5.4	4.5.1
Open TSDB	2.3.0	2.3.0	2.3.0	2.2.1

last-updated: Apr 11, 2022

Core 5.2.x with EEP 1.x

Ecosystem Component	EEP Series Versions					
	EEP-1.1.4	EEP-1.1.3	EEP-1.1.2	EEP-1.1.1	EEP-1.1.0	EEP-1.0.0
AsynchHBase	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0	1.7.0
Drill	1.8.0	1.8.0	1.8.0	1.8.0	1.8.0	1.6.0
Flume	1.6.0	1.6.0	1.6.0	1.6.0	1.6.0	1.6.0
HBase	1.1.8	1.1.8	1.1.8	1.1.1	1.1.1	1.1.1
Hive	1.2	1.2	1.2	1.2	1.2	1.2
HttpFS	1.0	1.0	1.0	1.0	1.0	1.0
Hue	3.9.0	3.9.0	3.9.0	3.9.0	3.9.0	3.9.0
Impala	2.5.0	2.5.0	2.5.0	2.5.0	2.5.0	2.5.0
Kafka Streams	0.9.0	0.9.0	0.9.0	0.9.0	0.9.0	0.9.0
Mahout	0.12.0	0.12.0	0.12.0	0.12.0	0.12.0	0.12.0
Myriad	0.1	0.1	0.1	0.1	0.1	0.1
Oozie	4.2.0	4.2.0	4.2.0	4.2.0	4.2.0	4.2.0
Pig	0.15	0.15	0.15	0.15	0.15	0.15
Sentry	1.6.0	1.6.0	1.6.0	1.6.0	1.6.0	1.6.0
Spark	1.6.1	1.6.1	1.6.1	1.6.1	1.6.1	1.6.1
Sqoop	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6
Sqoop2	1.99.6	1.99.6	1.99.6	1.99.6	1.99.6	1.99.6
Storm	0.10.0	0.10.0	0.10.0	0.10.0	0.10.0	0.10.0
Monitoring Components						
Collectd	5.7.1	5.7.1	5.5.1	5.5.1	5.5.1	5.5.1
Elasticsearch	2.3.3	2.3.3	2.3.3	2.3.3	2.3.3	2.3.3
Fluentd	0.14.00	0.14.00	0.14.00	0.14.00	0.14.00	0.14.00
Grafana	4.1.2	4.1.2	4.1.2	3.1.1	3.1.1	3.0.4
Kibana	4.5.4	4.5.4	4.5.4	4.5.1	4.5.1	4.5.1
Open TSDB	2.3.0	2.3.0	2.3.0	2.2.1	2.2.0	2.2.0

*Kafka Schema Registry 5.1.2.0 and later are provided as *general availability* software. Kafka Schema Registry 4.1.1 is *developer preview* software. Developer previews are not tested for production environments, and should be used with caution. Kafka Schema Registry packages are included in EEP 7.0.0 and later but not included in EEP 6.x.

**Beginning with EEP 4.0.0, Livy is included as its own package in MapR EEP repositories. Before EEP 4.0.0, Livy was included as mapr-hue-livy and released only as a part of Hue. For more information, see Livy on page 3839.

***Tez is supported only for use with Hive. Therefore, MapR documentation for Tez is limited when compared to the documentation for other ecosystem components. For release note information, see Tez Release Notes.

****Before using EEP 6.3.3 with release 6.1.0, you must apply the latest 6.1.0 patch.

last-updated: Apr 11, 2022

Related concepts

Understand MapR Versions on page 5519
Understanding the version numbers used by MapR core, MapR Ecosystem Packs (EEPs), EEP components, and MapR patches can help you keep your software up to date and plan for upgrades.

Related reference

EEP Support and Lifecycle Status on page 5531
This page shows the EEPs that are supported for different core releases and the current lifecycle status for each EEP.

CSI Version Compatibility

This matrix shows the released versions of the Container Storage Interface (CSI) storage plugin and their compatibility with other components in a Kubernetes environment.

CSI Storage Plugin	Version Compatibility						
	FUSE or Loopback NFS	Kubernetes	CSI Spec	MapR Core	OS for Kubernetes Nodes	OpenShift	HPE Ezmeral Runtime Enterprise
1.0.x[1]	Loopback NFS	1.17 and later[2]	1.3.0	Release 6.1.0 and later[3]	• RHEL • CentOS • Ubuntu	4.4, 4.5, 4.6, 4.7[2], 4.8[2], 4.9, 4.10	5.3.x, 5.4
1.2.x[1]	FUSE	1.17 and later[2]	1.3.0	Release 6.1.0 and later[3]	• RHEL • CentOS • Ubuntu	4.4, 4.5, 4.6, 4.7[2], 4.8[2], 4.9, 4.10	5.3.x, 5.4
1.1.0	FUSE	1.16 and later	1.3.0	Release 6.1.0 and later[3]	• RHEL • CentOS • Ubuntu	4.2 and 4.3	5.0 and 5.1
1.0.2	FUSE	1.13 and later	1.0.0	Release 6.1.0 and later[3]	• RHEL • CentOS • Ubuntu	4.1 and 4.2	5.0 and 5.1
1.0.0	FUSE	1.13 and later	1.0.0	Release 6.1.0 and later[3]	• RHEL • CentOS • Ubuntu	4.1 and 4.2	5.0 and 5.1

[1]Supports Raw Block Volumes on page 670.

[2]If your environment uses Kubernetes 1.20+ and OpenShift 4.7+, you must use FUSE 1.2.1+ and Loopback NFS 1.0.1+.

[3]In a release 7.0.0 environment, you must use Fuse 1.2.7+ and Loopback NFS 1.0.7+.

JDK Support Matrix

This matrix shows the Java Development Kit versions supported by different MapR Data Platform releases.

last-updated: Apr 11, 2022

Only the Oracle and OpenJDK Java engines are supported.

JDK/ MapR Release	MapR 6.1.x	MapR 6.0.x	MapR 5.2.x	MapR 5.1.0	MapR 5.0.0	MapR 4.1.0	MapR 4.0.x	MapR 3.1.x	MapR 3.0.x
JDK 8	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
JDK 7	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
JDK 6	No	No	No	No	No	No	No	Yes	Yes

Change Control Notes

Updated Feb. 21, 2021 (added MapR 6.1.x).

Updated Sep. 27, 2018 (added MapR 6.1.0).

JDK / JRE Support
This page shows the JDK and JRE versions for which data-fabric releases provide build-time and run-time support.

JDK Build-Time Support

	Release		
JDK Version	7.0.0	6.2.0	6.1.0
JDK 11	Yes	Yes	No
Amazon Corretto 11	Yes	Yes	No
JDK Azul	No	No	No
JDK 8	No	No	Yes
JDK 7	No	No	No
JDK 6	No	No	No

JRE Support (Run-Time Support)

JRE Version	Release 7.0.0	Release 6.2.0	Release 6.1.x	Release 6.0.x	Release 5.2.x
Open JRE 11	Yes	Yes	No	Yes	No
(Oracle Sun) JRE 11	Yes (implied)	Yes (implied)	No	No	No
Amazon Corretto (11)	Yes	Yes	No	No	No
JRE Azul	No	No	No	No	No
JRE 8	No	No	Yes	Yes	Yes
JRE 7	No	No	No	No	Yes
JRE 6	No	No	No	No	No

Hadoop Protocol Versions for MapR Software

Shows the Hadoop RPC protocol version and compatible MapR client versions for each MapR release.

Each MapR Cluster version is associated with a Hadoop RPC protocol version. The JobTrackers or ResourceManagers in a given cluster accept only the jobs submitted from clients with a compatible protocol version.

The following table shows the Hadoop RPC protocol version, and compatible MapR client versions for each MapR release:

Core Version	EEP	Hadoop Version	RPC Protocol	Compatible with Data Fabric Client Version
7.0.0	8.1.0	2.7.6.200	9	7.0.0.0
	8.0.0	2.7.6.100	9	7.0.0.0
6.2.x	8.1.0	2.7.6.200	9	6.2.x, 6.1.x[3]
	8.0.0	2.7.6.100	9	6.2.x, 6.1.x[3]
	7.1.1[1]	2.7.5.0	9	6.2.x, 6.1.x[3]
	7.1.0[1]	2.7.5.0	9	6.2.x, 6.1.x[3]
	7.0.1	2.7.4.100	9	6.2.x, 6.1.x[3]
	7.0.0	2.7.4.0	9	6.2.x, 6.1.x[3]
6.1.x	N/A	2.7.0	9	6.1.0, 6.0.x, 5.2.x
6.0.x	N/A	2.7.0	9	6.0.0, 5.2.0, 5.1.0, 5.0.0, 4.1.0, 4.0.2, 4.0.1[4]
5.2.x	N/A	2.7.0	9[2]	5.2.0, 5.1.0, 5.0.0, 4.1.0, 4.0.2, 4.0.1[4]
5.1.0	N/A	N/A	9[2]	5.1.0, 5.0.0, 4.1.0, 4.0.2, 4.0.1[4]
5.0.0	N/A	N/A	9[2]	5.0.0, 4.1.0, 4.0.2, 4.0.1[4]
4.1.0	N/A	N/A	9[2]	4.1, 4.0.2, 4.0.1[4]
4.0.x	N/A	N/A	9[2]	4.0.x[5]
3.1.x	N/A	N/A	4	3.1.x
3.0.x	N/A	N/A	5	3.0.x, 2.1.x
2.1.x	N/A	N/A	5	3.0.x, 2.1.x
2.0.x	N/A	N/A	4	3.1.x, 2.0.x

[1]Release 6.2.0.5 and later.

[2]MapReduce version 1 and version 2 use the same protocol version.

[3]Release 6.1 client support is limited if applications submitting jobs to release 6.2 clusters are developed in JDK 1.8. Jars used by the client application should not use features made obsolete in JDK 11.

[4]A release 4.0.2 client can submit MRv2 jobs to a release 4.1, 5.0, or 5.1 cluster configured with zero-configuration ResourceManager failover (RM HA) as long as the client is also configured with zero-configuration RM HA.

[5]If you want to submit MapReduce v2 jobs to a release 4.0.x cluster configured with zero-configuration RMHA, you must use a release 4.0.2 client on a release 4.0.2 cluster.

MapR Client Support Matrix

This matrix shows which MapR releases are compatible with different MapR client OS versions.

For a list of the MapR client versions that are compatible with each release, see Hadoop Protocol Versions for MapR Software on page 5597.

last-updated: Apr 11, 2022

Client OS	Version	MapR 6.1.x	MapR 6.0.x	MapR 5.2.x	MapR 5.1.0	MapR 5.0.0	MapR 4.1.0	MapR 4.0.x	MapR 3.1.x	MapR 3.0.x
Windows 64-bit	10	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	7	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	2012	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	2008	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	8	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Windows 32-bit	7	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	2008	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	8	No	No	Yes	Yes	Yes	Yes	Yes	No	No
Mac OS	10.8.x	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Linux 64-bit	See Operating System Support Matrix on page 5522	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Change Control Notes

Updated January 8, 2021 (changed "N/A" entries to "No" entries).

Updated June 27, 2018 (added column for MapR 6.1.x).

MapR Installer Support Matrix

This matrix shows the operating systems that are supported by the MapR Installer.

OS Version Support

Release Version/ Installer Version	Installer 1.17.0.0 Supported	Installer 1.16.0.x Supported	Installer 1.15.0.x Supported	Installer 1.14.0.x Supported	Installer 1.13.0.0 Supported	Installer 1.12.0.0 Supported	Installer 1.11.0.0 Supported

last-updated: Apr 11, 2022

RHEL / CentOS (64bit)	8.4	Yes[1]	Yes[1]	No	No	No	No	No
	8.3	Yes	Yes	Yes	No	No	No	No
	8.2	Yes	Yes	Yes	Yes	No	No	No
	8.1	Yes	Yes	Yes	Yes	No	No	No
	7.9	Yes[1]	Yes[1]	Yes[1]	Yes[1]	No	No	No
	7.8	Yes	Yes	Yes	Yes	No	No	No
	7.7	Yes	Yes	Yes	Yes	Yes	No	No
	7.6	Yes	Yes	Yes	Yes	Yes	No	No
	7.5	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	7.4	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	7.3	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	6.10	No[2]	No[2]	No[2]	No[2]	No[2]	No[2]	Yes
	6.9	No[2]	No[2]	No[2]	No[2]	No[2]	No[2]	Yes
	6.8	No[2]	No[2]	No[2]	No[2]	No[2]	No[2]	Yes
	6.7	No[2]	No[2]	No[2]	No[2]	No[2]	No[2]	Yes
	6.6	No[2]	No[2]	No[2]	No[2]	No[2]	No[2]	Yes
	6.5	No[2]	No[2]	No[2]	No[2]	No[2]	No[2]	Yes
Oracle Enterprise Linux	8.2	Yes	Yes	Yes	No	No	No	No
	7.8	Yes[5]	Yes[5]	Yes[5]	Yes[5]	No	No	No
	7.4	No	No	No	No	No	No	No
	7.3	No	No	No	No	No	No	No
Ubuntu (64bit)	20.04	Yes	No	No	Yes	No	No	No
	18.04	Yes	Yes	Yes	Yes	No	No	No
	16.04[3]	No	Yes	Yes	Yes	Yes	Yes	Yes
	14.04[4]	No	No	No	Yes	Yes	Yes	Yes
	12.04	No	No	No	No	No	Yes	Yes
	11.04	No	No	No	No	No	No	No
SLES (64bit)	15 SP2	Yes	Yes	No	No	No	No	No
	12 SP5	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	12 SP3	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	12 SP2	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	12 SP1	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	11 SP3	No	No	No	No	Yes	Yes	Yes
	11 SP2	No	No	No	No	No	No	No
	11 SP1	No	No	No	No	No	No	No

[1]Supported on RHEL but not on CentOS.

last-updated: Apr 11, 2022

[2]The Installer only supports upgrades for this OS version. See Selecting an Installer Version to Use on page 5402.

[3]Before using Installer 1.14 on Ubuntu 16.04 nodes, you must manually install Java JDK 11. If you are using Installer 1.14 on RHEL/CentOS or SLES, the Installer installs Java JDK11 for you.

[4]Before using the Installer to install Release 6.0 or later on Ubuntu 14.04, you must upgrade to Java 1.8 on the cluster nodes. See IN-553 in Installer Known Issues.

[5]Requires Installer 1.14.0.1 or later. Installer 1.14.0.1 supports Oracle Enterprise Linux 7.8 only on release 6.1.0 and does not support ecosystem components for release 6.1.0.

MapR Version Support

Data Fabric Version/ Installer Version	Installer 1.17.0.0 Supported	Installer 1.16.0.x Supported	Installer 1.15.0.x Supported	Installer 1.14.0.0 Supported	Installer 1.13.0.0 Supported	Installer 1.12.0.0 Supported	Installer 1.11.0.0 Supported
6.2.0	Yes	Yes	Yes	Yes	No	No	No
MapR 6.1.1	Yes*	Yes*	Yes*	Yes	Yes	Yes	Yes
MapR 6.1.0**	No	No	No	No	No	No	No
MapR 6.0.1	Yes	Yes	Yes	Yes	Yes	Yes	Yes
MapR 6.0.0	Yes	Yes	Yes	Yes	Yes	Yes	Yes
MapR 5.2.x	No	No	No	No	No	No***	Yes
MapR 5.1	No	No	No	No	No	No***	Yes
MapR 5.0	No	No	No	No	No	No***	Yes

*The Installer does not support new installations of release 6.1.0 on RHEL / CentOS 8.1.

**New installations of release 6.1.0 are no longer supported. Install release 6.1.1 or release 6.2.0 instead.

***The Installer only supports upgrades for this OS version. See Selecting an Installer Version to Use on page 5402.

Change Control Notes

- Updated October 15, 2021 (added Installer 1.17.0.0 information).
- Updated October 1, 2021 (added RHEL 8.4 support).
- Updated August 29, 2021 (added Installer 1.16.0.x information).
- Updated April 16, 2020 (added Installer 1.16.0.0 information).
- Updated February 20, 2020 (added Installer 1.15.0.1 and release 6.1.1 information).
- Updated January 16, 2020 (added Installer 1.15.0.0 information and support for Oracle Enterprise Linux 8.2).
- Updated October 28, 2020 (added support for RHEL 7.9 and removed column for Installer 1.8.0).
- Updated October 16, 2020 (added support for RHEL / CentOS 8.2 and Oracle Enterprise Linux 7.8).
- Updated September. 8, 2020 (Installer 1.14.0.0 added).

last-updated: Apr 11, 2022

- Updated June 17, 2020 (rows added for RHEL / CentOS 7.8). Changed RHEL / CentOS 6.x values from Yes to No where only upgrades are supported.
- Updated November 30, 2019 (Installer 1.13.0.0 added, rows for RHEL / CentOS 7.7 and Ubuntu 18.04 added).
- Updated May 1, 2019 (Installer 1.12.0.0 added, rows for RHEL / CentOS 7.0, 7.1, and 7.2 removed).
- Updated January 2, 2019 (Installer 1.11.0.0 added).
- Updated Jul 27, 2018 (Installer 1.10.0 added).
- Updated March 30, 2018 (Installer 1.9.0 added). Corrected support for Ubuntu 11.04.
- Updated January 7, 2018 (Installer 1.8.0 added).
- Updated September 8, 2017 (Installer 1.7 added, release 6.0.x added).
- Updated July 31, 2017 (Installer 1.6 added, SLES 12 SP2 support added, older installer information removed).
- Updated April 6, 2017 (Installer 1.5 added, RHEL / CentOS 7.3 added, SLES 12 SP1 added, releases 2.x, 3.x, and 4.x removed).
- Updated December 8, 2016 (Installer 1.4 added).
- Updated August 19, 2016 (Installer 1.3 and release 5.2 added).
- Updated February 5, 2016 (Installer 1.2 and release 5.1 added).

MapR Installer EEP Support

This matrix shows which MapR Installer versions support each MapR Ecosystem Pack (EEP) version.

MapR Installer EEP Support

To understand which EEP versions are supported with different versions of data-fabric core, see EEP Support and Lifecycle Status on page 5531.

EEP	Installer 1.17.0.x	Installer 1.16.0.x	Installer 1.15.0.x	Installer 1.14.0.0	Installer 1.13.0.0	Installer 1.12.0.0	Installer 1.11.0.0	Installer 1.10.0	Installer 1.9.0
8.x.y	Yes	No	No	No	No	No	No	No	No
7.x.y	Yes	Yes	Yes	Yes	No	No	No	No	No
6.x.y	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
5.x.y	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
4.x.y	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
3.x.y	No	No	No	No	No	No	Yes	Yes	Yes
2.x.y	No	No	No	No	No	No	Yes	Yes	Yes
1.x.y	No	No	No	No	No	No	Yes	Yes	Yes

For a list of released Installer versions, see MapR Installer Updates on page 5481.

MapR Security Support Matrix

The tables in this section show component support for authentication, impersonation, and wire-level encryption.

Information is provided for the MapR 6.1 and MapR 6.0.1 releases. See these sections:

- Table 1 - Authentication in MapR 6.1 on page 5603
- Table 2 - Impersonation and Wire-Level Encryption in MapR 6.1 on page 5609
- Table 3 - Authentication in MapR 6.0.1 on page 5614
- Table 4 - Impersonation and Wire-Level Encryption in MapR 6.0.1 on page 5619

Tables 1 and 3 show component support for authentication using MapR-SASL, Kerberos, and PAM.

Tables 2 and 4 show component support for impersonation and wire-level encryption.

Table Symbols

The tables in this section use dashes to indicate non-support and directional arrows to convey inbound and outbound communication:

- A dash (—) indicates that the feature is currently not supported, not needed, or not applicable.
- A right arrow (A ⇾ B) means OUTBOUND from A and INBOUND to B.
- A double arrow (A ⇾⇽ B) means OUTBOUND from A and INBOUND to B, and vice versa.
- No arrow indicates OUTBOUND communication from the subcomponent to all components with which it communicates.

Authentication in MapR 6.1

Table

Main Component	Subcomponent	Authentication		
		MapR-SASL	Kerberos	PAM[1]
CORE COMPONENTS				
Data Fabric for Kubernetes	N/A	—	—	—
FUSE POSIX Client	N/A	—	—	—
JobClient to Resource Manager	N/A	Yes	Yes	—
MapR Installer	N/A	—	—	Yes

Table (Continued)

Main Component	Subcomponent	Authentication		
		MapR-SASL	Kerberos	PAM[1]
MapR File System	FileClient C ⅓ MapR File System	Yes	—	—
	FileClient Java ⅓ MapR File System	Yes	Yes[2]	—
	MapR File System ⅓ MapR File System[3]	Yes	—	—
	CLDB ⅓ MapR File System[4]	Yes	—	—
	FileClient ⅓ CLDB[4]	Yes	Yes[2]	—
	NFSv3 ⅓ MapR File System	Yes	—	—
	NFSv3 ⅓ CLDB[5]	Yes	—	—
MapR Database	MapR Database Java Client ⅓ MapR Database[6]	Yes	Yes[2]	—
	MapR Database C Client ⅓ MapR Database[6]	Yes	—	—
	AsyncHBase Client ⅓ MapR Database[6]	Yes	Yes[2]	—
	Hive Job Using Connector to MapR Database[6]	Yes	—	—
	Spark Job Using Connector to MapR Database[6]	Yes	—	—
	Client ⅓ HBase Thrift Gateway[6]	—	—	Yes
	HBase Thrift Gateway for MapR Database (Binary)[7]	Yes	—	—
	Client ⅓ Data Access Gateway	—	—	Yes
	Data Access Gateway ⅓ MapR Database (JSON)	Yes	—	—
	Client ⅓ HBase REST Gateway	—	Yes	Yes
	HBase REST Gateway for MapR Database (Binary)	Yes	—	—

last-updated: Apr 11, 2022

Table (Continued)

Main Component	Subcomponent	Authentication		
		MapR-SASL	**Kerberos**	**PAM[1]**
MapR-Streams	Java Client ⅓ MapR Event Store For Apache Kafka	Yes	—	—
	librdkafka C/C#/ Python Client ⅓ MapR Event Store For Apache Kafka	Yes	—	—
	Client Kafka REST Gateway	—	—	Yes
	Kafka REST Gateway ⅓ MapR Event Store For Apache Kafka	Yes	—	—
	REST Client ⅓ Kafka Connect Gateway	—	—	—
	Kafka Connect Gateway ⅓ MapR Event Store For Apache Kafka	Yes	—	—
MapR Control System[8]	Control System CLI Command	Yes	Yes	—
	MCS Web Command (REST Interface)	—	Yes	Yes
NFSv3	N/A	—	—	—
NFSv4	N/A	—	Yes	—
ZooKeeper[9]	ZK client ½ ZK server	Yes	—	—
	ZK server ½ ZK server	Yes	—	—
BUNDLED CLIENTS[10]				
Data Science Refinery (DSR)	N/A	—	—	—
Persistent Application Client Container (PACC)	N/A	—	—	—
ECOSYSTEM COMPONENTS				

Table (Continued)

Main Component	Subcomponent	Authentication		
		MapR-SASL	Kerberos	PAM[1]
Drill[11]	Web client ⅓ Drillbit	—	Partial (using SPNEGO WIP)	Yes
	Drillbit ⅓ Drillbit	Yes	Yes	—
	Java/C++ Client/ JDBC/ODBC ⅓ Drillbit	Yes	Yes	Yes
	Drill ⅓ Hive Storage Plugin	Yes	—	—
Flume[12]	Thrift Client ⅓ Flume Agent	Yes	Yes	—
	Avro Client ⅓ Flume Agent (Netty)	—	—	—
	Flume Agent ⅓ MapR Streams	Yes	—	—
	Flume Agent ⅓ MapR Database	Yes	—	—
	Flume Agent ⅓ Hive Metastore	Yes	Yes	—
HBase	Client ⅓ HBase Thrift Gateway	Yes	Yes	Yes
	Client ⅓ HBase REST Gateway	Yes	Yes	Yes
	Hue ⅓ HBase Thrift	Yes	Yes	Yes
Hive	HiveServer2 ⅓ Metastore	Yes	Yes	—
	JDBC Client ⅓ HiveServer2	Yes	Yes	Yes
	ODBC Client ⅓ HiveServer2	—	Yes	Yes
	WebHCat ⅓ Metastore	—	Yes	—
	Hive Shell ⅓ MetaStore	Yes	Yes	—
	Beeline ⅓ HiveServer2	Yes	Yes	Yes
	Client (Browser) ⅓ HiveServer2 Web UI Server	—	—	Yes
	REST Client ⅓ WebHCat	—	Yes	—
HttpFS	Client (REST) ⅓ HttpFS	—	Yes	Yes
	HttpFS ⅓ MapR File System	Yes	—	—

last-updated: Apr 11, 2022

Table (Continued)

Main Component	Subcomponent	Authentication		
		MapR-SASL	Kerberos	PAM[1]
Hue	Hue ⅓ Oozie[13]	Yes	Yes	—
	Hue ⅓ YARN	Yes	Yes	—
	Hue ⅓ HbaseThrift	Yes	Yes	—
	Hue ⅓ Sqoop2	Yes	Yes	—
	Hue ⅓ HttpFS	Yes	Yes	—
	Hue ⅓ HiveServer2	Yes	Yes	Yes
	Hue ⅓ Livy Server	Yes	Yes	Yes
KSQL	KSQL ⅓ MapR Event Store For Apache Kafka (Java client)	—	—	—
	KSQL ⅓ Kafka Streams	—	—	—
Kafka Streams	Kafka Streams ⅓ MapR Event Store For Apache Kafka (Java client)	—	—	—
Livy	REST Client ⅓ Livy Server	Yes	Yes	Yes
S3 gateway	S3 gateway ⅓ MapR File System (via POSIX client)	Yes	—	—
	S3 gateway ⅓ MapR Event Store For Apache Kafka (via librdkafka)	Yes	—	—
	S3 gateway Client ⅓ S3 gateway (self-managed via credentials file)	—	—	—
Oozie	Oozie Client, REST API, Hue ⅓ Oozie Server[14]	Yes	Yes	—
	Oozie Server ⅓ Spark/Sqoop[15]	Yes	Yes	—
	Oozie Server ⅓ Beeline-HS2	Yes	Yes	Yes
	Oozie Server ⅓ Hive	Yes	Yes	—

last-updated: Apr 11, 2022

Table (Continued)

Main Component	Subcomponent	Authentication		
		MapR-SASL	Kerberos	PAM[1]
Spark	Web Clients ⅓ Spark Component UI	No, but uses Spark's shared secret with DIGEST-MD5		
	Spark Driver ⅓ Executor	No, but uses Spark's shared secret with DIGEST-MD5		
	Spark Job Using Connector ⅓ MapR Database	Yes	—	—
	Spark Job Using Connector ⅓ MapR-Streams	Yes	Yes	—
	JDBC Client ⅓ Spark Thrift Server	Yes	Yes	Yes
	ODBC Client ⅓ Spark Thrift Server	–	Yes	Yes
Sqoop 2[16]	REST API, Hue, Sqoop 2 Client ⅓ Sqoop 2 Server	Yes	Yes	—
YARN	REST/Browser ⅓ RM/JHS/ATS	–	Yes	Yes
	Internal communication (RM/NM/JHS)	Yes	Yes	—
	Containers ⅓ YARN Services (RM/NM)	No, but uses YARN's shared secret with DIGEST-MD5		
	Timeline Server	Yes	Yes	—

[1]If LDAP is required, LDAP can be supported through PAM.

[2] Kerberos support is provided by implicit conversion of Kerberos tickets to MapR tickets.

[3]Payload not encrypted by default.

[4]All data exchanged with CLDB is in protobufs only and hence encrypted in secure clusters.

[5]Only admin ops to CLDB are audited. NFSv3 communication with CLDB is usually not admin-related.

[6]Accessed through the MapR client, which reads security settings from `/opt/mapr/conf/mapr-clusters.conf`; hence, this interface follows the secure-by-default model.

[7]MapR-SASL is supported but not enabled during installation.

[8]The MapR Control System is secure between client and webserver (API Server). The server may invoke other commands through the `maprcli` interface that themselves do not use secure communication.

[9]MapR uses MapR-SASL for communication with ZooKeeper.

[10]Includes a FUSE POSIX client, YARN client, and other client components.

[11]Support for Kerberos has not been verified, but SPNEGO can be used in conjunction with HTTPS.

[12]Flume agents can't be started automatically after installation. Manual configuration is required.

[13]Auditing user administration operations with Hue.

[14]A custom authentication filter can be configured.

[15]Oozie orchestrates Spark/Sqoop jobs using the Spark/Sqoop native client, so security is the same as Spark/Sqoop.

[16]SSL added to Sqoop 1.99.7. Basic access authentication is enabled by default.

Impersonation and Wire-Level Encryption in MapR 6.1

Table

Main Component	Subcomponent	Impersonation	Wire-Level Encryption		
			MapR-SASL	Kerberos	SSL/TLS
CORE COMPONENTS					
Data Fabric for Kubernetes	N/A	—	—	—	—
FUSE POSIX Client	N/A	—	—	—	—
JobClient to Resource Manager	N/A	Yes	Yes	Yes	—
MapR Installer	N/A	—	—	—	Yes
MapR File System	FileClient C ⅓ MapR File System	Yes	Yes	—	—
	FileClient Java ⅓ MapR File System	Yes	Yes	—	—
	MapR File System ⅓ MapR File System	—	Yes	—	—
	CLDB ⅓ MapR File System	—	Yes	—	—
	FileClient ⅓ CLDB	Yes	Yes	—	—
	NFSv3 ⅓ MapR File System	Yes	Yes	—	—
	NFSv3 ⅓ CLDB	Yes	Yes	—	—

last-updated: Apr 11, 2022

Table (Continued)

Main Component	Subcomponent	Impersonation	Wire-Level Encryption		
			MapR-SASL	Kerberos	SSL/TLS
MapR Database	MapR Database Java Client ⅓ MapR Database	Yes	Yes	—	—
	MapR Database C Client ⅓ MapR Database	Yes	Yes	—	—
	AsyncHBase Client ⅓ MapR Database	Yes	Yes	—	—
	Hive Job Using Connector to MapR Database	Yes	Yes	—	—
	Spark Job Using Connector to MapR Database	Yes	Yes	—	—
	Client ⅓ HBase Thrift Gateway	—	—	—	Yes
	HBase Thrift Gateway for MapR Database (Binary)	Yes	Yes	—	—
	Client ⅓ Data Access Gateway	—	—	—	Yes
	Data Access Gateway ⅓ MapR Database (JSON)	Yes	Yes	—	—
	Client ⅓ HBase REST Gateway	—	—	—	Yes
	HBase REST Gateway for MapR Database (Binary)	Yes	Yes	—	—

Table (Continued)

Main Component	Subcomponent	Impersonation	Wire-Level Encryption		
			MapR-SASL	Kerberos	SSL/TLS
MapR-Streams	Java Client ⅓ MapR Event Store For Apache Kafka	Yes	Yes	—	—
	librdkafka C/C#/ Python Client ⅓ MapR Event Store For Apache Kafka	—	Yes	—	—
	Client ⅓ Kafka REST Gateway	—	—	—	Yes
	Kafka REST Gateway ⅓ MapR Event Store For Apache Kafka	Yes	Yes	—	—
	REST Client ⅓ Kafka Connect Gateway	—	—	—	—
	Kafka Connect Gateway ⅓ MapR Event Store For Apache Kafka	—	Yes	—	—
MapR Control System	MCS CLI Command	—	Yes	—	—
	MCS Web Command (REST Interface)	—	—	—	Yes
NFSv3	N/A	—	—	—	—
NFSv4	N/A	—	—	Yes	—
ZooKeeper	ZK client ⅓ ZK server	—	Yes	—	—
	ZK server ⅓ ZK server	—	—	—	—
BUNDLED CLIENTS[1]					
Data Science Refinery (DSR)	N/A	—	—	—	—
Persistent Application Client Container (PACC)	N/A	—	—	—	—
ECOSYSTEM COMPONENTS					
Drill	Web client ⅓ Drillbit	Yes	—	—	Yes
	Drillbit ⅓ Drillbit	Yes	Yes	Yes	—
	Java/C++ client ⅓ Drillbit	Yes	Yes	Yes	Yes
	Drill ⅓ Hive storage plugin	Yes	Yes	—	—

last-updated: Apr 11, 2022

Table (Continued)

Main Component	Subcomponent	Impersonation	Wire-Level Encryption		
			MapR-SASL	Kerberos	SSL/TLS
Flume	Thrift Client ⅓ Flume Agent	Yes	Yes	—	Yes
	Avro Client ⅓ Flume Agent (Netty)	—	—	—	Yes
	Flume Agent ⅓ MapR Streams	—	Yes	—	—
	Flume Agent ⅓ MapR Database	—	Yes	—	—
	Flume Agent ⅓ Hive Metastore	—	Yes	Yes	—
HBase	Client ⅓ HBase Thrift Gateway	Yes	Yes	Yes	Yes
	Client ⅓ HBase REST Gateway	Yes	—	—	Yes
	Hue ⅓ HBase Thrift	Yes	—	—	Yes
Hive	HiveServer2 ⅓ Metastore	Yes	Yes	Yes	Yes
	JDBC Client ⅓ HiveServer2	Yes	Yes	Yes	Yes
	ODBC Client ⅓ HiveServer2	Yes	—	Yes	Yes
	WebHCat ⅓ Metastore	Yes	—	Yes	—
	Hive Shell ⅓ MetaStore	Yes	Yes	Yes	—
	Beeline ⅓ HiveServer2	Yes	Yes	Yes	Yes
	Client (Browser) ⅓ HiveServer2 Web UI Server	—	—	—	Yes
	REST Client ⅓ WebHCat	Yes	—	Yes	—
HttpFS	Client (REST) ⅓ HttpFS	Yes	—	—	Yes
	HttpFS ⅓ MapR File System	Yes	Yes	—	—

last-updated: Apr 11, 2022

Table (Continued)

Main Component	Subcomponent	Impersonation	Wire-Level Encryption		
			MapR-SASL	Kerberos	SSL/TLS
Hue	Hue → Oozie	Yes	—	—	Yes
	Hue → YARN	Yes	—	—	Yes
	Hue → HBaseThrift	Yes	Yes	Yes	Yes
	Hue → Sqoop2	Yes	—	—	Yes
	Hue → HttpFS	Yes	—	—	Yes
	Hue → HiveServer2	Yes	Yes	Yes	Yes
	Hue → Livy Server	Yes	—	—	Yes
KSQL	KSQL → MapR Event Store For Apache Kafka (Java client)	—	—	—	—
	KSQL → Kafka Streams	—	—	—	—
Kafka Streams	Kafka Streams → MapR Event Store For Apache Kafka (Java client)	—	—	—	—
Livy	REST Client → Livy Server	Yes	—	—	Yes
S3 gateway	S3 gateway → MapR File System (via POSIX client)	Yes	Yes	—	—
	S3 gateway → MapR Event Store For Apache Kafka (via librdkafka)	—	Yes	—	—
	S3 gateway → S3 gateway (self-managed via credentials file)	—	—	—	Yes
Oozie	Oozie client, REST API, Hue → Oozie Server	Yes	Yes	Yes	Yes
	Oozie Server → Spark/Sqoop	Yes	Yes	Yes	—
	Oozie Server → Beeline-HS2	Yes	Yes	Yes	Yes
	Oozie Server → Hive	Yes	Yes	Yes	—

last-updated: Apr 11, 2022

Table (Continued)

Main Component	Subcomponent	Impersonation	Wire-Level Encryption		
			MapR-SASL	Kerberos	SSL/TLS
Spark	Web clients ⅓ Spark Component UI	—	—	—	Yes
	Spark Driver ⅓ Executor	—	When running Spark-on-YARN, Driver-To-Executor communication is through YARN (Hadoop protocol), so it is fully secured.		
	Spark Job Using Connector ⅓ MapR Database	—	Yes	—	—
	Spark Job Using Connector ⅓ MapR Event Store For Apache Kafka	—	Yes	—	Yes
Sqoop 2	REST API, Hue, Sqoop 2 Client ⅓ Sqoop 2 Server	Yes	Yes	Yes	Yes
Tez	Browser ⅓ Tez UI	—	—	—	Yes
	Tez UI ⅓ YARN RM	—	—	—	Yes
	Tez UI ⅓ Timeline Server	—	—	—	Yes
	Tez Containers ⅓ YARN ShuffleHandler Service	—	—	—	Yes
YARN	REST/Browser ⅓ RM/JHS/ATS	Yes	—	—	Yes
	Internal communication (RM/NM/JHS)	—	Yes	Yes	—
	Containers ⅓ YARN Services (RM/NM)	—	Yes	Yes	—
	Timeline Server	—	Yes	Yes	—

[1]Includes a FUSE POSIX client, YARN client, and other client components.

Authentication in MapR 6.0.1

Table

Main Component	Subcomponent	Authentication		
		MapR-SASL	Kerberos	PAM[1]
CORE COMPONENTS				
MapR Installer	N/A	—	—	Yes
JobClient to Resource Manager	N/A	Yes	Yes	—

last-updated: Apr 11, 2022

Table (Continued)

Main Component	Subcomponent	Authentication		
		MapR-SASL	Kerberos	PAM[1]
MapR File System	FileClient C ⅓ MapR File System	Yes	—	—
	FileClient Java ⅓ MapR File System	Yes	Yes[2]	—
	MapR File System ⅓ MapR File System[3]	Yes	—	—
	CLDB ⅓ MapR File System[4]	Yes	—	—
	FileClient ⅓ CLDB[4]	Yes	Yes[2]	—
	NFSv3 ⅓ MapR File System	Yes	—	—
	NFSv3 ⅓ CLDB[5]	Yes	—	—
MapR Database	MapR Database Java Client ⅓ MapR Database[6]	Yes	Yes[2]	—
	MapR Database C Client ⅓ MapR Database[6]	Yes	—	—
	AsyncHBase Client ⅓ MapR Database[6]	Yes	Yes[2]	—
	Hive Job Using Connector to MapR Database[6]	Yes	—	—
	Spark Job Using Connector to MapR Database[6]	Yes	—	—
	Client ⅓ HBase Thrift Gateway[6]	—	—	Yes
	HBase Thrift Gateway for MapR Database (Binary)[7]	Yes	—	—
	Client ⅓ Data Access Gateway	—	—	Yes
	Data Access Gateway ⅓ MapR Database (JSON)	Yes	—	—
	Client ⅓ HBase REST Gateway	—	Yes	Yes
	HBase REST Gateway for MapR Database (Binary)	Yes	—	—

last-updated: Apr 11, 2022

Table (Continued)

Main Component	Subcomponent	Authentication		
		MapR-SASL	**Kerberos**	**PAM[1]**
MapR-Streams	Java Client ⅓ MapR Event Store For Apache Kafka	Yes	Yes[2]	—
	librdkafka C/C#/ Python Client ⅓ MapR Event Store For Apache Kafka	Yes	—	—
	Client Kafka REST Gateway	—	—	Yes
	Kafka REST Gateway ⅓ MapR Event Store For Apache Kafka	Yes	—	—
	REST Client ⅓ Kafka Connect Gateway	—	—	—
	Kafka Connect Gateway ⅓ MapR Event Store For Apache Kafka	Yes	—	—
MapR Control System[8]	MCS CLI Command	Yes	Yes	—
	MCS Web Command (REST Interface)	—	Yes	Yes
ZooKeeper[9]	ZK client ½ ZK server	Yes	—	—
	ZK server ½ ZK server	—	—	—
ECOSYSTEM COMPONENTS				
Drill[10]	Web client ½ Drillbit	—	Partial (using SPNEGO WIP)	Yes
	Drillbit ½ Drillbit	Yes	Yes	—
	Java/C++ Client/ JDBC/ODBC ⅓ Drillbit	Yes	Yes	Yes
	Drill ⅓ Hive Storage Plugin	Yes	—	—

last-updated: Apr 11, 2022

Table (Continued)

Main Component	Subcomponent	Authentication		
		MapR-SASL	Kerberos	PAM[1]
Flume[11]	Thrift Client ⅓ Flume Agent	Yes	Yes	—
	Avro Client ⅓ Flume Agent (Netty)	—	—	—
	Flume Agent ⅓ MapR Streams	Yes	—	—
	Flume Agent ⅓ MapR Database	Yes	—	—
	Flume Agent ⅓ Hive Metastore	Yes	Yes	—
Hive	HiveServer2 ⅓ Metastore	Yes	Yes	—
	JDBC Client ⅓ HiveServer2	Yes	Yes	Yes
	ODBC Client ⅓ HiveServer2	—	Yes	Yes
	WebHCat ⅓ Metastore	Yes	Yes	—
	Hive Shell ⅓ MetaStore	Yes	Yes	—
	Beeline ⅓ HiveServer2	Yes	Yes	Yes
	REST API ⅓ WebHCat	—	Yes	—
HttpFS	Client (REST) ⅓ HttpFS	—	Yes	Yes
	HttpFS ⅓ MapR File System	Yes	—	—
Hue	Hue ⅓ Oozie[12]	Yes	Yes	—
	Hue ⅓ YARN	Yes	Yes	—
	Hue ⅓ HbaseThrift	Yes	Yes	—
	Hue ⅓ Sqoop2	Yes	Yes	—
	Hue ⅓ HttpFS	Yes	Yes	—
	Hue ⅓ HiveServer2	Yes	Yes	—
	Hue ⅓ Livy Server	—	—	—
Livy	REST Client ⅓ Livy Server	—	—	—

last-updated: Apr 11, 2022

Table (Continued)

Main Component	Subcomponent	Authentication		
		MapR-SASL	Kerberos	PAM[1]
Oozie	Oozie Client, REST API, Hue ⅓ Oozie Server[13]	Yes	Yes	—
	Oozie Server ⅓ Spark/Sqoop[14]	Yes	Yes	—
	Oozie Server ⅓ Beeline-HS2	Yes	Yes	Yes
	Oozie Server ⅓ Hive	Yes	Yes	—
Spark	Web Clients ⅓ Spark Component UI	No, but uses Spark's shared secret with DIGEST-MD5		
	Spark Driver ⅓ Executor	No, but uses Spark's shared secret with DIGEST-MD5		
	Spark Job Using Connector ⅓ MapR Database	Yes	—	—
	Spark Job Using Connector ⅓ MapR-Streams	Yes	Yes	—
Sqoop 2[15]	REST API, Hue, Sqoop 2 Client ⅓ Sqoop 2 Server	Yes	Yes	—
YARN	REST/Browser ⅓ RM/JHS/ATS	–	Yes	Yes
	Internal communication (RM/NM/JHS)	Yes	Yes	—
	Containers ⅓ YARN Services (RM/NM)	No, but uses YARN's shared secret with DIGEST-MD5		
	Timeline Server	Yes	Yes	—

[1]If LDAP is required, LDAP can be supported through PAM.

[2] Kerberos support is provided by implicit conversion of Kerberos tickets to MapR tickets.

[3]Payload not encrypted by default.

[4]All data exchanged with CLDB is in protobufs only and hence encrypted in secure clusters.

[5]Only admin ops to CLDB are audited. NFSv3 communication with CLDB is usually not admin-related.

[6]Accessed through the MapR client, which reads security settings from `/opt/mapr/conf/mapr-clusters.conf`; hence, this interface follows the secure-by-default model.

[7]MapR-SASL is supported but not enabled during installation.

[8]The MapR Control System is secure between client and webserver (API Server). The server may invoke other commands through the `maprcli` interface that themselves do not use secure communication.

[9]MapR uses MapR-SASL for communication with ZooKeeper.

[10]Support for Kerberos has not been verified, but SPNEGO can be used in conjunction with HTTPS.

[11]Flume agents can't be started automatically after installation. Manual configuration is required.

[12]Auditing user administration operations with Hue.

[13]A custom authentication filter can be configured.

[14]Oozie orchestrates Spark/Sqoop jobs using the Spark/Sqoop native client, so security is the same as Spark/Sqoop.

[15]SSL added to Sqoop 1.99.7. Basic access authentication is enabled by default.

Impersonation and Wire-Level Encryption in MapR 6.0.1

Table

Main Component	Subcomponent	Impersonation	Wire-Level Encryption		
			MapR-SASL	Kerberos	SSL/TLS
CORE COMPONENTS					
MapR Installer	N/A	—	—	—	Yes
JobClient to Resource Manager	N/A	Yes	Yes	Yes	—
MapR File System	FileClient C ⅓ MapR File System	Yes	Yes	—	—
	FileClient Java ⅓ MapR File System	Yes	Yes	—	—
	MapR File System ⅓ MapR File System	—	Yes	—	—
	CLDB ⅓ MapR File System	—	Yes	—	—
	FileClient ⅓ CLDB	Yes	Yes	—	—
	NFSv3 ⅓ MapR File System	Yes	Yes	—	—
	NFSv3 ⅓ CLDB	Yes	Yes	—	—

last-updated: Apr 11, 2022

Table (Continued)

Main Component	Subcomponent	Impersonation	Wire-Level Encryption		
			MapR-SASL	Kerberos	SSL/TLS
MapR Database	MapR Database Java Client ⅓ MapR Database	Yes	Yes	—	—
	MapR Database C Client ⅓ MapR Database	Yes	Yes	—	—
	AsyncHBase Client ⅓ MapR Database	Yes	Yes	—	—
	Hive Job Using Connector to MapR Database	Yes	Yes	—	—
	Spark Job Using Connector to MapR Database	Yes	Yes	—	—
	Client ⅓ HBase Thrift Gateway	—	—	—	Yes
	HBase Thrift Gateway for MapR Database (Binary)	Yes	Yes	—	—
	Client ½ Data Access Gateway	—	—	—	Yes
	Data Access Gateway ⅓ MapR Database (JSON)	Yes	Yes	—	—
	Client ⅓ HBase REST Gateway	—	—	—	Yes
	HBase REST Gateway for MapR Database (Binary)	Yes	Yes	—	—

last-updated: Apr 11, 2022

Table (Continued)

Main Component	Subcomponent	Impersonation	Wire-Level Encryption		
			MapR-SASL	Kerberos	SSL/TLS
MapR-Streams	Java Client ⅓ MapR Event Store For Apache Kafka	Yes	Yes	—	—
	librdkafka C/C#/ Python Client ⅓ MapR Event Store For Apache Kafka	—	Yes	—	—
	Client ⅓ Kafka REST Gateway	—	—	—	Yes
	Kafka REST Gateway ⅓ MapR Event Store For Apache Kafka	Yes	Yes	—	—
	REST Client ⅓ Kafka Connect Gateway	—	—	—	—
	Kafka Connect Gateway ⅓ MapR Event Store For Apache Kafka	—	Yes	—	—
MapR Control System	MCS CLI Command	—	Yes	—	—
	MCS Web Command (REST Interface)	—	—	—	Yes
ZooKeeper	ZK client ½ ZK server	—	Yes	—	—
	ZK server ½ ZK server	—	—	—	—
ECOSYSTEM COMPONENTS					
Drill	Web client ½ Drillbit	Yes	—	—	Yes
	Drillbit ½ Drillbit	Yes	Yes	Yes	—
	Java/C++ client ⅓ Drillbit	Yes	Yes	Yes	Yes
	Drill ⅓ Hive storage plugin	Yes	Yes	—	—

last-updated: Apr 11, 2022

Table (Continued)

Main Component	Subcomponent	Impersonation	Wire-Level Encryption		
			MapR-SASL	Kerberos	SSL/TLS
Flume	Thrift Client ⅓ Flume Agent	Yes	Yes	—	Yes
	Avro Client ⅓ Flume Agent (Netty)	—	—	—	Yes
	Flume Agent ⅓ MapR Streams	—	Yes	—	—
	Flume Agent ⅓ MapR Database	—	Yes	—	—
	Flume Agent ⅓ Hive Metastore	—	Yes	Yes	—
Hive	HiveServer2 ⅓ Metastore	Yes	Yes	Yes	Yes
	JDBC Client ⅓ HiveServer2	Yes	Yes	Yes	Yes
	ODBC Client ⅓ HiveServer2	Yes	—	Yes	Yes
	WebHCat ⅓ Metastore	Yes	—	Yes	—
	Hive Shell ⅓ MetaStore	Yes	Yes	Yes	—
	Beeline ⅓ HiveServer2	Yes	Yes	—	Yes
	REST API ⅓ WebHCat	Yes	—	Yes	—
HttpFS	Client (REST) ⅓ HttpFS	Yes	—	—	Yes
	HttpFS ⅓ MapR File System	Yes	Yes	—	—
Hue	Hue ⅓ Oozie	Yes	—	—	Yes
	Hue ⅓ YARN	Yes	—	—	Yes
	Hue ⅓ HBaseThrift	Yes	Yes	—	—
	Hue ⅓ Sqoop2	Yes	—	—	Yes
	Hue ⅓ HttpFS	Yes	—	—	Yes
	Hue ⅓ HiveServer2	Yes	Yes	Yes	Yes
	Hue ⅓ Livy Server	—	—	—	—
Livy	REST Client ⅓ Livy Server	—	—	—	—

Table (Continued)

Main Component	Subcomponent	Impersonation	Wire-Level Encryption		
			MapR-SASL	Kerberos	SSL/TLS
Oozie	Oozie client, REST API, Hue ⅓ Oozie Server	Yes	Yes	Yes	Yes
	Oozie Server ⅓ Spark/Sqoop	Yes	Yes	Yes	—
	Oozie Server ⅓ Beeline-HS2	Yes	Yes	—	Yes
	Oozie Server ⅓ Hive	Yes	Yes	Yes	—
Spark	Web clients ⅓ Spark Component UI	—	—	—	Yes
	Spark Driver ⅓ Executor	—	When running Spark-on-YARN, Driver-To-Executor communication is through YARN (Hadoop protocol), so it is fully secured.		
	Spark Job Using Connector ⅓ MapR Database	—	Yes	—	—
	Spark Job Using Connector ⅓ MapR Event Store For Apache Kafka	—	Yes	—	Yes
Sqoop 2	REST API, Hue, Sqoop 2 Client ⅓ Sqoop 2 Server	Yes	Yes	Yes	Yes
YARN	REST/Browser ⅓ RM/JHS/ATS	Yes	—	—	Yes
	Internal communication (RM/NM/JHS)	—	Yes	Yes	—
	Containers ⅓ YARN Services (RM/NM)	—	Yes	Yes	—
	Timeline Server	—	Yes	Yes	—

Release History for EEPs

This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

For EEP and core compatibility information, see EEP Support and Lifecycle Status on page 5531. For detailed EEP information, see EEP Release Notes on page 5658.

EEP	Release Date Identifier*	Release Date
8.1.0	2201	March 7, 2022
8.0.0	2110	November 1, 2021
7.1.1	2110	November 1, 2021

last-updated: Apr 11, 2022

EEP	Release Date Identifier*	Release Date
7.1.0	2104	June 2, 2021
7.0.1	2101	January 31, 2021
7.0.0	2009	September 18, 2020
6.3.6	2201	March 7, 2022
6.3.5	2110	November 1, 2021
6.3.4	2104	June 2, 2021
6.3.3	2103	March 29, 2021
6.3.2	2101	January 31, 2021
6.3.1	2009	September 18, 2020
6.3.0	1912	December 16, 2019
6.2.0	1904	May 30, 2019
6.1.1	1904	May 30, 2019
6.1.0	1901	February 6, 2019
6.0.2	1904	May 30, 2019
6.0.1	1901	February 6, 2019
6.0.0	1808	September 28, 2018
5.0.7	2104	June 2, 2021
5.0.6	2101	January 31, 2021
5.0.5	2009	September 18, 2020
5.0.4	1912	December 16, 2019
5.0.3	1904	May 30, 2019
5.0.2	1901	February 6, 2019
5.0.1	1808	September 28, 2018
5.0.0	1803	April 6, 2018
4.1.4	1904	May 30, 2019
4.1.3	1901	February 6, 2019
4.1.2	1808	September 28, 2018
4.1.1	1803	April 6, 2018
4.1.0	1801	February 2, 2018
4.0.0	1710	November 21, 2017
3.0.5	1901	February 6, 2019
3.0.4	1808	September 28, 2018
3.0.3	1803	April 6, 2018
3.0.2	1710	November 21, 2017
3.0.1	1707	August 2, 2017

last-updated: Apr 11, 2022

EEP	Release Date Identifier*	Release Date
3.0	1703	April 6, 2017
2.0.3	1710	November 21, 2017
2.0.2	1707	August 2, 2017
2.0.1	1703	April 6, 2017
2.0	1611	December 21, 2016
1.1.4	1710	November 21, 2017
1.1.3	1707	August 2, 2017
1.1.2	1703	April 6, 2017
1.1.1	1611	December 9, 2016
1.1.0	1609	September 29, 2016
1.0	1608	September 2, 2016

*The release date identifier is a four-digit number that appears in release notes and provides an approximate indication of the release date for a component. For example, in the string Hive 2.3.3 - 1808, 2.3.3 refers to the Hive version number, and 1808 indicates an August 2018 release. Note that last-minute changes in the release date can result in a slight mismatch between the release date identifier and the actual month of the release.

Ecosystem Support Matrix (Pre-5.2 releases)

This section provides support matrices for key ecosystem components for pre-5.2 releases.

The following table shows the compatibility of ecosystem products with MapR Converged Data Platform version 5.1 and below. For more recent versions of MapR, see the EEP Components and OS Support on page 5536.

Note: In the MapR Version column, JT stands for JobTracker.

Ecosystem Component/ MapR Version		MapR 3.0.x	MapR 3.1.x	MapR 4.0.1 (JT)	MapR 4.0.1 (YARN)	MapR 4.0.2 (JT)	MapR 4.0.2 (YARN)	MapR 4.1.0 (JT)	MapR 4.1.0 (YARN)	MapR 5.0.0 (JT)	MapR 5.0.0 (YARN)	MapR 5.1.0 (JT)	MapR 5.1.0 (YARN)
Apache MapReduce API	1.0.3	Yes	Yes	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A
	2.4.1	N/A	N/A	N/A	Yes	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	2.5.1	N/A	N/A	N/A	N/A	N/A	Yes	N/A	Yes	N/A	N/A	N/A	N/A
	2.7.0	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Yes	N/A	Yes

last-updated: Apr 11, 2022

Apache Hive	0.09	No	No	No	No	No	No	No	No	No	No	No	No
	0.10	Yes	No	No	No	No	No	No	No	No	No	No	No
	0.11	Yes	Yes	No	No	No	No	No	No	No	No	No	No
	0.12	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No
	0.13	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	0.14	No	No	No	No	No	No	No	No	No	No	No	No
	1.0	No	No	No	No	No	No	Yes	Yes	Yes	Yes	No	No
	1.2.1	No	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes
Apache Spark	0.9.1	Yes	Yes	No	No	No	No	No	No	No	No	No	No
	0.9.2	Yes	Yes	No	No	No	No	No	No	No	No	No	No
	1.0.2	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No
	1.1.0	No	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No
	1.2.1	No	No	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No
	1.3.1	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
	1.4.1	No	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
	1.5.2	No	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
	1.6.1	No	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes
Impala	1.1.1	Yes	No	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]
	1.2.3	Yes	Yes	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]
	1.4.1	No	No	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]
	2.2.0	No	No	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]
	2.5.0	No	No	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]	N/A[5]	N/A[6]
Apache Pig	10	Yes	No	No	No	No	No	No	No	No	No	No	No
	11	Yes	Yes	No	No	No	No	No	No	No	No	No	No
	12	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No
	13	No	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No
	14	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	15	No	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes
Apache Flume	1.3.1	Yes	No	No	No	No	No	No	No	No	N/A	No	N/A
	1.4.0	Yes	Yes	No	N/A	No	N/A	No	N/A	No	N/A	No	N/A
	1.5	No	Yes	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A	No	N/A
	1.6	No	No	No	No	No	No	Yes	N/A	Yes	N/A	Yes	N/A
Apache Sqoop	1.4.3	Yes	No	No	No	No	No	No	No	No	No	No	No
	1.4.4	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No
	1.4.5	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No
	1.4.6	No	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes

last-updated: Apr 11, 2022

Apache Sqoop2	1.99.0	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No
	1.99.3	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
	1.99.6	No	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes
Apache Mahout	0.7	Yes	No	No	No	No	No	No	No	No	No	No	No
	0.8	Yes	Yes	No	No	No	No	No	No	No	No	No	No
	0.9	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No
	0.10	No	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
	0.11	No	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
	0.12	No	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes
Apache Oozie	3.3.2	Yes	Yes	No	No	No	No	No	No	No	No	No	No
	4.0.0	Yes	Yes	No	No	No	No	No	No	No	No	No	No
	4.0.1	No	Yes[2]	Yes[2]	Yes[2]	Yes	Yes	Yes	Yes	No	No	No	No
	4.1.0	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	4.2.0	No	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes
Hue	2.5 Beta Only	Yes	Yes	No	No	No	No	No	No	No	No	No	No
	3.5	Yes	Yes	No	No	No	No	No	No	No	No	No	No
	3.6	No	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No
	3.7	No	Yes[1]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
	3.8.1	No	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
	3.9.0	No	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes
Apache HBase	0.92.2	No	No	No	No	No	No	No	No	No	No	No	No
	0.94.17	Yes	Yes	Yes	Yes	No	No	No	No	No	No	No	No
	0.94.21	Yes	Yes	Yes	Yes	No	No	No	No	No	No	No	No
	0.94.24	Yes	Yes	Yes	Yes	No	No	No	No	No	No	No	No
	0.98.4	No	No	Yes	Yes	Yes	Yes	No	No	No	No	No	No
	0.98.7	No	No	Yes	Yes	Yes	Yes	No	No	No	No	No	No
	0.98.9	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
	0.98.12	No	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
	1.1	No	No	No	No	No	No	No	No	No	No	Yes	Yes

last-updated: Apr 11, 2022

Apache Drill	1.0	No	Yes[3]	Yes	N/A	Yes	N/A	Yes	N/A	No	N/A	No	N/A
	1.1	No	Yes[3]	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A	No	N/A
	1.2	No	Yes[3]	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A	No	N/A
	1.3 dev preview	No	No	No	N/A	No	N/A	Yes	N/A	No	N/A	No	N/A
	1.4	No	Yes[3]	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A
	1.5 dev preview	No	Yes[3]	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A
	1.6	No	Yes[3]	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A
	1.7 dev preview	No	Yes[3]	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A
	1.8	No	Yes[3]	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A	Yes	Yes
AsyncHBase	1.4.1	Yes	Yes	No	No	No	No	No	No	No	No	No	No
	1.5	No	No	Yes	Yes	Yes	Yes	No	No	No	No	No	No
	1.6	No	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
	1.7	No	No	No	No	No	No	No	No	No	No	Yes	Yes
Cascading	2.1.6	Yes	Yes	No	No	No	No	No	No	No	No	No	No
	2.5	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
HTTPFS	1	Yes	Yes	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A
Apache Tez	0.4	N/A	N/A	N/A	Yes	N/A	Yes	N/A	Yes	N/A	No	No	No
(Developer Preview)	0.5.3	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Yes	N/A	No[4]	No[4]	No[4]
Storm	0.9.3	N/A	Yes	Yes	N/A	Yes	N/A	Yes	N/A	No	No	No	No
	0.9.4	N/A	Yes	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A	Yes	N/A
	0.10	No	No	No	No	No	No	No	No	No	No	Yes	N/A
Sentry	1.4.0	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
	1.6.0	No	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes
Apache Myriad	0.1	No	No	No	No	No	No	No	No	No	No	N/A	Yes

[1] Hue 3.7 with MapR 3.1.x does not work with MapR Security. Only CentOS 6.x and 7.x are supported.

[2] Please make sure to pick the latest patched Oozie 4.0.1 version from monthly release 1502 or later (Oozie Release Notes on page 6267).

[3] Drill 0.9 and later versions do not work with MapR Database in Version 3.1.x.

[4] The Tez development preview has been withdrawn.

[5] Impala does not use the JobTracker service, but can run in a cluster running JobTracker.

[6] You cannot run Impala as a YARN application, however Impala can run in a YARN cluster.

Change Control Notes

- This matrix was updated on February 8, 2017 (Apache MapReduce API correction).
- This matrix was updated on September 12, 2016 (Drill 1.8.0).
- This matrix was updated on April 4, 2016 (AsyncHBase 1.7).
- This matrix was updated on March 18, 2016 (Impala updates).
- This matrix was updated on February 29, 2016 (MapR 5.1 updates).
- This matrix was updated on Jan 20, 2016 (Drill 0.9 removed).
- This matrix was updated on Jan 12, 2016 (Drill 1.3, 1.4).
- This matrix was updated on Dec 21, 2015 (Spark 1.5.2).
- This matrix was updated on Oct 28, 2015 (Tez 0.5.3).
- This matrix was updated on Oct 4, 2015 (Flume 1.6 and Mahout 0.11).
- This matrix was updated on September 25, 2015 (Hive 1.2.1 and Oozie 4.2.0).
- This matrix was updated on September 9, 2015 (Pig 15 and Spark 1.4.1).
- This matrix was updated on August 4, 2015 (Hue 3.8.1 and Sqoop 2 1.99.6).
- This matrix was updated on July 24, 2015 (Sqoop and Sqoop2).

For interoperability among specific groups of ecosystem products, see the following subsections.

Drill Support Matrix
This matrix shows the interoperability between Drill and other ecosystem products.

For MapR Converged Data Platform releases beyond 5.1, you may also want to view EEP Components and OS Support on page 5536.

Compatible Product	Version	Drill 1.0	Drill 1.1	Drill 1.2	Drill 1.3 dev preview	Drill 1.4	Drill 1.5 dev preview	Drill 1.6	Drill 1.7 dev preview	Drill 1.8	Drill 1.9	Drill 1.10
MapR		4.x, 3.1.1	5.0, 4.x, 3.1.1	5.x, 4.x, 3.1.1	5.x, 4.x, 3.1.1	5.x, 4.x, 3.1.1	5.x, 4.x, 3.1.1	5.x, 4.x, 3.1.1	5.x, 4.x, 3.1.1	5.x, 4.x, 3.1.1	5.x, 4.x, 3.1.1	5.x
YARN		No	No	No	No	No	No	No	No	Yes[1]	Yes[1]	Yes[1]
JDK	6	No	No	No	No	No	No	No	No	No	No	No
	7	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	8	No	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes
Hive	12	No	No	No	No	No	No	No	No	No	No	No
	13	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
	1.0	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
	1.2.1	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

HBase	0.94.17, 0.94.21, 0.94.24	No	No	No	No	No	No	No	No	No	No	No
	0.98.7, 0.98.9	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
	0.98.12	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	1.1	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Note: As of the 1703 release of Drill 1.9 and 1.10, you can install Drill on RHEL, CentOS, Ubuntu, and SLES (with Open JDK 1.8 and Oracle JDK 1.7 or 1.8). Previous releases of Drill supported RHEL, CentOS, and Ubuntu platforms only.

[1] Drill running under YARN is supported on the MapR Converged Data Platform version 5.1 and 5.2 only.

Change Control Notes

- This matrix was updated April, 2017 (Drill 1.10-1703)
- This matrix was updated on December 7, 2016 (Drill 1.9-1611)
- This matrix was updated on September 29, 2016 (Drill 1.8-1609)
- This matrix was updated on September 12, 2016 (Drill 1.8)
- This matrix was updated on April 1, 2016 (Drill 1.6)
- This matrix was updated on February 28, 2016 (Drill 1.5)
- This matrix was updated on Jan 20, 2016 (Drill 0.x removed).
- This matrix was updated on Jan 12, 2016 (Drill 1.3, 1.4).

HBase Support Matrix

This matrix shows the interoperability between HBase and other ecosystem products for MapR versions 5.1 and below.

For more recent versions of MapR, see the EEP Components and OS Support on page 5536. Note that MapR 6.0.x and MapR 6.1 provide Apache HBase-compatible APIs and client interfaces but do not support HBase as a standalone ecosystem component.

Compatible Product	**Product Version**	**HBase 0.92.2**	**HBase 0.94.17**	**HBase 0.94.21**	**HBase 0.94.24**	**HBase 0.98.4**	**HBase 0.98.7**	**HBase 0.98.9**	**HBase 0.98.12**	**HBase 1.1**
Hive	See the Hive matrix.									
MapReduce	1.0.3	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	2.4.1	No	Yes	Yes	Yes	Yes	Yes	No	No	No
	2.5.1	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
	2.7	No	No	No	No	No	No	Yes	Yes	Yes

last-updated: Apr 11, 2022

AsyncHBase	1.4.1	Yes	Yes	Yes	Yes	No	No	No	No	No
	1.5.0	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
	1.6.0	No	No	No	No	No	No	Yes	Yes	No
	1.7.0	No	No	No	No	No	No	No	Yes	Yes
Hue	See the Hue matrix.									
Drill	1.0	No	No	No	No	Yes	Yes	Yes	No	No
	1.1	No	No	No	No	No	Yes	Yes	Yes	No
	1.2	No	No	No	No	No	Yes	Yes	Yes	No
	1.3	No	No	No	No	No	Yes	Yes	Yes	Yes
	1.4	No	No	No	No	No	Yes	Yes	Yes	Yes
	1.5	No	No	No	No	No	Yes	Yes	Yes	Yes
	1.6	No	No	No	No	No	Yes	Yes	Yes	Yes
	1.7	No	No	No	No	No	Yes	Yes	Yes	Yes
	1.8	No	No	No	No	No	Yes	Yes	Yes	Yes
	1.9	No	No	No	No	No	Yes	Yes	Yes	Yes
	1.10	No	No	No	No	No	No	No	Yes	Yes
Impala		No	1.2.3 only	1.2.3 only	1.2.3 only	1.4.1 only	1.4.1 only	1.4.1 only	1.4.1 only	2.2.0, 2.5.0
Spark-SQL	Spark-SQL goes through Hive to access HBase. See the Spark or Hive matrix pages.									
Flume	Flume uses the HBase client to use HBase. Flume supports all supported HBase clients.									
Storm	Storm uses the HBase client to use HBase. Storm supports HBase clients.									
Pig	13	No	Yes	Yes	Yes	No	No	No	No	No
	14	No	No	No	No	Yes	Yes	Yes	Yes	No
	15	No	No	No	No	No	No	Yes	Yes	Yes

Change Control Notes

- This matrix was updated April, 2017 (Drill 1.10-1703)
- This matrix was updated on April 4, 2016 (AsyncHBase 1.7).
- This matrix was updated on Feb 28, 2016 (Drill 1.5).
- This matrix was updated on Jan 20, 2016 (Drill 0.x removed).
- This matrix was updated on Jan 12, 2016 (Drill 1.3, 1.4).

Hive and HCatalog Support Matrix

This matrix shows the interoperability between Hive and HCatalog and other ecosystem products.

This page shows the compatibility of Hive and HCatalog with other ecosystem components and it applies to MapR versions 5.1 and below. For more recent versions of MapR, see the EEP Components and OS Support on page 5536.

Compatible Product	Product Version	Hive 0.09	Hive 0.10	Hive 0.11	Hive 0.12	Hive 0.13	Hive 1.0	Hive 1.2.1

last-updated: Apr 11, 2022

Hue	2.5	No	No	Yes	No	No	No	No
	3.5	No	No	No	No	Yes	No	No
	3.6	No	No	No	Yes	Yes	No	No
	3.7	No	No	No	Yes	Yes	Yes	No
	3.8.1	No	No	No	No	Yes	Yes	Yes
	3.9.0	No	No	No	No	Yes	Yes	Yes
HBase	0.92.2	Yes	Yes	No	No	No	No	No
	0.94.17	No	Yes	Yes	Yes	Yes	No	No
	0.94.21	No	Yes	Yes	Yes	Yes	No	No
	0.94.24	No	Yes	Yes	Yes	Yes	No	No
	0.98.4	No	No	No	No	Yes	No	No
	0.98.7	No	No	No	No	Yes	No	No
	0.98.9	No	No	No	No	Yes	Yes	Yes
	0.98.12	No	No	No	No	Yes	Yes	Yes
	1.1	No	No	No	No	No	No	Yes
Impala	1.1.1	No	No	No	Yes	No	No	No
	1.2.3	No	No	No	Yes	No	No	No
	1.4.1	No	No	No	No	Yes	No	No
	2.2.0	No	No	No	No	No	No	Yes
	2.5.0	No	No	No	No	No	No	Yes
Drill	1.0	No	No	No	No	Yes	No	No
	1.1	No	No	No	No	No	Yes	No
	1.2	No	No	No	No	Yes	Yes	Yes
	1.3	No	No	No	No	Yes	Yes	Yes
	1.4	No	No	No	No	Yes	Yes	Yes
	1.5	No	No	No	No	Yes	Yes	Yes
	1.6	No	No	No	No	Yes	Yes	Yes
	1.7	No	No	No	No	Yes	Yes	Yes
	1.8	No	No	No	No	Yes	Yes	Yes
	1.9	No	No	No	No	Yes	Yes	Yes
	1.10	No	No	No	No	No	No	Yes

last-updated: Apr 11, 2022

Spark	0.9.2	No	No	No	Yes	No	No	No
	1.0.2	No	No	No	Yes	No	No	No
	1.1.0	No	No	No	Yes	No	No	No
	1.2.1	No	No	No	No	Yes	No	No
	1.3.1	No	No	No	No	Yes	No	No
	1.4.1	No	No	No	No	Yes	No	No
	1.5.2	No	No	No	No	Yes[2]	Yes[2]	Yes
	1.6.1	No	No	No	No	Yes[2]	Yes[2]	Yes
Oozie	3.3.2	Yes	Yes	Yes	Yes	Yes	No	No
	4.0.1	No	No	No	Yes	Yes	No	No
	4.1.0	No	No	No	Yes[3]	Yes	Yes[3]	Yes[3]
	4.2.0	No	No	No	No	Yes[1]	Yes[1]	Yes
Sqoop	1.4.1	Yes	Yes	Yes	No	No	No	No
	1.4.4	No	No	No	Yes	Yes	No	No
	1.4.5	No	No	No	Yes	Yes	Yes	No
	1.4.6	No	No	No	No	Yes	Yes	Yes
Sqoop2	1.99.3	No	No	No	No	Yes	Yes	Yes
	1.99.6	No	No	No	No	Yes	Yes	Yes
Pig	11	No	No	No	No	No	No	No
	12	No	No	No	Yes	Yes	No	No
	13	No	No	No	Yes	Yes	No	No
	14	No	No	No	Yes	Yes	Yes	Yes
	15	No	No	No	No	Yes	Yes	Yes
Sentry	1.4	No	No	No	No	Yes	No	No
	1.6	No	No	No	No	No	No	Yes
Flume	1.6	No	No	No	No	Yes	Yes	Yes

[1] By default, Oozie 4.2.0 includes Hive 1.2.1 shared libraries. To use Oozie with other compatible versions of Hive, see MapR's Oozie documentation.

[2] When you use Spark 1.5.2 with Hive 0.13 or Hive 1.0, Spark SQL insert overwrite operations on Hive tables are not supported for the ORC, RC, and AVRO formats. For more information, see the Spark documentation.

[3] By default, Oozie 4.1.0 includes Hive 0.13 shared libraries. To use Oozie with other compatible versions of Hive, see MapR's Oozie documentation.

Change Control Notes

- This matric was last updated on April, 2017 (Drill 1.10 added)
- This matrix was last updated on February 29, 2016 (HBase 1.1, Drill 1.5, Impala 2.2, Sentry 1.6 added).
- This matrix was last updated on February 5, 2016 (Tez removed; Flume 1.6 added).

last-updated: Apr 11, 2022

- This matrix was last updated on Jan 20, 2016 (Drill 0.9 removed).
- This matrix was last updated on Jan 12, 2016 (Drill 1.3, 1.4).
- This matrix was last updated on Dec 21, 2015 (Spark 1.5.2).
- This matrix was last updated on Sept 25, 2015 (Hive 1.2.1 and Oozie 4.2.0).
- This matrix was updated on Sept 9, 2015 (Spark 1.4.1 and Pig 15).
- This matrix was updated on August 4, 2015 (Hue 3.8.1 and Sqoop2 1.99.6).
- This matrix was updated on November 20, 2015 (Hue 3.9.0).

Hue Support Matrix
This matrix shows the interoperability between Hue and other ecosystem products.

This page shows the versions of Hue that work other ecosystem components and it applies to MapR versions 5.1 and below. For more recent versions of MapR, see the EEP Components and OS Support on page 5536.

	Hue 2.5 Beta only	**Hue 3.5**	**Hue 3.6**	**Hue 3.7**	**Hue 3.8.1**	**Hue 3.8.1**	**Hue 3.9.0**
OS	CentOS[2] and Ubuntu	CentOS[2] and Ubuntu	CentOS[2] and Ubuntu	CentOS[2] and Ubuntu	CentOS[2] and Ubuntu	CentOS[2] and Ubuntu	CentOS[2] and Ubuntu
MapR Distribution	3.0.2, 3.1.0	3.0.3, 3.1.1	3.1.1, 4.0.x	3.1.1, 4.0.x, 4.1, 5.0	4.1	5.1.0, 5.0.0	5.1.0, 5.0.0
Hive	0.12	0.13	0.12, 0.13	0.12, 0.13, 1.0	0.13, 1.0	0.13, 1.0[6], 1.2.1	0.13, 1.0[6], 1.2.1
Impala	N/A	N/A	1.4.1 (Hive 0.13 only)	1.4.1 (Hive 0.13 only)	1.4.1 (Hive 0.13 only)	1.4.1 (Hive 0.13 only)	1.4.1 (Hive 0.13 only), 2.2.0 (Hive 1.2.1 only)
httpfs	1	1	1	1	1	1	1
Oozie	3.3.2	4.0.0	4.0.1	4.0.1, 4.1.0	4.1.0	4.1.0, 4.2.0	4.1.0, 4.2.0
Pig	11	12	12	12[3]	12[3]	12[3]	12[3]
HBase	No	0.94.17, 0.94.21, 0.98.7	0.94.17, 0.94.21, 0.98.7	0.94.x[1], 0.98.7, 0.98.9, 0.98.12	0.98.x	0.98.x	0.98.x, 1.1
MapR Database	No	Yes	Yes	Yes	Yes	Yes	Yes
Sentry	N/A	N/A	No	No	No	No	1.6
Solr	N/A	No	No	No	No	No	No
Spark[5]	N/A	N/A	N/A	N/A	1.3.1	1.3.1	1.3.1, 1.4.1,1.5.2, 1.6.1
Sqoop2	No	No	1.99.3	1.99.3	N/A[4]	1.99.6	1.99.6

[1] Loading examples with HBase 94.x do not work (because of the difference in Thrift version between HBase 94 and Hue).

[2] Not supported on CentOS 5.x.

[3] Pig jobs in Hue are run through Oozie. Oozie 4.0.1 and 4.1.0 bundle Pig 0.12 by default.

[4] On MapR 4.1, Hue 3.8.1 does not work with Sqoop2. To use Sqoop2 and Hue on MapR 4.1, consider using Hue 3.7.

[5] The Spark Notebook UI in Hue is a Beta feature.

[6]Hue 3.8/3.9 and Hive 1.0 require MapR 5.0.0. (This combination is not supported on MapR 5.1.0.)

Change Control Notes

- This matrix was last updated on February 29, 2016 (MapR 5.1 updates).
- This matrix was last updated on September 25, 2015 (Hue 3.8.1).
- This matrix was updated on August 4, 2015 (Hue 3.8.1).

Impala Support Matrix
This matrix shows the interoperability between Impala and other ecosystem products.

This page shows the versions of Impala that work with various versions of other ecosystem products and it applies to MapR versions 5.1 and below. For more recent versions of MapR, see the EEP Components and OS Support on page 5536.

Note: Impala 2.2.0 is supported on MapR 5.x. Impala 2.5.0 is supported on MapR 5.1 and later.

Product	Product Version	Impala 1.1.1	Impala 1.2.3	Impala 1.4.1	Impala 2.2.0	Impala 2.5.0
OS		RHEL/ CentOS6.x, Ubuntu 12 only	RHEL/ CentOS6.x, Ubuntu 12 only	RHEL/ CentOS6.x, Ubuntu 12 only	RHEL/CentOS 6.5, 6.6, 7.0,7.1 only (no Ubuntu, no SLES)	RHEL/CentOS 6.5, 6.6, 6.7, 6.8, 7.0, 7.1 only (no Ubuntu, no SLES)
Hive	12	Yes	Yes	No	No	No
	13	No	No	Yes	No	No
	1.2	No	No	No	Yes	Yes
Hue	2.5	No	No	No	No	No
	3.5	No	No	No	No	No
	3.6	No	No	Yes	No	No
	3.7	No	No	Yes	No	No
	3.8.1	No	No	Yes	No	No
	3.9.0	No	No	Yes	Yes	Yes
Sentry	1.4	No	No	Yes	No	No
	1.6	No	No	No	Yes	Yes
HBase		No	No	0.98 only	0.98.12, 1.1	1.1.x

Change Control Notes

- This matrix was last updated on July 7, 2016 (Impala 2.5.0).
- This matrix was last updated on February 29, 2016 (Impala 2.2.0).
- This matrix was last updated on September 25, 2015 (Hive 1.2.1).
- This matrix was updated on August 4, 2015 (Hue 3.8.1).

last-updated: Apr 11, 2022

Oozie Support Matrix

This matrix shows the interoperability between Oozie and other ecosystem products.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

This page shows the versions of Oozie that work with other ecosystem components and it applies to MapR versions 5.1 and below. For more recent MapR versions, see the EEP Components and OS Support on page 5536.

Compatible Product	Oozie 4.1.0	Oozie 4.2.0
Hive		0.13[1], 1.0[1], 1.2.1
Hue	3.8.1, 3.9.0	3.8.1, 3.9.0
Pig		0.14, 0.15
Spark	1.5.2, 1.6.1	1.3.1, 1.4.1, 1.5.2, 1.6.1
Sqoop		1.4.6

[1] By default, Oozie 4.2 includes Hive 1.2.1 shared libraries. To use Oozie with other compatible versions of Hive, see the MapR Oozie documentation.

Change Control Notes

- This matrix was last updated on February 5, 2016 (Oozie 4.1.0 column added).

Spark Support Matrix

This matrix shows the interoperability between Spark and other ecosystem products.

This page shows the version of Spark that works with other ecosystem components, and it applies to MapR versions 5.1 and below. See also Hive and HCatalog Support Matrix on page 5631. For more recent versions of MapR, see the EEP Components and OS Support on page 5536.

Compatible Product	Product Version	Spark 0.9.2	Spark 1.0.2	Spark 1.1	Spark 1.2.1	Spark 1.3.1	Spark 1.4.1	Spark 1.5.2	Spark 1.6.1
HBase	0.92.2	No	No	No	No	No	No	No	No
	0.94.17	Yes	Yes	Yes	No	No	No	No	No
	0.94.21	Yes	Yes	Yes	No	No	No	No	No
	0.94.24	Yes	Yes	Yes	Yes (4.0.1 only)	Yes (4.0.1 only)	No	No	No
	0.98.4	No	Yes	Yes	No	No	No	No	No
	0.98.7	No	Yes	Yes	Yes (4.0.x only)	Yes (4.0.x only)	No	No	No
	0.98.9	No	Yes	Yes	Yes (4.x)	Yes (4.x)	Yes (4.x)	No	No
	0.98.12	No	No	No	Yes (4.x, 5.0)	Yes (4.x, 5.0)	Yes (4.x, 5.x)	Yes (4.x, 5.x)	Yes (5.x)
	1.1	No	No	No	No	No	No	Yes (5.1.0)	Yes (5.1.0)
Hive[2]	See Hive and HCatalog Support Matrix on page 5631.								
Hue[1]	3.8.1					Yes	No	No	No
	3.9.0					Yes	Yes (5.x)	Yes (5.x)	Yes (5.x)

last-updated: Apr 11, 2022

Impala	1.1.1	No	No	No	No	No	No	No	No
	1.2.3	No	No	No	No	No	No	No	No
	1.4.1	No	No	No	No	No	No	No	No
	2.2.0	No	No	No	No	No	No	No	No
	2.5.0	No	No	No	No	No	No	No	No
Mahout	0.10					Yes	No	No	No
	0.11					Yes	Yes[3]	Yes[3]	Yes[3]
	0.12					No	No	No	Yes
Oozie	4.1.0					Yes	Yes	Yes	Yes
	4.2.0					Yes	Yes (5.x)	Yes (5.x)	Yes (5.x)

[1] The Spark Notebook UI in Hue is a beta feature.

[2] When you use Spark 1.5.2 with Hive 0.13 or Hive 1.0, Spark SQL insert overwrite operations on Hive tables are not supported for the ORC, RC, and AVRO formats. For more information, see the Spark documentation.

[3]This combination is supported as of Mahout 0.11.0-1601.

[4]Livy version is a snapshot.

[5]EEP 2.0 only.

Change Control Notes

- This matrix was last updated on December 8, 2016 (Mahout 0.12).

MapR Data Science Refinery Release Notes

This section contains release notes for the MapR Data Science Refinery.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Important: If you downloaded a copy of the MapR Data Science Refinery prior to **May 30, 2018**, you may need to update your Docker image. Read the following section to learn more.

Older Releases of the MapR Data Science Refinery

Version 1.0 and earlier instances of the 1.1 and 1.2 versions of the Data Science Refinery have a security vulnerability. The vulnerability allows users, other than the one you have configured, to access the Zeppelin UI. To determine if you are using an image that has this vulnerability, check the id of your Docker image by running the following command:

```
docker images
```

The following is sample output from the command. The third column corresponds to the image id:

```
REPOSITORY                          TAG                           IMAGE
ID            CREATED             SIZE
maprtech/data-science-refinery      v1.2_6.0.1_5.0.0_centos7
fb10d575b45f        5 days ago          3.94GB
```


last-updated: Apr 11, 2022

If your image id matches any of the ids shown in the following table, then you must pull a newer version of the Docker image.

Data Science Refinery Version Number	Operating System	Vulnerable Image Id
1.2	Ubuntu 16	fab3fc7a719e
	CentOS 7	0e746a584940
1.1	Ubuntu 16	310a209f8089
	CentOS 7	2caf2307bb0b
1.0	All operating systems	All ids

See Accessing the Zeppelin Docker Image on page 3034 for instructions on how to download the latest image.

For more information about this security vulnerability, see Zeppelin Authentication: Passwords in Shiro are overwritten.

MapR Data Science Refinery Support by MapR Core Version

This topic provides general guidance about the MapR core versions you can use with various versions of MapR Data Science Refinery. It also lists features that are exceptions to the general rule.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Each MapR Data Science Refinery release has a corresponding MapR Ecosystem Pack (EEP) release:

MapR Data Science Refinery Version Number	Corresponding EEP Version
1.4.1	6.3.0
1.4	6.2.0
1.3.2	6.1.0
1.3	6.0.0
1.2	5.0.0
1.1	4.1.0
1.0	4.0.0

In addition, each MapR core version supports a set of EEP releases. EEP Support and Lifecycle Status on page 5531 shows the support matrix.

In general, you can use MapR Data Science Refinery with the MapR core version supported by the corresponding EEP release, as well as newer MapR core versions. For example, because MapR Data Science Refinery 1.3 corresponds with EEP 6.0, which supports MapR core 6.1, unless otherwise noted, you can use MapR Data Science Refinery 1.3 to access a MapR core 6.1 (or later) cluster.

However, there are some exceptions to this general guidance. Some Zeppelin interpreters do not support certain features when using specific MapR core versions. The following table highlights those exceptions. It includes the version combinations, which are supported for those exception features, to show which MapR core version you need to use.

Zeppelin Interpreter Feature	MapR Data Science Refinery Version	MapR Core Version		
		6.0.0	6.0.1	6.1
Hive interpreter accessing a secure MapR cluster	1.0	Y	Y	N
	1.1	Y	Y	N
	1.2	N	Y	N
	1.3	N	N	Y
	1.4	N	N	Y
Livy and Spark interpreters accessing MapR Event Store	1.0 *	Y	Y	N
	1.1	Y	Y	N
	1.2	N	Y	N
	1.3	N	N	Y
	1.4	N	N	Y

* Not applicable for the Spark interpreter since it was introduced in MapR Data Science Refinery 1.1

What's New in MapR Data Science Refinery 1.4.1

Provides a summary of the new functionality in MapR Data Science Refinery 1.4.1.

MapR Data Science Refinery 1.4.1 includes the following new features:

- Zeppelin is updated to 0.8.2 patch release.
- Zeppelin is now compatible with Spark 2.4.4.
- Updated list of available Helium plugins to most recent version.
- Updated joda-time library to 2.10.3.

What's New in MapR Data Science Refinery 1.4

Provides a summary of the new functionality in MapR Data Science Refinery 1.4.

MapR Data Science Refinery 1.4 includes the following new features:

- Zeppelin has been updated to 0.8.1 patch release.
- Zeppelin is now compatible with Spark 2.4.
- Zeppelin can now be built from MapR Data Science Refinery Docker files that use Docker multi-stage build, rather than downloading Zeppelin RPM/DEB packages from corresponding repositories on https://package.mapr.hpe.com/.

What's New in MapR Data Science Refinery 1.3

MapR Data Science Refinery 1.3 introduces new features and some changes in behavior for existing features from prior releases. This release requires that you connect to a MapR 6.1.0 (or later) cluster.

New Features

The following are new features in this release:

Interpreter Lifecycle Management	Prior to the 1.3 release, if a Zeppelin interpreter is idle and using excessive resources, you must either restart or kill the interpreter to reclaim resources. Starting

last-updated: Apr 11, 2022

	in 1.3, MapR Data Science Refinery terminates interpreters that have been idle for an hour. You can configure this timeout threshold. For details, see the Idle Interpreter Timeout Threshold in Understanding Zeppelin Docker Parameters on page 3036.
Helium Repository Browser	Starting with the 1.3 release, MapR Data Science Refinery supports the Helium repository browser for enabling Zeppelin visualization packages. This provides a simpler procedure for enabling these packages. See Using Visualization Packages in Zeppelin on page 3084 for detailed instructions.
Configuration Storage	Starting with the MapR Data Science Refinery 1.3 release, you can store certain Zeppelin configuration files in MapR File System, which enables you to share them across multiple containers. For details, see Configuration Storage in Understanding Zeppelin Docker Parameters on page 3036.
Default Drill JDBC Connection String	Starting with MapR Data Science Refinery 1.3, you can configure the default Drill JDBC connection URL. For more information, see Default Drill JDBC Connection URL in Understanding Zeppelin Docker Parameters on page 3036.
Building your own Docker Image	Starting with the 1.3 release, you can build your own custom Docker image of MapR Data Science Refinery. See Building your own MapR Data Science Refinery Docker Image on page 3103 for more information.

Changes in Existing Features

The following describe changes in behavior from prior releases:

YARN Cluster Mode for Spark Interpreter Jobs	Prior to the 1.3 release, Spark interpreter jobs run in YARN client mode. The interpreter now runs in cluster mode. This mode reduces Spark resource utilization on the host machine of your MapR Data Science Refinery container. See Understanding Zeppelin Interpreters on page 3059 for details.
Shared Livy Sessions	In prior releases, the Livy interpreter uses separate Livy sessions for Spark, PySpark, and SparkR jobs. Starting in the 1.3 release, it uses a shared Livy session to run all Spark variations. This reduces resource utilization in your MapR cluster.
Sequential Execution of Notebook Paragraphs	Starting with the 1.3 release, MapR Data Science Refinery runs paragraphs in a notebook sequentially rather than in parallel. This allows paragraphs to run properly when they have dependencies on earlier paragraphs in the same notebook.
Hive JDBC Interpreter and Secure MapR Clusters	Starting with the 1.3 release, you must specify `ssl=true` in your Hive JDBC URL when connecting to a secure MapR cluster. See Installing Custom Packages for PySpark Using Conda on page 3068 for an example.
Python Versions with the Livy Interpreter	Starting with the 1.3 release, you no longer can run both Python 2 and Python 3 with the Livy interpreter. You can run only one or the other. By default, the

	interpreter runs Python 2. To switch to Python 3, see Python Version in Understanding Zeppelin Docker Parameters on page 3036. The limitation also applies if you are installing custom Python packages. See Installing Custom Packages for PySpark Using Conda on page 3068 for instructions on how to install Python 2 vs Python 3 custom packages.
Running Zeppelin as a Kubernetes Service	The `DEPLOY_MODE` parameter is your Kubernetes pod manifest file has been renamed to `ZEPPELIN_DEPLOY_MODE`. You can still use `DEPLOY_MODE`, but MapR Data Science Refinery 1.3 returns a warning, indicating the parameter is deprecated. See Running MapR Data Science Refinery as a Kubernetes Service on page 3049 for an example of a pod manifest file.
Notebook Storage Using MapR File System	Starting with the 1.3 release, to store your notebooks in MapR File System, you no longer need to use the FUSE-based POSIX client. See Understanding Zeppelin Docker Parameters on page 3036 for details.

Related concepts

Zeppelin 0.8.0-1808 Release Notes on page 5647

What's New in MapR Data Science Refinery 1.2

Provides a summary of the new functionality in MapR Data Science Refinery 1.2

Data Science Refinery 1.2 includes the following new features:

- Support for the MapR Database Shell interpreter that allows you to run MapR Database Shell commands in Zeppelin
- Support for deploying the Data Science Refinery on a MapR cluster node

What's New in MapR Data Science Refinery 1.1

Provides a summary of the new functionality in MapR Data Science Refinery 1.1

Data Science Refinery 1.1 includes the following new features in Zeppelin on MapR:

- Support for the Spark interpreter, configured to launch Spark jobs in YARN client mode
- Enhancements in installing custom Python environments for the Livy and Spark interpreters
- Improvements in launching multiple Zeppelin containers on the same host

Zeppelin Release Notes

The release notes for the Zeppelin component (included in the MapR Data Science Refinery) contain notes specific to the MapR release of Zeppelin.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following table shows the mapping between the Zeppelin and Data Science Refinery version numbers:

Zeppelin Version Number	MapR Data Science Refinery Version Number
0.8.2-1912	1.4.1
0.8.1-1904	1.4.0

last-updated: Apr 11, 2022

Zeppelin Version Number	MapR Data Science Refinery Version Number
0.8.0-1901	1.3.2
0.8.0-1808	1.3
0.7.2-1803	1.2
0.7.2-1801	1.1
0.7.2-1710	1.0

Zeppelin 0.8.2-1912 Release Notes

The notes below relate specifically to the MapR distribution of Apache Zeppelin. You may also be interested in the Apache Zeppelin project homepage and the following Apache Zeppelin 0.8.x changelogs:

- Apache Zeppelin 0.8.2 changelog
- Apache Zeppelin 0.8.1 changelog
- Apache Zeppelin 0.8.0 changelog

<table>
<tr><td>Version</td><td>0.8.2</td></tr>
<tr><td>Release Date</td><td>December 2019</td></tr>
<tr><td>Source on GitHub</td><td>Zeppelin: https://github.com/mapr/zeppelin
Livy: https://github.com/mapr/livy</td></tr>
<tr><td>GitHub Release Tag</td><td>Zeppelin: 0.8.2-mapr-1912
Livy: 0.5.0-mapr-1912</td></tr>
<tr><td>Docker Image Name and Tags</td><td>
<table>
<tr><th>Image Name</th><th>Operating System Version of the Running Container</th><th>Tag</th></tr>
<tr><td rowspan="2">maprtech/data-science-refinery</td><td>CentOS 7.7</td><td><code>v1.4.1_6.1.0_6.3.0_centos7</code></td></tr>
<tr><td>Ubuntu 16.04</td><td><code>v1.4.1_6.1.0_6.3.0_ubuntu16</code></td></tr>
</table>
</td></tr>
</table>

Zeppelin on MapR is a component of the MapR Data Science Refinery. This release of Zeppelin is in version 1.4.1 of the MapR Data Science Refinery.

The Data Science Refinery is packaged as a Docker container. MapR ecosystem components included in the Docker image are the same as those in the EEP 6.3.0 release. For product version details, see EEP 6.3.0 Components and OS Support on page 5549.

You can run the Docker image on the following operating systems:

- Linux (CentOS 7.x, Ubuntu 14, Ubuntu 16)
- Windows 10 Pro (64-bit)
- Mac OS X 10.11

The following are the verified browsers:

- Chrome 57

- Firefox 56.0
- Microsoft Edge 40
- Safari 9.0

The MapR product documentation is available at Zeppelin on MapR on page 3033.

New in this Release

Zeppelin in MapR Data Science Refinery 1.4.1 has updated dependencies to be compatible with EEP 6.3.0. Specifically:

- Zeppelin is updated to 0.8.2 patch release.
- Zeppelin is now compatible with Spark 2.4.4.
- Updated list of available Helium plugins to most recent version.
- Updated joda-time library to 2.10.3.

Fixes

This release of MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
c5f52f2	2019-10-22	Change zeppelin.server.addr back to 0.0.0.0 after ZEPPELIN-4166
fbba6a5	2019-11-18	[ZEPPELIN-4214]. Spark Web UI is displayed in the wrong paragraph
be3bea6	2019-11-19	CORE-332 Joda Time Library for Brazil
444c1c6	2019-11-19	DSR-94 Update list of Helium plugins to recent version
80a5809	2019-11-21	MZEP-182 Install git package in DSR images

Known Issues and Limitations

- MZEP-17: The HBase interpreter cannot be used to query MapR Database Binary tables
- MZEP-79: Legends in plots do not display correctly when running the Matplotlib (Python/PySpark) example from the Zeppelin Tutorial
- MD-2397: Zeppelin cannot connect to Drill through the JDBC driver on a secure MapR cluster when Zeppelin has Kerberos authentication enabled
- MZEP-86: You cannot run Zeppelin as user 'root'
- MZEP-110: You cannot use a custom R environment with Zeppelin
- DSR-27: You cannot access MapR Event Store For Apache Kafka using either the Livy or Spark interpreters with Data Science Refinery 1.2, if you are connecting to a MapR cluster older than version 6.0.1.

last-updated: Apr 11, 2022

- DSR-62: You cannot access MapR Event Store For Apache Kafka using either the Livy or Spark interpreters with MapR Data Science Refinery 1.1 and 1.2, if you are connecting to a MapR 6.1.0 cluster.
- ZEPPELIN-1904 - Pandas `dataframe.plot` does not work in the Python and PySpark interpreters

For issues that apply to running MapR Docker images, see MapR PACC Known Issues on page 417.

Zeppelin 0.8.1-1904 Release Notes

The notes below relate specifically to the MapR distribution of Apache Zeppelin. You may also be interested in the Apache Zeppelin 0.8. changelog and the Apache Zeppelin project homepage.

<table>
<tr><td>Version</td><td>0.8.1</td></tr>
<tr><td>Release Date</td><td>May 2019</td></tr>
<tr><td>Source on GitHub</td><td>https://github.com/mapr/zeppelin, https://github.com/mapr/livy</td></tr>
<tr><td>GitHub Release Tag</td><td>0.8.1-mapr-1904</td></tr>
<tr><td>Docker Image Name and Tags</td><td>
<table>
<tr><th>Image Name</th><th>Operating System Version of the Running Container</th><th>Tag</th></tr>
<tr><td rowspan="2">maprtech/data-science-refinery</td><td>CentOS 7.6</td><td>v1.4.0_6.1.0_6.2.0_centos7</td></tr>
<tr><td>Ubuntu 16.04</td><td>v1.4.0_6.1.0_6.2.0_ubuntu16</td></tr>
</table>
</td></tr>
</table>

Zeppelin on MapR is a component of the MapR Data Science Refinery. This release of Zeppelin is in version 1.4.0 of the MapR Data Science Refinery.

The Data Science Refinery is packaged as a Docker container. MapR ecosystem components included in the Docker image are the same as those in the EEP 6.2 release. See EEP 6.2.0 Components and OS Support on page 5550 for details on product version numbers.

You can run the Docker image on the following operating systems:

- Linux (CentOS 7.x, Ubuntu 14, Ubuntu 16)
- Windows 10 Pro (64-bit)
- Mac OS X 10.11

The following are the verified browsers:

- Chrome 57
- Firefox 56.0
- Microsoft Edge 40
- Safari 9.0

The MapR product documentation is available at Zeppelin on MapR on page 3033.

last-updated: Apr 11, 2022

New in this Release

Zeppelin in MapR Data Science Refinery 1.4.0 has updated dependencies to be compatible with MEP-6.2.0. Specifically:

- Zeppelin is updated to 0.8.1 patch release.
- Zeppelin is now compatible with Spark 2.4.
- Zeppelin can be built from MapR Data Science Refinery Docker files that use Docker multi-stage build instead of downloading Zeppelin RPM/DEB packages from corresponding repositories on https://package.mapr.hpe.com/.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
`706f3c3`	2019-03-14	DSR-84: Switched to Docker multi-stage build
`2a6ed4f`	2019-04-09	DSR-88: Refactored Spark MapR-DB Binary tutorials notebook according to ZEPPELIN-3587

Known Issues and Limitations

- MZEP-17: The HBase interpreter cannot be used to query MapR Database Binary tables
- MZEP-79: Legends in plots do not display correctly when running the Matplotlib (Python/PySpark) example from the Zeppelin Tutorial
- MD-2397: Zeppelin cannot connect to Drill through the JDBC driver on a secure MapR cluster when Zeppelin has Kerberos authentication enabled
- MZEP-86: You cannot run Zeppelin as user 'root'
- MZEP-110: You cannot use a custom R environment with Zeppelin
- DSR-27: You cannot access MapR Event Store using either the Livy or Spark interpreters with Data Science Refinery 1.2, if you are connecting to a MapR cluster older than version 6.0.1.
- DSR-62: You cannot access MapR Event Store using either the Livy or Spark interpreters with MapR Data Science Refinery 1.1 and 1.2, if you are connecting to a MapR 6.1.0 cluster.
- MZEP-154: You cannot log into the Zeppelin UI using Firefox on Ubuntu 18.04.
- ZEPPELIN-1904 - Pandas `dataframe.plot` does not work in the Python and PySpark interpreters

See MapR PACC Known Issues on page 417 for issues that apply to running MapR Docker images.

Zeppelin 0.8.0-1901 Release Notes

The notes below relate specifically to the MapR distribution of Apache Zeppelin. You may also be interested in the Apache Zeppelin 0.8.0 changelog and the Apache Zeppelin project homepage.

Version	0.8.0
Release Date	February 2019

<table>
<tr><td>Source on GitHub</td><td>https://github.com/mapr/zeppelin, https://github.com/mapr/livy</td></tr>
<tr><td>GitHub Release Tag</td><td>0.8.0-mapr-1901</td></tr>
<tr><td>Docker Image Name and Tags</td><td>
<table>
<tr><th>Image Name</th><th>Operating System Version of the Running Container</th><th>Tag</th></tr>
<tr><td rowspan="2">maprtech/data-science-refinery</td><td>CentOS 7.6</td><td>v1.3.2_6.1.0_6.1.0_centos7</td></tr>
<tr><td>Ubuntu 16.04</td><td>v1.3.2_6.1.0_6.1.0_ubuntu16</td></tr>
</table>
</td></tr>
</table>

Zeppelin on MapR is a component of the MapR Data Science Refinery. This release of Zeppelin is in version 1.3.2 of the MapR Data Science Refinery.

The Data Science Refinery is packaged as a Docker container. MapR ecosystem components included in the Docker image are the same as those in the EEP 6.1 release. See EEP 6.1.0 Components and OS Support on page 5552 for details on product version numbers.

You can run the Docker image on the following operating systems:

- Linux (CentOS 7.x, Ubuntu 14, Ubuntu 16)
- Windows 10 Pro (64-bit)
- Mac OS X 10.11

The following are the verified browsers:

- Chrome 57
- Firefox 56.0
- Microsoft Edge 40
- Safari 9.0

The MapR product documentation is available at .Zeppelin on MapR on page 3033

New in this Release

- Zeppelin in MapR Data Science Refinery 1.3.2 has updated dependencies to be compatible with MEP-6.1.0.
- Cron Scheduler feature is now enabled by default in Zeppelin UI.

See What's New in MapR Data Science Refinery 1.3 on page 5639 for the list of new Zeppelin features in 1.3 release.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
0345bf4	2018-09-25	DSR-70: Fix for DRILL_HOME warning

Commit	Date (YYYY-MM-DD)	Comment
97da759	2018-10-01	DSR-70: Use HTTPs APT repositories on Ubuntu
9bc5343	2018-10-29	DSR-74: New Drill Simba Driver for MapR Data Science Refinery 1.3
8878608	2019-01-25	DSR-76: Enable the cron scheduler in Zeppelin

Known Issues

See Known Issues and Limitations on page 5649 in the Zeppelin 0.8.0-1808 Release Notes on page 5647 for the list of known issues and limitations in this release. See also MapR PACC Known Issues on page 417 for issues that apply to running MapR Docker images.

Zeppelin 0.8.0-1808 Release Notes

The notes below relate specifically to the MapR distribution of Apache Zeppelin. You may also be interested in the Apache Zeppelin 0.8.0 changelog and the Apache Zeppelin project homepage.

<table>
<tr><td>Version</td><td>0.8.0</td></tr>
<tr><td>Release Date</td><td>September 2018</td></tr>
<tr><td>Source on GitHub</td><td>https://github.com/mapr/zeppelin, https://github.com/mapr/livy</td></tr>
<tr><td>GitHub Release Tag</td><td>0.8.0-mapr-1808</td></tr>
<tr><td>Docker Image Name and Tags</td><td>
<table>
<tr><th>Image Name</th><th>Operating System Version of the Running Container</th><th>Tag</th></tr>
<tr><td rowspan="2">maprtech/data-science-refinery</td><td>CentOS 7.4</td><td>v1.3_6.1.0_6.0.0_centos7</td></tr>
<tr><td>Ubuntu 16.04</td><td>v1.3_6.1.0_6.0.0_ubuntu16</td></tr>
</table>
</td></tr>
</table>

Zeppelin on MapR is a component of the MapR Data Science Refinery. This release of Zeppelin is in version 1.3 of the Data Science Refinery.

The Data Science Refinery is packaged as a Docker container. MapR ecosystem components included in the Docker image are the same as those in the EEP 6.0 release. See EEP 6.0.0 Components and OS Support on page 5555 for details on product version numbers.

You can run the Docker image on the following operating systems:

- Linux (CentOS 7.x, Ubuntu 14, Ubuntu 16)
- Windows 10 Pro (64-bit)
- Mac OS X 10.11

The following are the verified browsers:

- Chrome 57
- Firefox 56.0
- Microsoft Edge 40

- Safari 9.0

The MapR product documentation is available at Zeppelin on MapR on page 3033.

New in this Release

See What's New in MapR Data Science Refinery 1.3 on page 5639 for the list of new Zeppelin features in this release.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

The following table lists the fixes for Zeppelin:

Commit	Date (YYYY-MM-DD)	Comment
29b0c56	2018-07-25	MZEP-145: Configure interpreter callback host and ports
2fd4269	2018-08-09	MZEP-152: Fix Spark interpreter start-up when Spark is not installed on the cluster
a010939	2018-08-09	MZEP-151: Enable multi-command Drill and Hive SQL execution
3155d5a	2018-08-09	MZEP-156: Fix problem in Spark interpreter start-up, due to timeout
07887f9	2018-08-09	MZEP-153: Pre-populate `default.user` for Drill and Hive in the interpreter page
0c718e4	2018-08-10	MZEP-146: Fix issues with Zeppelin tutorial examples
6bf403d	2018-08-16	MZEP-149: Update Hive JDBC driver version
4913932	2018-08-20	MZEP-146: Fix issues with Zeppelin tutorial examples
dd49cfa	2018-08-21	MZEP-148: Fix start-up failures with `%livy.pyspark` and `%spark.pyspark` interpreters
e971e68	2018-08-21	MZEP-149: Fix issue with Hive interpreter
a9cf841	2018-08-21	MZEP-161: Disable interpreter support in Helium
3d5a928	2018-08-21	MZEP-164: Disable Zeppelin credential page
df5c720	2018-08-21	MZEP-162: Disable Livy UI in MapR Data Science Refinery
dd29624	2018-08-23	MZEP-160: Fix error when enabling Helium plugins on Ubuntu
2dcc495	2018-08-23	DSR-42: Properly set Spark interpreter `deploy-mode` in a Kubernetes environment
3d53808	2018-08-28	MZEP-147: Correct redirection in Spark job UI
40e62a9	2018-08-29	MZEP-137: Add IPython interpreter
0ba1ac0	2018-08-31	MZEP-52: Use Hadoop libraries from `/opt/mapr/lib`
eded152	2018-09-03	MZEP-140: Add support for Zeppelin config storage
0a1cbaf	2018-09-03	MZEP-52 Remove R package from MapR Data Science Refinery
6af200a	2018-09-03	MZEP-140: Add `HADOOP_CONF_DIR` to `zeppelin-env.sh`
a31dd4	2018-09-04	MZEP-140: Support `ZEPPELIN_CONFIG_FS_DIR` and `ZEPPELIN_NOTEBOOK_DIR` with MapR File System
c493470	2018-09-05	MZEP-139: Enable Interpreter Lifecycle Manager by default

Commit	Date (YYYY-MM-DD)	Comment
8256a32	2018-09-05	MZEP-165: Fix failure to initialize `matplotlib` in Spark
89b1edd	2018-09-06	MZEP-137: Disable IPython by default
c1c9047	2018-09-06	DSR-61: Support configuration of the Drill interpreter connection string
0897390	2018-09-06	DSR-61: Update default hostnames in Drill and Hive JDBC URLs
8ebcd0e	2018-09-07	DSR-61: Fix incorrect setting of Drill JDBC URL
81e67fb	2018-09-07	DSR-67: Correct empty Helium page when config storage is enabled and there are no available plugins
78a5aef	2018-09-11	MZEP-171: Permanent fix for MZEP-59
2062eb7	2018-09-11	MZEP-167: Support only verified and working versions of Helium plugins
0f22f82	2018-09-11	DSR-61: Fix Drill JDBC URL for Zookeeper connections
1c84a99	2018-09-12	MZEP-140: Ensure that config storage directory exists in local filesystem
8ba3312	2018-09-12	DSR-67: Use system wide Java TrustStore
633f5a5	2018-09-13	ZEP-172: Fix Zeppelin internal tests so they run on MapR
70a5a2c	2018-09-14	Replace Hadoop `SNAPSHOT` dependencies with release versions
e3d53ad	2018-09-20	DSR-70: Prepare Docker files for publishing
b00a771	2018-09-20	DSR-61: Support `MAPR_DRILLBITS_HOSTS` and `MAPR_ZK_QUORUM` without ports
ac5da95	2018-09-20	MZEP-174: Change default name of Zeppelin Spark session

Known Issues and Limitations

- MZEP-17: The HBase interpreter cannot be used to query MapR Database Binary tables
- MZEP-79: Legends in plots do not display correctly when running the Matplotlib (Python/PySpark) example from the Zeppelin Tutorial
- MD-2397: Zeppelin cannot connect to Drill through the JDBC driver on a secure MapR cluster when Zeppelin has Kerberos authentication enabled
- MZEP-86: You cannot run Zeppelin as user 'root'
- MZEP-110: You cannot use a custom R environment with Zeppelin
- DSR-27: You cannot access MapR Event Store using either the Livy or Spark interpreters with Data Science Refinery 1.2, if you are connecting to a MapR cluster older than version 6.0.1.
- DSR-62: You cannot access MapR Event Store using either the Livy or Spark interpreters with MapR Data Science Refinery 1.1 and 1.2, if you are connecting to a MapR 6.1.0 cluster.
- MZEP-154: You cannot log into the Zeppelin UI using Firefox on Ubuntu 18.04.
- ZEPPELIN-1904 - Pandas `dataframe.plot` does not work in the Python and PySpark interpreters
- See MapR PACC Known Issues on page 417 for issues that apply to running MapR Docker images.

last-updated: Apr 11, 2022

Zeppelin 0.7.2-1803 Release Notes

The notes below relate specifically to the MapR distribution of Apache Zeppelin. You may also be interested in the Apache Zeppelin 0.7.2 changelog and the Apache Zeppelin project homepage.

<table>
<tr><td>Version</td><td>0.7.2</td></tr>
<tr><td>Release Date</td><td>March 2018</td></tr>
<tr><td>Source on GitHub</td><td>https://github.com/mapr/zeppelin, https://github.com/mapr/livy</td></tr>
<tr><td>GitHub Release Tag</td><td>0.7.2-mapr-1803</td></tr>
<tr><td>Docker Image Name and Tags</td><td>
<table>
<tr><th>Image Name</th><th>Operating System Version of the Running Container</th><th>Tag</th></tr>
<tr><td rowspan="2">maprtech/data-science-refinery</td><td>CentOS 7.4</td><td>v1.2_6.0.1_5.0.0_centos7</td></tr>
<tr><td>Ubuntu 16.04</td><td>v1.2_6.0.1_5.0.0_ubuntu16</td></tr>
</table>
</td></tr>
</table>

Zeppelin on MapR is a component of the MapR Data Science Refinery. This release of Zeppelin is in version 1.2 of the Data Science Refinery.

The Data Science Refinery is packaged as a Docker container. MapR ecosystem components included in the Docker image are the same as those in the EEP 5.0 release. See EEP 5.0.0 Components and OS Support on page 5563 for details on product version numbers.

You can run the Docker image on the following operating systems:

- Linux (CentOS 7.x, Ubuntu 14, Ubuntu 16)
- Windows 10 Pro (64-bit)
- Mac OS X 10.11

The following are the verified browsers:

- Chrome 57
- Firefox 56.0
- Microsoft Edge 40
- Safari 9.0

The MapR product documentation is available at Zeppelin on MapR on page 3033.

New in this Release

- Support for the MapR Database Shell Interpreter that allows you to run MapR Database Shell commands in Zeppelin
- Support for deploying the Data Science Refinery on a MapR cluster node

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

The following table lists the fixes for Zeppelin:

Commit	Date (YYYY-MM-DD)	Comment
6ad375f	2018-03-26	MZEP-124: Fix Livy and Spark R interpreters so they work with 'googleVis' package
ef817af	2018-03-23	MZEP-124: Fix package and namespace load failures for 'googleVis' when using `livy.sparkr` and `spark.r` interpreters
20c571c	2018-03-16	DSR-21: Fix Spark interpreter so it does not fail when `zeppelin.spark.useHiveContext` is set to true
552b84b	2018-03-13	ZEPPELIN-2377: Fix Hive support so it can be enabled in Spark master
8d02402	2018-03-07	DSR-20: Fix `NullPointerException` when starting Spark shell inside PACC
45973ee	2018-03-05	DSR-18: Update DSR image with the latest Drill JDBC 1.13 driver
a87ebdb	2018-02-27	MZEP-121: Fix Zeppelin so it can launch Spark jobs with Spark 2.2.1
eee0429	2018-02-27	DSR-17: Create maven profile for MEP-5.0.0
93563b7	2018-02-27	DSR-17: Update version to v1.2_6.0.1_5.0.0
dad374e	2018-02-22	DSR-15: Move `spark-defaults` modification to Livy
06bc368	2018-02-21	MZEP-11: Support MapR Database shell in Zeppelin
21a3343	2018-02-08	DSR-12: Fix Spark interpreter so it works with bridge networking
223945b	2018-02-02	MZEP-116: Add MapR Database Binary RDD example
4cf60c8	2018-01-29	MZEP-116: Add Spark MapR Database examples
cd11f79	2018-01-29	DSR-2: Fix DSR so it does not exit immediately in Kubernetes
8d0e8d9	2018-01-24	MZEP-110: Include R interpreter in DSR container

The following table lists fixes for Livy:

Commit	Date (YYYY-MM-DD)	Comment
a556425	2018-03-16	MLIVY-21: Fix errors in Livy after installation
2f51b41	2018-03-15	MLIVY-20: Fix failure to start after upgrade

Commit	Date (YYYY-MM-DD)	Comment
d8463f2	2018-03-14	MZEP-127: Fixes to allow Livy to work correctly when DSR is installed on a MapR cluster node
addd85	2018-03-09	DSR-9: Fix spelling error in console output
4141b70	2018-03-02	IN-1317: Implement packaging improvements for restart and security
d778404	2018-02-27	MLIVY-19: Fix file path to `warden.livy.conf`
f41ff88	2018-02-26	DSR-15: Log an error message when Python archive setup fails
73f2e22	2018-02-22	Clean up `configure.sh`
df7a030	2018-02-22	DSR-15: Propagate `spark-defaults` modifications to Livy
7f690bf	2018-02-22	LIVY-19: Fix Livy so it can start with spark 2.2.1

Known Issues and Limitations

- MZEP-17: The HBase interpreter cannot be used to query MapR Database Binary tables
- MZEP-79: Legends in plots do not display correctly when running the Matplotlib (Python/PySpark) example from the Zeppelin Tutorial
- MD-2397: Zeppelin cannot connect to Drill through the JDBC driver on a secure MapR cluster when Zeppelin has Kerberos authentication enabled
- MZEP-86: You cannot run Zeppelin as user 'root'
- MZEP-110: You cannot use a custom R environment with Zeppelin
- DSR-27: You cannot access MapR Event Store using either the Livy or Spark interpreters with Data Science Refinery 1.2, if you are connecting to a MapR cluster older than version 6.0.1.
- See MapR PACC Known Issues on page 417 for issues that apply to running MapR Docker images.

Zeppelin 0.7.2-1801 Release Notes

The notes below relate specifically to the MapR distribution of Apache Zeppelin. You may also be interested in the Apache Zeppelin 0.7.2 changelog and the Apache Zeppelin project homepage.

Version	0.7.2
Release Date	February 2018
Source on GitHub	https://github.com/mapr/zeppelin, https://github.com/mapr/livy
GitHub Release Tag	0.7.2-mapr-1801

Docker Image Name and Tags

Image Name	Operating System Version of the Running Container	Tag
maprtech/ data-science-refinery	CentOS 7.4	v1.1_6.0.0_4.1.0 _centos7
	Ubuntu 16.04	v1.1_6.0.0_4.1.0 _ubuntu16

Zeppelin on MapR is a component of the MapR Data Science Refinery. This release of Zeppelin is in version 1.1 of the Data Science Refinery.

The Data Science Refinery is packaged as a Docker container. MapR ecosystem components included in the Docker image are the same as those in the EEP 4.1 release. See EEP 4.1.0 Components and OS Support on page 5568 for details on product version numbers.

You can run the Docker image on the following operating systems:

- Linux (CentOS 7.x, Ubuntu 14, Ubuntu 16)
- Windows 10 Pro (64-bit)
- Mac OS X 10.11

The following are the verified browsers:

- Chrome 57
- Firefox 56.0
- Microsoft Edge 40
- Safari 9.0

The MapR product documentation is available at Zeppelin on MapR on page 3033.

New in this Release

This release of Zeppelin on MapR includes the following new features:

- Support for the Spark interpreter, configured to launch Spark jobs in YARN client mode
- Enhancements in installing custom Python environments for the Livy and Spark interpreters
- Improvements in launching multiple Zeppelin containers on the same host

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

The following table lists the fixes for Zeppelin:

Commit	Date (YYYY-MM-DD)	Comment
625acf6	2018-01-11	MZEP-94: Support for different versions of Python for the Spark interpreter
aaa8e92	2018-01-11	PACC-18: Fix regression for PACC-12

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
dc9cb61	2018-01-09	PACC-18: Fix incorrect home directory after `docker exec` run on Ubuntu
8297a9e	2017-12-25	PACC-12: Remove obsolete fix for MZEP-66
3698203	2017-12-25	MZEP-97: Enable Spark interpreter to run in YARN client mode
1e6ccc0	2017-12-14	Restore Core 6.0.0 version in Dockerfile
0418ced	2017-12-13	Update EEP and MapR core versions in Dockerfiles
c6538ab	2017-12-12	MZEP-99: Fix no graphs being displayed the first time the Spark Pyspark Tutorial examples are opened
ee11e2f	2017-12-07	MZEP-98: Restore default examples in the Zeppelin Tutorial
7d15fd9	2017-12-05	MZEP-63: Explicitly set Zeppelin working directory
0fc3079	2017-12-01	Fix `mapr-setup.sh` URL in PACC build scripts
b8152f2	2017-11-30	MZEP-63: Enable Spark interpreter
b8e6693	2017-11-27	Refactor the script that builds Docker images

The following table lists fixes for Livy:

Commit	Date (YYYY-MM-DD)	Comment
bbf0155	2018-01-15	MZEP-102: Fix issue when no `LIVY_RSC_PORT_RANGE` is specified
90bd035	2018-01-12	MZEP-102: Provide ability to specify custom port range in `livy.rsc.launcher.port.range`
528e2b7	2018-01-11	MZEP-94: Support different versions of Python for the Spark interpreter
bef6778	2017-12-07	MZEP-65: Minor fix in `start-in-container.sh`
ccaed0f	2017-12-05	MZEP-63: Explicitly set Livy working directory
c2ddf74	2017-11-24	Fixes needed to set up a Conda environment for PySpark

Known Issues and Limitations

- MZEP-17: The HBase interpreter cannot be used to query MapR Database Binary tables
- MZEP-79: Legends in plots do not display correctly when running the Matplotlib (Python/PySpark) example from the Zeppelin Tutorial

- MD-2397: Zeppelin cannot connect to Drill through the JDBC driver on a secure MapR cluster when Zeppelin has Kerberos authentication enabled
- MZEP-86: You cannot run Zeppelin as user 'root'
- MZEP-110: You cannot use a custom R environment with Zeppelin
- See MapR PACC Known Issues on page 417 for issues that apply to running MapR Docker images.

Zeppelin 0.7.2-1710 Release Notes

The notes below relate specifically to the MapR distribution of Apache Zeppelin. You may also be interested in the Apache Zeppelin 0.7.2 changelog and the Apache Zeppelin project homepage.

<table>
<tr><td>Version</td><td>0.7.2</td></tr>
<tr><td>Release Date</td><td>November 2017</td></tr>
<tr><td>Source on GitHub</td><td>https://github.com/mapr/zeppelin</td></tr>
<tr><td>GitHub Release Tag</td><td>0.7.2-mapr-1710</td></tr>
<tr><td>Docker Image Name and Tags</td><td>
<table>
<tr><th>Image Name</th><th>Operating System Version of the Running Container</th><th>Tag</th></tr>
<tr><td rowspan="2">maprtech/data-science-refinery</td><td>CentOS 7.4</td><td>v1.0_6.0.0_4.0.0_centos7</td></tr>
<tr><td>Ubuntu 16.04</td><td>v1.0_6.0.0_4.0.0_ubuntu16</td></tr>
</table>
</td></tr>
</table>

New in this Release

This is the first MapR release of Apache Zeppelin. The following are important product notes:

- Apache Zeppelin on MapR is packaged as a Docker container.
- The Docker container includes the following preconfigured interpreters:
  - Livy
  - JDBC (Drill and Hive)
  - Pig
  - Shell
- MapR does not support the HBase and Spark interpreters. Support for Spark is available through the Livy interpreter.
- MapR ecosystem components included in the Docker image are the same as those in the EEP 4.0 release. See EEP 4.0.0 Components and OS Support on page 5569 for details on product version numbers.
- You can run the Zeppelin Docker image on the following operating systems:
  - Linux (CentOS 7.x, Ubuntu 14, Ubuntu 16)
  - Windows 10 Pro (64-bit)
  - Mac OS X 10.11

last-updated: Apr 11, 2022

- Verified browsers:
  - Chrome 57
  - Firefox 56.0
  - Microsoft Edge 40
  - Safari 9.0

The MapR product documentation is available at Zeppelin on MapR on page 3033.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
31a59b9	2017-11-16	MZEP-40: Update Docker files to use default MapR repositories
f4b2a58	2017-11-13	MZEP-40: Update Docker files to use stable Zeppelin repository
43afedd	2017-11-13	MZEP-67: Fix Zeppelin Pig Tutorial so it works on Ubuntu
fed5270	2017-11-13	MZEP-66: Set default working directory to avoid errors in PySpark and Spark Shell
ed7c061	2017-11-11	MZEP-67: Fix failures in second paragraph of Pig Tutorial
fb8df9a	2017-11-10	MZEP-73: Set default username for Drill interpreter on non-secure clusters
d19616a	2017-11-10	MZEP-72: Fix failures in PySpark example in Zeppelin Tutorial
f78e4e4	2017-11-03	MZEP-46: Explicitly set interpreters in code snippets from Zeppelin Tutorial
647a137	2017-11-03	MZEP-60: Increase Livy timeout parameters for Spark YARN configuration
d9e4a06	2017-11-03	MZEP-59: Suppress `DRILL_HOME` warning
54cfeef	2017-11-01	MZEP-46: Modify output from PySpark and SparkR examples
f74e216	2017-10-27	MZEP-46: Fix the Docker image so it runs on Ubuntu16
b90a313	2017-10-27	MZEP-46: Fix the Python, PySpark, and SparkR tutorials so they run in a MapR environment
fcf1cce	2017-10-23	MZEP-48: Include `mapr-hbase` package so Spark Connector for MapR Database runs with the Livy interpreter

Commit	Date (YYYY-MM-DD)	Comment
e33773e	2017-10-23	MZEP-46: Modify Zeppelin Tutorial so it works in a MapR environment
c5f8024	2017-10-22	MSPARK-99: Add `SparkVersion` file to `/opt/mapr/spark`
8aba877	2017-10-21	MZEP-39: Include Kafka client jar
39c74c6	2017-10-18	MZEP-29: Include `ZEPPELIN_DRILL_CLASSPATH` in Drill interpreter settings
da4603d	2017-10-18	MZEP-40: Publish Zeppelin Docker image on dockerhub
185b313	2017-10-17	MZEP-29: Include Drill and Hive JDBC
66acd0a	2017-10-10	MZEP-38: Fix dependency problems to address build issues
0cdd0c9	2017-10-10	MZEP-38: Fix dependency problems to address build issues
7bbf433	2017-10-10	MZEP-14: Fix port collisions for certain components
2e15609	2017-10-10	MZEP-38: Add Hive and Drill as separate interpreters to interpreters list
4babf22	2017-10-04	MZEP-30: Enable HTTPS by default for Zeppelin UI
4af55cc	2017-10-02	MZEP-31: Move maven arguments to profile
9cfc4bd	2017-10-02	MZEP-31: Move maven arguments to profile
2eefc6d	2017-09-21	Fix `shiro.ini` to allow users to log in to Docker environment
462889f	2017-09-01	MZEP-18: Add PID for Zeppelin service
733a624	2017-08-28	MZEP-13: Fix Zeppelin service so it starts with MapR core 6.0
3ccf3ee	2017-08-18	MZEP-4: Add MapR classpath to interpreter classpath
4e26f4b	2017-08-18	MZEP-4 : Implement impersonation in the interpreters using `sudo` instead of `ssh`
5d6384c	2017-08-18	MZEP-4: Use PAM by default
6432602	2017-08-17	MZEP-4: Add `configure.sh` and Warden configuration
6e69cc4	2017-08-17	MZEP-4: Allow root user to build Zeppelin
1a36f87	2017-08-17	MZEP-4: Add MapR repository to `pom.xml`
3ccf3ee	2017-08-17	MZEP-4: Package Zeppelin to run against MapR clusters

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
462889ff	2017-09-01	MZEP-18: Add PID for Zeppelin service

Known Issues and Limitations

- MZEP-17: The HBase interpreter cannot be used to query MapR Database Binary tables.
- MZEP-79: Legends in plots do not display correctly when running the Matplotlib (Python/PySpark) example from the Zeppelin Tutorial.
- MD-2397: Zeppelin cannot connect to Drill through the JDBC driver on a secure MapR cluster when Zeppelin has Kerberos authentication enabled.
- MZEP-86: You cannot run Zeppelin as user 'root'.
- See MapR PACC Known Issues on page 417 for issues that apply to running MapR Docker images.

Ecosystem Component Release Notes

The following release notes contain information for the components included in the MapR Data Platform.

Note that the *MapR Ecosystem Pack (MEP)* has been renamed as the *Ezmeral Ecosystem Pack (EEP)*. For more information about MapR Data Platform terminology, see Documentation Enhancements in What's New in Version 6.1.0 on page 34.

Be sure to review these considerations for using ecosystem component release notes:

- Ecosystem component release notes contain HPE-specific information; they do not duplicate release note information on open-source websites.
- Ecosystem component release notes are not necessarily cumulative in nature. For example, if you need to upgrade from version 1.x of a MapR product to version 4.x, be sure to review the release notes for versions 2.x, 3.x, and 4.x to become familiar with new features and known issues that might be relevant to version 4.x.
- To view individual product versions organized by EEP and core version, see Component Versions for Released MEPs.

EEP Release Notes

Provides release-note information for MapR Ecosystem Packs (EEPs).

MapR Ecosystem Pack 8.1.0 Release Notes
This topic contains information about the components included in MapR Ecosystem Pack 8.1.0.

Release Date	January 2022
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]EEPs are contained in the `MEP-<version>` directory. The `MEP-version` directory can be represented by a 2-digit number or a 3-digit number (for example, `MEP-6.0` or `MEP-6.0.0`). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 8.1.0 can be used with core 7.0.0 and core 6.2.0. For more information about EEP and core version support, see EEP Support and Lifecycle Status on page 5531.

last-updated: Apr 11, 2022

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.8-2104, 2.3.8 refers to the Hive version number, and 2104 typically indicates an April 2021 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 8.1.0 did not introduce any changes that affect application backward compatibility.

EEP 8.1.0 Components

The EEP 8.1.0 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
Airflow 2.2.1.0	Airflow 2.2.1.0 - 2201 (EEP 6.2.0-8.1.0) Release Notes on page 5738
AsyncHBase 1.8.2	AsyncHBase 1.8.2-2009 Release Notes on page 5739
Data Access Gateway 4.0.0.0	Data Access Gateway 4.0 Release Notes on page 5743
Drill 1.16.1.400[1]	Drill 1.16.1.400-2201 (EEP 8.1.0) Release Notes on page 5747
Hadoop 2.7.6.200	Hadoop 2.7.6.200 - 2201 (EEP 8.1.0) Release Notes on page 5834
HBase 1.4.13.200	HBase 1.4.13.200 - 2201 (EEP 8.1.0) Release Notes on page 5855
Hive 2.3.9	Hive 2.3.9.0 - 2201 (EEP 8.1.0) Release Notes on page 5883
HttpFS 1.1.0.200	HttpFS 1.1.0.200 - 2201 (EEP 8.1.0) Release Notes on page 6050
Hue 4.6.0.300[2]	Hue 4.6.0.300 - 2201 (EEP 8.1.0) Release Notes on page 6070
Livy 0.7.0.100	Livy 0.7.0.200 - 2201 (EEP 8.1.0) Release Notes on page 6155
Monitoring	Monitoring Components - EEP 8.1.0 Release Notes on page 6234
Oozie 5.2.1.200	Oozie 5.2.1.200 - 2201 (EEP 8.1.0) Release Notes on page 6267
S3 Gateway 2.2.0.0	S3 Gateway 2.2.0.0 - 2110 (EEP 8.0.0) Release Notes on page 6260
Spark 3.2.0.0	Spark 3.2.0.0 - 2201 (EEP 8.1.0) Release Notes on page 6344
Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167

last-updated: Apr 11, 2022

Component	Release Notes
Streams Tools	Kafka Streams 2.6.1.100 - 2201 (EEP 8.1.0) Release Notes on page 6170 KSQL 6.0.0.100 - 2201 (EEP 8.1.0) Release Notes on page 6183 Kafka Connect HDFS 10.0.0.100 - 2201 (EEP 8.1.0) Release Notes on page 6192 Kafka Connect JDBC 10.0.1.100 - 2201 (EEP 8.1.0) Release Notes on page 6201 Kafka Connect 10.0.0.100 - 2201 (EEP 8.1.0) Release Notes on page 6207 Kafka REST Proxy 6.0.0.100 - 2201 (EEP 8.1.0) Release Notes on page 6213 Kafka Schema Registry 6.0.0.100 - 2201 (EEP 8.1.0) Release Notes on page 6228
Tez 0.9.2	Tez 0.9.2 - 2201 (EEP 8.1.0) Release Notes on page 6481

[1]Support for this component is subject to your license agreement.

[2]The Spark Notebook UI in Hue is a beta feature.

The EEP 8.1.0 repository contains the following ecosystem components that are supported for internal data-fabric monitoring use cases only:

- Collectd 5.12.0.400
- Elasticsearch 6.8.8.500
- Fluentd 1.10.3.400
- Grafana 7.5.10.400
- Kibana 6.8.8.500
- OpenTSDB 2.4.1.400

MapR Ecosystem Pack 8.0.0 Release Notes
This topic contains information about the components included in MapR Ecosystem Pack 8.0.0.

Notice: HPE recommends using EEP 8.1.0 instead of EEP 8.0.0. For more information about EEP 8.1.0, see EEP 8.1.0 Reference Information on page 6504.

Release Date	October 2021
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]EEPs are contained in the `MEP-<version>` directory. The `MEP-version` directory can be represented by a 2-digit number or a 3-digit number (for example, `MEP-6.0` or `MEP-6.0.0`). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 8.0.0 can be used with core 6.2.0. For more information about EEP and core version support, see EEP Support and Lifecycle Status on page 5531.

last-updated: Apr 11, 2022

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.8-2104, 2.3.8 refers to the Hive version number, and 2104 typically indicates an April 2021 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 8.0.0 did not introduce any changes that affect application backward compatibility.

EEP 8.0.0 Components

The EEP 8.0.0 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.8.2	AsyncHBase 1.8.2-2009 Release Notes on page 5739
Data Access Gateway 3.0.0.0	Data Access Gateway 3.0 Release Notes on page 5744
Drill 1.16.1.300[1]	Drill 1.16.1.300-2110 (EEP 8.0.0) Release Notes on page 5748
Flume 1.9.0.200	Flume 1.9.0.200-2110 (EEP 8.0.0) Release Notes on page 5819
Hadoop 2.7.6.100	Hadoop 2.7.6.100 - 2110 (EEP 8.0.0) Release Notes on page 5836
HBase 1.4.13.100	HBase 1.4.13.100 - 2110 (EEP 8.0.0) Release Notes on page 5856
Hive 2.3.9	Hive 2.3.9 - 2110 (EEP 8.0.0) Release Notes on page 5890
HttpFS 1.1.0.100	HttpFS 1.1.0.100 - 2110 (EEP 8.0.0) Release Notes on page 6051
Hue 4.6.0.200[2]	Hue 4.6.0.200 - 2110 (EEP 8.0.0) Release Notes on page 6072
Livy 0.7.0.100	Livy 0.7.0.100 - 2110 (EEP 8.0.0) Release Notes on page 6156
Monitoring	Monitoring Components - EEP 8.0.0 Release Notes on page 6235
Oozie 5.2.1.100	Oozie 5.2.1.100 - 2110 (EEP 8.0.0) Release Notes on page 6270
Pig 0.17.0.100	Pig 0.17.0.100 - (EEP 8.0.0) 2110 Release Notes on page 6314
S3 Gateway	S3 Gateway 2.2.0.0 - 2110 (EEP 8.0.0) Release Notes on page 6260
Spark 3.1.2.0	Spark 3.1.2.0 - 2110 (EEP 8.0.0) Release Notes on page 6347
Sqoop 1.4.7	Sqoop 1.4.7 - 2110 (EEP 8.0.0) Release Notes on page 6448

last-updated: Apr 11, 2022

Component	Release Notes
Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
Streams Tools	Kafka Streams 2.6.1.0 - 2110 (EEP 8.0.0) Release Notes on page 6172 KSQL 6.0.0.0 - 2110 (EEP 8.0.0) Release Notes on page 6185 Kafka Connect HDFS 10.0.0.0 - 2110 (EEP 8.0.0) Release Notes on page 6193 Kafka Connect JDBC 10.0.1.0 - 2110 (EEP 8.0.0) Release Notes on page 6202 Kafka REST Proxy 6.0.0.0 - 2110 (EEP 8.0.0) Release Notes on page 6214 Kafka Schema Registry 6.0.0.0 - 2110 (EEP 8.0.0) Release Notes on page 6229
Tez 0.9.2	Tez 0.9.2 - 2110 (EEP 8.0.0) Release Notes on page 6482

[1]Support for this component is subject to your license agreement.

[2]The Spark Notebook UI in Hue is a beta feature.

The EEP 8.0.0 repository contains the following ecosystem components that are supported for internal data-fabric monitoring use cases only:

- Collectd 5.12.0.300
- Elasticsearch 6.8.8.400
- Fluentd 1.10.3.300
- Grafana 7.5.10.300
- Kibana 6.8.8.400
- OpenTSDB 2.4.1.300

MapR Ecosystem Pack 7.1.1 Release Notes
This topic contains information about the components included in MapR Ecosystem Pack 7.1.1.

Release Date	October 2021
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]EEPs are contained in the `MEP-<version>` directory. The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-6.0 or MEP-6.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 7.1.1 can be used with core 6.2.0. For more information about EEP and core version support, see EEP Support and Lifecycle Status on page 5531.

EEP 7.1.1 is identical to EEP 7.1.0 except for changes to the monitoring (Spyglass) components. For a list of monitoring fixes in EEP 7.1.1, see Monitoring Components - EEP 7.1.1 Release Notes on page 6236.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.8-2104, 2.3.8 refers to the Hive version number, and 2104 typically indicates an April 2021 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 7.1.1 did not introduce any changes that affect application backward compatibility.

Impala SLES 15 Support

Though there are no code changes for Impala in this release, new Impala packages are available. These new Impala packages include support for SLES 15.

EEP 7.1.1 Components

The EEP 7.1.1 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.8.2	AsyncHBase 1.8.2-2009 Release Notes on page 5739
Data Access Gateway 3.0.0.0	Data Access Gateway 3.0 Release Notes on page 5744
Drill 1.16.1.200[1]	Drill 1.16.1.200-2104 (MEP 7.1.0) Release Notes
Flume 1.9.0.100	Flume 1.9.0.100-2101 (EEP 7.0.1) Release Notes on page 5820
Hadoop 2.7.5.0	Hadoop 2.7.5.0 - 2104 (MEP 7.1.0) Release Notes
HBase 1.4.13.0	HBase 1.4.13.0 - 2104 (MEP 7.1.0) Release Notes
Hive 2.3.8	Hive 2.3.8 - 2104 (MEP 7.1.0) Release Notes
HttpFS 1.1.0.0	HttpFS 1.1.0.0 - 2104 (MEP 7.1.0) Release Notes
Hue 4.6.0.0[2]	Hue 4.6.0.0 - 2104 (MEP 7.1.0) Release Notes
Impala 2.12.0.500	Impala 2.12.0.400 - 2101 (EEP 7.0.1) Release Notes on page 6139
Livy 0.7.0.0	Livy 0.7.0.0 - 2104 (MEP 7.1.0) Release Notes
Monitoring	Monitoring Components - EEP 7.1.1 Release Notes on page 6236
S3 Gateway 2.1.0.0	Object Store with S3-Compatible API 2.1.0.0 - 2104 (MEP 7.1.0) Release Notes
Oozie 5.2.1.0	Oozie 5.2.1.0 - 2104 (MEP 7.1.0) Release Notes
Pig 0.17.0.0	Pig 0.17.0.0 Release Notes on page 6314
Sentry 1.7.0	Sentry 1.7.0 - 2101 (MEP 7.0.1) Release Notes on page 6330
Spark 2.4.7.100	Spark 2.4.7.100 - 2104 (MEP 7.1.0) Release Notes
Sqoop 1.4.7	Sqoop 1.4.7 - 2104 (MEP 7.1.0) Release Notes

last-updated: Apr 11, 2022

Component	Release Notes
Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
Streams Tools	Kafka Streams 2.1.1.200 - 2104 (MEP 7.1.0) Release Notes KSQL 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes Kafka Connect 5.1.2.0 - 2009 (EEP 7.0.0) Release Notes on page 6208 Kafka Connect HDFS 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes Kafka Connect JDBC 5.1.2.0 - 2009 (MEP 7.0.0) Release Notes on page 6203 Kafka REST Proxy 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes Kafka Schema Registry 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes
Tez 0.9.2	Tez 0.9.2 - 2104 (MEP 7.1.0) Release Notes

[1]Support for this component is subject to your license agreement.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]Support for Sentry is limited to Impala users.

The EEP 7.1.1 repository contains the following ecosystem components that are supported for internal data-fabric monitoring use cases only:

- Collectd 5.10.0.0
- Elasticsearch 6.8.8.300
- Fluentd 1.10.3.200
- Grafana 7.5.2.200
- Kibana 6.8.8.300
- OpenTSDB 2.4.0

MapR Ecosystem Pack 7.1.0 Release Notes

This topic contains information about the components included in MapR Ecosystem Pack 7.1.0.

Release Date	April 2021
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-6.0 or MEP-6.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 7.1.0 can be used with core 6.2.0. For more information about EEP and core version support, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.8-2104, 2.3.8 refers to the Hive version number, and 2104 typically indicates an April 2021 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 7.1.0 did not introduce any changes that affect application backward compatibility.

Impala SLES 15 Support

Though there are no code changes for Impala in this release, new Impala packages are available. These new Impala packages include support for SLES 15.

EEP 7.1.0 Components

The EEP 7.1.0 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.8.2	AsyncHBase 1.8.2-2009 Release Notes on page 5739
Data Access Gateway 3.0	Data Access Gateway 3.0 Release Notes on page 5744
Drill 1.16.1.200[1]	Drill 1.16.1.200-2104 (MEP 7.1.0) Release Notes
Flume 1.9.0.100	Flume 1.9.0.100-2101 (EEP 7.0.1) Release Notes on page 5820
Hadoop 2.7.5.0	Hadoop 2.7.5.0 - 2104 (MEP 7.1.0) Release Notes
HBase 1.4.13.0	HBase 1.4.13.0 - 2104 (MEP 7.1.0) Release Notes
Hive 2.3.8	Hive 2.3.8 - 2104 (MEP 7.1.0) Release Notes
HttpFS 1.1.0.0	HttpFS 1.1.0.0 - 2104 (MEP 7.1.0) Release Notes
Hue 4.6.0.0[2]	Hue 4.6.0.0 - 2104 (MEP 7.1.0) Release Notes
Impala 2.12.0.500	Impala 2.12.0.400 - 2101 (EEP 7.0.1) Release Notes on page 6139
Livy 0.7.0.0	Livy 0.7.0.0 - 2104 (MEP 7.1.0) Release Notes
Monitoring	Monitoring Components - MEP 7.1.0 Release Notes
S3 Gateway 2.1.0.0	Object Store with S3-Compatible API 2.1.0.0 - 2104 (MEP 7.1.0) Release Notes
Oozie 5.2.1.0	Oozie 5.2.1.0 - 2104 (MEP 7.1.0) Release Notes
Pig 0.17.0.0	Pig 0.17.0.0 Release Notes on page 6314
Sentry 1.7.0	Sentry 1.7.0 - 2101 (MEP 7.0.1) Release Notes on page 6330
Spark 2.4.7.100	Spark 2.4.7.100 - 2104 (MEP 7.1.0) Release Notes
Sqoop 1.4.7	Sqoop 1.4.7 - 2104 (MEP 7.1.0) Release Notes

last-updated: Apr 11, 2022

Component	Release Notes
Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
Streams Tools	Kafka Streams 2.1.1.200 - 2104 (MEP 7.1.0) Release Notes KSQL 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes Kafka Connect 5.1.2.0 - 2009 (EEP 7.0.0) Release Notes on page 6208 Kafka Connect HDFS 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes Kafka Connect JDBC 5.1.2.0 - 2009 (MEP 7.0.0) Release Notes on page 6203 Kafka REST Proxy 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes Kafka Schema Registry 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes
Tez 0.9.2	Tez 0.9.2 - 2104 (MEP 7.1.0) Release Notes

[1]Support for this component is subject to your license agreement.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]Support for Sentry is limited to Impala users.

The EEP 7.1.0 repository contains the following ecosystem components that are supported for internal data-fabric monitoring use cases only:

- Collectd 5.10.0.0
- Elasticsearch 6.8.8.300
- Fluentd 1.10.3.0
- Grafana 7.5.2.200
- Kibana 6.8.8.300
- OpenTSDB 2.4.0

MapR Ecosystem Pack 7.0.1 Release Notes

This topic contains information about the components included in MapR Ecosystem Pack 7.0.1.

Release Date	January 2021
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-6.0 or MEP-6.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 7.0.1 can be used with core 6.2.0. For more information about EEP and core version support, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 7.0.1 did not introduce any changes that affect application backward compatibility.

EEP 7.0.1 Components

The EEP 7.0.1 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.8.2.0	AsyncHBase 1.8.2-2009 Release Notes on page 5739
Data Access Gateway 3.0.0.0	Data Access Gateway 3.0 Release Notes on page 5744
Drill 1.16.1.100[1]	Drill 1.16.1.100-2101 (EEP 7.0.1) Release Notes on page 5751
Flume 1.9.0.100	Flume 1.9.0.100-2101 (EEP 7.0.1) Release Notes on page 5820
Hadoop 2.7.4.100	Hadoop 2.7.4.100 - 2101 (EEP 7.0.1) Release Notes on page 5842
HBase 1.4.12.100	HBase 1.4.12.100 - 2101 (EEP 7.0.1) Release Notes on page 5860
Hive 2.3.7	Hive 2.3.7 - 2101 (EEP 7.0.1) Release Notes on page 5902
HttpFS 1.0	HttpFS 1.0 - 2101 (EEP 7.0.1) Release Notes on page 6054
Hue 4.6.0.0[2]	Hue 4.6.0.0 - 2009 (EEP 7.0.0) Release Notes on page 6075
Impala 2.12.0.400	Impala 2.12.0.400 - 2101 (EEP 7.0.1) Release Notes on page 6139
Monitoring	MapR Monitoring Components - EEP 7.0.1 Release Notes on page 6239
S3 Gateway 2.0.0.0	S3 Gateway 2.0.0.0 - 2009 (EEP 7.0.0) Release Notes on page 6262
Oozie 5.2.0.100	Oozie 5.2.0.100 - 2101 (EEP 7.0.1) Release Notes on page 6272
Pig 0.17.0.0	Pig 0.17.0.0 Release Notes on page 6314
Sentry 1.7.0	Sentry 1.7.0 - 2101 (MEP 7.0.1) Release Notes on page 6330
Spark 2.4.7.0	Spark 2.4.7.0 - 2101 (EEP 7.0.1) Release Notes on page 6353
Sqoop 1.4.7	Sqoop 1.4.7 - 2101 (EEP 7.0.1) Release Notes on page 6451

last-updated: Apr 11, 2022

Component	Release Notes
Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
Streams Tools	Kafka Streams 2.1.1.100 - 2101 (MEP 7.0.1) Release Notes on page 6175 KSQL 5.1.2.100 - 2101 (MEP 7.0.1) Release Notes on page 6187 Kafka Connect HDFS 5.1.2.100 - 2101 (EEP 7.0.1) Release Notes on page 6195 Kafka Connect JDBC 5.1.2.0 - 2009 (MEP 7.0.0) Release Notes on page 6203 Kafka REST Proxy 5.1.2.100 - 2101 (MEP 7.0.1) Release Notes on page 6216 Kafka Schema Registry 5.1.2.100 - 2101 (MEP 7.0.1) Release Notes on page 6231
Tez 0.9.2	Tez 0.9.2 - 2101 (EEP 7.0.1) Release Notes on page 6485

[1]Support for this component is subject to your license agreement.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]Support for Sentry is limited to Impala users.

The EEP 7.0.1 repository contains the following ecosystem components that are supported for internal data-fabric monitoring use cases only:

- Collectd 5.10.0.0
- Elasticsearch 6.8.8.0
- Fluentd 1.10.3.0
- Grafana 6.7.4.0
- Kibana 6.8.8.0
- OpenTSDB 2.4.0

MapR Ecosystem Pack 7.0.0 Release Notes

This topic contains information about the components included in MapR Ecosystem Pack 7.0.0.

Release Date	September 2020
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-6.0 or MEP-6.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 7.0.0 can be used with core 6.2.0. For more information about EEP and core version support, see EEP Support and Lifecycle Status on page 5531.

last-updated: Apr 11, 2022

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 7.0.0 did not introduce any changes that affect application backward compatibility.

EEP 7.0.0 Components

The EEP 7.0.0 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.8.2.0	AsyncHBase 1.8.2-2009 Release Notes on page 5739
Data Access Gateway 3.0.0.0	Data Access Gateway 3.0 Release Notes on page 5744
Drill 1.16.1.0[1]	Drill 1.16.1.0-2009 (EEP 7.0.0) Release Notes on page 5753
Flume 1.9.0.0	Flume 1.9.0.0-2009 Release Notes on page 5821
Hadoop 2.7.4.0	Hadoop 2.7.4.0-2009 (EEP 7.0.0) Release Notes on page 5843
HBase 1.4.12.0	HBase 1.4.12.0-2009 (EEP 7.0.0) Release Notes on page 5861
Hive 2.3.7	Hive 2.3.7-2009 (EEP 7.0.0) Release Notes on page 5906
HttpFS 1.0	HttpFS 1.0 - 2009 (EEP 7.0.0) Release Notes on page 6055
Hue 4.6.0.0[2]	Hue 4.6.0.0 - 2009 (EEP 7.0.0) Release Notes on page 6075
Impala 2.12.0.200	Impala 2.12.0.200 - 2009 Release Notes on page 6140
Monitoring	MapR Monitoring Components - EEP 7.0.0 Release Notes on page 6240
S3 Gateway 2.0.0.0	S3 Gateway 2.0.0.0 - 2009 (EEP 7.0.0) Release Notes on page 6262
Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167

last-updated: Apr 11, 2022

Component	Release Notes
Streams Tools	Kafka Connect 5.1.2.0 - 2009 (EEP 7.0.0) Release Notes on page 6208 Kafka Connect HDFS 5.1.2.0 - 2009 (EEP 7.0.0) Release Notes on page 6195 Kafka Connect JDBC 5.1.2.0 - 2009 (MEP 7.0.0) Release Notes on page 6203 Kafka REST Proxy 5.1.2.0 - 2009 (MEP 7.0.0) Release Notes on page 6217 Kafka Schema Registry 5.1.2.0 - 2009 (MEP 7.0.0) Release Notes on page 6232 Kafka Streams 2.1.1.0 - 2009 (MEP 7.0.0) Release Notes on page 6176 KSQL 5.1.2.0 - 2009 (MEP 7.0.0) Release Notes on page 6188
Oozie 5.2.0.0	Oozie 5.2.0.0 - 2009 (EEP 7.0.0) Release Notes on page 6274
Pig 0.17.0.0	Pig 0.17.0.0 Release Notes on page 6314
Sentry 1.7.0	Sentry 1.7.0 - 2009 (MEP 7.0.0) Release Notes on page 6331
Spark 2.4.5.0	Spark 2.4.5-2009 (EEP 7.0.0) Release Notes on page 6355
Sqoop 1.4.7	Sqoop 1.4.7 - 2009 (EEP 7.0.0) Release Notes on page 6452
Tez 0.9.2.0	Tez 0.9.2-2009 (EEP 7.0.0) Release Notes on page 6487

[1]Support for this component is subject to your license agreement.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]Support for Sentry is limited to Impala users.

The EEP 7.0.0 repository contains the following ecosystem components that are supported for internal data-fabric monitoring use cases only:

- Collectd 5.10.0.0
- Elasticsearch 6.8.8.0
- Fluentd 1.10.3.0
- Grafana 6.7.4.0
- Kibana 6.8.8.0
- OpenTSDB 2.4.0

MapR Ecosystem Pack 6.3.6 Release Notes
This topic contains information about the components included in MapR Ecosystem Pack 6.3.6.

CAUTION: EEP 6.3.6 requires a core patch to resolve a Warden defect. The defect is fixed in `mapr-patch-6.1.0.20180926230239.GA-20210512163609.x86_64` and later. Before upgrading to EEP 6.3.6, you must apply the patch. See Patches and Documentation for MapR 6.1.1 on page 96.

last-updated: Apr 11, 2022


Notice: Currently, no version of the Installer supports installing or upgrading to EEP 6.3.6. Manual installations and upgrades are required until a new version of the Installer is released.

Release Date	January 2022
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]EEPs are contained in the `MEP-<version>` directory. The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-6.0 or MEP-6.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 6.x can be used with MapR Core 6.1.x. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 6.3.6 did not introduce any changes that affect application backward compatibility.

EEP 6.3.6 Components

The EEP 6.3.6 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7.0	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Data Access Gateway 2.0	MapR Data Access Gateway 2.0 Release Notes on page 5745
Drill 1.16.0.400[1]	Drill 1.16.0.400-2201 (EEP 6.3.6) Release Notes on page 5756
Flume 1.8.0	Flume 1.8.0-2201 (EEP 6.3.6) Release Notes on page 5822
HBase 1.1.13.500	HBase 1.1.13.500-2201 (EEP 6.3.6) Release Notes on page 5863
Hive 2.3 (with patches from 2.3.6)	Hive 2.3.6 - 2201 (EEP 6.3.6) Release Notes on page 5916
HttpFS 1.0	HttpFS 1.0 - 2201 (EEP 6.3.6) Release Notes on page 6056
Hue 4.3.0.500[2]	Hue 4.3.0.500 - 2201 (EEP 6.3.6) Release Notes on page 6078
Impala 2.12.0.650[1]	Impala 2.12.0.600 - 2110 (EEP 6.3.5) Release Notes on page 6138
Livy 0.5.0	Livy 0.5.0 - 2201 (EEP 6.3.6) Release Notes on page 6159
Monitoring	Monitoring Components - EEP 6.3.6 Release Notes on page 6241

last-updated: Apr 11, 2022

Component	Release Notes
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
MapR Streams Tools	Kafka Streams 1.1.1 - 2201 (EEP 6.3.6) Release Notes on page 6177 Kafka Connect HDFS 4.1.0 - 2201 (EEP 6.3.6) Release Notes on page 6196 Kafka Connect JDBC 4.1.0 - 2201 (EEP 6.3.6) Release Notes on page 6204 Kafka REST 4.1.0 - 2201 (EEP 6.3.6) Release Notes on page 6219 KSQL 4.1.1-2201 (EEP 6.3.6) Release Notes on page 6190
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 5.1.0.800	Oozie 5.1.0.800 - 2201 (EEP 6.3.6) Release Notes on page 6275
Pig 0.16	Pig 0.16.0 - 2110 (EEP 6.3.5) Release Notes on page 6316
S3 Gateway 1.0.1 (formerly MapR Object Store)	S3 Gateway 1.0.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6264
Sentry 1.7.0[3]	Sentry 1.7.0 - 2101 (MEP 6.3.2) Release Notes on page 6332
Spark 2.4.4.500	Spark 2.4.4.500 - 2201 (EEP 6.3.6) Release Notes on page 6359
Sqoop 1.4.7	Sqoop 1.4.7 - 2201 (EEP 6.3.6) Release Notes on page 6453
Tez 0.9.1	Tez 0.9.1 - 2201 (EEP 6.3.6) Release Notes on page 6489

[1]Support for this component is subject to your license agreement.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

The EEP 6.3.6 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.8.1.210
- Elasticsearch 6.8.8.110
- Fluentd 1.10.3.110
- Grafana 7.5.2.110
- Kibana 6.8.8.110
- OpenTSDB 2.4.0

last-updated: Apr 11, 2022

MapR Ecosystem Pack 6.3.5 Release Notes

This topic contains information about the components included in MapR Ecosystem Pack 6.3.5.

CAUTION: EEP 6.3.5 requires a core patch to resolve a Warden defect. The defect is fixed in `mapr-patch-6.1.0.20180926230239.GA-20210512163609.x86_64` and later. Before upgrading to EEP 6.3.5, you must apply the patch. See Patches and Documentation for MapR 6.1.1 on page 96.

Release Date	October 2021
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]EEPs are contained in the `MEP-<version>` directory. The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-6.0 or MEP-6.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 6.x can be used with MapR Core 6.1.x. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 6.3.5 did not introduce any changes that affect application backward compatibility.

EEP 6.3.5 Components

The EEP 6.3.5 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7.0	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Data Access Gateway 2.0	MapR Data Access Gateway 2.0 Release Notes on page 5745
Drill 1.16.0.300[1]	Drill 1.16.0.300-2110 (EEP 6.3.5) Release Notes on page 5757
Flume 1.8.0	Flume 1.8.0-2104 (EEP 6.3.4) Release Notes on page 5823
HBase 1.1.13.400	HBase 1.1.13.400-2110 (EEP 6.3.5) Release Notes on page 5864
Hive 2.3 (with patches from 2.3.6)	Hive 2.3.6 - 2110 (EEP 6.3.5) Release Notes on page 5920
HttpFS 1.0	HttpFS 1.0 - 2104 (EEP 6.3.4) Release Notes on page 6057
Hue 4.3.0.400[2]	Hue 4.3.0.400 - 2104 (EEP 6.3.4) Release Notes on page 6080
Impala 2.12.0.600[1]	Impala 2.12.0.600 - 2110 (EEP 6.3.5) Release Notes on page 6138
Livy 0.5.0	Livy 0.5.0 - 2104 (EEP 6.3.4) Release Notes on page 6160

last-updated: Apr 11, 2022

Component	Release Notes
Monitoring	Monitoring Components - EEP 6.3.5 Release Notes on page 6243
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
MapR Streams Tools	Kafka Streams 1.1.1 - 2104 (MEP 6.3.4) Release Notes on page 6178 Kafka Connect HDFS 4.1.0 - 2104 (EEP 6.3.4) Release Notes on page 6197 Kafka Connect JDBC 4.1.0 - 2101 (EEP 6.3.2) Release Notes on page 6204 Kafka REST 4.1.0 - 2101 (MEP 6.3.2) Release Notes on page 6219 KSQL 4.1.1-1808 Release Notes on page 6191
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 5.1.0.700	Oozie 5.1.0.700 - 2110 (EEP 6.3.5) Release Notes on page 6277
Pig 0.16	Pig 0.16.0 - 2110 (EEP 6.3.5) Release Notes on page 6316
S3 Gateway 1.0.1 (formerly MapR Object Store)	S3 Gateway 1.0.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6264
Sentry 1.7.0[3]	Sentry 1.7.0 - 2101 (MEP 6.3.2) Release Notes on page 6332
Spark 2.4.4.400	Spark 2.4.4.400 - 2110 (EEP 6.3.5) Release Notes on page 6360
Sqoop 1.4.7	Sqoop 1.4.7 - 2110 (EEP 6.3.5) Release Notes on page 6454
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469
Tez 0.9.1	Tez 0.9.1 - 2104 (EEP 6.3.4) Release Notes on page 6491

[1]Support for this component is subject to your license agreement.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

The EEP 6.3.5 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.8.1.201
- Elasticsearch 6.8.8.100
- Fluentd 1.10.3.100
- Grafana 7.5.2.100

last-updated: Apr 11, 2022

- Kibana 6.8.8.100
- OpenTSDB 2.4.0

MapR Ecosystem Pack 6.3.4 Release Notes
This topic contains information about the components included in MapR Ecosystem Pack 6.3.4.

CAUTION: EEP 6.3.4 requires a core patch to resolve a Warden defect. The defect is fixed in `mapr-patch-6.1.0.20180926230239.GA-20210512163609.x86_64` and later. Before upgrading to EEP 6.3.4, you must apply the patch. See Patches and Documentation for MapR 6.1.1 on page 96.

Release Date	April 2021
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-6.0 or MEP-6.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 6.x can be used with MapR Core 6.1.x. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 6.3.4 did not introduce any changes that affect application backward compatibility.

EEP 6.3.4 Components

The EEP 6.3.4 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7.0	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Data Access Gateway 2.0	MapR Data Access Gateway 2.0 Release Notes on page 5745
Drill 1.16.0.200[1]	Drill 1.16.0.200-2104 (EEP 6.3.4) Release Notes on page 5758
Flume 1.8.0	Flume 1.8.0-2104 (EEP 6.3.4) Release Notes on page 5823
HBase 1.1.13.300	HBase 1.1.13.300-2104 (EEP 6.3.4) Release Notes on page 5865
Hive 2.3 (with patches from 2.3.6)	Hive 2.3.6 - 2104 (EEP 6.3.4) Release Notes on page 5924
HttpFS 1.0	HttpFS 1.0 - 2104 (EEP 6.3.4) Release Notes on page 6057
Hue 4.3.0.400[2]	Hue 4.3.0.400 - 2104 (EEP 6.3.4) Release Notes on page 6080

last-updated: Apr 11, 2022

Component	Release Notes
Impala 2.12.0.300[1]	Impala 2.12.0.300 - 2101 (EEP 6.3.2) Release Notes on page 6139
Livy 0.5.0	Livy 0.5.0 - 2104 (EEP 6.3.4) Release Notes on page 6160
Monitoring	MapR Monitoring Components - EEP 6.3.4 Release Notes on page 6243
MapR Object Store 1.0.1	S3 Gateway 1.0.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6264
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
MapR Streams Tools	Kafka Streams 1.1.1 - 2104 (MEP 6.3.4) Release Notes on page 6178 Kafka Connect HDFS 4.1.0 - 2104 (EEP 6.3.4) Release Notes on page 6197 Kafka Connect JDBC 4.1.0 - 2101 (EEP 6.3.2) Release Notes on page 6204 Kafka REST 4.1.0 - 2101 (MEP 6.3.2) Release Notes on page 6219 KSQL 4.1.1-1808 Release Notes on page 6191
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 5.1.0.600	Oozie 5.1.0.600 - 2104 (EEP 6.3.4) Release Notes on page 6278
Pig 0.16	Pig 0.16.0 - 2009 (EEP 6.3.1) Release Notes on page 6317
Sentry 1.7.0[3]	Sentry 1.7.0 - 2101 (MEP 6.3.2) Release Notes on page 6332
Spark 2.4.4.300	Spark 2.4.4.300 - 2104 (EEP 6.3.4) Release Notes on page 6362
Sqoop 1.4.7	Sqoop 1.4.7 - 2101 (EEP 6.3.2) Release Notes on page 6455
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469
Tez 0.9.1	Tez 0.9.1 - 2104 (EEP 6.3.4) Release Notes on page 6491

[1]Support for this component is subject to your license agreement.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

The EEP 6.3.4 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.8.1.201
- Elasticsearch 6.8.8.100

last-updated: Apr 11, 2022

- Fluentd 1.10.3.100
- Grafana 7.5.2.100
- Kibana 6.8.8.100
- OpenTSDB 2.4.0

MapR Ecosystem Pack 6.3.3 Release Notes
This topic contains information about the components included in MapR Ecosystem Pack 6.3.3.

Release Date	March 2021
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-6.0 or MEP-6.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 6.x can be used with MapR Core 6.1.x. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 6.3.3 did not introduce any changes that affect application backward compatibility.

EEP 6.3.3 Components

The EEP 6.3.3 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7.0	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Data Access Gateway 2.0	MapR Data Access Gateway 2.0 Release Notes on page 5745
Drill 1.16.0.100[1]	Drill 1.16.0.100-2101 (EEP 6.3.2) Release Notes on page 5759
Flume 1.8.0	Flume 1.8.0-2101 (EEP 6.3.2) Release Notes on page 5824
HBase 1.1.13.200	HBase 1.1.13.200-2101 (EEP 6.3.2) Release Notes on page 5867
Hive 2.3 (with patches from 2.3.6)	Hive 2.3.6 - 2101 (EEP 6.3.2) Release Notes on page 5928
HttpFS 1.0	HttpFS 1.0 - 2101 (EEP 6.3.2) Release Notes on page 6058
Hue 4.3.0.300[2]	Hue 4.3.0.300 - 2101 (EEP 6.3.2) Release Notes on page 6081
Impala 2.12.0.300[1]	Impala 2.12.0.300 - 2101 (EEP 6.3.2) Release Notes on page 6139

last-updated: Apr 11, 2022

Component	Release Notes
Monitoring	MapR Monitoring Components - EEP 6.3.3 Release Notes on page 6245
MapR Object Store 1.0.1	S3 Gateway 1.0.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6264
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
MapR Streams Tools	Kafka Streams 1.1.1 - 2101 (MEP 6.3.2) Release Notes on page 6179 Kafka Connect HDFS 4.1.0 - 2101 (EEP 6.3.2) Release Notes on page 6198 Kafka Connect JDBC 4.1.0 - 2101 (EEP 6.3.2) Release Notes on page 6204 Kafka REST 4.1.0 - 2101 (MEP 6.3.2) Release Notes on page 6219 KSQL 4.1.1-1808 Release Notes on page 6191
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 5.1.0.500	Oozie 5.1.0.500 - 2101 (EEP 6.3.2) Release Notes on page 6279
Pig 0.16	Pig 0.16.0 - 2009 (EEP 6.3.1) Release Notes on page 6317
Sentry 1.7.0[3]	Sentry 1.7.0 - 2101 (MEP 6.3.2) Release Notes on page 6332
Spark 2.4.4.200	Spark 2.4.4.200 - 2101 (EEP 6.3.2) Release Notes on page 6363
Sqoop 1.4.7	Sqoop 1.4.7 - 2101 (EEP 6.3.2) Release Notes on page 6455
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469
Tez 0.9.1	Tez 0.9.1 - 2101 (EEP 6.3.2) Release Notes on page 6492

[1]Support for this component is subject to your license agreement.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

The EEP 6.3.3 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.8.1.201
- Elasticsearch 6.8.8.100
- Fluentd 1.10.3.100
- Grafana 6.7.4.100

last-updated: Apr 11, 2022

- Kibana 6.8.8.100
- OpenTSDB 2.4.0

MapR Ecosystem Pack 6.3.2 Release Notes
This topic contains information about the components included in MapR Ecosystem Pack 6.3.2.

Release Date	January 2021
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-6.0 or MEP-6.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 6.x can be used with MapR Core 6.1.0. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 6.3.2 did not introduce any changes that affect application backward compatibility.

EEP 6.3.2 Components

The EEP 6.3.2 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7.0	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Data Access Gateway 2.0	MapR Data Access Gateway 2.0 Release Notes on page 5745
Drill 1.16.0.100[1]	Drill 1.16.0.100-2101 (EEP 6.3.2) Release Notes on page 5759
Flume 1.8.0	Flume 1.8.0-2101 (EEP 6.3.2) Release Notes on page 5824
HBase 1.1.13.200	HBase 1.1.13.200-2101 (EEP 6.3.2) Release Notes on page 5867
Hive 2.3 (with patches from 2.3.6)	Hive 2.3.6 - 2101 (EEP 6.3.2) Release Notes on page 5928
HttpFS 1.0	HttpFS 1.0 - 2101 (EEP 6.3.2) Release Notes on page 6058
Hue 4.3.0.300[2]	Hue 4.3.0.300 - 2101 (EEP 6.3.2) Release Notes on page 6081
Impala 2.12.0.300[1]	Impala 2.12.0.300 - 2101 (EEP 6.3.2) Release Notes on page 6139
Monitoring	MapR Monitoring Components - EEP 6.3.2 Release Notes on page 6246
MapR Object Store 1.0.1	S3 Gateway 1.0.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6264

last-updated: Apr 11, 2022

Component	Release Notes
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
MapR Streams Tools	Kafka Streams 1.1.1 - 2101 (MEP 6.3.2) Release Notes on page 6179 Kafka Connect HDFS 4.1.0 - 2101 (EEP 6.3.2) Release Notes on page 6198 Kafka Connect JDBC 4.1.0 - 2101 (EEP 6.3.2) Release Notes on page 6204 Kafka REST 4.1.0 - 2101 (MEP 6.3.2) Release Notes on page 6219 KSQL 4.1.1-1808 Release Notes on page 6191
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 5.1.0.500	Oozie 5.1.0.500 - 2101 (EEP 6.3.2) Release Notes on page 6279
Pig 0.16	Pig 0.16.0 - 2009 (EEP 6.3.1) Release Notes on page 6317
Sentry 1.7.0[3]	Sentry 1.7.0 - 2101 (MEP 6.3.2) Release Notes on page 6332
Spark 2.4.4.200	Spark 2.4.4.200 - 2101 (EEP 6.3.2) Release Notes on page 6363
Sqoop 1.4.7	Sqoop 1.4.7 - 2101 (EEP 6.3.2) Release Notes on page 6455
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469
Tez 0.9.1	Tez 0.9.1 - 2101 (EEP 6.3.2) Release Notes on page 6492

[1]Support for this component is subject to your license agreement.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

The EEP 6.3.2 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.8.1.201
- Elasticsearch 6.8.8.100
- Fluentd 1.10.3.100
- Grafana 6.7.4.100
- Kibana 6.8.8.100
- OpenTSDB 2.4.0

MapR Ecosystem Pack 6.3.1 Release Notes

This topic contains information about the components included in MapR Ecosystem Pack 6.3.1.

Release Date	September 2020
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-6.0 or MEP-6.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 6.x can be used with MapR Core 6.1.0. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 6.3.1 did not introduce any changes that affect application backward compatibility.

EEP 6.3.1 Components

The EEP 6.3.1 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7.0	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Data Access Gateway 2.0	MapR Data Access Gateway 2.0 Release Notes on page 5745
Drill 1.16.0.22[1]	Drill 1.16.0.22-2009 (EEP 6.3.1) Release Notes on page 5761
Flume 1.8	Flume 1.8.0-1904 Release Notes on page 5825
HBase 1.1.13.0	HBase 1.1.13.100-2009 (EEP 6.3.1) Release Notes on page 5868
Hive 2.3 (with patches from 2.3.6)	Hive 2.3.6-2009 (EEP 6.3.1) Release Notes on page 5931
HttpFS 1.0	HttpFS-1.0-1904 Release Notes on page 6060
Hue 4.3.0.200[2]	Hue 4.3.0.200 - 2009 (EEP 6.3.1) Release Notes on page 6083
Impala 2.12.0.100[1]	Impala 2.12.0.100-1912 Release Notes on page 6141
MapR Monitoring	MapR Monitoring Components - EEP 6.3.1 Release Notes on page 6246
MapR Object Store 1.0.1	S3 Gateway 1.0.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6264

last-updated: Apr 11, 2022

Component	Release Notes
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
MapR Streams Tools	Kafka Connect JDBC 4.1.0-1808 Release Notes on page 6205 Kafka Connect HDFS 4.1.0-1808 Release Notes on page 6199 Kafka Connect 4.1.0-1808 Release Notes on page 6209 Kafka REST Proxy 4.1.0 - 2009 (MEP 6.3.1) Release Notes on page 6221 Kafka Schema Registry 4.1.1-1901 Release Notes on page 6233 Kafka Streams 1.1 - 2009 (MEP 6.3.1) Release Notes on page 6180 KSQL 4.1.1-1808 Release Notes on page 6191
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 5.1.0.300	Oozie 5.1.0.400 - 2009 (EEP 6.3.1) Release Notes on page 6280
Pig 0.16	Pig 0.16.0 - 2009 (EEP 6.3.1) Release Notes on page 6317
Sentry 1.7.0[3]	Sentry 1.7.0-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6335
Spark 2.4.4.0	Spark 2.4.4.0-1912 Release Notes on page 6368
Sqoop 1.4.7	Sqoop 1.4.7 - 2009 (EEP 7.0.0) Release Notes on page 6452
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469
Tez 0.9.1	Tez 0.9.1-2009 (EEP 6.3.1) Release Notes on page 6493

[1]Support for this component is subject to your license agreement.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

The EEP 6.3.1 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.8.1.200
- Elasticsearch 6.8.8.100
- Fluentd 1.10.3.100
- Grafana 6.0.2.200
- Kibana 6.8.8.100
- OpenTSDB 2.4.0

MapR Ecosystem Pack 6.3.0 Release Notes

This topic contains information about the components included in MapR Ecosystem Pack 6.3.0.

Release Date	December 2019
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-6.0 or MEP-6.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 6.x can be used with MapR Core 6.1.0. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 6.3.0 did not introduce any changes that affect application backward compatibility.

EEP 6.3.0 Components

The EEP 6.3.0 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.16.0.10[1]	Drill 1.16.0.10-1912 (EEP 6.3.0) Release Notes on page 5762
Flume 1.8	Flume 1.8.0-1904 Release Notes on page 5825
HBase 1.1.13.0	HBase 1.1.13.0-1912 (EEP 6.3.0) Release Notes on page 5871
Hive 2.3.6	Hive 2.3.6-1912 (EEP 6.3.0) Release Notes on page 5936
HttpFS 1.0	HttpFS-1.0-1904 Release Notes on page 6060
Hue 4.3.0.100[2]	Hue 4.3.0.100-1912 (EEP 6.3.0) Release Notes on page 6085
Impala 2.12.0.100[1]	Impala 2.12.0.100-1912 Release Notes on page 6141
MapR Data Access Gateway 2.0	MapR Data Access Gateway 2.0 Release Notes on page 5745
MapR Monitoring	MapR Monitoring Components - EEP 6.3.0 Release Notes on page 6247
MapR Object Store 1.0.1	S3 Gateway 1.0.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6264

last-updated: Apr 11, 2022

Component	Release Notes
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
MapR Streams Tools	Kafka Connect JDBC 4.1.0-1808 Release Notes on page 6205 Kafka Connect HDFS 4.1.0-1808 Release Notes on page 6199 Kafka Connect 4.1.0-1808 Release Notes on page 6209 Kafka REST Proxy 4.1.0-1912 Release Notes on page 6221 Kafka Schema Registry 4.1.1-1901 Release Notes on page 6233 Kafka Streams 1.1-1912 Release Notes on page 6181 KSQL 4.1.1-1808 Release Notes on page 6191
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 5.1.0.300	Oozie 5.1.0.300-1912 (EEP 6.3.0) Release Notes on page 6281
Pig 0.16	Pig 0.16.0-1912 Release Notes on page 6318
Sentry 1.7.0[3]	Sentry 1.7.0-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6335
Spark 2.4.4.0	Spark 2.4.4.0-1912 Release Notes on page 6368
Sqoop 1.4.7	Sqoop 1.4.7-1904 Release Notes on page 6457
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469
Tez 0.9.1	Tez 0.9.1-1912 (EEP 6.3.0) Release Notes on page 6494

[1]Support for this component is subject to your license agreement.

[2]The Spark Notebook UI in Hue is a beta feature.

[3]MapR support for Sentry is limited to Impala users.

The EEP 6.3.0 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.8.1.200
- Elasticsearch 6.5.3.200
- Fluentd 1.4.0.100
- Grafana 6.0.2.100
- Kibana 6.5.3.200
- OpenTSDB 2.4.0

MapR Ecosystem Pack 6.2.0 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 6.2.0.

Release Date	May 2019
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-6.0 or MEP-6.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 6.x can be used with MapR Core 6.1.0. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 6.2.0 did not introduce any changes that would impact application backward compatibility.

EEP 6.2.0 Components

The EEP 6.2.0 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.16.0.0[1]	Drill 1.16.0.0-1904 (EEP 6.2.0) Release Notes on page 5763
Flume 1.8	Flume 1.8.0-1904 Release Notes on page 5825
HBase-compatible API[2]	HBase 1.1.8-1904 Release Notes on page 5874
Hive 2.3.3	Hive 2.3.3-1904 (EEP 6.2.0, EEP 6.1.1, and EEP 6.0.2) Release Notes on page 5943
HttpFS 1.0	HttpFS-1.0-1904 Release Notes on page 6060
Hue 4.3.0.0[3]	Hue 4.3.0-1904 (EEP 6.2.0) Release Notes on page 6087
Impala 2.12.0.0[1]	Impala 2.12-1904 Release Notes on page 6142
MapR Data Access Gateway 2.0	MapR Data Access Gateway 2.0 Release Notes on page 5745
MapR Monitoring	MapR Monitoring Components - EEP 6.2.0 Release Notes on page 6248
MapR Object Store 1.0.1	S3 Gateway 1.0.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6264
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167

last-updated: Apr 11, 2022

Component	Release Notes
MapR Streams Tools	Kafka Connect JDBC 4.1.0-1808 Release Notes on page 6205 Kafka Connect HDFS 4.1.0-1808 Release Notes on page 6199 Kafka Connect 4.1.0-1808 Release Notes on page 6209 Kafka REST 4.1.0-1808 Release Notes on page 6222 Kafka Schema Registry 4.1.1-1901 Release Notes on page 6233 Kafka Streams 1.1-1808 Release Notes on page 6182 KSQL 4.1.1-1808 Release Notes on page 6191
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 5.1.0.200	Oozie 5.1.0.200-1904 (EEP 6.2.0) Release Notes on page 6282
Pig 0.16	Pig 0.16.0-1901 Release Notes on page 6319
Sentry 1.7.0[4]	Sentry 1.7.0-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6335
Spark 2.4.0.0[1]	Spark 2.4.0.0-1904 (EEP 6.2.0) Release Notes on page 6370
Sqoop 1.4.7	Sqoop 1.4.7-1904 Release Notes on page 6457
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469

[1]Support for this component is subject to your license agreement.

[2]For use only with MapR Database binary tables.

[3]The Spark Notebook UI in Hue is a beta feature.

[4]MapR support for Sentry is limited to Impala users.

The EEP 6.2.0 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.8.1.100
- Elasticsearch 6.5.3.100
- Fluentd 1.4.0.0
- Grafana 6.0.2.0
- Kibana 6.5.3.100
- OpenTSDB 2.4.0

MapR Ecosystem Pack 6.1.1 Release Notes

This topic contains information about the components included with the MapR Ecosystem Pack 6.1.1.

Release Date	May 2019
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-6.0 or MEP-6.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

last-updated: Apr 11, 2022

EEP 6.x can be used with MapR Core 6.1.0. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 6.1.1 did not introduce any changes that would impact application backward compatibility.

EEP 6.1.1 Components

The EEP 6.1.1 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.15.0.7[1]	Drill 1.15.0.0-1901 (EEP 6.1.0) Release Notes on page 5768
Flume 1.8	Flume 1.8.0-1904 Release Notes on page 5825
HBase-compatible API[1]	HBase 1.1.8-1904 Release Notes on page 5874
Hive 2.3.3	Hive 2.3.3-1904 (EEP 6.2.0, EEP 6.1.1, and EEP 6.0.2) Release Notes on page 5943
HttpFS 1.0	HttpFS-1.0-1904 Release Notes on page 6060
Hue 4.2	Hue 4.2.0-1904 Release Notes on page 6090
Impala 2.10.0[1]	Impala 2.10.0-1808 Release Notes on page 6144
MapR Data Access Gateway 2.0	MapR Data Access Gateway 2.0 Release Notes on page 5745
MapR Monitoring	MapR Monitoring Components - EEP 6.1.0 Release Notes on page 6249
MapR Object Store 1.0.1	S3 Gateway 1.0.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6264
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167

last-updated: Apr 11, 2022

Component	Release Notes
MapR Streams Tools	Kafka Connect JDBC 4.1.0-1808 Release Notes on page 6205 Kafka Connect HDFS 4.1.0-1808 Release Notes on page 6199 Kafka Connect 4.1.0-1808 Release Notes on page 6209 Kafka REST 4.1.0-1808 Release Notes on page 6222 Kafka Schema Registry 4.1.1-1901 Release Notes on page 6233 Kafka Streams 1.1-1808 Release Notes on page 6182 KSQL 4.1.1-1808 Release Notes on page 6191
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 5.1.0.100	Oozie 5.1.0-1904 (EEP 6.1.1) Release Notes on page 6283
Pig 0.16	Pig 0.16.0-1901 Release Notes on page 6319
Sentry 1.7.0[2]	Sentry 1.7.0-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6335
Spark 2.3.3.100[1]	Spark 2.3.3.0-1904 (EEP 6.1.1 and EEP 6.0.2) Release Notes on page 6374
Sqoop 1.4.7	Sqoop 1.4.7-1904 Release Notes on page 6457
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469

[1]For use only with MapR Database binary tables.

[2]MapR support for Sentry is limited to Impala users.

The EEP 6.1.0 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.8.1.1
- Elasticsearch 6.5.3.1
- Fluentd 1.3.2.1
- Grafana 5.4.2.1
- Kibana 6.5.3.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 6.1.0 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 6.1.0.

Release Date	February 2019
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 6.x can be used with MapR Core 6.1.0. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

last-updated: Apr 11, 2022

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 6.1.0 Components

The EEP 6.1.0 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.15.0.0[1]	Drill 1.15.0.0-1901 (EEP 6.1.0) Release Notes on page 5768
Flume 1.8	Flume 1.8.0-1901 Release Notes on page 5825
HBase-compatible API[2]	HBase 1.1.8-1901 Release Notes on page 5875
Hive 2.3.3[3]	Hive 2.3.3-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 5947
HttpFS 1.0	HttpFS-1.0-1901 (EEP 6.1.0) Release Notes on page 6061
Hue 4.2	Hue 4.2.0-1901 (EEP 6.1.0) Release Notes on page 6092
Impala 2.10.0[1]	Impala 2.10.0-1808 Release Notes on page 6144
MapR Data Access Gateway 2.0	MapR Data Access Gateway 2.0 Release Notes on page 5745
MapR Monitoring	MapR Monitoring Components - EEP 6.1.0 Release Notes on page 6249
MapR Object Store 1.0.1	S3 Gateway 1.0.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6264
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
MapR Streams Tools	Kafka Connect JDBC 4.1.0-1808 Release Notes on page 6205 Kafka Connect HDFS 4.1.0-1808 Release Notes on page 6199 Kafka Connect 4.1.0-1808 Release Notes on page 6209 Kafka REST 4.1.0-1808 Release Notes on page 6222 Kafka Schema Registry 4.1.1-1901 Release Notes on page 6233 Kafka Streams 1.1-1808 Release Notes on page 6182 KSQL 4.1.1-1808 Release Notes on page 6191
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265

last-updated: Apr 11, 2022

Component	Release Notes
Oozie 5.1.0.0	Oozie 5.1.0.0-1901 (EEP 6.1.0) Release Notes on page 6284
Pig 0.16	Pig 0.16.0-1901 Release Notes on page 6319
Sentry 1.7.0[4]	Sentry 1.7.0-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6335
Spark 2.3.2.0[1]	Spark 2.3.2.0-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6383
Sqoop 1.4.7	Sqoop 1.4.7-1808 (EEP 6.0.0) Release Notes on page 6458
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469

[1]Support for this component is subject to your license agreement.

[2]For use only with MapR Database binary tables.

[3]MapR Hive 2.3.3 is equivalent to Apache Hive 2.3.4.

[4]MapR support for Sentry is limited to Impala users.

The EEP 6.1.0 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.8.1.0
- Elasticsearch 6.5.3.0
- Fluentd 1.3.2.0
- Grafana 5.4.2.0
- Kibana 6.5.3.0
- OpenTSDB 2.4.0

MapR Ecosystem Pack 6.0.2 Release Notes

This topic contains information about the components included with the MapR Ecosystem Pack 6.0.2.

Release Date	May 2019
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-6.0 or MEP-6.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 6.x can be used with MapR Core 6.1.0. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

EEP 6.0.2 did not introduce any changes that would impact application backward compatibility.

EEP 6.0.2 Components

The EEP 6.0.2 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.14	Drill 1.14.0-1901 (EEP 6.0.1) Release Notes on page 5773
Flume 1.8	Flume 1.8.0-1904 Release Notes on page 5825
HBase-compatible API[1]	HBase 1.1.8-1904 Release Notes on page 5874
Hive 2.3.3	Hive 2.3.3-1904 (EEP 6.2.0, EEP 6.1.1, and EEP 6.0.2) Release Notes on page 5943
HttpFS 1.0	HttpFS-1.0-1904 Release Notes on page 6060
Hue 4.2	Hue 4.2.0-1904 Release Notes on page 6090
Impala 2.10.0	Impala 2.10.0-1808 Release Notes on page 6144
MapR Data Access Gateway	MapR Data Access Gateway 2.0 Release Notes on page 5745
MapR Monitoring	MapR Monitoring Components - EEP 6.0.0 Release Notes on page 6250
MapR Object Store 1.0.1	S3 Gateway 1.0.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6264
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
MapR Streams Tools	Kafka Connect JDBC 4.1.0-1808 Release Notes on page 6205 Kafka Connect HDFS 4.1.0-1808 Release Notes on page 6199 Kafka Connect 4.1.0-1808 Release Notes on page 6209 Kafka REST 4.1.0-1808 Release Notes on page 6222 Kafka Streams 1.1-1808 Release Notes on page 6182 KSQL 4.1.1-1808 Release Notes on page 6191
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1904 (EEP 6.0.2) Release Notes on page 6287
Pig 0.16	Pig 0.16.0-1901 Release Notes on page 6319
Sentry 1.7.0[2]	Sentry 1.7.0-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6335
Spark 2.3.3.0	Spark 2.3.3.0-1904 (EEP 6.1.1 and EEP 6.0.2) Release Notes on page 6374
Sqoop 1.4.7	Sqoop 1.4.7-1904 Release Notes on page 6457
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469

last-updated: Apr 11, 2022

[1]For use only with MapR Database binary tables.

[2]MapR support for Sentry is limited to Impala users.

The EEP 6.0.1 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.8.0
- Elasticsearch 6.2.3
- Fluentd 1.1.2
- Grafana 4.6.5
- Kibana 6.2.3
- OpenTSDB 2.4.0

MapR Ecosystem Pack 6.0.1 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 6.0.1.

Release Date	February 2019
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 6.x can be used with MapR Core 6.1.0. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 6.0.1 Components

The EEP 6.0.1 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.14[1]	Drill 1.14.0-1901 (EEP 6.0.1) Release Notes on page 5773
Flume 1.8	Flume 1.8.0-1901 Release Notes on page 5825
HBase-compatible API[2]	HBase 1.1.8-1901 Release Notes on page 5875
Hive 2.3[3]	Hive 2.3.3-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 5947
HttpFS 1.0	HttpFS-1.0-1901 (EEP 6.1.0) Release Notes on page 6061
Hue 4.2	Hue 4.2.0-1808 (EEP 6.0.0) Release Notes on page 6093
Impala 2.10.0[1]	Impala 2.10.0-1808 Release Notes on page 6144

Component	Release Notes
MapR Data Access Gateway	MapR Data Access Gateway 2.0 Release Notes on page 5745
MapR Monitoring	MapR Monitoring Components - EEP 6.0.0 Release Notes on page 6250
MapR Object Store 1.0.1	S3 Gateway 1.0.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6264
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
MapR Streams Tools	Kafka Connect JDBC 4.1.0-1808 Release Notes on page 6205 Kafka Connect HDFS 4.1.0-1808 Release Notes on page 6199 Kafka Connect 4.1.0-1808 Release Notes on page 6209 Kafka REST 4.1.0-1808 Release Notes on page 6222 Kafka Streams 1.1-1808 Release Notes on page 6182 KSQL 4.1.1-1808 Release Notes on page 6191
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1901 Release Notes on page 6290
Pig 0.16	Pig 0.16.0-1901 Release Notes on page 6319
Sentry 1.7.0[4]	Sentry 1.7.0-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6335
Spark 2.3.2.0[1]	Spark 2.3.2.0-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6383
Sqoop 1.4.7	Sqoop 1.4.7-1808 (EEP 6.0.0) Release Notes on page 6458
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469

[1]Support for this component is subject to your license agreement.

[2]For use only with MapR Database binary tables.

[3]MapR Hive 2.3.0 is equivalent to Apache Hive 2.3.3.

[4]MapR support for Sentry is limited to Impala users.

The EEP 6.0.1 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.8.0
- Elasticsearch 6.2.3
- Fluentd 1.1.2
- Grafana 4.6.1
- Kibana 6.2.3

last-updated: Apr 11, 2022

- OpenTSDB 2.4.0

MapR Ecosystem Pack 6.0.0 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 6.0.0.

Release Date	September 2018
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

EEP 6.x can be used with MapR Core 6.1.0. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 6.0.0 Components

The EEP 6.0.0 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.14[1]	Drill 1.14.0-1808 (EEP 6.0.0) Release Notes on page 5775
Flume 1.8	Flume 1.8.0-1808 Release Notes on page 5826
HBase-compatible API[2]	HBase 1.1.8-1808 Release Notes on page 5876
Hive 2.3[3]	Hive 2.3.3-1808 (EEP 6.0.0) Release Notes on page 5953
HttpFS 1.0	HttpFS-1.0-1808 (EEP 6.0.0) Release Notes on page 6061
Hue 4.2	Hue 4.2.0-1808 (EEP 6.0.0) Release Notes on page 6093
Impala 2.10.0[1]	Impala 2.10.0-1808 Release Notes on page 6144
MapR Data Access Gateway	MapR Data Access Gateway 2.0 Release Notes on page 5745
MapR Monitoring	MapR Monitoring Components - EEP 6.0.0 Release Notes on page 6250
MapR Object Store 1.0.0	S3 Gateway 1.0.0-1808 (EEP 6.0.0) Release Notes on page 6264
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167

Component	Release Notes
MapR Streams Tools	Kafka Connect JDBC 4.1.0-1808 Release Notes on page 6205 Kafka Connect HDFS 4.1.0-1808 Release Notes on page 6199 Kafka Connect 4.1.0-1808 Release Notes on page 6209 Kafka REST 4.1.0-1808 Release Notes on page 6222 Kafka Streams 1.1-1808 Release Notes on page 6182 KSQL 4.1.1-1808 Release Notes on page 6191
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1808 (EEP 6.0.0) Release Notes on page 6291
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[4]	Sentry 1.7.0-1808 (EEP 6.0.0) Release Notes on page 6337
Spark 2.3.1[1]	Spark 2.3.1-1808 (EEP 6.0.0) Release Notes on page 6385
Sqoop 1.4.7	Sqoop 1.4.7-1808 (EEP 6.0.0) Release Notes on page 6458
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469

[1]Support for this component is subject to your license agreement.

[2]For use only with MapR Database binary tables.

[3]MapR Hive 2.3.0 is equivalent to Apache Hive 2.3.3.

[4]MapR support for Sentry is limited to Impala users.

The EEP 6.0.0 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.8.0
- Elasticsearch 6.2.3
- Fluentd 1.1.2
- Grafana 4.6.1
- Kibana 6.2.3
- OpenTSDB 2.4.0

MapR Ecosystem Pack 5.0.7 Release Notes

This topic contains information about the components included with the MapR Ecosystem Pack 5.0.7.

Release Date	May 2021
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-5.0 or MEP-5.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

last-updated: Apr 11, 2022

To understand which core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

MEP 5.0.7 did not introduce any changes that would impact application backward compatibility.

EEP 5.0.7 Components

The EEP 5.0.7 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.13.0	Drill 1.13.0-2009 (EEP 5.0.5) Release Notes on page 5779
Flume 1.8	Flume 1.8.0-1904 Release Notes on page 5825
HBase-compatible API[1]	HBase 1.1.8-2101 (EEP 5.0.6) Release Notes on page 5870
Hive 2.1.1	Hive 2.1.1-2009 (EEP 5.0.5) Release Notes on page 5958
HttpFS 1.0	HttpFS 1.0 - 2101 (EEP 5.0.6) Release Notes on page 6058
Hue 3.12.0	Hue 3.12.0 - 2009 (EEP 5.0.5) Release Notes on page 6098
Impala 2.10.0	Impala 2.10.0-1803 Release Notes on page 6145
Monitoring	MapR Monitoring Components - EEP 5.0.6 Release Notes on page 6252
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0 - 2009 (EEP 5.0.5) Release Notes on page 6286
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[2]	Sentry 1.7.0 - 2009 (MEP 5.0.5) Release Notes on page 6335
Spark 2.2.1	Spark 2.2.1-2009 (EEP 5.0.5) Release Notes on page 6378
Sqoop 1.4.6	Sqoop 1.4.6-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6460
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469

last-updated: Apr 11, 2022

Component	Release Notes
Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
Streams Tools	Kafka Connect HDFS 4.0.0-1808 Release Notes on page 6199 Kafka Connect JDBC 4.0.0-1803 Release Notes on page 6206 Kafka REST 4.0.0-1803 Release Notes on page 6223
Tez 0.8.4	Tez 0.8.4-2009 (EEP 5.0.5) Release Notes on page 6498

[1]For use only with MapR Database binary tables.

[2]Support for Sentry is limited to Impala users.

The EEP 5.0.7 repository contains the following ecosystem components that are supported for internal Monitoring use cases only:

- Collectd 5.7.2
- Elasticsearch 5.4.1
- Fluentd 0.14.20
- Grafana 7.5.2.0
- Kibana 5.4.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 5.0.6 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 5.0.6.

Release Date	January 2021
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-5.0 or MEP-5.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

MEP 5.0.6 did not introduce any changes that would impact application backward compatibility.

EEP 5.0.6 Components

The EEP 5.0.6 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.13.0	Drill 1.13.0-2009 (EEP 5.0.5) Release Notes on page 5779
Flume 1.8	Flume 1.8.0-1904 Release Notes on page 5825
HBase-compatible API[1]	HBase 1.1.8-2101 (EEP 5.0.6) Release Notes on page 5870
Hive 2.1.1	Hive 2.1.1-2009 (EEP 5.0.5) Release Notes on page 5958
HttpFS 1.0	HttpFS 1.0 - 2101 (EEP 5.0.6) Release Notes on page 6058
Hue 3.12.0	Hue 3.12.0 - 2009 (EEP 5.0.5) Release Notes on page 6098
Impala 2.10.0	Impala 2.10.0-1803 Release Notes on page 6145
Monitoring	MapR Monitoring Components - EEP 5.0.6 Release Notes on page 6252
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0 - 2009 (EEP 5.0.5) Release Notes on page 6286
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[2]	Sentry 1.7.0 - 2009 (MEP 5.0.5) Release Notes on page 6335
Spark 2.2.1	Spark 2.2.1-2009 (EEP 5.0.5) Release Notes on page 6378
Sqoop 1.4.6	Sqoop 1.4.6-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6460
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469
Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
Streams Tools	Kafka Connect HDFS 4.0.0-1808 Release Notes on page 6199 Kafka Connect JDBC 4.0.0-1803 Release Notes on page 6206 Kafka REST 4.0.0-1803 Release Notes on page 6223
Tez 0.8.4	Tez 0.8.4-2009 (EEP 5.0.5) Release Notes on page 6498

[1]For use only with MapR Database binary tables.

[2]Support for Sentry is limited to Impala users.

last-updated: Apr 11, 2022

The EEP 5.0.6 repository contains the following ecosystem components that are supported for internal Monitoring use cases only:

- Collectd 5.7.2
- Elasticsearch 5.4.1
- Fluentd 0.14.20
- Grafana 4.6.5
- Kibana 5.4.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 5.0.5 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 5.0.5.

Release Date	September 2020
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-5.0 or MEP-5.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

MEP 5.0.5 did not introduce any changes that would impact application backward compatibility.

EEP 5.0.5 Components

The EEP 5.0.5 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.13.0	Drill 1.13.0-2009 (EEP 5.0.5) Release Notes on page 5779
Flume 1.8	Flume 1.8.0-1904 Release Notes on page 5825
HBase-compatible API[1]	HBase 1.1.8-2009 (EEP 5.0.5) Release Notes on page 5871
Hive 2.1.1	Hive 2.1.1-2009 (EEP 5.0.5) Release Notes on page 5958
HttpFS 1.0	HttpFS-1.0-1904 Release Notes on page 6060
Hue 3.12.0	Hue 3.12.0 - 2009 (EEP 5.0.5) Release Notes on page 6098

last-updated: Apr 11, 2022

Component	Release Notes
Impala 2.10.0	Impala 2.10.0-1803 Release Notes on page 6145
Monitoring	MapR Monitoring Components - EEP 5.0.5 Release Notes on page 6253
Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
Streams Tools	Kafka Connect HDFS 4.0.0-1808 Release Notes on page 6199 Kafka Connect JDBC 4.0.0-1803 Release Notes on page 6206 Kafka REST 4.0.0-1803 Release Notes on page 6223
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0 - 2009 (EEP 5.0.5) Release Notes on page 6286
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[2]	Sentry 1.7.0 - 2009 (MEP 5.0.5) Release Notes on page 6335
Spark 2.2.1	Spark 2.2.1-2009 (EEP 5.0.5) Release Notes on page 6378
Sqoop 1.4.6	Sqoop 1.4.6-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6460
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469
Tez 0.8.4	Tez 0.8.4-2009 (EEP 5.0.5) Release Notes on page 6498

[1]For use only with MapR Database binary tables.

[2]Support for Sentry is limited to Impala users.

The EEP 5.0.5 repository contains the following ecosystem components that are supported for internal Monitoring use cases only:

- Collectd 5.7.2
- Elasticsearch 5.4.1
- Fluentd 0.14.20
- Grafana 4.6.5
- Kibana 5.4.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 5.0.4 Release Notes

This topic contains information about the components included with the MapR Ecosystem Pack 5.0.4.

Release Date	December 2019

last-updated: Apr 11, 2022

Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-5.0 or MEP-5.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

MEP 5.0.4 did not introduce any changes that would impact application backward compatibility.

EEP 5.0.4 Components

The EEP 5.0.4 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.13	Drill 1.13.0-1912 (EEP 5.0.4) Release Notes on page 5780
Flume 1.8	Flume 1.8.0-1904 Release Notes on page 5825
HBase-compatible API[1]	HBase 1.1.8-1904 Release Notes on page 5874
Hive 2.1.1	Hive 2.1.1-1912 (EEP 5.0.4) Release Notes on page 5960
HttpFS 1.0	HttpFS-1.0-1904 Release Notes on page 6060
Hue 3.12	Hue 3.12.0-1912 (EEP 5.0.4) Release Notes on page 6100
Impala 2.10.0	Impala 2.10.0-1803 Release Notes on page 6145
MapR Monitoring	MapR Monitoring Components - EEP 5.0.0 Release Notes on page 6254
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
MapR Streams Tools	Kafka Connect HDFS 4.0.0-1808 Release Notes on page 6199 Kafka Connect JDBC 4.0.0-1803 Release Notes on page 6206 Kafka REST 4.0.0-1803 Release Notes on page 6223
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265

last-updated: Apr 11, 2022

Component	Release Notes
Oozie 4.3.0	Oozie 4.3.0-1912 (EEP 5.0.4) Release Notes on page 6286
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[2]	Sentry 1.7.0-1901 (EEP 5.0.1, EEP 4.1.3, and EEP 3.0.5) Release Notes on page 6336
Spark 2.2.1	Spark 2.2.1-1912 (EEP 5.0.4) Release Notes on page 6379
Sqoop 1.4.6	Sqoop 1.4.6-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6460
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469
Tez 0.8.4	Tez 0.8.4-1912 (EEP 5.0.4) Release Notes on page 6498

[1]For use only with MapR Database binary tables.

[2]MapR support for Sentry is limited to Impala users.

The EEP 5.0.4 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.2
- Elasticsearch 5.4.1
- Fluentd 0.14.20
- Grafana 4.6.5
- Kibana 5.4.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 5.0.3 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 5.0.3.

Release Date	May 2019
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-5.0 or MEP-5.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

MEP 5.0.3 did not introduce any changes that would impact application backward compatibility.

EEP 5.0.3 Components

The EEP 5.0.3 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.13	Drill 1.13.0-1901 (EEP 5.0.2) Release Notes on page 5782
Flume 1.8	Flume 1.8.0-1904 Release Notes on page 5825
HBase-compatible API[1]	HBase 1.1.8-1904 Release Notes on page 5874
Hive 2.1.1	Hive 2.1.1-1904 (EEP 5.0.3 and EEP 4.1.4) Release Notes on page 5962
HttpFS 1.0	HttpFS-1.0-1904 Release Notes on page 6060
Hue 3.12	Hue 3.12.0-1901 (EEP 4.1.3 and EEP 5.0.2) Release Notes on page 6101
Impala 2.10.0	Impala 2.10.0-1803 Release Notes on page 6145
MapR Monitoring	MapR Monitoring Components - EEP 5.0.0 Release Notes on page 6254
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
MapR Streams Tools	Kafka Connect HDFS 4.0.0-1808 Release Notes on page 6199 Kafka Connect JDBC 4.0.0-1803 Release Notes on page 6206 Kafka REST 4.0.0-1803 Release Notes on page 6223
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1904 (EEP 5.0.3) Release Notes on page 6288
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[2]	Sentry 1.7.0-1901 (EEP 5.0.1, EEP 4.1.3, and EEP 3.0.5) Release Notes on page 6336
Spark 2.2.1	Spark 2.2.1-1904 (EEP 5.0.3) Release Notes on page 6381
Sqoop 1.4.6	Sqoop 1.4.6-1904 (EEP 5.0.3 and EEP 4.1.4) Release Notes on page 6459
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469

[1]For use only with MapR Database binary tables.

[2]MapR support for Sentry is limited to Impala users.

The EEP 5.0.3 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.2

last-updated: Apr 11, 2022

- Elasticsearch 5.4.1
- Fluentd 0.14.20
- Grafana 4.6.5
- Kibana 5.4.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 5.0.2 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 5.0.2.

Release Date	February 2019
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 5.0.2 Components

The EEP 5.0.2 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.13	Drill 1.13.0-1901 (EEP 5.0.2) Release Notes on page 5782
Flume 1.8	Flume 1.8.0-1803 Release Notes on page 5827
HBase-compatible API[1]	HBase 1.1.8-1901 Release Notes on page 5875
Hive 2.1	Hive 2.1.1-1901 (EEP 4.1.3 and EEP 5.0.2) Release Notes on page 5964
HttpFS 1.0	HttpFS-1.0-1901 (EEP 6.1.0) Release Notes on page 6061
Hue 3.12	Hue 3.12.0-1901 (EEP 4.1.3 and EEP 5.0.2) Release Notes on page 6101
Impala 2.10.0	Impala 2.10.0-1803 Release Notes on page 6145
MapR Monitoring	MapR Monitoring Components - EEP 5.0.0 Release Notes on page 6254

Component	Release Notes
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
MapR Streams Tools	Kafka Connect HDFS 4.0.0-1808 Release Notes on page 6199 Kafka Connect JDBC 4.0.0-1803 Release Notes on page 6206 Kafka REST 4.0.0-1803 Release Notes on page 6223
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1901 Release Notes on page 6290
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[2]	Sentry 1.7.0-1901 (EEP 5.0.1, EEP 4.1.3, and EEP 3.0.5) Release Notes on page 6336
Spark 2.2.1	Spark 2.2.1-1901 (EEP 5.0.2) Release Notes on page 6389
Sqoop 1.4.6	Sqoop 1.4.6-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6460
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469

[1]For use only with MapR Database binary tables.

[2]MapR support for Sentry is limited to Impala users.

The EEP 5.0.2 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.2
- Elasticsearch 5.4.1
- Fluentd 0.14.20
- Grafana 4.6.1
- Kibana 5.4.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 5.0.1 Release Notes

This topic contains information about the components included with the MapR Ecosystem Pack 5.0.1.

Release Date	September 2018
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

last-updated: Apr 11, 2022

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 5.0.1 Components

The EEP 5.0.1 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.13	Drill 1.13.0-1808 Release Notes on page 5783
Flume 1.8	Flume 1.8.0-1803 Release Notes on page 5827
HBase-compatible API[1]	HBase 1.1.8-1808 Release Notes on page 5876
Hive 2.1	Hive 2.1.1-1808 (EEP 4.1.2 and EEP 5.0.1) Release Notes on page 5971
HttpFS 1.0	HttpFS-1.0-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes on page 6063
Hue 3.12	Hue 3.12.0-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6102
Impala 2.10.0	Impala 2.10.0-1803 Release Notes on page 6145
MapR Monitoring	MapR Monitoring Components - EEP 5.0.0 Release Notes on page 6254
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 and MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 and MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167
MapR Streams Tools	Kafka Connect HDFS 4.0.0-1808 Release Notes on page 6199 and Kafka Connect JDBC 4.0.0-1803 Release Notes on page 6206 and Kafka REST 4.0.0-1803 Release Notes on page 6223
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6292
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[2]	Sentry 1.7.0-1803 Release Notes on page 6338
Spark 2.2.1	Spark 2.2.1-1808 (EEP 5.0.1) Release Notes on page 6391
Sqoop 1.4.6	Sqoop 1.4.6-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6460
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469

[1]For use only with MapR Database binary tables.

last-updated: Apr 11, 2022

[2]MapR support for Sentry is limited to Impala users.

The EEP 5.0.1 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.2
- Elasticsearch 5.4.1
- Fluentd 0.14.20
- Grafana 4.6.1
- Kibana 5.4.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 5.0.0 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 5.0.0.

Release Date	March 2018
Repository Location	`https://package.mapr.hpe.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 5.0.0 Components

The EEP 5.0.0 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.13	Drill 1.13-1803 Release Notes on page 5784
Flume 1.8	Flume 1.8.0-1803 Release Notes on page 5827
HBase-compatible API[1]	HBase 1.1.8-1710 Release Notes on page 5877
Hive 2.1	Hive 2.1.1-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes on page 5978
HttpFS 1.0	HttpFS-1.0-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes on page 6063
Hue 3.12	Hue 3.12.0-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes on page 6104
Impala 2.10.0	Impala 2.10.0-1803 Release Notes on page 6145
MapR Monitoring	MapR Monitoring Components - EEP 5.0.0 Release Notes on page 6254

Component	Release Notes
MapR Streams Tools	Kafka Connect HDFS 4.0.0-1803 Release Notes on page 6200 and Kafka Connect JDBC 4.0.0-1803 Release Notes on page 6206 and Kafka REST 4.0.0-1803 Release Notes on page 6223
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1803 (EEP 5.0.0) Release Notes on page 6293
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[2]	Sentry 1.7.0-1803 Release Notes on page 6338
Spark 2.2.1	Spark 2.2.1-1803 (EEP 5.0.0) Release Notes on page 6397
Sqoop 1.4.6	Sqoop 1.4.6-1803 Release Notes on page 6461
Sqoop2 1.99.7	Sqoop2 1.99.7-1803 Release Notes on page 6469
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes on page 6166 and MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes on page 6167 and MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes on page 6167

[1]For use only with MapR Database binary tables.

[2]MapR support for Sentry is limited to Impala users.

The EEP 5.0.0 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.2
- Elasticsearch 5.4.1
- Fluentd 0.14.20
- Grafana 4.6.1
- Kibana 5.4.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 4.1.4 Release Notes

This topic contains information about the components included with the MapR Ecosystem Pack 4.1.4.

Release Date	May 2019
Repository Location	`https://package.mapr.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-4.0 or MEP-4.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on the version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release,

but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

Backward Compatibility

MEP 4.1.4 did not introduce any changes that would impact application backward compatibility.

EEP 4.1.4 Components

The EEP 4.1.4 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.12	Drill 1.12.0-1901 (EEP 4.1.3) Release Notes on page 5788
Flume 1.7	Flume 1.7.0-1703 Release Notes on page 5828
HBase-compatible API[1]	HBase 1.1.8-1904 Release Notes on page 5874
Hive 2.1.1	Hive 2.1.1-1904 (EEP 5.0.3 and EEP 4.1.4) Release Notes on page 5962
HttpFS 1.0	HttpFS-1.0-1904 Release Notes on page 6060
Hue 3.12	Hue 3.12.0-1901 (EEP 4.1.3 and EEP 5.0.2) Release Notes on page 6101
Impala 2.7.0	Impala 2.7.0-1710 Release Notes on page 6146
MapR Monitoring	MapR Monitoring Components - EEP 4.1.0 Release Notes on page 6255
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.9.1 - 1703 Release Notes on page 6168 MapR Event Store For Apache Kafka Python Client 0.9.2-1707 Release Notes on page 6169
MapR Streams Tools	Kafka Connect 2.0.1-1801 Release Notes on page 6210 Kafka REST 2.0.1-1803 Release Notes on page 6224
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1904 (EEP 4.1.4) Release Notes on page 6289
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[2]	Sentry 1.7.0-1901 (EEP 5.0.1, EEP 4.1.3, and EEP 3.0.5) Release Notes on page 6336
Spark 2.1.1	Spark 2.1.0-1904 (EEP 4.1.4) Release Notes on page 6393
Sqoop 1.4.6	Sqoop 1.4.6-1904 (EEP 5.0.3 and EEP 4.1.4) Release Notes on page 6459
Sqoop2 1.99.7	Sqoop2 1.99.7-1710 Release Notes on page 6470

[1]For use only with MapR Database binary tables.

[2]MapR support for Sentry is limited to Impala users.

The EEP 4.1.4 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

last-updated: Apr 11, 2022

- Collectd 5.7.2
- Elasticsearch 5.4.1
- Fluentd 0.14.20
- Grafana 4.6.5
- Kibana 5.4.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 4.1.3 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 4.1.3.

Release Date	February 2019
Repository Location	`https://package.mapr.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 4.1.3 Components

The EEP 4.1.3 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.12	Drill 1.12.0-1901 (EEP 4.1.3) Release Notes on page 5788
Flume 1.7	Flume 1.7.0-1703 Release Notes on page 5828
HBase-compatible API[1]	HBase 1.1.8-1901 Release Notes on page 5875
Hive 2.1	Hive 2.1.1-1901 (EEP 4.1.3 and EEP 5.0.2) Release Notes on page 5964
HttpFS 1.0	HttpFS-1.0-1901 (EEP 6.1.0) Release Notes on page 6061
Hue 3.12	Hue 3.12.0-1901 (EEP 4.1.3 and EEP 5.0.2) Release Notes on page 6101
Impala 2.7.0	Impala 2.7.0-1710 Release Notes on page 6146
MapR Monitoring	MapR Monitoring Components - EEP 4.1.0 Release Notes on page 6255

Component	Release Notes
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.9.1 - 1703 Release Notes on page 6168 MapR Event Store For Apache Kafka Python Client 0.9.2-1707 Release Notes on page 6169
MapR Streams Tools	Kafka Connect 2.0.1-1801 Release Notes on page 6210 Kafka REST 2.0.1-1803 Release Notes on page 6224
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1901 Release Notes on page 6290
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[2]	Sentry 1.7.0-1901 (EEP 5.0.1, EEP 4.1.3, and EEP 3.0.5) Release Notes on page 6336
Spark 2.1	Spark 2.1.0-1901 (EEP 4.1.3 and EEP 3.0.5) Release Notes on page 6394
Sqoop 1.4.6	Sqoop 1.4.6-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6460
Sqoop2 1.99.7	Sqoop2 1.99.7-1710 Release Notes on page 6470

[1]For use only with MapR Database binary tables.

[2]MapR support for Sentry is limited to Impala users.

The EEP 4.1.3 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.2
- Elasticsearch 5.4.1
- Fluentd 0.14.20
- Grafana 4.4.2
- Kibana 5.4.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 4.1.2 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 4.1.2.

Release Date	September 2018
Repository Location	`https://package.mapr.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but

last-updated: Apr 11, 2022

last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 4.1.2 Components

The EEP 4.1.2 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.12	Drill 1.12.0-1808 Release Notes on page 5789
Flume 1.7	Flume 1.7.0-1703 Release Notes on page 5828
HBase-compatible API[1]	HBase 1.1.8-1808 Release Notes on page 5876
Hive 2.1	Hive 2.1.1-1808 (EEP 4.1.2 and EEP 5.0.1) Release Notes on page 5971
HttpFS 1.0	HttpFS-1.0-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes on page 6063
Hue 3.12	Hue 3.12.0-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6102
Impala 2.7.0	Impala 2.7.0-1710 Release Notes on page 6146
MapR Monitoring	MapR Monitoring Components - EEP 4.1.0 Release Notes on page 6255
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.9.1 - 1703 Release Notes on page 6168 and MapR Event Store For Apache Kafka Python Client 0.9.2-1707 Release Notes on page 6169
MapR Streams Tools	Kafka Connect 2.0.1-1801 Release Notes on page 6210 and Kafka REST 2.0.1-1803 Release Notes on page 6224
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6292
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[2]	Sentry 1.7.0-1803 Release Notes on page 6338
Spark 2.1	Spark 2.1.0-1808 (EEP 3.0.4 and EEP 4.1.2) Release Notes on page 6395
Sqoop 1.4.6	Sqoop 1.4.6-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6460
Sqoop2 1.99.7	Sqoop2 1.99.7-1710 Release Notes on page 6470

[1]For use only with MapR Database binary tables.

[2]MapR support for Sentry is limited to Impala users.

The EEP 4.1.2 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.2
- Elasticsearch 5.4.1
- Fluentd 0.14.20

- Grafana 4.4.2
- Kibana 5.4.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 4.1.1 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 4.1.1.

Release Date	March 2018
Repository Location	`https://package.mapr.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 4.1.1 Components

The EEP 4.1.1 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.12	Drill 1.12.0-1801 Release Notes on page 5790
Flume 1.7	Flume 1.7.0-1703 Release Notes on page 5828
HBase-compatible API[1]	HBase 1.1.8-1710 Release Notes on page 5877
Hive 2.1	Hive 2.1.1-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes on page 5978
HttpFS 1.0	HttpFS-1.0-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes on page 6063
Hue 3.12	Hue 3.12.0-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes on page 6104
Impala 2.7.0	Impala 2.7.0-1710 Release Notes on page 6146
MapR Monitoring	MapR Monitoring Components - EEP 4.1.0 Release Notes on page 6255
MapR Streams Tools	Kafka Connect 2.0.1-1801 Release Notes on page 6210 and Kafka REST 2.0.1-1803 Release Notes on page 6224
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1803 (EEP 4.1.1) Release Notes on page 6294
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[2]	Sentry 1.7.0-1803 Release Notes on page 6338

last-updated: Apr 11, 2022

Component	Release Notes
Spark 2.1	Spark 2.1.0-1803 (EEP 4.1.1) Release Notes on page 6400
Sqoop 1.4.6	Sqoop 1.4.6-1803 Release Notes on page 6461
Sqoop2 1.99.7	Sqoop2 1.99.7-1710 Release Notes on page 6470
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.9.1 - 1703 Release Notes on page 6168 and MapR Event Store For Apache Kafka Python Client 0.9.2-1707 Release Notes on page 6169

[1]For use only with MapR Database binary tables.

[2]MapR support for Sentry is limited to Impala users.

The EEP 4.1.1 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.2
- Elasticsearch 5.4.1
- Fluentd 0.14.20
- Grafana 4.4.2
- Kibana 5.4.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 4.1.0 Release Notes

This topic contains information about the components included with the MapR Ecosystem Pack 4.1.0.

Release Date	February 2018
Repository Location	`https://package.mapr.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 4.1.0 Components

The EEP 4.1.0 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.12	Drill 1.12.0-1801 Release Notes on page 5790
Flume 1.7	Flume 1.7.0-1703 Release Notes on page 5828

Component	Release Notes
HBase-compatible API[1]	HBase 1.1.8-1710 Release Notes on page 5877
Hive 2.1	Hive 2.1.1-1710 Release Notes on page 5984
HttpFS 1.0	HttpFS-1.0-1710 Release Notes on page 6065
Hue 3.12	Hue 3.12.0-1710 Release Notes on page 6106
Impala 2.7.0	Impala 2.7.0-1710 Release Notes on page 6146
MapR Monitoring	MapR Monitoring Components - EEP 4.1.0 Release Notes on page 6255
MapR Streams Tools	Kafka Connect 2.0.1-1801 Release Notes on page 6210 and Kafka REST 2.0.1-1710 Release Notes on page 6225
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1801 Release Notes on page 6296
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[2]	Sentry 1.7.0-1703 Release Notes on page 6339
Spark 2.1	Spark 2.1.0-1801 Release Notes on page 6403
Sqoop 1.4.6	Sqoop 1.4.6-1710 Release Notes on page 6462
Sqoop2 1.99.7	Sqoop2 1.99.7-1710 Release Notes on page 6470
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.9.1 - 1703 Release Notes on page 6168 and MapR Event Store For Apache Kafka Python Client 0.9.2-1707 Release Notes on page 6169

[1]For use only with MapR Database binary tables.

[2]MapR support for Sentry is limited to Impala users.

The EEP 4.1.0 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.2
- Elasticsearch 5.4.1
- Fluentd 0.14.20
- Grafana 4.4.2
- Kibana 5.4.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 4.0.0 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 4.0.0.

Release Date	November 2017
Repository Location	`https://package.mapr.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

last-updated: Apr 11, 2022

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 4.0.0 Components

The EEP 4.0.0 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.11	Drill 1.11.0-1710 Release Notes on page 5794
Flume 1.7	Flume 1.7.0-1703 Release Notes on page 5828
HBase-compatible API[1]	HBase 1.1.8-1710 Release Notes on page 5877
Hive 2.1	Hive 2.1.1-1710 Release Notes on page 5984
HttpFS 1.0	HttpFS-1.0-1710 Release Notes on page 6065
Hue 3.12	Hue 3.12.0-1710 Release Notes on page 6106
Impala 2.7.0	Impala 2.7.0-1710 Release Notes on page 6146
MapR Monitoring	MapR Monitoring Components - EEP 4.0 Release Notes on page 6256
MapR Streams Tools	Kafka Connect HDFS 2.0.1-1710 Release Notes on page 6201 and Kafka REST 2.0.1-1710 Release Notes on page 6225
Myriad 0.2	Myriad 0.2-1710 Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1710 Release Notes on page 6297
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[2]	Sentry 1.7.0-1703 Release Notes on page 6339
Spark 2.1	Spark 2.1.0-1710 Release Notes on page 6405
Sqoop 1.4.6	Sqoop 1.4.6-1710 Release Notes on page 6462
Sqoop2 1.99.7	Sqoop2 1.99.7-1710 Release Notes on page 6470
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.9.1 - 1703 Release Notes on page 6168 and MapR Event Store For Apache Kafka Python Client 0.9.2-1707 Release Notes on page 6169

[1]For use only with MapR Database binary tables.

[2]MapR support for Sentry is limited to Impala users.

The EEP 4.0.0 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.2
- Elasticsearch 5.4.1

- Fluentd 0.14.20
- Grafana 4.4.2
- Kibana 5.4.1
- OpenTSDB 2.4.0

MapR Ecosystem Pack 3.0.5 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 3.0.5.

Release Date	February 2019
Repository Location	`https://package.mapr.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 3.0.5 Components

The EEP 3.0.5 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.10	Drill 1.10.0-1808 Release Notes on page 5799
Flume 1.7	Flume 1.7.0-1703 Release Notes on page 5828
HBase 1.1.8	HBase 1.1.8-1703 Release Notes on page 5877
Hive 2.1	Hive 2.1.1-1901 (EEP 3.0.5) Release Notes on page 5968
HttpFS 1.0	HttpFS-1.0-1803 (EEP 3.0.3) Release Notes on page 6064
Hue 3.12	Hue 3.12.0-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6102
Impala 2.7.0	Impala 2.7.0-1710 Release Notes on page 6146
Mahout 0.12.0[2]	Mahout 0.12.0-1611 Release Notes on page 6161
MapR Monitoring	MapR Monitoring Components - EEP 3.x.x Release Notes on page 6257
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.9.1 - 1703 Release Notes on page 6168 MapR Event Store For Apache Kafka Python Client 0.9.2-1707 Release Notes on page 6169

last-updated: Apr 11, 2022

Component	Release Notes
MapR Streams Tools	Kafka Connect 2.0.1-1707 Release Notes on page 6211 Kafka REST 2.0.1-1707 Release Notes on page 6226
Myriad 0.1	Myriad Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1901 Release Notes on page 6290
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[3]	Sentry 1.7.0-1901 (EEP 5.0.1, EEP 4.1.3, and EEP 3.0.5) Release Notes on page 6336
Spark 2.1	Spark 2.1.0-1901 (EEP 4.1.3 and EEP 3.0.5) Release Notes on page 6394
Sqoop 1.4.6	Sqoop 1.4.6-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6460
Sqoop2 1.99.7	Sqoop2 1.99.7-1611 Release Notes on page 6471
Storm 0.10.0[4]	Storm 0.10.0-1703 Release Notes on page 6475

[2]Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

[3]MapR support for Sentry is limited to Impala users.

[4]Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

The EEP 3.0.5 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.1
- Elasticsearch 2.3.3
- Fluentd 0.14.00
- Grafana 4.1.2
- Kibana 4.5.4
- OpenTSDB 2.3.0

MapR Ecosystem Pack 3.0.4 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 3.0.4.

Release Date	September 2017
Repository Location	`https://package.mapr.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 3.0.4 Components

The EEP 3.0.4 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1808 Release Notes on page 5739
Drill 1.10	Drill 1.10.0-1808 Release Notes on page 5799
Flume 1.7	Flume 1.7.0-1703 Release Notes on page 5828
HBase 1.1.8	HBase 1.1.8-1703 Release Notes on page 5877
Hive 2.1	Hive 2.1.1-1808 (EEP 3.0.4) Release Notes on page 5975
HttpFS 1.0	HttpFS-1.0-1803 (EEP 3.0.3) Release Notes on page 6064
Hue 3.12	Hue 3.12.0-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6102
Impala 2.7.0	Impala 2.7.0-1710 Release Notes on page 6146
Mahout 0.12.0[2]	Mahout 0.12.0-1611 Release Notes on page 6161
MapR Monitoring	MapR Monitoring Components - EEP 3.x.x Release Notes on page 6257
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.9.1 - 1703 Release Notes on page 6168 and MapR Event Store For Apache Kafka Python Client 0.9.2-1707 Release Notes on page 6169
MapR Streams Tools	Kafka Connect 2.0.1-1707 Release Notes on page 6211 and Kafka REST 2.0.1-1707 Release Notes on page 6226
Myriad 0.1	Myriad Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6292
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[3]	Sentry 1.7.0-1803 Release Notes on page 6338
Spark 2.1	Spark 2.1.0-1808 (EEP 3.0.4 and EEP 4.1.2) Release Notes on page 6395
Sqoop 1.4.6	Sqoop 1.4.6-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes on page 6460
Sqoop2 1.99.7	Sqoop2 1.99.7-1611 Release Notes on page 6471
Storm 0.10.0[4]	Storm 0.10.0-1703 Release Notes on page 6475

[2]Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

[3]MapR support for Sentry is limited to Impala users.

[4]Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

The EEP 3.0.4 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.1
- Elasticsearch 2.3.3

- Fluentd 0.14.00
- Grafana 4.1.2
- Kibana 4.5.4
- OpenTSDB 2.3.0

MapR Ecosystem Pack 3.0.3 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 3.0.3.

Release Date	March 2017
Repository Location	`https://package.mapr.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 3.0.3 Components

The EEP 3.0.3 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.10	Drill 1.10.0-1707 Release Notes on page 5801
Flume 1.7	Flume 1.7.0-1703 Release Notes on page 5828
HBase 1.1.8	HBase 1.1.8-1703 Release Notes on page 5877
Hive 2.1	Hive 2.1.1-1803 (EEP 3.0.3) Release Notes on page 5982
HttpFS 1.0	HttpFS-1.0-1803 (EEP 3.0.3) Release Notes on page 6064
Hue 3.12	Hue 3.12.0-1803 (EEP 3.0.3) Release Notes on page 6105
Impala 2.7.0	Impala 2.7.0-1710 Release Notes on page 6146
Mahout 0.12.0[2]	Mahout 0.12.0-1611 Release Notes on page 6161
MapR Monitoring	MapR Monitoring Components - EEP 3.x.x Release Notes on page 6257
MapR Streams Tools	Kafka Connect 2.0.1-1707 Release Notes on page 6211 and Kafka REST 2.0.1-1707 Release Notes on page 6226
Myriad 0.1	Myriad Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1803 (EEP 3.0.3) Release Notes on page 6295

Component	Release Notes
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[3]	Sentry 1.7.0-1803 Release Notes on page 6338
Spark 2.1	Spark 2.1.0-1803 (EEP 3.0.3) Release Notes on page 6401
Sqoop 1.4.6	Sqoop 1.4.6-1803 Release Notes on page 6461
Sqoop2 1.99.7	Sqoop2 1.99.7-1611 Release Notes on page 6471
Storm 0.10.0[4]	Storm 0.10.0-1703 Release Notes on page 6475
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.9.1 - 1703 Release Notes on page 6168 and MapR Event Store For Apache Kafka Python Client 0.9.2-1707 Release Notes on page 6169

[2]Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

[3]MapR support for Sentry is limited to Impala users.

[4]Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

The EEP 3.0.3 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.1
- Elasticsearch 2.3.3
- Fluentd 0.14.00
- Grafana 4.1.2
- Kibana 4.5.4
- OpenTSDB 2.3.0

MapR Ecosystem Pack 3.0.2 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 3.0.2.

Release Date	November 2017
Repository Location	`https://package.mapr.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 3.0.2 Components

The EEP 3.0.2 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.10	Drill 1.10.0-1707 Release Notes on page 5801
Flume 1.7	Flume 1.7.0-1703 Release Notes on page 5828
HBase 1.1.8	HBase 1.1.8-1703 Release Notes on page 5877
Hive 2.1	Hive 2.1.1-1710 Release Notes on page 5984
HttpFS 1.0	HttpFS-1.0-1710 Release Notes on page 6065
Hue 3.12	Hue 3.12.0-1710 Release Notes on page 6106
Impala 2.7.0	Impala 2.7.0-1710 Release Notes on page 6146
Mahout 0.12.0[2]	Mahout 0.12.0-1611 Release Notes on page 6161
MapR Monitoring	MapR Monitoring Components - EEP 3.x.x Release Notes on page 6257
MapR Streams Tools	Kafka Connect 2.0.1-1707 Release Notes on page 6211 and Kafka REST 2.0.1-1707 Release Notes on page 6226
Myriad 0.1	Myriad Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1710 Release Notes on page 6297
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[3]	Sentry 1.7.0-1703 Release Notes on page 6339
Spark 2.1	Spark 2.1.0-1710 Release Notes on page 6405
Sqoop 1.4.6	Sqoop 1.4.6-1710 Release Notes on page 6462
Sqoop2 1.99.7	Sqoop2 1.99.7-1611 Release Notes on page 6471
Storm 0.10.0[4]	Storm 0.10.0-1703 Release Notes on page 6475
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.9.1 - 1703 Release Notes on page 6168 and MapR Event Store For Apache Kafka Python Client 0.9.2-1707 Release Notes on page 6169

[2]Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

[3]MapR support for Sentry is limited to Impala users.

[4]Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

The EEP 3.0.2 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.1
- Elasticsearch 2.3.3
- Fluentd 0.14.00
- Grafana 4.1.2
- Kibana 4.5.4
- OpenTSDB 2.3.0

last-updated: Apr 11, 2022

MapR Ecosystem Pack 3.0.1 Release Notes

This topic contains information about the components included with the MapR Ecosystem Pack 3.0.1.

Release Date	August 2017
Repository Location	`https://package.mapr.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 3.0.1 Components

The EEP 3.0.1 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.10	Drill 1.10.0-1707 Release Notes on page 5801
Flume 1.7	Flume 1.7.0-1703 Release Notes on page 5828
HBase 1.1.8	HBase 1.1.8-1703 Release Notes on page 5877
Hive 2.1	Hive 2.1.1-1707 Release Notes on page 5987
HttpFS 1.0	HttpFS 1.0-1703 Release Notes on page 6066
Hue 3.12	Hue 3.12.0-1707 Release Notes on page 6109
Impala 2.7.0	Impala 2.7.0-1707 Release Notes on page 6147
Mahout 0.12.0[2]	Mahout 0.12.0-1611 Release Notes on page 6161
MapR Monitoring	MapR Monitoring Components - EEP 3.x.x Release Notes on page 6257
MapR Streams Tools	Kafka Connect 2.0.1-1707 Release Notes on page 6211 and Kafka REST 2.0.1-1707 Release Notes on page 6226
Myriad 0.1	Myriad Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1707 Release Notes on page 6298
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[3]	Sentry 1.7.0-1703 Release Notes on page 6339
Spark 2.1	Spark 2.1.0-1707 Release Notes on page 6407
Sqoop 1.4.6	Sqoop 1.4.6-1707 Release Notes on page 6464
Sqoop2 1.99.7	Sqoop2 1.99.7-1611 Release Notes on page 6471
Storm 0.10.0[4]	Storm 0.10.0-1703 Release Notes on page 6475

last-updated: Apr 11, 2022

Component	Release Notes
MapR Streams Clients	MapR Event Store For Apache Kafka C Client 0.9.1 - 1703 Release Notes on page 6168 and MapR Event Store For Apache Kafka Python Client 0.9.2-1707 Release Notes on page 6169

[2]Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

[3]MapR support for Sentry is limited to Impala users.

[4]Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

The EEP 3.0.1 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.1
- Elasticsearch 2.3.3
- Fluentd 0.14.00
- Grafana 4.1.2
- Kibana 4.5.4
- OpenTSDB 2.3.0

MapR Ecosystem Pack 3.0 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 3.0.

Release Date	April 2017
Repository Location	https://package.mapr.com/releases/MEP/[1]

[1]The MEP-<version> directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). The three-digit EEP directory contains a fixed EEP version with patches. Use this EEP version if you prefer to do manual installs and do not require patch updates.

The two-digit EEP directory contains the latest EEP and patches and is continuously updated. The MapR installer uses the two-digit EEP directory to make new patches available automatically without the need for system reconfiguration.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.3.3-1808, 2.3.3 refers to the Hive version number, and 1808 typically indicates an August 2018 release, but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 3.0 Components

The EEP 3.0 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.10	Drill 1.10.0-1703 Release Notes on page 5802

Component	Release Notes
Flume 1.7	Flume 1.7.0-1703 Release Notes on page 5828
HBase 1.1.8	HBase 1.1.8-1703 Release Notes on page 5877
Hive 2.1	Hive 2.1-1703 Release Notes on page 5992
HttpFS 1.0	HttpFS 1.0-1703 Release Notes on page 6066
Hue 3.12	Hue 3.12.0-1703 Release Notes on page 6111
Impala 2.7.0	Impala 2.7.0 - 1703 Release Notes on page 6148
Mahout 0.12.0[2]	Mahout 0.12.0-1611 Release Notes on page 6161
MapR Monitoring	MapR Monitoring Components - EEP 3.x.x Release Notes on page 6257
MapR Event Store For Apache Kafka Tools	Kafka Connect 2.0.1-1703 (EEP 3.x) Release Notes on page 6211 and Kafka REST 2.0.1-1703 Release Notes on page 6227
Myriad 0.1	Myriad Release Notes on page 6265
Oozie 4.3.0	Oozie 4.3.0-1703 Release Notes on page 6298
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.7.0[3]	Sentry 1.7.0-1703 Release Notes on page 6339
Spark 2.1	Spark 2.1.0-1703 Release Notes on page 6412
Sqoop 1.4.6	Sqoop 1.4.6-1703 Release Notes on page 6465
Sqoop2 1.99.7	Sqoop2 1.99.7-1611 Release Notes on page 6471
Storm 0.10.0[4]	Storm 0.10.0-1703 Release Notes on page 6475
MapR Event Store For Apache Kafka Clients	MapR Event Store For Apache Kafka C Client 0.9.1 - 1703 Release Notes on page 6168 and MapR Event Store For Apache Kafka Python Client 0.9.2 - 1703 Release Notes on page 6169

[2]Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

[3]MapR support for Sentry is limited to Impala users.

[4]Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

The EEP 3.0 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.1
- Elasticsearch 2.3.3
- Fluentd 0.14.00
- Grafana 4.1.2
- Kibana 4.5.4
- OpenTSDB 2.3.0

MapR Ecosystem Pack 2.0.3 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 2.0.3.

Release Date	November 2017

last-updated: Apr 11, 2022

Repository Location	`https://package.mapr.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.1.1-1803, 2.1.1 refers to the Hive version number and 1803 typically indicates a March 2018 release but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 2.0.3 Components

The EEP 2.0.3 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.9	Drill 1.9.0-1703 Release Notes on page 5803
Flume 1.6	Flume 1.6.0-1602 Release Notes on page 5829
HBase 1.1.8	HBase 1.1.8-1703 Release Notes on page 5877
Hive 1.2.1[2]	Hive 1.2.1-1710 Release Notes on page 5995
HttpFS 1.0	HttpFS 1.0-1703 Release Notes on page 6066
Hue 3.10	Hue 3.10.0-1707 Release Notes on page 6115
Impala 2.5.0	Impala 2.5.0-1703 Release Notes on page 6149
Mahout 0.12.0[3]	Mahout 0.12.0-1611 Release Notes on page 6161
MapR Streams Tools	Kafka Connect 2.0.1-1707 Release Notes on page 6211 and Kafka REST 2.0.1-1707 Release Notes on page 6226
Myriad 0.1	Myriad Release Notes on page 6265
Oozie 4.2.0	Oozie 4.2.0-1710 (EEP 2.x) Release Notes on page 6299
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.6.0	Sentry 1.6.0-1707 Release Notes on page 6340
Spark 2.0.1[4]	Spark 2.0.1-1707 Release Notes on page 6416
Sqoop 1.4.6	Sqoop 1.4.6-1710 Release Notes on page 6462
Sqoop2 1.99.7	Sqoop2 1.99.7-1611 Release Notes on page 6471
Storm 0.10.0[5]	Storm 0.10.0-1611 Release Notes on page 6476

[2]Hive 1.2.1 includes backports of specific patches contained in Apache Hive 1.2.2. For details see the Hive 1.2.1-1703 Release Notes on page 5999.

[3]Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

[4]Spark 2.0.1-1703 includes backports of all the patches contained in Apache Spark 2.0.2.

[5]Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

The EEP 2.0.2 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.1
- Elasticsearch 2.3.3
- Fluentd 0.14.00
- Grafana 4.1.2
- Kibana 4.5.4
- OpenTSDB 2.3.0

MapR Ecosystem Pack 2.0.2 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 2.0.2.

Release Date	August 2017
Repository Location	`https://package.mapr.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.1.1-1803, 2.1.1 refers to the Hive version number and 1803 typically indicates a March 2018 release but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 2.0.2 Components

The EEP 2.0.2 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.9	Drill 1.9.0-1703 Release Notes on page 5803
Flume 1.6	Flume 1.6.0-1602 Release Notes on page 5829
HBase 1.1.8	HBase 1.1.8-1703 Release Notes on page 5877
Hive 1.2.1[2]	Hive 1.2.1-1707 Release Notes on page 5996
HttpFS 1.0	HttpFS 1.0-1703 Release Notes on page 6066
Hue 3.10	Hue 3.10.0-1707 Release Notes on page 6115
Impala 2.5.0	Impala 2.5.0-1703 Release Notes on page 6149
Mahout 0.12.0[3]	Mahout 0.12.0-1611 Release Notes on page 6161
MapR Streams Tools	Kafka Connect 2.0.1-1707 Release Notes on page 6211 and Kafka REST 2.0.1-1707 Release Notes on page 6226
Myriad 0.1	Myriad Release Notes on page 6265

Component	Release Notes
Oozie 4.2.0	Oozie 4.2.0-1707 (EEP 2.x) Release Notes on page 6301
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.6.0	Sentry 1.6.0-1707 Release Notes on page 6340
Spark 2.0.1[4]	Spark 2.0.1-1707 Release Notes on page 6416
Sqoop 1.4.6	Sqoop 1.4.6-1707 Release Notes on page 6464
Sqoop2 1.99.7	Sqoop2 1.99.7-1611 Release Notes on page 6471
Storm 0.10.0[5]	Storm 0.10.0-1611 Release Notes on page 6476

[2]Hive 1.2.1 includes backports of specific patches contained in Apache Hive 1.2.2. For details see the Hive 1.2.1-1703 Release Notes on page 5999.

[3]Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

[4]Spark 2.0.1-1703 includes backports of all the patches contained in Apache Spark 2.0.2.

[5]Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

The EEP 2.0.2 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.1
- Elasticsearch 2.3.3
- Fluentd 0.14.00
- Grafana 4.1.2
- Kibana 4.5.4
- OpenTSDB 2.3.0

MapR Ecosystem Pack 2.0.1 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 2.0.1.

Release Date	April 2017
Repository Location	https://package.mapr.com/releases/MEP/[1]

[1]The MEP-<version> directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). The three-digit EEP directory contains a fixed EEP version with patches. Use this EEP version if you prefer to do manual installs and do not require patch updates.

The two-digit EEP directory contains the latest EEP and patches and is continuously updated. The MapR installer uses the two-digit EEP directory to make new patches available automatically without the need for system reconfiguration.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.1.1-1803, 2.1.1 refers to the Hive version number and 1803 typically indicates a March 2018 release but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 2.0.1 Components

The EEP 2.0.1 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.9	Drill 1.9.0-1703 Release Notes on page 5803
Flume 1.6	Flume 1.6.0-1602 Release Notes on page 5829
HBase 1.1.8	HBase 1.1.8-1703 Release Notes on page 5877
Hive 1.2.1[2]	Hive 1.2.1-1703 Release Notes on page 5999
HttpFS 1.0	HttpFS 1.0-1703 Release Notes on page 6066
Hue 3.10	Hue 3.10.0-1703 Release Notes on page 6116
Impala 2.5.0	Impala 2.5.0-1703 Release Notes on page 6149
Mahout 0.12.0[3]	Mahout 0.12.0-1611 Release Notes on page 6161
MapR Event Store For Apache Kafka Tools	Kafka Connect 2.0.1-1703 (EEP 2.x) Release Notes on page 6212 and Kafka REST 2.0.1-1703 Release Notes on page 6227
Myriad 0.1	Myriad Release Notes on page 6265
Oozie 4.2.0	Oozie 4.2.0-1703 (EEP 2.x) Release Notes on page 6302
Pig 0.16	Pig 0.16.0-1703 Release Notes on page 6320
Sentry 1.6.0	Sentry 1.6.0-1606 Release Notes on page 6341
Spark 2.0.1[4]	Spark 2.0.1-1703 Release Notes on page 6419
Sqoop 1.4.6	Sqoop 1.4.6-1703 Release Notes on page 6465
Sqoop2 1.99.7	Sqoop2 1.99.7-1611 Release Notes on page 6471
Storm 0.10.0[5]	Storm 0.10.0-1611 Release Notes on page 6476

[2]Hive 1.2.1 includes backports of specific patches contained in Apache Hive 1.2.2. For details see the Hive 1.2.1-1703 Release Notes on page 5999.

[3]Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

[4]Spark 2.0.1-1703 includes backports of all the patches contained in Apache Spark 2.0.2.

[5]Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

The EEP 2.0.1 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.5.1
- Elasticsearch 2.3.3
- Fluentd 0.14.00
- Grafana 4.1.2
- Kibana 4.5.4
- OpenTSDB 2.3.0

MapR Ecosystem Pack 2.0 Release Notes

This topic contains information about the components included with the MapR Ecosystem Pack 2.0.

last-updated: Apr 11, 2022

Release Date	December 8, 2016
Repository Location	https://package.mapr.com/releases/MEP/[1]

[1]The MEP-<version> directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). The three-digit EEP directory contains a fixed EEP version with patches. Use this EEP version if you prefer to do manual installs and do not require patch updates.

The two-digit EEP directory contains the latest EEP and patches and is continuously updated. The MapR installer uses the two-digit EEP directory to make new patches available automatically without the need for system reconfiguration.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.1.1-1803, 2.1.1 refers to the Hive version number and 1803 typically indicates a March 2018 release but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 2.0 Components

The EEP 2.0 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.9	Drill 1.9 Release Notes
Flume 1.6	Flume 1.6.0-1602 Release Notes on page 5829
HBase 1.1.1	HBase 1.1-1602 Release Notes on page 5878
Hive 1.2.1	Hive 1.2.1-1611 Release Notes
HttpFS 1.0	HttpFS 1.0-1609 Release Notes
Hue 3.10	Hue 3.10.0-1611 Release Notes
Impala 2.5.0	Impala 2.5.0 - 1606 Release Notes on page 6149
Mahout 0.12.0[2]	Mahout 0.12.0-1611 Release Notes
MapR Event Store For Apache Kafka Tools	Kafka REST 2.0.1-1611 Release Notes on page 6227 Kafka Connect 2.0.1-1611 Release Notes on page 6212
Myriad 0.1	Myriad Release Notes on page 6265
Oozie 4.2.0	Oozie 4.2.0-1611 (EEP 2.x) Release Notes
Pig 0.16	Pig 0.16.0-1611 Release Notes
Sentry 1.6.0	Sentry 1.6.0-1606 Release Notes on page 6341
Spark 2.0.1	Spark 2.0.1-1611 Release Notes
Sqoop 1.4.6	Sqoop 1.4.6-1611 Release Notes
Sqoop2 1.99.7	Sqoop2 1.99.7-1611 Release Notes
Storm 0.10.0[3]	Storm 0.10.0-1611 Release Notes

[2]Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

[3]Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

The EEP 2.x repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.5.1
- Elasticsearch 2.3.3
- Fluentd 0.14.60
- Grafana 3.1.1
- Kibana 4.5.1
- OpenTSDB 2.2.1

MapR Ecosystem Pack 1.1.4 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 1.1.4.

Release Date	November 2017
Repository Location	`https://package.mapr.com/releases/MEP/`[1]

[1]The `MEP-<version>` directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.1.1-1803, 2.1.1 refers to the Hive version number and 1803 typically indicates a March 2018 release but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 1.1.4 Components

The EEP 1.1.3 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.8	Drill 1.8.0-1703 Release Notes on page 5806
Flume 1.6	Flume 1.6.0-1602 Release Notes on page 5829
HBase 1.1.8	HBase 1.1.8-1703 Release Notes on page 5877
Hive 1.2.1[2]	Hive 1.2.1-1703 Release Notes on page 5999
HttpFS 1.0	HttpFS 1.0-1703 Release Notes on page 6066
Hue 3.9	Hue 3.9.0-1707 Release Notes on page 6118
Impala 2.5.0	Impala 2.5.0-1703 Release Notes on page 6149
Mahout 0.12.0[3]	Mahout 0.12.0-1609 Release Notes on page 6161
Myriad 0.1	Myriad Release Notes on page 6265
Oozie 4.2.0	Oozie 4.2.0-1710 (EEP 1.x) Release Notes on page 6300
Pig 0.15	Pig 0.15.0-1703 Release Notes on page 6322

Component	Release Notes
Sentry 1.6.0	Sentry 1.6.0-1707 Release Notes on page 6340
Spark 1.6.1	Spark 1.6.1-1707 Release Notes on page 6418
Sqoop 1.4.6	Sqoop 1.4.6-1710 Release Notes on page 6462
Sqoop2 1.99.6	Sqoop2 1.99.6-1607 Release Notes on page 6472
Storm 0.10.0[4]	Storm 0.10.0-1611 Release Notes on page 6476

[2]Hive 1.2.1 includes backports of specific patches contained in Apache Hive 1.2.2. For details see the Hive 1.2.1-1703 Release Notes on page 5999.

[3]Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

[4]Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

The EEP 1.1.3 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.1
- Elasticsearch 2.3.3
- Fluentd 0.14.00
- Grafana 4.1.2
- Kibana 4.5.4
- OpenTSDB 2.3.0

MapR Ecosystem Pack 1.1.3 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 1.1.3.

Release Date	August 2017
Repository Location	https://package.mapr.com/releases/MEP/[1]

[1]The MEP-<version> directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). See Understanding Two-Digit and Three-Digit EEPs on page 5450.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.1.1-1803, 2.1.1 refers to the Hive version number and 1803 typically indicates a March 2018 release but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 1.1.3 Components

The EEP 1.1.3 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.8	Drill 1.8.0-1703 Release Notes on page 5806

Component	Release Notes
Flume 1.6	Flume 1.6.0-1602 Release Notes on page 5829
HBase 1.1.8	HBase 1.1.8-1703 Release Notes on page 5877
Hive 1.2.1[2]	Hive 1.2.1-1707 Release Notes on page 5996
HttpFS 1.0	HttpFS 1.0-1703 Release Notes on page 6066
Hue 3.9	Hue 3.9.0-1707 Release Notes on page 6118
Impala 2.5.0	Impala 2.5.0-1703 Release Notes on page 6149
Mahout 0.12.0[3]	Mahout 0.12.0-1609 Release Notes on page 6161
Myriad 0.1	Myriad Release Notes on page 6265
Oozie 4.2.0	Oozie 4.2.0-1707 (EEP 1.x) Release Notes on page 6301
Pig 0.15	Pig 0.15.0-1703 Release Notes on page 6322
Sentry 1.6.0	Sentry 1.6.0-1707 Release Notes on page 6340
Spark 1.6.1	Spark 1.6.1-1707 Release Notes on page 6418
Sqoop 1.4.6	Sqoop 1.4.6-1707 Release Notes on page 6464
Sqoop2 1.99.6	Sqoop2 1.99.6-1607 Release Notes on page 6472
Storm 0.10.0[4]	Storm 0.10.0-1611 Release Notes on page 6476

[2]Hive 1.2.1 includes backports of specific patches contained in Apache Hive 1.2.2. For details see the Hive 1.2.1-1703 Release Notes on page 5999.

[3]Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

[4]Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

The EEP 1.1.3 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.7.1
- Elasticsearch 2.3.3
- Fluentd 0.14.00
- Grafana 4.1.2
- Kibana 4.5.4
- OpenTSDB 2.3.0

MapR Ecosystem Pack 1.1.2 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 1.1.2.

Release Date	April 2017
Repository Location	https://package.mapr.com/releases/MEP/[1]

[1]The MEP-<version> directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-2.0 or MEP-2.0.0). The three-digit EEP directory contains a fixed EEP version with patches. Use this EEP version if you prefer to do manual installs and do not require patch updates.

last-updated: Apr 11, 2022

The two-digit EEP directory contains the latest EEP and patches and is continuously updated. The MapR installer uses the two-digit EEP directory to make new patches available automatically without the need for system reconfiguration.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.1.1-1803, 2.1.1 refers to the Hive version number and 1803 typically indicates a March 2018 release but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 1.1.2 Components

The EEP 1.1.2 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.8	Drill 1.8.0-1703 Release Notes on page 5806
Flume 1.6	Flume 1.6.0-1602 Release Notes on page 5829
HBase 1.1.8	HBase 1.1.8-1703 Release Notes on page 5877
Hive 1.2.1[2]	Hive 1.2.1-1703 Release Notes on page 5999
HttpFS 1.0	HttpFS 1.0-1703 Release Notes on page 6066
Hue 3.9	Hue 3.9.0-1703 Release Notes on page 6119
Impala 2.5.0	Impala 2.5.0-1703 Release Notes on page 6149
Mahout 0.12.0[3]	Mahout 0.12.0-1609 Release Notes on page 6161
Myriad 0.1	Myriad Release Notes on page 6265
Oozie 4.2.0	Oozie 4.2.0-1703 (EEP 1.x) Release Notes on page 6303
Pig 0.15	Pig 0.15.0-1703 Release Notes on page 6322
Sentry 1.6.0	Sentry 1.6.0-1606 Release Notes on page 6341
Spark 1.6.1	Spark 1.6.1-1703 Release Notes on page 6427
Sqoop 1.4.6	Sqoop 1.4.6-1703 Release Notes on page 6465
Sqoop2 1.99.6	Sqoop2 1.99.6-1607 Release Notes on page 6472
Storm 0.10.0[4]	Storm 0.10.0-1611 Release Notes on page 6476

[2]Hive 1.2.1 includes backports of specific patches contained in Apache Hive 1.2.2. For details see the Hive 1.2.1-1703 Release Notes on page 5999.

[3]Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

[4]Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

The EEP 1.1.2 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.5.1
- Elasticsearch 2.3.3

- Fluentd 0.14.00
- Grafana 4.1.2
- Kibana 4.5.4
- OpenTSDB 2.3.0

MapR Ecosystem Pack 1.1.1 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 1.1.1.

Release Date	December 9, 2016
Repository Location	https://package.mapr.com/releases/MEP/[1]

[1]The MEP-<version> directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-1.1 or MEP-1.1.0). The three-digit EEP directory contains a fixed EEP version with patches. Use this EEP version if you prefer to do manual installs and do not require patch updates.

The two-digit EEP directory contains the latest EEP and patches and is continuously updated. The MapR installer uses the two-digit EEP directory to make new patches available automatically without the need for system reconfiguration.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.1.1-1803, 2.1.1 refers to the Hive version number and 1803 typically indicates a March 2018 release but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 1.1.1 Components

The EEP 1.1.1 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740
Drill 1.8	Drill 1.8.0-1609 Release Notes
Flume 1.6	Flume 1.6.0-1602 Release Notes on page 5829
HBase 1.1	HBase 1.1-1602 Release Notes on page 5878
Hive 1.2	Hive 1.2.1-1611 Release Notes on page 6003
HttpFS 1.0	HttpFS 1.0-1609 Release Notes
Hue 3.9.0	Hue 3.9.0-1609 Release Notes
Impala 2.5.0	Impala 2.5.0 - 1606 Release Notes on page 6149
Mahout 0.12.0[2]	Mahout 0.12.0-1611 Release Notes on page 6161
Myriad 0.1	Myriad Release Notes on page 6265
Oozie 4.2.0	Oozie 4.2.0-1608 Release Notes on page 6305 and Oozie 4.2.0-1609 Release Notes and Oozie 4.2.0-1611 Release Notes
Pig 0.15	Pig 0.15.0-1611 Release Notes on page 6322
Sentry 1.6.0	Sentry 1.6.0-1606 Release Notes on page 6341

last-updated: Apr 11, 2022

Component	Release Notes
Spark 1.6.1	Spark 1.6.1-1608 Release Notes on page 6434 and Spark 1.6.1-1609 Release Notes and Spark 1.6.1-1611 Release Notes
Sqoop 1.4.6	Sqoop 1.4.6-1611 Release Notes on page 6465
Sqoop2 1.99.6	Sqoop2 1.99.6-1607 Release Notes on page 6472
Storm 0.10.0[3]	Storm 0.10.0-1611 Release Notes on page 6476

[2]Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

[3]Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

The EEP 1.1 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.5.1
- Elasticsearch 2.3.3
- Fluentd 0.14.00
- Grafana 3.1.1
- Kibana 4.5.1
- OpenTSDB 2.2.1

MapR Ecosystem Pack 1.1.0 Release Notes
This topic contains information about the components included with the MapR Ecosystem Pack 1.1.0.

Release Date	September 30, 2016
Repository Location	https://package.mapr.com/releases/MEP/[1]

[1]The MEP-<version> directory can be represented by a 2-digit number or a 3-digit number (for example, MEP-1.1 or MEP-1.1.0). The three-digit EEP directory contains a fixed EEP version with patches. Use this EEP version if you prefer to do manual installs and do not require patch updates.

The two-digit EEP directory contains the latest EEP and patches and is continuously updated. The MapR installer uses the two-digit EEP directory to make new patches available automatically without the need for system reconfiguration.

To understand which MapR Core versions can use this MEP, see EEP Support and Lifecycle Status on page 5531.

Release Note Naming Convention

The release note naming convention is based on version number and release date. For Hive 2.1.1-1803, 2.1.1 refers to the Hive version number and 1803 typically indicates a March 2018 release but last-minute changes in the release date can result in a slight mismatch between the naming convention and the actual release.

EEP 1.1.0 Components

The EEP 1.1.0 repository contains the following ecosystem components that are fully supported:

Component	Release Notes
AsyncHBase 1.7	AsyncHBase 1.7.0-1607 Release Notes on page 5740

Component	Release Notes
Drill 1.8	Drill 1.8.0-1609 Release Notes
Flume 1.6	Flume 1.6.0-1602 Release Notes on page 5829
HBase 1.1	HBase 1.1-1602 Release Notes on page 5878
Hive 1.2	Hive 1.2.1-1608 Release Notes on page 6005 and Hive 1.2.1-1609 Release Notes on page 6004
HttpFS 1.0	HttpFS 1.0-1609 Release Notes
Hue 3.9.0	Hue 3.9.0-1609 Release Notes
Impala 2.5.0	Impala 2.5.0 - 1606 Release Notes on page 6149
Mahout 0.12.0	Mahout 0.12.0-1609 Release Notes on page 6161
Myriad 0.1	Myriad Release Notes on page 6265
Oozie 4.2.0	Oozie 4.2.0-1608 Release Notes on page 6305 and Oozie 4.2.0-1609 Release Notes and Oozie 4.2.0-1611 Release Notes
Pig 0.15	Pig 0.15.0-1608 Release Notes on page 6323
Sentry 1.6.0	Sentry 1.6.0-1606 Release Notes on page 6341
Spark 1.6.1	Spark 1.6.1-1608 Release Notes on page 6434 and Spark 1.6.1-1609 Release Notes
Sqoop 1.4.6	Sqoop1.4.6-1609 Release Notes on page 6466
Sqoop2 1.99.6	Sqoop2 1.99.6-1607 Release Notes on page 6472
Storm 0.10.0	Storm 0.10.0-1609 Release Notes on page 6476

The EEP 1.1 repository contains the following ecosystem components that are supported for internal MapR Monitoring use cases only:

- Collectd 5.5.1
- Elasticsearch 2.3.3
- Fluentd 0.14.00
- Grafana 3.1.1
- Kibana 4.5.1
- OpenTSDB 2.2.0

Package Names for MapR Ecosystem Packs (EEPs)
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

To view the package names for an EEP:

1. Use a browser to navigate to https://package.mapr.hpe.com/releases/MEP/. A list of EEP links is displayed.
2. Click the link for your EEP. A list of operating system links is displayed.
3. Click the link for your operating system. The list of package names is displayed.

For more information about the supported EEPs, see EEP Components and OS Support on page 5536.

last-updated: Apr 11, 2022

For information about packages and dependencies, see Packages and Dependencies for MapR Software on page 68.

Airflow Release Notes

The release notes for the Airflow component included in the MapR Data Platform contain notes specific to data-fabric only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536 or EEP Support and Lifecycle Status on page 5531. To view release notes for prior data-fabric releases, see Previous Versions on page 6578.

Airflow 2.2.1.0 - 2201 (EEP 6.2.0-8.1.0) Release Notes

The following notes relate specifically to the MapR Data Platform Distribution for Apache Airflow. You may also be interested in the Apache Airflow home page.

Airflow Version	2.2.1.0
Release Date	January 2022
HPE Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/airflow
GitHub Release Tag	2.2.1.0-eep-810
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to http://package.mapr.hpe.com/releases/MEP/, and select your EEP(MEP) and OS to view the list of package names.
Documentation	• Apache Airflow • Installing Airflow • Airflow Providers

New in This Release

This is the first release of the Airflow component. Starting from EEP 8.1.0, the MapR Data Platform supports Apache Airflow in core releases 7.0.0 and 6.2.0. You can use Airflow to:

- Define, schedule, and monitor workflows.
- Orchestrate third-party systems to execute tasks.
- Analyze and manage workflows using the Airflow web interface.

A variety of operators and sensors are provided to integrate Airflow with the MapR Data Platform Database. In addition, release 7.0.0 includes operators, sensors, and transfers that enable Airflow to create and interact with S3 buckets. For more information, see Airflow Providers.

Fixes

None.

Known Issues and Limitations

- Airflow is not supported with FIPS-enabled nodes.

last-updated: Apr 11, 2022

- The Installer can install Airflow, but cannot set up MySQL as the backend database for Airflow. The default Airflow database is SQLite.
- S3 operators can be used with Airflow in release 7.0.0 but not in release 6.2.0 because native S3 support is not implemented in release 6.2.0.
- Airflow Amazon S3 providers might work with Minio S3 API, but HPE does not guarantee this functionality.
- Airflow requires a patch to operate in a release 6.2.0 cluster with security enabled.

Resolved Issues

- None.

AsyncHBase Release Notes

The release notes for AsyncHBase component contains notes specific to MapR only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

AsyncHBase 1.8.2-2009 Release Notes

The notes below relate to the MapR Data Platform. You may also be interested in the AsyncHBase Github page.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.8.2
Release Date	September 2020
MapR Version Interoperability	Component Versions for Released EEPs
Source on GitHub	https://github.com/mapr/asynchbase
GitHub Release Tag	v1.8.2-mapr-2009
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

Fixes

None.

Known Issues and Limitations

None.

Resolved Issues

None.

AsyncHBase 1.7.0-1808 Release Notes

last-updated: Apr 11, 2022

The notes below relate to the MapR Converged Data Platform. You may also be interested in the AsyncHBase Github page.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.7.0
Release Date	September 2018
MapR Version Interoperability	Pre-MapR 5.2: Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 MapR 5.2 and later: Component Versions for Released EEPs
Source on GitHub	https://github.com/mapr/asynchbase/tree/1.7.0-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

Fixes

This release by MapR includes the following fixes on the base release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
e2a97c2	2018-07-11	MAPRDB-1243: Fixed AsyncHBase not sorting column qualifiers in lexicographical order.

AsyncHBase 1.7.0-1607 Release Notes

The notes below relate to the MapR Converged Data Platform. You may also be interested in the AsyncHBase github page.

Version	1.7.0
Release Date	July 29, 2016
MapR Version Interoperability	Pre-MapR 5.2: Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 MapR 5.2 and later: EEP Components and OS Support
Source on GitHub	https://github.com/mapr/asynchbase/tree/v1.7.0-mapr-1607
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

Fixes

This release by MapR includes the following fixes on the base release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
0917694	2016-07-109	MapR-23932: Free memoryArena after performing GET from AsyncHBase. This fixes an issue where FreeArena() wasn't called for GET operations.
a0e33ad	2016-07-07	MapR-23881: Fixed an issue where OpenTSDB daemon gets stuck while running queries with large number of rows.
9faa290	2016-06-09	MapR-22952: Fixed Maven Protocol Buffers plugin in AsyncHBase pom file. Previously, the asynchbase build was broken because maven-protoc-plugin was not available in Maven Central anymore.
11006f0	2016-05-26	MapR-20677: Added support to delete exact cell version through AsyncHBase.

AsyncHBase 1.7.0-1603 Release Notes

The notes below relate to the MapR Converged Data Platform. You may also be interested in the AsyncHBase github page.

Version	1.7.0
Release Date	April 4, 2016
MapR Version Interoperability	See the Ecosystem Support Matrix.
Source on GitHub	https://github.com/mapr/asynchbase/tree/v1.7.0-mapr-1603
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-asynchbase-1.7.0.201603301412-1.noarch.rpm • mapr-asynchbase_1.7.0.201603301412_all.deb

New Features

This release of AsyncHBase includes the following behavior change that is specific to MapR:

Ability to define the default database that AsyncHBase connects to.	A new parameter `mapr.hbase.default.db` can be added to the asynchbase.conf file. You can configure this parameter to specify if AsyncHBase accesses HBase tables or MapR Database tables by default. If this value is not configured, AsyncHBase will determine which type of table is being accessed based on the table name.

Fixes

This release by MapR includes the following fixes on the base release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
f1d9c46	2016-03-18	MapR-22861: `mapr.hbase.default.db` parameter can now be set to indicate whether AsyncHBase connects to MapR Database tables or HBase tables by default.
7f563c4	2016-03-15	MapR-22797: Implemented the locateRegions functionality for MapR Database tables
f2b0a70	2016-03-09	MapR 22778: The AppendRequest behavior against HBase tables was modified so that it returns ArrayList<KeyValue> whenever it is used to return the new values after the append operation.

Known Issue

- 22911: In MapR 5.1, when the DeleteRequest API calls setDeleteAtTimestampOnly for a MapR Database table, the delete occurs for timestamps less than or equal to the given timestamp.

AsyncHBase 1.6.0-1504 Release Notes

Version	1.6.0-mapr-1504
Release Date	May 6, 2015
MapR Version Interoperability	See the Ecosystem Support Matrix.
Source on GitHub	https://github.com/mapr/asynchbase
GitHub Release Tag	v1.6.0-mapr-1504
Maven Artifacts	https://repository.mapr.com/maven/

Fixes

This release by MapR includes the following fixes on the base release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
b1535ac	2015-04-21	MapR 17453: JNI memory is no longer released when data from a scan is still being cached.

AsyncHBase 1.6.0-1503 Release Notes

You may also be interested in the AsyncHBase github page.

Version	1.6.0-mapr-1503
Release Date	March 27, 2015
MapR Version Interoperability	See the Ecosystem Support Matrix.
Source on GitHub	https://github.com/mapr/asynchbase
GitHub Release Tag	v1.6.0-mapr-1503
Maven Artifacts	https://repository.mapr.com/maven/

New in this Release

- Comparison filters for HBase and MapR Database are now supported.

Data Access Gateway Release Notes

This section includes the release notes for the MapR Data Access Gateway.

Data Access Gateway 4.0 Release Notes

These notes describe the 4.0 release of the Data Access Gateway.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.0
Release Date	January 2022
MapR Version Interoperability	Compatible with release 6.2 with EEP 7.0.0 and later*
Package Name	`mapr-data-access-gateway` To view the full list of package names, navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS.
Documentation	Administering the MapR Data Access Gateway on page 1492

*The latest core 6.2.0 EBF patch must be applied before you can use Data Access Gateway 4.0 on release 6.2.0.

New in This Release

Only Data Access Gateway 4.0 can be used with release 7.0.0 in FIPS mode. This is the only significant difference between Data Access Gateway 3.0 and 4.0. Both versions 3.0 and 4.0 can be used in non-FIPS mode.

The MapR Data Platform Data Access Gateway is included in EEP repositories beginning with EEP 5.0.0. The Data Access Gateway is a service that acts as a proxy and gateway for translating requests between lightweight client applications and the data-fabric cluster.

Client	Supported EEPs
MapR Database JSON REST API	EEP 5.0.0 and later
Python OJAI client	EEP 6.0.0 and later
Node.js OJAI client	EEP 6.0.0 and later
C# OJAI client	EEP 6.0.0 and later
Go OJAI client	EEP 6.0.0 and later
Java OJAI Thin Client	EEP 6.3.0 and later

Configuring the Maximum Message Size for the gRPC Service

EEP 7.1.0 and later support a new configuration option (`grpc.service.max-message-size`) that allows you to change the maximum message size that the gRPC service accepts. For details, see Administering the MapR Data Access Gateway on page 1492.

last-updated: Apr 11, 2022

In EEP 7.1.0 and later, the Java OJAI Thin Client supports an OJAI Connection String option (`maxmsgsize`) to change the maximum message size that the gRPC client accepts. For details, see Using the Java OJAI Thin Client on page 2670.

Fixes

None.

Known Issues and Limitations

None.

Resolved Issues

None.

Data Access Gateway 3.0 Release Notes

These notes describe the 3.0 release of the Data Access Gateway.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	3.0
Release Date	Setember 2020
MapR Version Interoperability	Compatible with release 6.2 with EEP 7.0.0 and later
Package Name	`mapr-data-access-gateway` To view the full list of package names, navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS.
Documentation	Administering the MapR Data Access Gateway on page 1492

New in This Release

The MapR Data Platform Data Access Gateway is included in EEP repositories beginning with EEP 5.0.0. The Data Access Gateway is a service that acts as a proxy and gateway for translating requests between lightweight client applications and the data-fabric cluster.

Client	Supported EEPs
MapR Database JSON REST API	EEP 5.0.0 and later
Python OJAI client	EEP 6.0.0 and later
Node.js OJAI client	EEP 6.0.0 and later
C# OJAI client	EEP 6.0.0 and later
Go OJAI client	EEP 6.0.0 and later
Java OJAI Thin Client	EEP 6.3.0 and later

Configuring the Maximum Message Size for the gRPC Service

EEP 7.1.0 and later support a new configuration option (`grpc.service.max-message-size`) that allows you to change the maximum message size that the gRPC service accepts. For details, see Administering the MapR Data Access Gateway on page 1492.

In EEP 7.1.0 and later, the Java OJAI Thin Client supports an OJAI Connection String option (`maxmsgsize`) to change the maximum message size that the gRPC client accepts. For details, see Using the Java OJAI Thin Client on page 2670.

Fixes

None.

Known Issues and Limitations

None.

Resolved Issues

None.

MapR Data Access Gateway 2.0 Release Notes

These notes describe the 2.0 release of the MapR Data Access Gateway.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	2.0
Release Date	December 2019
MapR Version Interoperability	Compatible with MapR 6.1 with EEP 6.0.0 and later
Package Name	`mapr-data-access-gateway` To view the full list of package names, navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS.
Documentation	Administering the MapR Data Access Gateway on page 1492

New in This Release

The MapR Data Access Gateway is included in EEP repositories beginning with EEP 5.0.0. The MapR Data Access Gateway is a service that acts as a proxy and gateway for translating requests between lightweight client applications and the MapR cluster. The version of the MapR Data Access Gateway remains unchanged for EEP 6.3.0, but the Java OJAI Thin Client is now supported:

Client	Supported EEPs
MapR Database JSON REST API	EEP 5.0.0 and later
Python OJAI client	EEP 6.0.0 and later
Node.js OJAI client	EEP 6.0.0 and later
C# OJAI client	EEP 6.0.0 and later
Go OJAI client	EEP 6.0.0 and later
Java OJAI Thin Client	EEP 6.3.0 and later

Configuring the Maximum Message Size for the gRPC Service

last-updated: Apr 11, 2022

EEP 6.3.4 and later support a new configuration option (`grpc.service.max-message-size`) that allows you to change the maximum message size that the gRPC service accepts. For details, see Administering the MapR Data Access Gateway on page 1492.

In EEP 6.3.4 and later, the Java OJAI Thin Client supports an OJAI Connection String option (`maxmsgsize`) to change the maximum message size that the gRPC client accepts. For details, see Using the Java OJAI Thin Client on page 2670.

Fixes

None.

Known Issues and Limitations

None.

Resolved Issues

None.

MapR Data Access Gateway 1.0 Release Notes

These notes describe the first release of the MapR Data Access Gateway.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.0
Release Date	April 2018
MapR Version Interoperability	Compatible with MapR 6.0.1 and later
Package Name	`mapr-data-access-gateway` To view the full list of package names, navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS.
Documentation	Administering the MapR Data Access Gateway on page 1492

New in This Release

This is the first release of the MapR Data Access Gateway. The MapR Data Access Gateway is included in EEP repositories beginning with EEP 5.0.0. The MapR Data Access Gateway is a service that acts as a proxy and gateway for translating requests between lightweight client applications and the MapR cluster. For the EEP 5.0.0 release, the MapR Database JSON REST API uses this service.

Fixes

None.

Known Issues and Limitations

None.

Resolved Issues

None

Drill Release Notes

The release notes for Apache Drill contains notes specific to MapR only. Release notes for prior releases are posted on the Apache Drill web site.

Drill 1.16.1.400-2201 (EEP 8.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Drill. You may also be interested in the Apache Drill homepage and the Apache Drill release notes:

Version	1.16.1.400
Release Date	January 2022
HPE Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Drill 1.16.1.400-2201 introduces the following enhancements or HPE platform-specific behavior changes:

- None

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
33a7d24ea2	2022-01-10	MD-6251: Secondary index error when reading from mapr-db json (#575)
8365703194	2021-12-30	MD-5516: Add https support for Drill-on-YARN (#566)
d492882d29	2021-12-30	MD-6191: Allow partition pruning with dynamic CURRENT_DATE operator.
a6ab48aede	2021-12-20	MD-6196: Medial CVE fixes (jquery, bootstrap, data tables)
5ee9b79204	2021-12-20	MD-6196: Medial CVE fixes (junit, bcpkix-jdk15on)
4f92206395	2021-12-08	MD-6202: Excluding transitive netty dependency from zookeeper
aeb91d399c	2021-12-07	MD-6161: fix skipping of types written not in uppercase (#565)
cbbceb728b	2021-11-26	DRILL-8009: DrillConnectionImpl#isValid() doesn't correspond JDBC API
a5b79c9e67	2021-11-21	MD-6196: Fix the CVEs with the high severity
bddeab869d	2021-11-16	DRILL-7586: Fix loading incorrect version of commons-lang3
ec1051c0ec	2021-11-09	MD-6182: Fixed problem with starting drill on the fips cluster (#564)

Known Issues

- You cannot connect to Drill from SQLLine with the `auth` argument set to `maprsasl`, as shown:

```
/opt/mapr/drill/drill-1.16.0/bin/sqlline -u
jdbc:drill:zk=localhost:5181;auth=maprsasl
```

- Due to Drill version changes (3-digit to 4-digit), you cannot upgrade from Drill in EEP 7.0.0 (Drill 1.16.1) to Drill in EEP 7.0.1 (Drill 1.16.1.5) or later. You must perform a new installation of Drill. Alternatively, if you are running Drill on CentOS or RHEL, you can issue the following command as a workaround to upgrade Drill:

```
rpm -Uv --<old-package> <path/to/packages>/*.rpm
```

See Upgrading Drill on page 351.

Limitations

- Older versions of Drill, such as Drill 1.10.0, supported the HBase plug-in, but Drill versions 1.11.0 through 1.16.0.x do not support queries on HBase tables.

Drill 1.16.1.300-2110 (EEP 8.0.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Drill. You may also be interested in the Apache Drill homepage and the Apache Drill release notes:

Version	1.16.1.300
Release Date	October 2021
HPE Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Drill 1.16.1.300-2110 introduces the following enhancements or HPE platform-specific behavior changes:

- None

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
e1578d9aed	2021-09-06	MD-6167: Update the maven artifact version strings to eep
7dd73ee1a9	2021-08-27	MD-6163: Fixed bug with returning null values when selecting by key without tests
4c322e45ce	2021-08-18	MD-6165: Downgrading jackson libs to be consistent with core (from 2.12.1 to 2.11.1)
67619111f2	2021-08-12	MD-6162: Security vulnerability found in jars used in Drill
1c657eff14	2021-08-12	MD-6153: CVE-2019-10172, CVE-2019-10202 vulnerabilities in jackson-mapper-asl-1.9.13.jar

ffe1f0b175	2021-08-06	DRILL-7934: Fix NullPointerException error when reading parquet files
5c58e3c598	2021-07-28	MD-6144: Jetty security vulnerability
ee6525a53f	2021-07-26	MD-6145: CVE-2020-13956, WS-2017-3734 vulnerabilities in http-client
597c32929e	2021-07-15	MD-6140: mapr-drill failing with java.lang.NoClassDefFoundError: org/apache/zookeeper/Environment
4d04fcdd74	2021-07-07	MD-6143: Commons-codec vulnerability WS-2019-0379
d9c5a9f006	2021-06-03	MD-6130: Different jackson jar versions in 3rdparty libs
4e673662e6	2021-05-27	MD-6126: CVE-2020-13936 velocity-engine-core vulnerability
f80f67df1e	2021-05-19	DRILL-7372: MethodAnalyzer consumes too much memory
c023fa7705	2021-05-19	MD-6022: Column names are not flipped when running a query

Known Issues

- You cannot connect to Drill from SQLLine with the `auth` argument set to `maprsasl`, as shown:

```
/opt/mapr/drill/drill-1.16.0/bin/sqlline -u
jdbc:drill:zk=localhost:5181;auth=maprsasl
```

- Due to Drill version changes (3-digit to 4-digit), you cannot upgrade from Drill in EEP 7.0.0 (Drill 1.16.1) to Drill in EEP 7.0.1 (Drill 1.16.1.5) or later. You must perform a new installation of Drill. Alternatively, if you are running Drill on CentOS or RHEL, you can issue the following command as a workaround to upgrade Drill:

```
rpm -Uv --<old-package> <path/to/packages>/*.rpm
```

See Upgrading Drill on page 351.

Limitations

- Older versions of Drill, such as Drill 1.10.0, supported the HBase plug-in, but Drill versions 1.11.0 through 1.16.0.x do not support queries on HBase tables.

Drill 1.16.1.200-2104 (EEP 7.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Drill. You may also be interested in the Apache Drill homepage and the Apache Drill release notes:

Version	1.16.1.200
Release Date	April 2021
HPE Version Interoperability	See Component Versions for Released MEPs.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your MEP and OS to view the list of package names.

New in This Release

Drill 1.16.1.200-2104 introduces the following enhancements or HPE platform-specific behavior changes:

- Service verifier

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
85f32639d	2021-05-09	MD-6133: Fixed issue with Decimal type
8af5a9b51	2021-04-30	MD-6128: Fixed differences between jetty versions and updated to 9.4.40
4324b0753	2021-04-16	MD-6120: Fixed issues with parquet while using hive-storage
2f111fcfa	2021-04-14	MD-6120: Fixed issues with hive (#548)
746179715	2021-04-12	MD-6120: Fixed issue with builds
bf8f0febd	2021-04-12	MD-6102: Removed hasSecurityMechanism function because if there is no mechanism drill will use plain
6963a5b61	2021-04-12	MD-6102: Cannot connect by MAPRSASL without username field (#547)
8b3a076e5	2021-03-31	MD-6120: Update Drill dependencies to the latest artifacts for MEP7.1.0 release (#545)
a467f6327	2021-03-22	MD-6114: Add service verifier to drill package (#543)
9ccf98f66	2021-03-22	MD-5787: Improved kafka-storage plugin to handle topics from non default stream and added support for headers
a9bc026f3	2021-03-18	MD-6101: CVE-2019-16869, CVE-2014-3488 etc. Netty vulnerabilities (#542)
5c878b763	2021-03-10	MD-6117: The Drill cannot run after updates Jetty vulnerability in secure cluster
9e3c48030	2021-02-26	MD-6077: Excluded Log4jplugins.dat file from hive-exec-shaded.jar
0ab22dc82	2021-02-22	MD-6105: Fixed issue so Drill does not return a parsing error for queries on Parquet files.
64d3e225b	2021-02-22	DRILL-7361: Support MAP (DICT) type in schema file
e7f3b6133	2021-02-22	DRILL-7509: Incorrect TupleSchema is created for DICT column when querying Parquet files
27099bcda	2021-02-22	Similar changes to DRILL-7359
0bec91f7c	2021-02-15	MD-6110: CVE-2020-27216: Jetty vulnerability (#538)
5828c1c13	2021-02-10	MD-6111: CVE-2020-25649 Jackson vulnerability (#537)

Known Issues and Limitations

- You cannot connect to Drill from SQLLine with the `auth` argument set to `maprsasl`, as shown:

```
/opt/mapr/drill/drill-1.16.0/bin/sqlline -u
jdbc:drill:zk=localhost:5181;auth=maprsasl
```

- Due to Drill version changes (3-digit to 4-digit), you cannot upgrade from Drill in MEP 7.0.0 (Drill 1.16.1) to Drill in MEP 7.0.1 (Drill 1.16.1.5) or later. You must perform a new installation of Drill. Alternatively, if you are running Drill on CentOS or RHEL, you can issue the following command as a workaround to upgrade Drill:

```
rpm -Uv --<old-package> <path/to/packages>/*.rpm
```

  See Upgrading Drill on page 351.

- Older versions of Drill, such as Drill 1.10.0, supported the HBase plug-in, but Drill versions 1.11.0 through 1.16.0.x do not support queries on HBase tables.

Drill 1.16.1.100-2101 (EEP 7.0.1) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Drill. You may also be interested in the Apache Drill homepage and the Apache Drill release notes:

Version	1.16.1.100
Release Date	January 2021
HPE Version Interoperability	See Component Versions for Released MEPs.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your MEP and OS to view the list of package names.

New in This Release

Drill 1.16.1.100-2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
90e5ecb	2021-01-04	MD-6096: Partition pruning not happening after Upgrade and causing OversizedAllocationException failure
94675b6 b898b86	2020-12-15 2020-12-11	MD-6086:CVE-2019-14262: metadata extractor vulnerability fixed
dd051ab	2020-12-11	MD-6084: CVE-2017-12610,CVE-2018-17196: kafka-clients vulnerability (kafka-clients updated to 2.3.1)
f4bf0cb	2020-12-10	MD-6094: WS-2019-0490: jcommander vulnerability

last-updated: Apr 11, 2022

0386778	2020-12-10	MD-6095:CVE-2019-17359: bcprov-jdk15on-1.60.jar library vulnerability
9f0ac5a	2020-12-09	MD-6091: CVE-2017-5929: logback vulnerability
55b4ad0	2020-12-07	MD-6089:CVE-2018-1272 etc. vulnerabilities in Spring
8457e19	2020-11-26	MD-6072: SchemaChangeException due to a REPEATED MAP field
7c6f90e	2020-11-25	MD-6069: updated not equal ComparisonPredicate because of wrong logic inside
228775b	2020-11-17	MD-6070: Fixed problem with NPE in index picking by adding extra checking
c420ba6	2020-11-17	MD-6057: Positional alias is not working in Group by clause for Nested CASE expressions
58047ca	2020-11-13	MD-6066: Changes due to fix performance issue.
fe00ddf	2020-10-29	MD-6073:CVE-2019-10172 and CVE-2019-10202 - jackson-mapper-asl vulnrability

Known Issues

MapR Tracking Number	Description
MD-6052	Prttn_Prnng_TC1.sql.explain test failed.
MD-6028	Property 'MAPR_JMXSSL' in 'env_override.sh' by default set as "false"
MD-6020	Error while using Drill
MD-5792	TPCH query 5 runs 10-20% slower at sf100/sf1000, possibly due to hash join ordering
MD-5786	TPCDS query 98 is 2x slower with Statistics enabled due to hash join order for sf100 and sf1000
MD-5770	TPCH query 9 runs 18% slower at sf 100/sf1000, possibly due to hash join
MD-5758	TPCDS query 78 runs 30x slower with Statistics enabled at sf100
MD-5694	select query takes too long after metadata refresh
MD-5684	Drill timeout when querying a large number of files
MD-5646	Modification needed in Drill dependencies/scripts to support hadoop decoupling
MD-5518	Semi-Join performance is slower with default settings for TPCH 16

MD-5309	Query on complex data fails with SYSTEM ERROR: SchemaChangeException: Failure while materializing expression. Error in expression at index -1. Error: Missing function implementation: [min(MAP-REPEATED)]. Full expression: --UNKNOWN EXPRESSION--.
MD-4906	SI Selectivity Queries hit ForemanException: One more more nodes lost connectivity during query
MD-4738	Incorrect parallelism determination for index plans (SI and Complex SI)
MD-4504	DRILL-6465: Transitive closure is not working in Drill for Join with multiple local conditions
MD-2294	DRILL-1162: 25 way join ended up with OOM

Known Issues and Limitations

- Due to Drill version changes (3-digit to 4-digit), you cannot upgrade from Drill in MEP 7.0.0 (Drill 1.16.1) to Drill in MEP 7.0.1 (Drill 1.16.1.5) or later. You must perform a new installation of Drill. Alternatively, if you are running Drill on CentOS or RHEL, you can issue the following command as a workaround to upgrade Drill:

  ```
  rpm -Uv --<old-package> <path/to/packages>/*.rpm
  ```

  See Upgrading Drill on page 351.

- Older versions of Drill, such as Drill 1.10.0, supported the HBase plug-in, but Drill versions 1.11.0 through 1.16.0.x do not support queries on HBase tables.

Drill 1.16.1.0-2009 (EEP 7.0.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Drill 1.16.1.0-2009 in EEP 7.0.0.

Below are release notes for the Drill component included in the MapR Converged Data Platform. You may also be interested in the Apache Drill homepage and the Apache Drill release notes:

Version	1.16.1.0-2009
Release Date	September 2020
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your MEP and OS to view the list of package names.

Feature Support

- None.

New Features

- None.

Resolved Issues

Drill 1.16.1.0-2009 provides the following resolved issues and improvements:

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
dade023	2020-03-24	DRILL-7405: Avoiding download of TPC-H data
5f44f00	2020-06-22	DRILL-7750: Drill fails to read KeyStore password from Credential provider
065ac75	2020-06-18	DRILL-7749: Drill-on-Yarn Application Master UI is broken due to bootstrap update
28d3f4c	2020-04-30	DRILL-7705: Updated jQuery and Bootstrap libraries
11fa657	2020-04-21	DRILL-7713: Upgrade misc libraries which outdated versions have reported vulnerabilities
a28c844	2020-04-17	DRILL-7704: Update Maven to 3.6.3
dff80b1	2020-04-16	DRILL-7702: Update shaded guava
ab18420	2020-04-16	DRILL-7702: Update ZooKeeper and Curator, exclude org.codehaus.jackson
8479c37	2020-04-16	DRILL-7702: Update httpclient, libthrift, httpdlog-parser, jetty, derby, exclude and ban log4j, jasper-runtime, commons-httpclient
302fc35	2020-04-16	Revert "DRILL-7189: Revert DRILL-7105 Error while building the Drill native client"
59856bc	2020-04-16	DRILL-7693: Updated protobuf version to 3.11.1
abfc879	2020-04-14	DRILL-7294: Prevent generating java beans using protostuff to avoid overriding classes with the same simple name declared as nested in the proto files
bb4069e	2020-04-14	Revert "DRILL-7188: Revert DRILL-6642: Update protocol-buffers version"
892d164	2020-03-24	DRILL-7650: Add option to enable Jetty's dump for troubleshooting
493e65e	2020-03-19	DRILL-7351: Added tokens to Web forms to prevent CSRF attacks
97c1d22	2020-03-18	DRILL-7644: Log SSL protocol version at Drill start up
3b0c61f	2020-03-18	DRILL-7626: Add ability to set HTTP response headers
f46e0c9	2020-03-13	DRILL-7637: Add an option to retrieve MapR SSL truststore/keystore credentials using MapR Web Security Manager
d0daa60	2020-03-13	DRILL-7367: Remove Server details from response headers

Drill 1.16.1.0-2009 also includes the following resolved issues and improvements:

MapR Tracking Number	Description
MD-6051	Drill Build is failing on CentOS 8 while Building "libdrillClient.so"
MD-6048	DRILL-7774: NPE during planning phase of query in 1.16.0.21
MD-6043	Create get-jars-list.sh script to be used by mapr-config.sh (CORE-221)
MD-6037	Cleanup `GuavaPatcher`'s warning messages in log
MD-6036	Drill logging system is broken after the latest build
MD-6035	Drill logging system is broken after the latest build
MD-6034	Janino compiler 3.0.8 causing drill query throwing exception org.codehaus.commons.compiler.CompileException
MD-6033	Fix unit tests failures caused by latest changes in Hive

MapR Tracking Number	Description
MD-6018	Querying empty Parquet files throws error after upgrade to 1.16
MD-6003	DRILL-7694: CollectD warnings on collecting queries.running and queries.completed metrics
MD-6004	UNION query returns "VALIDATION ERROR" if try union varchar output and decimals/ integer/date
MD-5998	DRILL-7649: Replace maprfs.version property usage by mapr.release.version
MD-5996	Scrypt j_security_check works without security headers
MD-5990	Drill Java JMX Server Insecure Configuration Remote Code Execution Vulnerability
MD-5950	Drill-on-Yarn: NPE while querying a column in Hive table
MD-5920	drill.exec.ssl.protocol option is not honored by Webserver
MD-5908	Drillbit not starting with error: "Caused by: java.io.IOException: No FileSystem for scheme: maprfs"
MD-5839	DRILL-7615: UNION ALL query returns the wrong result for the decimal value

Known Issues

The following table lists the known issues for Drill 1.16.1.0:

MapR Tracking Number	Description
MD-6052	Prttn_Prnng_TC1.sql.explain test failed.
MD-6028	Property 'MAPR_JMXSSL' in 'env_override.sh' by default set as "false"
MD-6027	Cannot change value of "MAPR_JMXSSL=" in "env_override.sh"
MD-6020	Error while using Drill
MD-5792	TPCH query 5 runs 10-20% slower at sf100/sf1000, possibly due to hash join ordering
MD-5786	TPCDS query 98 is 2x slower with Statistics enabled due to hash join order for sf100 and sf1000
MD-5770	TPCH query 9 runs 18% slower at sf 100/sf1000, possibly due to hash join
MD-5758	TPCDS query 78 runs 30x slower with Statistics enabled at sf100
MD-5694	select query takes too long after metadata refresh
MD-5684	Drill timeout when querying a large number of files
MD-5646	Modification needed in Drill dependencies/scripts to support hadoop decoupling
MD-5518	Semi-Join performance is slower with default settings for TPCH 16
MD-5309	Query on complex data fails with SYSTEM ERROR: SchemaChangeException: Failure while materializing expression. Error in expression at index -1. Error: Missing function implementation: [min(MAP-REPEATED)]. Full expression: --UNKNOWN EXPRESSION--.
MD-4906	SI Selectivity Queries hit ForemanException: One more more nodes lost connectivity during query
MD-4738	Incorrect parallelism determination for index plans (SI and Complex SI)
MD-4504	DRILL-6465: Transitive closure is not working in Drill for Join with multiple local conditions
MD-2294	DRILL-1162: 25 way join ended up with OOM

last-updated: Apr 11, 2022

Limitations

Older versions of Drill, such as Drill 1.10.0, supported the HBase plug-in, but Drill versions 1.11.0 through 1.16.0.x do not support querying HBase tables.

Drill 1.16.0.400-2201 (EEP 6.3.6) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Drill. You may also be interested in the Apache Drill homepage and the Apache Drill release notes.

Version	1.16.0.400
Release Date	January 2022
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Drill 1.16.0.400-2201 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
bf345798f7	2022-02-10	MD-6277: Remove mapr-log4j-slf4j-impl module introduced in Hive
da06a2908d	2022-01-10	MD-6251: Secondary index error at reading from mapr-db json (#575)
4faf5f4870	2021-12-30	MD-6191: Change to allow partition prunning with dynamic CURRENT_DATE operator.
14840a0f91	2021-12-20	MD-6196: Medial CVE fixes (jquery, bootstrap, data tables)
6573c1a091	2021-12-20	DRILL-7705: Updated jQuery and Bootstrap libraries
616fd1943e	2021-12-20	MD-6196: Medial CVE fixes (junit, derby, bcpkix-jdk15on)
a16e1268e9	2021-12-07	MD-6161: fix skipping of types written not in uppercase (#565)
40019935b2	2021-11-26	DRILL-8009: DrillConnectionImpl#isValid() doesn't correspond JDBC API
f60bc99104	2021-11-21	MD-6196: Fix the CVEs with the high severity
182dcaf124	2021-11-16	DRILL-7586: Fix loading incorrect version of commons-lang3

Known Issues

- You cannot connect to Drill from SQLLine with the `auth` argument set to `maprsasl`, as shown:

```
/opt/mapr/drill/drill-1.16.0/bin/sqlline -u
jdbc:drill:zk=localhost:5181;auth=maprsasl
```

Limitations

- Older versions of Drill, such as Drill 1.10.0, supported the HBase plug-in, but Drill versions 1.11.0 through 1.16.0.x do not support queries on HBase tables.

Drill 1.16.0.300-2110 (EEP 6.3.5) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Drill. You may also be interested in the Apache Drill homepage and the Apache Drill release notes.

Version	1.16.0.300
Release Date	October 2021
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Drill 1.16.0.300-2110 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
ebcc3f82ed	2021-09-01	DRILL-7294: Prevent generating java beans using protostuff to avoid overriding classes with the same simple name declared as nested in the proto files
aee600f190	2021-08-27	MD-6163: Fixed bug with returning null values when selecting by key without tests
e8efefe649, 082c21f6a6, 4942eaaf62	2021-08-18	MD-6165: Downgrading jackson libs to v2.11.1 to be consistent with core version
6b0e8685e8	2021-08-12	MD-6162: Security vulnerability found in jars used in Drill
84d23a2698	2021-08-12	MD-6153: CVE-2019-10172, CVE-2019-10202 vulnerabilities in jackson-mapper-asl-1.9.13.jar
86ba2964bd, 6a1a3eabec	2021-08-12	MD-6146: CVE-2012-5783 vulnerability in commons-httpclient
17d0259f5e	2021-08-06	DRILL-7934: Fix NullPointerException error when reading parquet files
1c68da948a	2021-07-28	MD-6144: Jetty security vulnerability
8ee8583eb0	2021-07-27	MD-6145: CVE-2020-13956, WS-2017-3734 vulnerabilities in http-client
36cf7751de	2021-07-15	MD-6140: mapr-drill failing with java.lang.NoClassDefFoundError: org/apache/zookeeper/Environment
df4362ab48	2021-07-07	MD-6143: Commons-codec vulnerability, WS-2019-0379
74b5cccf95	2021-06-03	MD-6130: Different jackson jar versions in 3rdparty libs
073696be4d	2021-05-27	MD-6126: CVE-2020-13936 velocity-engine-core vulnerability

694139b41d	2021-05-19	DRILL-7372: MethodAnalyzer consumes too much memory
b1c05a7849	2021-05-19	MD-6022: Column names are not flipped when running a query

Known Issues

- You cannot connect to Drill from SQLLine with the `auth` argument set to `maprsasl`, as shown:

```
/opt/mapr/drill/drill-1.16.0/bin/sqlline -u
jdbc:drill:zk=localhost:5181;auth=maprsasl
```

Limitations

- Older versions of Drill, such as Drill 1.10.0, supported the HBase plug-in, but Drill versions 1.11.0 through 1.16.0.x do not support queries on HBase tables.

Drill 1.16.0.200-2104 (EEP 6.3.4) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Drill. You may also be interested in the Apache Drill homepage and the Apache Drill release notes.

Version	1.16.0.200
Release Date	April 2021
MapR Version Interoperability	See Component Versions for Released MEPs.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your MEP and OS to view the list of package names.

New in This Release

Drill 1.16.0.200-2104 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
85f32639d	2021-05-09	MD-6133: Fixed issue with Decimal type
8af5a9b51	2021-04-30	MD-6128: Fixed differences between jetty versions and updated to 9.4.40
5772a10ad	2021-04-12	MD-6102 Removed hasSecurityMechanism function because if there is no mechanism drill will use plain
ec9554e91	2021-04-12	MD-6102: Cannot connect by MAPRSASL without username field (#547)
a40a3f242	2021-03-22	MD-5787 Improved kafka-storage plugin to handle topics from non default stream and added support for headers
7b16dd6e3	2021-03-18	MD-6101: CVE-2019-16869, CVE-2014-3488 etc. Netty vulnerabilities (#542)
8d67a1902	2021-03-10	MD-6117 The Drill cannot run after updates Jetty vulnerability in secure cluster

23e61b94c	2021-02-26	MD-6077 Excluded Log4jplugins.dat file from hive-exec-shaded.jar
0f2d562be	2021-02-22	MD-6105 Checkstyle fixes and resolved problems with tests
5257dedf1	2021-02-22	DRILL-7361: Support MAP (DICT) type in schema file
335e11089	2021-02-22	DRILL-7509: Incorrect TupleSchema is created for DICT column when querying Parquet files
b30e485e1	2021-02-22	Similar changes to DRILL-7359
b1ce9720b	2021-02-15	MD-6110: CVE-2020-27216: Jetty vulnerability (#538)
720b7437e	2021-02-10	MD-6111: CVE-2020-25649 Jackson vulnerability (#537)

Known Issues

- You cannot connect to Drill from SQLLine with the `auth` argument set to `maprsasl`, as shown:

```
/opt/mapr/drill/drill-1.16.0/bin/sqlline -u
jdbc:drill:zk=localhost:5181;auth=maprsasl
```

Limitation

Older versions of Drill, such as Drill 1.10.0, supported the HBase plug-in, but Drill versions 1.11.0 through 1.16.0.x do not support queries on HBase tables.

Drill 1.16.0.100-2101 (EEP 6.3.2) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Drill 1.16.0.100 in MEP 6.3.2.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Drill. You may also be interested in the Apache Drill homepage and the Apache Drill release notes.

Version	1.16.0.100
Release Date	January 2021
MapR Version Interoperability	See Component Versions for Released MEPs.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your MEP and OS to view the list of package names.

New in This Release

Drill 1.16.0.100-2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
9e9dbd3	2020-12-02	EMEP-85:CVE-2017-7656,CVE-2017-7657,CVE-2017-7658, etc : jetty vulnerabilities
0256dda	2020-12-02	EMEP-77: CVE-2019-0205: potential DoS when processing untrusted Thrift payloads

31a7739	2020-12-07	MD-6089:CVE-2018-1272 etc. vulnerabilities in Spring
09b2f4a	2020-12-07	MD-6090: CVE-2014-0107: xalan vulnerability
c55d81b	2020-12-08	MD-6087:CVE-2018-1000844,CVE-2018-1000850: retrofit vulnerabilities
d2e977d	2020-12-08	MD-6091: CVE-2017-5929: logback vulnerability
bc498b8	2020-12-10	MD-6095:CVE-2019-17359: bcprov-jdk15on-1.60.jar library vulnerability
808b7c2	2020-12-10	MD-6094: WS-2019-0490: jcommander vulnerability
8746484	2020-12-11	MD-6084: CVE-2017-12610,CVE-2018-17196: kafka-clients vulnerability (kafka-clients updated to 2.3.1)
866078d	2020-12-11	CVE-2019-14262: metadata extractor vulnerability fixed
75dece5	2020-12-15	MD-6086:CVE-2019-14262: metadata extractor vulnerability fixed
43a0034	2020-12-29	MD-6096: Partition pruning not happening after Upgrade and causing OversizedAllocationException failure
3221da2 4914e4e	2020-10-22 2021-01-05	MD-6071: Update commons-beanutils, xercesImpl
fd96c36	2020-10-15	MD-6061:CVE-2019-14262 - mapr ojai driver and mapr drill
65ff5ec	2020-10-23	MD-6073: Update snakeyaml
A33c78e 235d54b	2020-10-29 2020-11-02	MD-6074:CVE-2019-10172 and CVE-2019-10202 - jackson-mapper-asl vulnerability
1edf007	2020-11-11	MD-6066: Changes due to performance issue, to allow the setting of variable output_batch_size on both system and session level
87ca6c8	2020-11-17	MD-6070: Fixed problem with NPE in index picking by adding extra checking
69f6ca3	2020-11-25	MD-6069: updated not equal ComparisonPredicate because of wrong logicinside
7daaab9 1d7e83d	2020-11-25 2020-11-26	MD-6072: SchemaChangeException due to a REPEATED MAP field
3530e89	2020-12-01	MD-6083: Updated derby version to 10.12.1.1 (CVE-2015-1832)

last-updated: Apr 11, 2022

Known Issues

None.

Limitation

Older versions of Drill, such as Drill 1.10.0, supported the HBase plug-in, but Drill versions 1.11.0 through 1.16.0.x do not support queries on HBase tables.

Drill 1.16.0.22-2009 (EEP 6.3.1) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Drill 1.16.0.22-2009 in EEP 6.3.1.

Below are release notes for the Drill component included in the MapR Converged Data Platform. You may also be interested in the Apache Drill homepage and the Apache Drill release notes:

Version	1.16.0.22-2009
Release Date	September 2020
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.
Package Names	Navigate to https://package.mapr.com/releases/MEP/, and select your MEP and OS to view the list of package names.

New Features

- None.

Resolved Issues

Drill 1.16.0.22-2009 provides the following resolved issues and improvements:

Commit	Date (YYYY-MM-DD)	Comment
dade023	2020-03-24	DRILL-7405: Avoiding download of TPC-H data
892d164	2020-03-24	DRILL-7650: Add option to enable Jetty's dump for troubleshooting
493e65e	2020-03-19	DRILL-7351: Added tokens to Web forms to prevent CSRF attacks
97c1d22	2020-03-18	DRILL-7644: Log SSL protocol version at Drill start up
3b0c61f	2020-03-18	DRILL-7626: Add ability to set HTTP response headers
f46e0c9	2020-03-13	DRILL-7637: Add an option to retrieve MapR SSL truststore/keystore credentials using MapR Web Security Manager
d0daa60	2020-03-13	DRILL-7367: Remove Server details from response headers

Drill 1.16.0.22-2009 also includes the following resolved issues and improvements:

MapR Tracking Number	Description
MD-6048	DRILL-7774: NPE during planning phase of query in 1.16.0.21
MD-6035	DRILL-7761: Drill 1.16.0 fails with OOM in PLANNING and drillbit hung
MD-6034	Janino compiler 3.0.8 causing drill query throwing exception org.codehaus.commons.compiler.CompileException
MD-6018	Querying empty Parquet files throws error after upgrade to 1.16

last-updated: Apr 11, 2022

MapR Tracking Number	Description
MD-6004	UNION query returns "VALIDATION ERROR" if try union varchar output and decimals/ integer/date
MD-5998	DRILL-7649: Replace maprfs.version property usage by mapr.release.version
MD-5996	Scrypt j_security_check works without security headers
MD-5990	Drill Java JMX Server Insecure Configuration Remote Code Execution Vulnerability
MD-5950	Drill-on-Yarn: NPE while querying a column in Hive table
MD-5920	drill.exec.ssl.protocol option is not honored by Webserver
MD-5839	DRILL-7615: UNION ALL query returns the wrong result for the decimal value

Known Issues

- None.

Limitations

Older versions of Drill, such as Drill 1.10.0, supported the HBase plug-in, but Drill versions 1.11.0 through 1.16.0.x do not support querying HBase tables.

Drill 1.16.0.10-1912 (EEP 6.3.0) Release Notes

This section provides reference information for fixes in Drill 1.16.0.10-1912.

The following release note applies to the 1.16.0.10-1912 version of the Drill component:

Important: This release note describes only the issues resolved since the 1.16.0.0 release note. For a listing of 1.16.0.0 resolved issues, see Drill 1.16.0.0-1904 (EEP 6.2.0) Release Notes on page 5763.

Version	1.16.0.10-1912
Release Date	December 2019
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.
Package Names	Navigate to https://package.mapr.com/releases/MEP/, and select your MEP and OS to view the list of package names, for example: • mapr-drill-1.16.0.10.201912111313-1.noarch.rpm • mapr-drill-internal-1.16.0.10.201912111313-1.noarch.rpm • mapr-drill-yarn-1.16.0.10.201912111313-1.noarch.rpm

Resolved Issues

Drill 1.16.0.10-1912 provides the following resolved issues and improvements:

Apache Drill Issue	MapR Tracking Number	Description
DRILL-7453	MD-5936	Updated the JODA-Time API to ensure correct time zone info
DRILL-7417	MD-5914	Add user logged in/out events in info-level logs

last-updated: Apr 11, 2022

DRILL-7391	MD-5895	Wrong result when doing left outer join on CSV table
DRILL-6711	N/A	Use jetpack repository for Drill Calcite project artifact
DRILL-7373	MD-5534	Fix problems reading from DICT type
DRILL-7096	MD-5534	Develop vector for canonical Map<K,V>
DRILL-7376	N/A	Drill ignores Hive schema for MaprDB tables when group scan has star column
DRILL-7341	N/A	Vector reAlloc can fail after exchange
DRILL-7369	MD-5801	Schema for MaprDB tables is not used when several fields are queried
DRILL-7338	MD-5262	REST API calls to Drill fail dure to insufficient heap memory
DRILL-7313	MD-5801	Query on a Hive external table in MaprDB fails while reading vector
DRILL-7250	MD-5707	Query with CTE fails when its name matches the table name without access
DRILL-7237	MD-5732	Fix single_value aggregate function for variable length types
DRILL-7050	MD-5732	RexNode convert exception in sub-query

Limitations

Older versions of Drill, such as Drill 1.10.0, supported the HBase plug-in, but Drill versions 1.11.0 through 1.16.0.x do not support querying HBase tables.

Drill 1.16.0.0-1904 (EEP 6.2.0) Release Notes

This section provides reference information, including new features, improvements, resolved issues, known issues, and limitations for Drill 1.16.0.0-1904.

These release notes contain MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

The following release notes apply to the 1.16.0.0 version of the Drill component:

Version	1.16.0.0
Release Date	May 2019
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names, for example: • mapr-drill-1.16.0.0.201905231125-1.noarch.rpm • mapr-drill-internal-1.16.0.0.201905231125-1.noarch.rpm • mapr-drill-yarn-1.16.0.0.201905231125-1.noarch.rpm

New in this Release

Drill 1.16.0.0 includes the following new features and improvements in the following areas:

SQL

- ANALYZE TABLE COMPUTE STATISTICS generates table statistics for more efficient query plans. (DRILL-1328)

last-updated: Apr 11, 2022

	• ANALYZE TABLE REFRESH METADATA generates metadata cache files for specific columns instead of an entire table or directory. (DRILL-7058) • CREATE OR REPLACE SCHEMA command defines schema for text files. (MD-5202, DRILL-6964) • NEARESTDATE function for time series analysis. • NEAREST DATE function to facilitate time series analysis (DRILL-7077) • By default, Drill no longer writes the profiles for SET queries to the persistent store. Setting the `exec.query_profile.alter_session.skip` option to false reverts this behavior.
Storage	• SYSLOG (RFC-5424) Format Plugin (DRILL-6582) • Format plugin for LTSV files (DRILL-7014) • A new maprdb format plugin option, `readTimestampWithZoneOffset`, converts timestamp values from UTC to local time zone when values are read from MapR Database. This option is disabled by default. (MD-5272) • Drill can query views defined in Hive similarly to querying Hive tables in a hive schema, for example: `SELECT * FROM hive.`hive_view`;` (DRILL-540)
Configuration Options	• A new Drill configuration option, `store.hive.maprdb_json.read_timestamp_with_timezone_offset`, enables Drill to read timestamp values with a timezone offset when using the hive plugin with the Drill native MaprDB JSON reader enabled. This option is disabled by default. (MD-5272)
Web UI	Several Web UI improvements, including: • Storage plugin management improvements (DRILL-6562) • Query progress indicators and warnings (DRILL-6879) • Ability to limit the result size for better UI response (DRILL-6050) • Ability to sort the list of profiles in the Drill Web UI (DRILL-6942) • Display query state in query result page (DRILL-6939) • Button to reset the options filter (DRILL-6921)

SQLLine (Drill shell)	• Upgrade to SQLLine 1.7. (DRILL-6989)
Calcite	• Upgrade to Calcite 1.18.0. (MD-5050)

For a list of additional features and improvements, see the Apache Drill 1.16 release notes.

Resolved Issues

Drill 1.16.0.0 includes the following resolved issues and improvements:

MapR Tracking Number	Resolved Issue
MD-5673	DRILL-7150: Drill timestamp timezone conversion uses current daylight savings time instead of the one active during timestamp date
MD-5647	DRILL-7118: Filter not getting pushed down on MapR-DB tables.
MD-5638	DRILL-7130: IllegalStateException: Read batch count [0] should be greater than zero
MD-5630	DRILL-7113: Drill on MapRDB can not understand null value
MD-5624	DRILL-7125: REFRESH TABLE METADATA fails after upgrade from Drill 1.13.0 to Drill 1.15.0
MD-5623	Unable to connect to Drill 1.15 through ZK
MD-5609	DRILL-7079: Drill can't query views from the S3 storage when plain authentication is enabled
MD-5606	DRILL-7100: parquet RecordBatchSizerManager : IllegalArgumentException: the requested size must be non-negative
MD-5561	DRILL-7060: Query on audit logs fails by DATA_READ ERROR Error Parsing JSON - Unrecognized character escape 'S' (code 83)
MD-5552	DRILL-7119: Modify selectivity calculations to use histograms
MD-5550	DRILL-7048: Implement JDBC Statement.setMaxRows() with System Option
MD-5523	Physical plan generation failure after upgrade from 1.10 to 1.14
MD-5490	DRILL-7117: Support creation of column Histograms for numeric data types
MD-5428	Include links to pre and post procedures in Drill upgrade documentation
MD-5379	DRILL-7018: Drill Query (when store.parquet.reader.int96_as_timestamp=true) on Parquet File fails with Error: SYSTEM ERROR: IndexOutOfBoundsException: readerIndex: 0, writerIndex: 372 (expected: 0 <= readerIndex <= writerIndex <= capacity(256))
MD-5374	DRILL-6971: Display query state in query result page of Web UI
MD-5369	DRILL-7115: Improve Hive schema show tables performance
MD-5368	DRILL-4858: repeated_count on JSON array of objects (maps) implementation is **missing** in Drill 1.14
MD-5363	DRILL-4858: Missing function implementation: [repeated_count(LIST-REPEATED)]

last-updated: Apr 11, 2022

MD-5356	DRILL-4858: Implement - Missing function implementation: [repeated_count(MAP-REPEATED)].
MD-5348	DRILL-6997: TPCDS queries 56, 60, 83 are slower with plan change
MD-5330	DRILL-6967: TIMESTAMPDIFF returns incorrect value for SQL_TSI_QUARTER
MD-5319	DRILL-6997: TPCDS query 95 slower with plan change
MD-5278	DRILL-6931: Drill "SHOW FILES" command duplicates empty S3 folders as subfolders
MD-5277	DRILL-6928: exec.query.return_result_set_for_ddl does not affect Web UI query results
MD-5272	DRILL-6969: Drill on maprdb native reader reads a wrong timezone comparing to hive
MD-5253	to_timestamp function is losing precision for milliseconds
MD-5251	DRILL-6894: CTAS and CTTAS are not working on S3 storage when cache is disabled
MD-5236	DRILL-7023: Tableau query fails with IndexOutOfBoundsException after upgrade from drill 1.13.0 to drill 1.14.0
MD-5226	DRILL-6918: Querying empty topics fails with "NumberFormatException"
MD-5198	DRILL-6880: TPCDS query 35 slower due to nulls
MD-5179	DRILL-6874: CTAS from json to parquet is not working on S3 storage
MD-5095	DRILL-7051: Update Drill's Jetty Server to 9.3
MD-4863	Simba JDBC driver does not return some values
MD-4862	Simba JDBC driver returns incorrect time value
MD-4826	The COALESCE function returns results when the columns referenced in the function do not exist in the files being queried. You do not have to CAST the columns to a specific data type for the COALESCE function to return results.
MD-4617	"direct.used" metrics(jvm_direct_current) doesn't catch the direct memory usage.
MD-4362	Query on data containing reserved word 'date' as column name fails to generate non-covering index plan
MD-3723	Querying Hbase row_key column with non- existing column returns different results in different Drill Versions
MD-1585	Need More Accurate Filter Estimation Before Running a Query
MD-1008	DRILL-7038: Performance - Queries on partitioned columns currently scan the entire datasets
MD-880	HashJoin's not fully parallelized in query plan
MD-680	DRILL-7069: Planning time unaccounted for query with longer planning time

Known Issues

Drill 1.16.0.0 has the following known issues:

last-updated: Apr 11, 2022

MapR Tracking Number	Known Issue
MD-5792	TPCH query 5 runs 10-20% slower at sf100/sf1000, possibly due to hash join ordering
MD-5786	TPCDS query 98 is 2x slower with Statistics enabled due to hash join order for sf100 and sf1000
MD-5782	Need better error message when analyze command fails due to schema change
MD-5770	TPCH query 9 runs 18% slower at sf 100/sf1000, possibly due to hash join
MD-5758	TPCDS query 78 runs 30x slower with Statistics enabled at sf100
MD-5755	DirectScan lists all partitions in explain plan, even for full table scan
MD-5744	[DRILL-7216] Auto limit is happening on the Drill Web-UI while the limit check box is unchecked
MD-5740	REFRESH TABLE METADATA does not count null values for decimal, varchar, and interval data types.
MD-5694	The first query to use a new metadata cache file may take a while to run because the first query triggers a refresh of the metadata cache file.
MD-5684	Drill timeout when querying a large number of files
MD-5676	Drill parquet file may not have statistics for decimal and varchar data types.
MD-5608	Running analyze command on a view fails correctly but the error is confusing
MD-5528	Compute stats on non existent columns fails with exception
MD-5388	Running analyze cmd on duplicate column names is resulting in IndexOutOfBoundsException
MD-5371	Error msg not clear when analyze cmd is run on table with complex types
MD-5342	DRILL-6839 : regarding aggs in cross join queries

Fixes

None.

Limitations

None.

Drill 1.15.0.7-1904 (EEP 6.1.1) Release Notes

This section provides reference information for fixes in Drill 1.15.0.7-1904.

The following release notes apply to the 1.15.0.7-1904 version of the Drill component:

Version	1.15.0.7-1904
Release Date	May 2019
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.

last-updated: Apr 11, 2022

Package Names	Navigate to https://package.mapr.com/releases/MEP/, and select your MEP and OS to view the list of package names, for example: • mapr-drill-1.15.0.7.201905031823-1.noarch.rpm • mapr-drill-internal-1.15.0.7.201905031823-1.noarch.rpm • mapr-drill-yarn-1.15.0.7.201905031823-1.noarch.rpm

Fixes

Drill 1.15.0.7-1904 provides the following fixes:

Fix Tracking Number	Description
MD-5523	Physical plan generation failure after upgrade from 1.10 to 1.14.
DRILL-7100	Fixed IllegalArgumentException when reading Parquet data.
MD-4166	Fix creation of filter conditions for IS NULL and IS NOT NULL.
DRILL-7118	Filter not getting pushed down on MapR-DB tables.
DRILL-7125	REFRESH TABLE METADATA fails after upgrade from Drill 1.13 to 1.15.0.0.
DRILL-7130	Fixed IllegalStateException while reading Parquet data.
DRILL-7150	Fix timezone conversion for timestamp from MaprDB after the transition from PDT to PST.
DRILL-7050	RexNode convert exception in sub-query.
MD-5623	Unable to connect to Drill 1.15 through ZK.
MD-5771	Username / password authentication for JDBC is not working anymore in Drill 1.15.
DRILL-7237	Fix single_value aggregate function for variable length types.

Known Issues and Limitations

None.

Drill 1.15.0.0-1901 (EEP 6.1.0) Release Notes

This section provides reference information, including new features, improvements, resolved issues, known issues, and limitations for Drill 1.15.0.0-1901.

These release notes contain MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

The following release notes apply to the 1.15.0.0 version of the Drill component:

Version	1.15.0.0
Release Date	February 2019
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

ODBC/JDBC Drivers	Drill 1.15.0.0 works with the following MapR Drill drivers:
	• ODBC Drivers
	  • Mac OS X
	  • Linux
	  • 32-bit (Unsupported)
	  • 64-bit
	  • Windows
	  • 32-bit (Unsupported)
	  • 64-bit
	• JDBC Driver
	For additional driver information, see Drill Drivers.

New in this Release

Drill 1.15.0.0 includes the following new features and improvements in the following areas:

Storage

- Ability to query data in S3 via the new cloud storage plugin for S3. (MD-4541)
- Credential Provider API support in the S3 storage plugin. (MD-4543, DRILL-6662)
- The INFORMATION_SCHEMA FILES table provides information about directories and files stored in workspaces configured within S3 and filesystem storage plugin configurations. (MD-4544, DRILL-6680)

SQL

- TIMESTAMPADD and TIMESTAMPDIFF datetime functions. (MD-342, DRILL-3610)
- CROSS JOIN in the FROM clause. (MD-4359, DRILL-786)
- Support for column aliases and positional aliases in the GROUP BY, HAVING, and ORDER BY clauses. (MD-4825, DRILL-1248)
- Lateral join functionality is enabled by default. (DRILL-6729)
- Storage plugin names are case-insensitive. (MD-4589, MD-4663, DRILL-6492, DRILL-6768)
- All cast and data type conversion functions return null for an empty string ('') when the `drill.exec.functions.cast_empty_string_to_null` option is enabled. (DRILL-6817)
- A new option, `exec.query.return_result_set_for_ddl`, prevents Drill from returning a result set for DDL statements when set to "false." Useful for clients tools that connect to Drill (via JDBC) if they do not expect a result set. (MD-4757, DRILL-6834)

last-updated: Apr 11, 2022

Query optimization	• MapR Database secondary index support in Apache Drill. (DRILL-6381) • Improved query performance with semi-join functionality inside the Hash-Join operator. (MD-4877, DRILL-6735)
Parquet	• Parquet filter pushdown supports VARCHAR and DECIMAL data types (MD-1266, DRILL-6744)
Security	• Secure znodes with custom Access Control Lists. (MD-4476, DRILL-5671)
Web UI	Multiple Web UI improvements that simplify use and administrative tasks, including: • The ability to cancel multiple queries at once via a check box next to executing queries. • The auto prompt in the Web UI SQL box now works on UDFs. • An enhanced Options page that includes detailed descriptions of each option. • Categorized options via Quick Filters on the Options page. (DRILL-5735) • Default button (DRILL-6668) • Web display options and updated option names (MD-4835, DRILL-6544, DRILL-6715) • Meta+Enter key combination to submit queries (DRILL-6611)
SQLLine (Drill shell)	• Upgrade to SQLLine 1.6, includes the ability to add a custom configuration. (MD-2682, MD-5151, DRILL-3853) • Additional SQLLine connection parameters. (DRILL-3933)
System tables	• System options table includes option descriptions. (DRILL-6684) • System functions table exposes the available SQL functions in Drill and also detects UDFs that have been dynamically loaded into Drill. (MD-4941, DRILL-3988)

For a list of additional features and improvements, see the Apache Drill 1.15 Release Notes.

For additional information, read the MapR Drill 1.15 blog post.

Default Configuration Changes

Starting in MEP 6.1.0, certain ecosystem components, including Drill, have four digit version numbers. For example, when you install Drill, the Drill version appears as 1.15.0.0.

last-updated: Apr 11, 2022

Resolved Issues

Drill 1.15.0.0 includes the following resolved issues:

MapR Tracking Number	Resolved Issue
MD-5253	The TO_TIMESTAMP function no longer loses precision for milliseconds.
MD-5226	DRILL-6918: Querying empty topics no longer fails with a "NumberFormatException."
MD-5208	Semi-join queries on sf1 data no longer return an UNSUPPORTED_OPERATION error.
MD-5149	DRILL-6869: Drill no longer allows views created outside of workspaces.
MD-5147	DRILL-6865: Queries no longer return the wrong results when filter pruning occurs.
MD-5142	DRILL-6857: Limit is pushed into the scan when selecting from a Parquet file with multiple row groups.
MD-5138	DRILL-6863: Drop table works when the path is within a workspace that starts with '/'.
MD-5105	DRILL-6818: Secondary index options have descriptions.
MD-5076	DRILL-6811: Type inference returns the correct data mode for boolean functions.
MD-5062	DRILL-6664: Drill returns the same results for queries on the same column of a Parquet table with the same predicate.
MD-5057	Queries against new files created on a volume that has reached the name container data threshold no longer fail with I/O exceptions.
MD-5023	DRILL-6810: Drill does not support nullable complex types (maps, arrays); therefore, UDFs cannot return null if @Output is of a complex type. Null handling of complex types is disabled by default. NullHandling.INTERNAL should be used instead of NullHandling.NULL_IF_NULL in UDFs. When using NullHandling.INTERNAL, the UDF must handle null input(s) on its own.
MD-5019	DRILL-6797: Drill aggregation queries that use custom UDF to split column data in a Parquet file no longer fail with a compilation error if the filter criteria results in no rows.
MD-5014	DRILL-6776: The Drill Web UI page source no longer has links to external sites.
MD-5008	Drill can leverage indexes for queries with ORDER BY DESC on indexed columns; for example, if a secondary index is defined on column a, Drill leverages the index for the query: SELECT a FROM t ORDER BY a DESC;
MD-5002	DRILL-6764: Queries no longer fail with IOB exception when UNNEST references a deep nested field, like t.c_orders.o_lineitems.
MD-5001	DRILL-6766 : Queries with LATERAL UNNEST no longer return the following system error when one batch of streaming aggregates is split across multiple output batches : IllegalStateException: Unexpected case where rowId [] in right batch of lateral is smaller than rowId [] in left batch being processed.
MD-4968	DRILL-6762: Dynamic UDFs registered on one Drillbit are now visible on other Drillbits.
MD-4956	DRILL-6755: Hash-Join does not build hash tables when the probe side is empty.

MD-4938	The option planner.index.covering_selectivity_threshold takes effect in execution plan when set to values < 1.0 for complex data.
MD-4936	DRILL-6773: Queries, with column aliases, that returned inconsistent results after a Drill patch was applied no longer return inconsistent results.
MD-4867	DRILL-6726: Drill can query views created before DRILL-6492 when impersonation is enabled.
MD-4852	DRILL-6819: The invisible "back" link on the query results page in the Drill WebUI has been removed.
MD-4851	Drill queries on MapRDB JSON tables no longer fail on schema change when the only distinct values are null and text.
MD-4801	DRILL-6724: Operator context is dumped to logs when errors occur during query execution.
MD-4780	DRILL-6746: Queries no longer hang when one of the nodes encounters a connection error.
MD-4755	DRILL-6709: The batch statistics logging utility has been extended to mid-stream operators.
MD-4643	DRILL-6732: Disabled plugins no longer work as enabled.
MD-4617	"direct.used" metrics(jvm_direct_current) catches the direct memory usage.
MD-4585	The drillbit.log file includes query user information.
MD-4567	DRILL-6517 : Hash-Join no longer accesses batches after query cancellation.
MD-4499	DRILL-5365: File not found exception no longer results from reading a Parquet file.
MD-4362	Drill no longer fails to generate a non-covering index plan for queries on data with the reserved word 'date' as a column name.
MD-3879	DRILL-6039: Graceful shutdown waits for fragments to complete before stopping the Drillbit.
MD-2979	DRILL-6410: The Parquet reader no longer leaks memory during query cancellation.

Known Issues

Drill 1.15.0.0 has the following known issues:

MapR Tracking Number	Known Issue
MD-5390	Drill does not return an error message when impersonation prevents access to a directory.
MD-5376	DRILL-6991: The Kerberos ticket is being dumped in the log if the log level is "debug" for stdout.
MD-5348	TPCDS queries 56, 60, 83 are slower with plan change.
MD-5319	TPCDS query 95 slower with plan change.
MD-5250	TPCDS queries 56 and 60 have regressed.
MD-5232	TPCH query 4 has regressed for scale factor 1000.
MD-5231	DRILL-6896: TPCH 13 has "regressed".
MD-5204	DRILL-6845: TPCDS query 95 slower with semi-join enabled. You can disable semi-join through the planner.enable_semijoin option.

MD-5063	TPCH queries hit an IndexOutOfBoundsException when planner.enable_demux_exchange = true.

Fixes

None.

Limitations

None.

Drill 1.14.0-1904 (EEP 6.0.2) Release Notes

This section provides reference information for fixes in Drill 1.14.0-1904.

The following release notes apply to the 1.14.0-1904 version of the Drill component:

Version	1.14.0-1904
Release Date	May 2019
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.
Package Names	Navigate to https://package.mapr.com/releases/MEP/, and select your MEP and OS to view the list of package names, for example: • mapr-drill-1.14.0.201905031749-1.noarch.rpm • mapr-drill-internal-1.14.0.201905031749-1.noarch.rpm • mapr-drill-yarn-1.14.0.201905031749-1.noarch.rpm

Fixes

Drill 1.14.0-1904 provides the following fixes:

Fix Tracking Number	Description
MD-5523	Physical plan generation failure after upgrade from 1.10 to 1.14.
MD-4166	Fix creation of filter conditions for IS NULL and IS NOT NULL.
DRILL-6721	Fix SchemalessScan plan serialization / deserialization.
DRILL-7050	RexNode convert exception in sub-query.
DRILL-7125	REFRESH TABLE METADATA fails after upgrade from Drill 1.13 to 1.15.0.0.
MD-5771	Username / password authentication for JDBC is not working anymore in Drill 1.15.
DRILL-7237	Fix single_value aggregate function for variable length types.

Known Issues and Limitations

None.

Drill 1.14.0-1901 (EEP 6.0.1) Release Notes

This section provides reference information for fixes in Drill 1.14.0-1901.

The following release notes apply to the 1.14.0-1901 version of the Drill component:

last-updated: Apr 11, 2022

Version	1.14.0-1901
Release Date	February 2019
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.
Package Names	Navigate to https://package.mapr.com/releases/MEP/, and select your MEP and OS to view the list of package names, for example: • mapr-drill-1.14.0.201901161034-1.noarch.rpm • mapr-drill-internal-1.14.0.201901161034-1.noarch.rpm • mapr-drill-yarn-1.14.0.201901161034-1.noarch.rpm

Fixes

Drill 1.14.0-1901 provides the following fixes:

Fix Tracking Number	Description
MD-5379	DRILL-7018: Fixed Parquet buffer overflow when reading timestamp column.
MD-5272	DRILL-6969: Drill no longer returns inconsistent results when reading MapR Database JSON tables using hive plugin when the native reader is enabled.
MD-5257	DRILL-6942: Sort query profiles by duration in the Drill Web UI under Profiles tab.
MD-5062	DRILL-6664: Drill returns the same results for queries on the same column of a Parquet table with the same predicate.
MD-5019	DRILL-6797: Drill aggregation queries that use custom UDF to split column data in a Parquet file no longer fail with a compilation error if the filter criteria results in no rows.
MD-5014	DRILL-6776: The Drill Web UI page source no longer has links to external sites.
MD-5008	Drill can leverage indexes for queries with ORDER BY DESC on indexed columns; for example, if a secondary index is defined on column a, Drill leverages the index for the query: SELECT a FROM t ORDER BY a DESC;
MD-5002	DRILL-6764: Queries no longer fail with IndexOutOfBoundsException when UNNEST references a deep nested field, like t.c_orders.o_lineitems.
MD-5001	DRILL-6766: Queries with LATERAL UNNEST no longer return the following system error when one batch of streaming aggregates is split across multiple output batches: IllegalStateException: Unexpected case where rowId [] in right batch of lateral is smaller than rowId [] in left batch being processed.

MD-4938	The option planner.index.covering_selectivity_threshold takes effect in execution plan when set to values < 1.0 for complex data.
MD-4936	DRILL-6773: Queries, with column aliases, that returned inconsistent results after a Drill patch was applied no longer return inconsistent results.
MD-4902	Complex secondary index plans are parallelized when queries have AND conditions.
MD-4851	Drill queries on MapRDB JSON tables no longer fail on schema change when the only distinct values are null and text.
MD-4780	DRILL-6746: Queries no longer hang when one of the nodes encounters a connection error.

Known Issues and Limitations

None.

Drill 1.14.0-1808 (EEP 6.0.0) Release Notes

This section provides reference information, including new features, improvements, resolved issues, known issues, and limitations for Drill 1.14.0-1808.

These release notes contain MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

The following release notes apply to the 1.14.0 version of the Drill component:

Version	1.14.0
Release Date	September 2018
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

New in this Release

Drill 1.14.0 includes the following new features and improvements:

Query Planning	The query planner in Drill can leverage indexes created on MapR Database JSON document fields with array fields. See Writing Drill Queries that Leverage Indexes on Array Fields. (MD-3742)
	When you query a view created on a MapR Database table, the query planner in Drill creates the same query plan it would create if you ran the underlying query directly against the MapR Database table. (MD-3360, MD-3313)
SQL	Drill supports lateral joins. You must enable the lateral join functionality in Drill. See Lateral Join.
	The DECIMAL data type is enabled by default and has enhanced support. (MD-3522, MD-1138, MD-640)
	Hive schema and table names are case-insensitive. See Case-Sensitivity. (MD-948)
	Support for the ANY_VALUE function; based on CALCITE-2366. (MD-4498)

last-updated: Apr 11, 2022

	Drill allows comparisons between date and timestamp values. (MD-4220)
Resource management	Drill can directly manage the CPU resources through the Drill start-up script, drill-env.sh. See Configuring cgroups to Control CPU Usage. (MD-3862)
Storage plugin management	Ability to export and save your storage plugin configurations to a JSON file for reuse. (MD-4621) Ability to manage storage plugin configurations in the Drill configuration file storage-plugins-override.conf. (MD-2732)
Integration with Hive and Hue	Hue integration with Drill is officially supported. See Integrate Hue with Drill. (MD-2382) Ability to specify Hive properties at the session level through the store.hive.conf.properties option. See Setting Hive Properties. (MD-4370) A new option, store.hive.maprdb_json.optimize_scan_with_native_reader, enables Drill to use the native Drill reader to read Hive MapR Database JSON tables. When you enable this option, Drill typically performs faster reads of data and applies filter pushdown optimizations. See SET maprdb Format Plugin Options. (MD-3662)
Parquet filter pushdown and partition pruning	Drill can infer filter conditions for join queries and push the filter conditions down to the data source. (MD-3542, MD-2181, MD-1399, MD-1272) Parquet filter pushdown pushes filters past the ITEM operator; useful for queries on complex fields. (MD-1271) Parquet performance improvements. (MD-2518, MD-1582) Drill uses a native reader to read Hive tables when the store.hive.optimize_scan_with_native_readers option is enabled; Drill reads data faster and applies filter pushdown optimizations. (MD-1267) Drill can pushdown filters into the Systems table. (MD-4055)
Query profiles	Profiles in Drill show the amount of memory used by the Unordered Receiver operator. (MD-4260)
Drivers	Drill 1.14 requires new versions of the ODBC and JDBC drivers. You can download the MapR ODBC and JDBC drivers for Drill 1.14. Earlier versions of the drivers do not work with Apache Drill 1.14.

For a list of additional features and improvements, see the Apache Drill 1.14 Release Notes.

Default Configuration Changes

Note the following changes to default configurations in Drill 1.14.0:

- Warden manages memory as a percentage of total system memory. See Configuring Drill Memory. (MD-2850)
- Spillable operators spill data to the /tmp/drill/spill directory on the MapR Filesystem. You can override this setting in drill-override.conf. Refer to the examples in drill-override-example.conf. (MD-4775)
- Query profile data is stored in maprfs:///apps/drill/profiles. The drill-override.conf file includes the sys.store.provider.zk.blobroot property that you can use to override the default location. See Configuring the ZooKeeper PStore Location. (MD-3527)

last-updated: Apr 11, 2022

Resolved Issues

Drill 1.14.0 includes the following resolved issues:

MapR Tracking Numbers	Resolved Issues
MD-4871	When querying a target MapR stream, the query stops running, and Drill prints a message stating "Failed to fetch messages within 200 milliseconds."
MD-4831	The Drill-on-Yarn package has inconsistent libMapRClient.so versions.
MD-4721	The following error should be logged as a warning instead of an error: "ERROR o.a.d.e.p.index.IndexDiscoverBase - No index returned from Admin.getTableIndexes"
MD-4666	DRILL-6612: Drill logs an assertion error when a query joins on a temporary table.
MD-4643	DRILL-6732: Disabled storage plugins work as if they are enabled.
MD-4607	DRILL-6557: Scanning input splits in Hive table causes an exceptionally long planning phase.
MD-4535	DRILL-6513: Drill should only allow valid values when users set planner.memory.max_query_memory_per_node. This option should be limited by direct memory; otherwise, there can be memory pressure and out-of-memory errors.
MD-4422	DRILL-6468: The Drillbit stays in QUIESCENT mode after an out-of-memory condition.
MD-4371	A specific query returns an exception when using the "equals" operator to filter on a boolean column.
MD-4251	DRILL-6474: Queries with ORDER BY and OFFSET (without LIMIT) do not return any rows.
MD-4156	When selecting a column from a Parquet file, a query may stop running and return an error stating "ArrayIndexOutOfBoundsException,"
MD-4133	The INFO log level provides excessive logging information.
MD-4107	Queries on Hive data sources may stop running amd return an error stating “UnsupportedOperationException: org.apache.hadoop.hive.ql.io.parquet.convert.ETypeConverter$8$1.”
MD-4102	UNION ALL queries return a UNION-ALLNumberFormatException
MD-4065	The Hash Aggregate operator uses ~2X memory.
MD-4048	DRILL-6282: Update Drill's metric dependencies.
MD-4033	Drill does not return results for some queries with an inner join.
MD-4017	DRILL-6254: Flatten queries may stop running due to an error that states “IllegalArgumentException: the requested size must be non-negative.”
MD-4005	The HashJoinSpill operator does not use memory efficiently.
MD-3997	DRILL-6250: The SQLLine start command and password appears in sqlline.log.
MD-3984	DRILL-6241: The Saffron properties configuration file has excessive permissions.
MD-3886	DRILL-6199: Filter push down does not work with more than one nested subquery.
MD-3716	DRILL-6223: Queries stop running if they select on all columns from a set of Parquet files.

last-updated: Apr 11, 2022

MD-3688	Impersonating a view owner does not work.
MD-3656	DRILL-6132 The HashPartitionSender leaks memory.
MD-3541	If Drill encounters JRE SIGSEGV, the Drillbit stops running.
MD-3525	Drill queries fails on function LOG10.
MD-2883	DRILL-4337: Querying Hive tables with INT96 fields causes Drill to fail.
MD-2082	DRILL 4807: An ORDER BY aggregate function in a window definition results in an assertion error.
MD-2048	The JSESSIONID cookie is not set with the HttpOnly flag.
MD-1549	DRILL-5188: TPC-DS query16 fails with the following exception: “IllegalArgumentException: Target must be less than target count”
MD-1487	DRILL-3855: Predicate pushdown does not occur for the UNION ALL operator.

Known Issues

Drill 1.14.0 has the following known issues:

MapR Tracking Numbers	Known Issues
MD-4938	The planner.index.covering_selectivity_threshold does not take effect in the execution plan when the option is set to values less than 1.0 for complex data.
MD-4906	For selectivity queries on secondary indexes, Drill may return an exception that states “ForemanException: One more more nodes lost connectivity during query.”
MD-4902	Queries with AND conditions on indexed complex type fields are not parallelized.
MD-4894	Queries with nested FLATTEN functions may stop running and return an error that states “Error: UNSUPPORTED_OPERATION ERROR: Hash aggregate does not support schema change.”
MD-4890	The query planner in Drill does not create an index-based query plan for queries with multi-level flattens or queries with intermediate filters that reference multi-level flattens.
MD-4865	Certain queries with AND conditions on alphanumeric data, such as keys, stop running, and Drill returns an error that states “UNSUPPORTED_OPERATION ERROR: In a list of type BIT, encountered a value of type FLOAT4.”
MD-4860	Simple select star queries return a NullPointerException when the data is highly complex.
MD-4846	When operators hit the maximum buffer size, Drill returns an OversizedAllocationException that states "Unable to expand the buffer. Max allowed buffer size is reached."
MD-4827	The ODBC driver returns INFINITY in capital letters instead of mixed case.
MD-4821	DRILL-6707: A query with a 10-way merge join fails with an IllegalArgumentException.
MD-4799	Data batches for the Project operator exceed the specified maximum.
MD-4773	A data verification failure in Functional/tpch/sf0dot01/smoke/parquet/join10-hash.q needs to be resolved.
MD-4759	An orderby on a field with [][] throws a NullPointerException.
MD-4739	Parallelism for complex secondary index plans are unrestricted.
MD-4738	The query planner incorrectly determines parallelism for secondary index plans.
MD-4736	Queries with multiple flatten functions may hang.

MD-4730	Drill may log the following exception during index planning: java.lang.ClassCastException: org.apache.drill.common.expression.FunctionCall cannot be cast to org.apache.drill.common.expression.SchemaPath
MD-4709	Queries pick non-covering index plans if the Streaming Aggregate operator is disabled.
MD-4704	Queries that have an exact equality filter with a map JSON literal stop running and return an error that states "SchemaChangeException - Error: Missing function implementation: [equal(MAP-REQUIRED, VARBINARY-REQUIRED)]."
MD-4577	The HashJoin operator allocates too much memory and slows down queries (TPCH 16) when spill to disk is enabled.
MD-4574	MapR Database cannot push filters on non-rowkey columns down to the data source when using a convert function with the byte_substr manipulation function; for example: … where convert_from(byte_substr(t.cf1.ADDR_WORK_OPT_OUT_DATE_DM,1,8),'UTF8')='20130402'
MD-4531	The total batches for the Project operator are not properly split and exceed the maximum specified.
MD-4518	The total batch size for the Project operator exceeds the maximum specified.
MD-4504	[DRILL-6465] Transitive closure is not working for joins with multiple local conditions.
MD-4479	The error message for group by queries on a complex type needs to be updated to state that they are unsupported.
MD-4404	The Datediff function returns the wrong result if Drill uses a timezone with DST.
MD-4377	Queries on complex data may stop running and return a NumberFormatException.
MD-4376	Queries on complex data may return a NullPointerException.
MD-4375	Queries with invalid filters on fields with complex types hang during the planning phase.
MD-4264	HashAgg Batch throws an IllegalStateException when asserts are enabled for a non-covering index plan.
MD-4229	DRILL-6329 : TPC-DS Query 66 failed due to out-of-memory errors.

Fixes

None

Limitations

None

Drill 1.13.0-2009 (EEP 5.0.5) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Drill Drill 1.13.0-2009 in EEP5.0.5.

Below are release notes for the Drill component included in the MapR Converged Data Platform. You may also be interested in the Apache Drill homepage and the Apache Drill release notes:

Version	Drill 1.13.0-2009
Release Date	September 2020

last-updated: Apr 11, 2022

MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.
Package Names	Navigate to https://package.mapr.com/releases/MEP/, and select your MEP and OS to view the list of package names.

New Features

- None.

Resolved Issues

Drill 1.13.0-2009 provides the following resolved issues and improvements:

Commit	Date (YYYY-MM-DD)	Comment
9d1841f	2020-06-05	DEVOPS-4070: Fix repository.mapr.com/nexus repository
8857248	2020-06-03	DRILL-7405: Avoiding download of TPC-H data
628933e	2020-06-04	DRILL-6470: Remove defunct repository

Drill 1.13.0-2009 also includes the following resolved issues and improvements:

MapR Tracking Number	Description
MD-6000	Query hung after upgrade from 1.10 to 1.13

Known Issues

- None.

Limitations

Older versions of Drill, such as Drill 1.10.0, supported the HBase plug-in, but Drill versions 1.11.0 through 1.16.0.x do not support querying HBase tables.

Drill 1.13.0-1912 (EEP 5.0.4) Release Notes

This section provides reference information for fixes in Drill 1.13.0-1912.

The following release notes apply to the 1.13.0-1912 version of the Drill component:

Version	1.13.0-1912
Release Date	December 2019
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.

last-updated: Apr 11, 2022

Package Names	Navigate to https://package.mapr.com/releases/MEP/, and select your MEP and OS to view the list of package names. For example: CentOS/Red Hat: • `mapr-drill-1.13.0.201910091439-1.noarch.rpm` • `mapr-drill-internal-1.13.0.201910091439-1.noarch.rpm` • `mapr-drill-yarn-1.13.0.201910091439-1.noarch.rpm` Ubuntu/Debian: • `mapr-drill_1.13.0.201910091438_all.deb` • `mapr-drill-internal_1.13.0.201910091438_all.deb` • `mapr-drill-yarn_1.13.0.201910091438_all.deb`

Fixes

Drill 1.13.0-1912 provides the following fixes:

Apache Drill Issue	MapR Tracking Number	Description
N/A	MD-5842	Drill confuses the data type when inserted both from OJAI API and dbshell
DRILL-7113	MD-5630	Drill on MapRDB cannot understand a null value
DRILL-7052	MD-5541	High risk phishing vulnerability in Web UI
DRILL-6664	MD-5077	Backport fix for DRILL-6664 into 1.13-R1 and 1.14-R1 branches
N/A	MD-5057	Drill Query on New File created on Volume that has reached namecontainerdatathreshold fails with gets I/O Exception (Failure while trying to get block map)
DRILL-6557	MD-4607	Drill should not get ALL input splits during planning phase for Hive partition table

Known Issues and Limitations

Older versions of Drill, such as Drill 1.10.0, supported the HBase plug-in, but Drill versions 1.11.0 through 1.16.0.x do not support querying HBase tables.

Drill 1.13.0-1904 (EEP 5.0.3) Release Notes

This section provides reference information for fixes in Drill 1.13.0-1904.

The following release notes apply to the 1.13.0-1904 version of the Drill component:

Version	1.13.0-1904
Release Date	May 2019
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.

last-updated: Apr 11, 2022

Package Names	Navigate to https://package.mapr.com/releases/MEP/, and select your MEP and OS to view the list of package names, for example: • mapr-drill-1.13.0.201905031711-1.noarch.rpm • mapr-drill-internal-1.13.0.201905031711-1.noarch.rpm • mapr-drill-yarn-1.13.0.201905031711-1.noarch.rpm

Fixes

Drill 1.13.0-1904 provides the following fixes:

Fix Tracking Number	Description
MD-4166	Fix creation of filter conditions for IS NULL and IS NOT NULL.
DRILL-6721	Fix SchemalessScan plan serialization / deserialization.
MD-5771	Username / password authentication for JDBC is not working anymore in Drill 1.15.

Known Issues and Limitations

None.

Drill 1.13.0-1901 (EEP 5.0.2) Release Notes
This section provides reference information for fixes in Drill 1.13.0-1901.

The following release notes apply to the 1.13.0-1901 version of the Drill component:

Version	1.13.0-1901
Release Date	February 2019
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.
Package Names	Navigate to https://package.mapr.com/releases/MEP/, and select your MEP and OS to view the list of package names, for example: • mapr-drill-1.13.0.201901241316-1.noarch.rpm • mapr-drill-internal-1.13.0.201901241316-1.noarch.rpm • mapr-drill-yarn-1.13.0.201901241316-1.noarch.rpm

Fixes

Drill 1.13.0-1901 provides the following fixes:

Fix Tracking Number	Description
MD-5379	DRILL-7018: Fixed Parquet buffer overflow when reading timestamp column.
MD-5062	DRILL-6664: Drill returns the same results for queries on the same column of a Parquet table with the same predicate.
MD-5019	DRILL-6797: Drill aggregation queries that use custom UDF to split column data in a Parquet file no longer fail with a compilation error if the filter criteria results in no rows.

MD-5014	DRILL-6776: The Drill Web UI page source no longer has links to external sites.
MD-5008	Drill can leverage indexes for queries with ORDER BY DESC on indexed columns; for example, if a secondary index is defined on column a, Drill leverages the index for the query: SELECT a FROM t ORDER BY a DESC;
MD-4936	DRILL-6773: Queries, with column aliases, that returned inconsistent results after a Drill patch was applied no longer return inconsistent results.
MD-4851	Drill queries on MapRDB JSON tables no longer fail on schema change when the only distinct values are null and text.
MD-4780	DRILL-6746: Queries no longer hang when one of the nodes encounters a connection error.

Known Issues

See the Drill 1.13-1803 release notes.

Drill 1.13.0-1808 Release Notes

This section provides reference information for fixes in Drill 1.13.0-1808.

The following release notes apply to the 1.13.0-1808 version of the Drill component:

Version	1.13.0-1808
Release Date	September 2018
MapR Version Interoperability	Component Versions for Released EEPs
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names, for example: • mapr-drill-1.13.0.201809261426-1.noarch.rpm • mapr-drill-internal-1.13.0.201809261426-1.noarch.rpm • mapr-drill-yarn-1.13.0.201809261426-1.noarch.rpm

Fixes

Drill 1.13.0-1808 provides the following fixes:

Fix Tracking Numbers	Description
MD-4156	Queries on a single Parquet column no longer fail with an ArrayIndexOutOfBoundsException.
MD-4248	(DRILL-6442) Hbase disk cost and row count estimates are adjusted when filter push down is applied; also, includes fixes for issues in the HBaseTestsSuite's init method.
MD-4259	LIMIT queries on MapR Database JSON tables with indexes now return the correct results.
MD-4403	The compiler no longer returns an 'inline' specifier error when compiling the Drill native client on CentOS 7, Ubuntu 14, and Windows.
MD-4444	Issuing the CTAS command against a MapR Database table no longer returns a NullPointerException.

last-updated: Apr 11, 2022

Fix Tracking Numbers	Description
MD-4509	Query plans with CAST indexes are generated for LIKE filters with CAST functions.
MD-4969	The client POM file now includes the org.apache.drill.contrib dependency.

Known Issues and Limitations

See the Drill 1.13-1803 release notes.

Drill 1.13-1803 Release Notes

This section provides reference information, including new features, improvements, resolved issues, known issues, and limitations for Drill 1.13.0-1803.

The following release notes apply to the 1.13.0 version of the Apache Drill component included in the MapR Converged Data Platform.

Version	1.13.0
Release Date	April 2018
MapR Version Interoperability	See the EEP Components and OS Support, Interoperability Matrix, and Drill Support Matrix.
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in this Release

This release of Drill includes a fix for CVE-2017-12197 and the following new features:

- Parquet planning improvements, including:
  - (MD-3445, MD-1312) Performance improvements with support for project push down, filter push down, and partition pruning on star queries in common table expressions (CTEs), views, and subqueries.
- Parquet filter pushdown improvements, including:
  - (MD-1264) Filtering on the IS [NOT] NULL and IS [NOT] TRUE|FALSE operators.
  - (MD-3821) Filtering on boolean values.
  - (MD-1270) Casting between date, time, and timestamp values.
  - (MD-1269) Support for files with multiple row groups.
- Runtime filter pushdown support for queries with join conditions on rowkeys.
  - (MD-1293) Drill can push down rowkey join filters to MapR Database at runtime.
- (MD-1947) Upgrade to Calcite version 1.15.
- (MD-1086) Ability to configure cgroups to control Drill CPU usage when Drill runs under YARN.
- (MD-2234) Ability to gracefully shut down a Drillbit.
- (MD-2404) Support for SPNEGO to extend the Kerberos-based single sign-on authentication mechanism to HTTP.
- SQL syntax highlighting for queries and snippets (SQL templates).

last-updated: Apr 11, 2022

- Ability to define the maximum amount of cumulative memory allocated to the Drill process during startup.

For a list of additional features and improvements, see the Apache Drill 1.13 Release Notes.

Improvements

This release of Drill includes the following improvements:

MapR Tracking Numbers	Improvements
MD-3836	DRILL-6174 Parquet pushdown planning improvements
MD-3711	Batch Sizing for Operators in EEP 5.0
MD-3657	DRILL-6071 Limit batch size for flatten operator
MD-3611	DRILL-6145 Implement Hive MapR Database JSON handler
MD-3428	DRILL-6115 SingleMergeExchange is not scaling up when many minor fragments are allocated for a query
MD-3340	DRILL-6147: Limit batch size for Flat Parquet Reader
MD-3312	Proposal Auto Mem Allocation
MD-3265	Query with union on empty directory on any side fails
MD-2920	Drill contains updated Hive libraries with Hive version 2.1.1-mapr-1710. With the update, Drill supports queries on transactional (ACID) and non-transactional Hive bucketed ORC tables. The updated libraries are backward compatible with earlier versions of the Hive server and metastore.
MD-2074	Memory Fragmentation Fundamentals Completion
MD-1360	Projection push down using WITH clause needs to work on Parquet files in addition to views
MD-1297	Improve Drill Join Performance on MapR Database Tables On Primary Indexes

Resolved Issues

This release of Drill includes the following resolved issues:

MapR Tracking Numbers	Resolved Issues
MD-4026	Drillbit failed to start after v6.0.0 -> v6.0.1 core upgrade.
MD-3997	DRILL-6250: Sqlline start command with password appears in the sqlline.log.
MD-4110	OJAI query through Drill fails due to impersonation.
MD-4099	Fix protocol incompatibility between 1.12.0-mapr and 1.13.0-mapr client and bit.
MD-4049	Query with runtime filter pushdown returns incorrect results.
MD-4074	HashAggBatch.innerNext() OOM error.
MD-4108	Queries on wide column parquet dataset return incorrect results.
MD-4101	Queries going to Drill fail with java.lang.IllegalArgumentException: org.apache.zookeeper.KeeperException$NoAuthException: KeeperErrorCode = NoAuth for /drill/cluster114-drillbits/2ffc1e3c-0ff7-490c-bf23-98365cc7eaf2.
MD-4100	Update the default Heap for Drill to 4GB.
MD-4098	Drillbit crashes when canceling a hash agg query.
MD-4082	Query fails with IndexOutOfBoundsException: Index: 0, Size: 0.
MD-4077	OJAI is unable to query QueryService (Drill) as root.

last-updated: Apr 11, 2022

MD-4071	Update the Hadoop and MapR versions in the pom.xml file.
MD-4056	Drillbit startup fails with "No configuration setting found for key 'drill.exec.options.store.parquet.reader.enableFSRetry'".
MD-4054	Queries with runtime filter pushdown fail with NPE when subquery contains filter on nested fields.
MD-4039	Query encounters INTERNAL_ERROR ERROR: index: -25614, length: 0 (expected: range(0, 65536)).
MD-4017	DRILL-6254: IllegalArgumentException: the requested size must be non-negative.
MD-4013	DRILL-6275: Drillbit direct_current memory usage is not populated/updated.
MD-4002	Query is not honoring the batch size limit.
MD-3986	Queries are hanging.
MD-3984	DRILL-6241: Saffron properties config file still has the excessive permissions.
MD-3957	Error in parquet record reader.
MD-3956	Query hangs.
MD-3948	Allocated batch size is larger than requested.
MD-3947	Record counts are the same but batch sizes are different.
MD-3946	Batch Sizes are inconsistent when record sizes are the same.
MD-3945	MapR Drill does not compile because JDBC driver JAR is too large.
MD-3925	Query using index plan fails with "DrillRuntimeException: Join only supports implicit casts between <types>. Add explicit casts to avoid this error".
MD-3886	DRILL-6199: Filter push down doesn't work with more than one nested subqueries
MD-3880	RowKeyJoin caused IndexOutOfBoundsException.
MD-3875	DRILL-6151 Fragment executors may terminate without sending final batch to a downstream causing query to hang.
MD-3865	DRILL-6217: NaN/Inf: NestedLoopJoin processes NaN values incorrectly.
MD-3864	Error is displaying while accessing query profiles via the Web-UI.
MD-3863	DRILL-6187: Exception in RPC communication between data server and data client.
MD-3857	NullPointerException occurs on HiveDrillNativeParquetScan.
MD-3824	dir0 does not work when the directory structure contains Avro files(DRILL-4120).
MD-3822	DRILL-6216: Metadata mismatch for sys.options table.
MD-3802	DRILL-6023 Graceful shutdown hardening.
MD-3784	DRILL-6189: User credentials appear in the Drill logs if the log level is set to ALL.
MD-3783	DRILL-6189: Drill config files have the excessive permissions.
MD-3726	Simple Order by queries (without limit) when an index is used are showing regression.
MD-3670	Fix NPE during physical plan submission for various storage plugins.
MD-3634	Find on a table using Dill fails with ClassNotFoundException: org.ojai.store.QueryResult.
MD-3610	Drill query hanging.
MD-3598	Address bugs related to Nan / Inf w/ MD-2745.
MD-3455	Queries with GROUP BY w/o WHERE clause using index plans returns rows with possible incorrect precision.

last-updated: Apr 11, 2022

MD-3575	DRILL-6192: Drill is vulnerable to CVE-2017-12197.
MD-3208	Misleading message when multi-column subquery is projecting within an IN clause.
MD-3188	DRILL-5903: Query encounters "Waited for 15000ms, but tasks for 'Fetch parquet metadata' are not complete."
MD-3187	DRILL-5902: Regression: Queries encounter random failure due to RPC connection timed out.
MD-2922	Predicates joined by AND/OR do not return right results for time and timestamp with millisec or higher precision.
MD-2790	DRILL-5902 Queries encounter random failure due to RPC connection timed out.
MD-2684	DRILL-4708: connection closed unexpectedly.
MD-2544	Drillbit JVM dumps core when querying a table with SI.
MD-2041	Potential wrong result from spillable hash agg.
MD-1848	Drill not able to read MapR Database when FQDN is longer than 64 characters.
MD-1575	Not able to set exec.impersonation.inbound_policies from web interface.

Known Issues

This release of Drill includes the following known issues:

MapR Tracking Numbers	Known Issues
MD-4106	DRILL-6293: Unable to read Apache Hive(2.1.1) tables using Drill 1.13.0.
MD-3988	Query runs out of memory.
MD-4153	Add support for runtime filter pushdown when _id occurs on the right side of the join.
MD-4133	Excessive logging at INFO level.
MD-4109	Drillbit crashed when concurrency is high (48 clients) and -Dio.netty.buffer.bytebuf.checkAccessible=true not in the Drillbit start-up command line.
MD-4107	Queries on Hive datasource fail with UnsupportedOperationException: org.apache.hadoop.hive.ql.io.parquet.convert.ETypeConverter$8$1.
MD-4102	NumberFormatException from UNION-ALL query.
MD-4086	Clean up messages going into drillbit.out.
MD-4065	HashAgg uses ~2X memory.
MD-4048	DRILL-6283: Excluding io.dropwizard.metrics dependencies.
MD-4047	Queries frequently fail with scan errors when performing runtime filter pushdown.
MD-4046	Drill returns different query plans across runs for queries performing runtime filter pushdown.
MD-4031	Drill does not pushdown runtime filters for queries with IN subquery unless threshold is over 10%.
MD-4001	select count(*) the Count from sys.drillbits works fine from sqlline but fails from drillSession.executeStatement().
MD-3981	Runtime filter pushdown does not happen for equality join with scalar subquery.
MD-3929	DRILL-6219: Filter pushdown does not work with star operator if there is a subquery with its own filter.
MD-3924	Query with a nested sub-query having a filter on a non-existent field fails with NPE.
MD-3138	Invalid expected results for Functional/table_function/positive/drill-3149_10.q.

last-updated: Apr 11, 2022

Fixes

None

Limitations

None

Drill 1.12.0-1904 (EEP 4.1.4) Release Notes

This section provides reference information for fixes in Drill 1.12.0-1904.

The following release notes apply to the 1.12.0-1904 version of the Drill component:

Version	1.12.0-1904
Release Date	May 2019
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.
Package Names	Navigate to https://package.mapr.com/releases/MEP/, and select your MEP and OS to view the list of package names, for example: • mapr-drill-1.12.0.201904160030-1.noarch.rpm • mapr-drill-internal-1.12.0.201904160030-1.noarch.rpm • mapr-drill-yarn-1.12.0.201904160030-1.noarch.rpm

Fixes

Drill 1.12.0-1904 provides the following fixes:

Fix Tracking Number	Description
DRILL-6721	Fix SchemalessScan plan serialization / deserialization.

Known Issues and Limitations

None.

Drill 1.12.0-1901 (EEP 4.1.3) Release Notes

This section provides reference information for fixes in Drill 1.12.0-1901.

The following release notes apply to the 1.12.0-1901 version of the Drill component:

Version	1.12.0-1901
Release Date	February 2019
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586.
Package Names	Navigate to https://package.mapr.com/releases/MEP/, and select your MEP and OS to view the list of package names, for example: • mapr-drill-1.12.0.201901030219-1.noarch.rpm • mapr-drill-internal-1.12.0.201901030219-1.noarch.rpm • mapr-drill-yarn-1.12.0.201901030219-1.noarch.rpm

Fixes

Drill 1.12.0-1901 provides the following fixes:

Fix Tracking Numbers	Description
MD-5264	DRILL-5902: Queries no longer encounter a random failure due to RPC connection timeouts.
MD-4936	DRILL-6773: Queries, with column aliases, that returned inconsistent results after a Drill patch was applied no longer return inconsistent results.

Known Issues and Limitations

See the Drill 1.12.0-1801 release notes.

Drill 1.12.0-1808 Release Notes

This section provides reference information for fixes in Drill 1.12.0-1808.

The following release notes apply to the 1.12.0-1808 version of the Drill component:

Version	1.12.0-1808
Release Date	September 2018
MapR Version Interoperability	Component Versions for Released EEPs.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names, for example: • mapr-drill-1.12.0.201807100435-1.noarch.rpm • mapr-drill-internal-1.12.0.201807100435-1.noarch.rpm • mapr-drill-yarn-1.12.0.201807100435-1.noarch.rpm

Fixes

Drill 1.12.0-1808 provides the following fixes:

Fix Tracking Numbers	Descriptions
MD-4509	Query plans with CAST indexes are generated for LIKE filters with CAST functions.
MD-4444	Issuing the CTAS command against a MapR Database table no longer returns a NullPointerException.
MD-4156	Queries on a single Parquet column no longer fail with an ArrayIndexOutOfBoundsException.
MD-4048	The io.dropwizard.metrics have been removed from Drill.
MD-3979	Queries that use CAST index plans no longer fail with a NullPointerException.
MD-3716	(DRILL-6223) Queries on complex nested data in Parquet files no longer fail when selecting on all columns.
MD-3544	Queries on complex nested data in Parquet files no longer fail.
DRILL-6557	Drill no longer fetches all the input splits when Hive statistics provide the number of bytes, which improves query planning time for queries on Hive data.

last-updated: Apr 11, 2022

Fix Tracking Numbers	Descriptions
DRILL-6204	When the store.hive.optimize_scan_with_native_readers option is enabled, Drill separates the partitioned columns from the non-partition columns when reading Hive data directly from the filesystem. This prevents Drill from returning an error if the reader cannot find partitioned columns in the table schema.
DRILL-6195	Drill can query Hive partitioned bucketed tables.
DRILL-6164	Drill no longer returns out-of-memory errors due to memory leaks caused by the buildup of Parquet reader references that the scanner does not remove.
DRILL-5978	Hive libraries are updated to version 2.1.2-mapr-1710.
DRILL-4185	Drill supports queries with JOIN and UNION [ALL] operators on empty directories.

Known Issues and Limitations

See the Drill 1.12.0-1801 release notes.

Drill 1.12.0-1801 Release Notes

This section provides reference information, including new features, default configuration changes, improvements, resolved issues, known issues, and limitations for Drill 1.12.0-1801.

The following release notes apply to the 1.12.0 version of the Apache Drill component included in the MapR Converged Data Platform.

Version	1.12.0
Release Date	February 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536, Interoperability Matrix, and Drill Support Matrix.
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in this Release

- (MD-1649) Index-based query plans for queries without filters, including queries with GROUP BY, JOIN, and DISTINCT projections. See Index Planning in Drill.
- (MD-2301) Ability to submit queries from the REST API when impersonation is enabled and authentication is disabled. See Submitting Queries.
- (MD-2745) Support for NaN (Not-a-Number) and Infinity (both POSITIVE and NEGATIVE) as numeric values. See JSON Data Model.
- (MD-2490, DRILL-5723) System options improvements, including a new internal system options table.

Default Configuration Changes

The default size for the `store.parquet.block-size` parameter is now 268435456 (256 MB), the same size as MapR filesystem chunk sizes. Previously, the default size was 536870912 (512 MB). See Configuring the Parquet Block Size for more information.

Improvements

In addition to the features mentioned above, Drill 1.12 includes the following improvements:

MapR Tracking Numbers	Improvements
MD-3365	Bounds checking for direct memory is now disabled by default. You can enable bounds checking, as explained in Configuring Drill Memory.
MD-3314	Unable to re-run queries from Profiles tab with impersonation and without authentication
MD-3086	Improve the row count estimates for LIMIT queries to avoid over-parallelization
MD-3053	MapR Database Statistics Improvements
MD-2867	DRILL-5855 Provide a useful error message when the SystemOptionManager is queried for a non-existent option
MD-2842	Projecting a Few Columns with DISTINCT, LIMIT and ORDER BY without a Filter Specified should use a Secondary Index Tables for Fast Query Response Times
MD-2758	Drill performance improvement for text search vs Impala
MD-2745	Error parsing JSON containing token 'NaN'
MD-2647	Enable UTF-8 support in query string by default
MD-2518	Parquet Scan Performance Improvement
MD-2403	Drillbit fails to start when only keystore path is provided without keystore password.
MD-2323	Review & update the default values of index planning config options
MD-2319	Query Containing Wildcard in String Matching Filter Runs 4x Slower Than Impala
MD-2301	Support Impersonation without authentication for REST API
MD-2196	DRILL-4264: Allow field names to include dots
MD-2002	Operator Specific Handle for Empty Batches
MD-1752	Throttling-based per-query memory assignment
MD-1650	Drill Join query with Secondary indexes
MD-1649, MD-3267	Support Index Planning for Queries with GROUP BY w/o WHERE clause
MD-1431	Improve Performance of Drill's Parquet Scan Operator (Amadeus)
MD-1247, MD-1810	Drill only initializes enabled storage plugins and registers schemas for workspaces that relate directly to a query, which improves query time for small queries that run concurrently. The Foreman in Drill caches the root schema of the JSON document for use in subsequent queries. Previously, Drill initialized all enabled storage plugins and registered workspace schemas for each query without caching the root schema, which caused an increase in query time if a storage plugin was misconfigured or the underlying data source was down or slow.
MD-580	Improve performance of Parquet scan operator

For a list of additional features and improvements, see Apache Drill 1.12 Release Notes.

Resolved Issues

This release of Drill includes the following resolved issues:

last-updated: Apr 11, 2022

MapR Tracking Numbers	Resolved Issues
MD-3471	Intermittent permission denied errors with Drill queries on Views
MD-3423	DRILL-5880 : Advanced tests fail with error: "UNSUPPORTED_OPERATION ERROR: This query cannot be planned possibly due to either a cartesian join or an inequality join"
MD-3322	You no longer need to enable the planner.index.force_sort_noncovering option for Drill to correctly return the results of a non-covering query in sorted order.
MD-3249	Query with union on 520 columns failing with compilation exception
MD-3246	Investigate jackson-databind vulnerabilities CVE-2017-15095 & CVE-2017-7525
MD-3196	Query hangs when we concat many items
MD-3155	Simba JDBC driver dependency issues with third party applications
MD-3131	DRILL-5967 Memory leak by HashPartitionSender
MD-3071	Queries (intermittently) fail with SYSTEM ERROR: RpcException: Data not accepted downstream.
MD-3066	Queries with join on two or more tables fail to pick hash index plans for all eligible tables
MD-3062	DRILL-5822: The query with "SELECT *" with "ORDER BY" clause and `planner.slice_target`=1 doesn't preserve column order
MD-3055	Remove hbase and hbase_describe plugin templates from 1.11.0-mapr branch
MD-3031	Duplicate columns projected from MapR Database Scan for large IN condition
MD-2987	DRILL-5771: Fix serDe errors when format plugin is used with table function
MD-2983	Using option `planner.index.use_hashjoin_noncovering`=true is causing NPE
MD-2979	Memory leak by ParquetRowGroupScan
MD-2978	Drill isn't honoring option not to save query profiles
MD-2894	Order by doesn't sort columns when window function is involved in the query
MD-2843	MIN MAX DIR tests fail and return incorrect results
MD-2807	DRILL-5922 Intermittent Memory Leaks in the ROOT allocator
MD-2771	skip.header.line.count does not work for Hive table with data file size > chunk size
MD-2770	java.lang.NoClassDefFoundError exception not handled in org.apache.drill.exec.rpc.security.ClientAuthenticatorProvider
MD-2769	DRILL-5819: Default value of security.admin.user_groups and security.admin.users is "true"
MD-2730	Null Pointer Exception with query using table function
MD-2723	Client to Bit Encryption shown as disabled in the drill webserver when SSL encryption is enabled
MD-2712	Queries failed with AssertionError: rel has lower cost than the best cost of subset

MD-2702	Rename ssl property enableHostVerification as disableHostVerification
MD-2635	Enabling SSL using hadoop config doesn't honor "ssl.server.keystore.keypassword" property
MD-2632	DRILL-5775: Select * query on a maprdb binary table fails
MD-2621	Fix XSS vulnerabilities in Drill
MD-2619	Drill fails to start when https is enabled with a keystore having keyPassword different from keyStorePassword
MD-2617	NullPointerException: Querying maprdb json tables
MD-2528	Parquet int_64 causes UnsupportedOperationException in Drill
MD-2516	Hash function produces skewed results on String values with same leading prefix
MD-2509	Some queries in concurrent execution get stuck in STARTING phase
MD-2483	exec.hashagg.num_partitions cannot be set by "alter session" or "alter system"
MD-2464	DRILL-5715: Performance of refactored HashAgg operator regressed
MD-2459	DRILL-5714: Fix NPE when MapR Database plugin is used in table function
MD-2436	Issue with drill-config.sh checking java version
MD-2412	Query with 2-way JOIN fails with "Hash join does not support schema changes"
MD-2379	drill.connections.rpc.user.<encrypted/unencrypted> metric can result in a neg value
MD-2375	RowKeyJoin is generated instead of IndexScan for queries with derived table/ table functions
MD-2292	Query hangs with bit to bit authentication
MD-2216	DRILL-5660: Drill 1.10 queries fail due to Parquet Metadata "corruption" from DRILL-3867
MD-2201	DRILL-4469: SUM window query returns incorrect results over integer data
MD-2130	Drill wrong old date-time values displaying
MD-2125	Queries fail with "Failed to create schema tree." when Impersonation is enabled and logins are anonymous
MD-2041	Potential wrong result from spillable hash agg
MD-1945	DRILL-4139: Fix parquet partition pruning for BIT, INTERVAL and DECIMAL types
MD-1828	DRILL-5507 Millions of "Failure finding Drillbit running on host" info messages in foreman logs
MD-1586	Pam authentication with Centrify doesn't work by JPam restriction
MD-1455	Hash aggregate does not support schema changes
MD-1205	Simple query with only one join condition failed with "Hash join does not support schema changes"
MD-1171	Security: Stored XSS (Drillbit>>Query) vulnerability

last-updated: Apr 11, 2022

MD-1168	Drill Web UI Page Source Has Links To External Sites
MD-1122	Fix the Hive default storage plugin template.
MD-1115	DRILL-5464: HashAgg throws a SchemaChangeException when atleast 1 scan fragment reads 0 rows
MD-965	UnsupportedOperationException: Unable to get holder type for minor type [LATE] and mode [OPTIONAL].
MD-853	Query on TPC-H SF100 dataset fails with "Hash join does not support schema changes"

Known Issues

This release of Drill includes the following known issues:

MapR Tracking Numbers	Known Issues
MD-3564	Min/Max functions logic should be improved
MD-3563	Order by clause works inconsistently when sorting columns with NaN
MD-3534	Query runs out of memory when using SI indices
MD-3521	Nan/Inf data types: strange query result with INNER JOIN operator when selecting 1 column
MD-3497	NaN/Infinity: some functions don't work as expected
MD-3449	Queries using non-covering index plans with larger rowkey join batch size fail with "CONNECTION ERROR: Exceeded timeout while waiting send intermediate work fragments to remote nodes"
MD-3345	Query using index plans (intermittently) fails with IllegalStateException: #Scan ranges should be greater than 0 since estimated rowcount=[non-zero]
MD-3344	TPC-H Query 1 returns rows with incorrect precision on Parquet SF100 dataset
MD-3201	Multiple occurrences of "AnalyzerException: Error at instruction 98: Expected an object reference, but found ." seen in drillbit.out
MD-2455	Query with INTERVAL in predicate does not return any rows

Fixes

None

Limitation

(Ubuntu systems only) If you uninstall Drill 1.11 or upgrade from Drill 1.11 to Drill 1.12, you must manually backup log files, third-party JAR files, and configuration files and settings prior to performing the uninstall or upgrade procedure. Failure to do so results in the loss of the files and configurations.

Drill 1.11.0-1710 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Drill 1.11.0-1710.

The following release notes apply to the 1.11.0 version of the Apache Drill component included in the MapR Converged Data Platform.

Version	1.11.0

Release Date	November 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536, Interoperability Matrix, and Drill Support Matrix.
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in this Release

- Secondary index support on MapR Database for JSON tables.
- MapR default security configuration that secures the cluster and ecosystem components using MapR security (mapr tickets) to provide authentication, authorization, and encryption.
- Client to Drillbit Encryption using SSL/TLS.
- Spill to disk for the Hash Aggregate operator (in addition to the Sort operator).
- Throttle-based resource assignment (queuing feature).
- Drill integration with Zeppelin.

In addition to the features mentioned above, Drill 1.11 also includes the following features and improvements:

MapR Tracking Numbers	Features
MD-2490	Addition of an internal options table.
MD-1813	Multiplexing session over a connection.
MD-1655	Provide an option to persist Query Profiles in a Drillbit server's memory
MD-1245	MapR-SASL Plugin for ODBC/Drill (Authentication & Encryption)
MD-536	SASL support between Drillclient and Drillbit

MapR Tracking Numbers	Improvements
MD-2962	Add a UDF to decode encoded fieldpath
MD-2871	Restricted scan (for non-covering) gets over-parallelized due to scan range size
MD-2799	DRILL-5811 Large number of "Failure finding Drillbit" messages when using MapR File System
MD-2789	Fix correct batch size for merge back during non-covering query.
MD-2693	Drill should use `com.mapr.db.index.IndexDesc.getNullOrdering()` to decide if an index can be used for order by
MD-2656	Drill needs to return more specific error for DB
MD-2647	Enable UTF-8 support in query string by default
MD-2506	Standardize on JDBC driver that gets shipped with Drill and sqlline
MD-2422	Drill performance - doing less than 40 queries/sec
MD-2416	RestrictedJsonTableGroupScan should support projection pushdown
MD-2403	Drillbit fails to start when only keystore path is provided without keystore password.
MD-2339	Additional log events for debugging index selection

MD-2319	Query Containing Wildcard in String Matching Filter Runs 4x Slower Than Impala
MD-2301	Support Impersonation without authentication for REST API
MD-2278	Include flags controlling secure Drill ZK ACLs in the secure installer
MD-2225	Review the names of index planning config options
MD-2196	DRILL-4264: Allow field names to include dots
MD-2192	Provide log tracing of a query submitted by an OJAI client for user debugging
MD-2074	Memory Fragmentation Fundamentals Completion
MD-2067	Handle the case of order-by with no filter conditions
MD-2002	Operator Specific Handle for Empty Batches
MD-1987	Drill should read MapR Database tablets in parallel
MD-1962	Create configuration file for distribution specific configuration
MD-1836	Leverage hash index when doing secondary index planning
MD-1823	Add logging information for listing candidate indexes for a given query
MD-1815	Leverage MapR Database batch gets for better performance.
MD-1789	Cast function support pull request and code review process
MD-1786	Push down limit on MapR Database table scan
MD-1752	Throttling-based per-query memory assignment
MD-1673	Add Statistics support for MapR Database
MD-1665	Provide query level option to pick specific indexes
MD-1643	Disable table caching by default
MD-1498	Plan queries against MapR Database JSON faster
MD-1318	Improve Low Latency Performance of Drill
MD-1209	Configuring ZooKeeper Access Controls for Drill on a Secure Cluster

Fixes

None

Known Issues and Limitations

This release of Drill includes the following known issues:

MapR Tracking Numbers	Known Issues
MD-3171	Drillbit down due to fs client assertion in Java_com_mapr_fs_jni_MapRTableTools_GetScanner
MD-3162	DRILL-5946: Document maprdb format plugin options
MD-3160	Hit ExpressionParsingException: Expression has syntax error! line 1:0:no viable alternative at input 'TIME'
MD-3155	Simba JDBC driver dependency issues with third party applications
MD-3131	Memory leak by HashPartitionSender
MD-3108	planner.memory.max_query_memory_per_node should be default to 4GB

MD-3089	Some queries with join on two or more tables only pick non-covering index plans
MD-3071	Queries using hash index plans (intermittently) fail with SYSTEM ERROR: RpcException: Data not accepted downstream.
MD-3047	Apache Drill JDBC driver from mapr-drill package not works
MD-3041	Identify closed jdbc connection
MD-3040	need a better error message - RpcException: Drillbit (demo-179.lab) does not require auth, but auth is enabled.
MD-3037	Performance issues with queries against MapR Database JSON tables (full table scans)
MD-3031	Duplicate columns projected from MapR Database Scan for large IN condition
MD-2979	Memory leak by ParquetRowGroupScan
MD-2950	Full outer join on two empty hive tables throws IllegalArgumentException: Full outer join not currently supported
MD-2948	SI: Non-covering queries hang after drillbit restart
MD-2848	IteratorValidatorBatchIterator is throwing exception when assertions are enabled.
MD-2838	After upgrading from MapR Drill 1.10.0 to 1.11.0, drillbit hangs on startup
MD-2837	Queries on MapR Database 5.x tables hang
MD-2809	Scalar replacement failures
MD-2791	Cancelled query using index plan logs several errors
MD-2790	Queries fail with various CONNECTION ERRORs
MD-2695	Test Framework - Test fails with DATA_READ_ERROR : Failed to load table into HBase
MD-2589	Drillbit startup has errors: "Can't load log handler java.util.logging.FileHandler"
MD-2544	drillbit jvm dumps core when querying a table with SI
MD-2528	Parquet int_64 causes UnsupportedOperationException in Drill
MD-2515	Evaluate Codegen Compilers for performance overhead
MD-2513	Querying last column in CSV causes IndexOutOfBoundsException
MD-2455	Query with INTERVAL in predicate does not return any rows
MD-2420	External sort with higher memory takes longer to finish vs. smaller memory
MD-2419	Query fails with OOM with queuing enabled and exchanges disabled
MD-2415	Resource leak during Hash Agg spill to disk longevity run
MD-2400	Random Jenkins unit test failure due to ".crc" files in spill directory
MD-2397	Zeppelin cannot connect to Drill through the JDBC driver on a secure MapR cluster when Zeppelin has Kerberos authentication enabled.
MD-2285	Filter with interval was not fully pushed down
MD-2218	Direct memory fragmentation cause OOM while running OJAI queries
MD-2141	Predicates on indexed field, joined by OR, do not return the correct results for time and timestamp data types
MD-2120	Order by queries on short and byte datatypes fail
MD-2112	Query with LIKE Predicate on Non-Varchar Fields Returns No Results
MD-2060	Hash Partitioner Sender taking unusual amount of time

last-updated: Apr 11, 2022

MD-2041	Potential wrong result from spillable hash agg
MD-1964	OOM when Hash Aggr outputs large aggregate data types
MD-1923	Simple query failed when planner.use_simple_optimizer=true
MD-1867	SYSTEM ERROR: CannotPlanException when query MDB Json table

Resolved Issues

This release of Drill includes the following resolved issues:

MapR Tracking Numbers	Resolved Issues
MD-2825	initcap function returning incorrect result
MD-2769	Default value of security.admin.user_groups and security.admin.users is "true"
MD-2738	Memory Limit Exception in External Sort
MD-2621	Fix XSS vulnerabilities in Drill
MD-2617	NullPointerException: Querying maprdb json tables
MD-2531	Error connecting to Drill from Data Stage through Simba ODBC driver
MD-2436	Issue with drill-config.sh checking java version
MD-2216	DRILL-5660: Drill 1.10 queries fail due to Parquet Metadata "corruption" from DRILL-3867
MD-2197	DRILL-4511: refresh over empty folder results in error, we need a better error message
MD-2195	DRILL-4755: StringIndexOutOfBoundsException seen with CONVERT_FROM function
MD-2146	DRILL-4970: Wrong results when casting double to bigint or int
MD-2144	DRILL-4720: MINDIR() and IMINDIR() functions return no results with metadata cache
MD-2130	Drill wrong old date-time values displaying
MD-2125	Queries fail with "Failed to create schema tree." when Impersonation is enabled and logins are anonymous
MD-1988	Spillable Hash Agg : OOM while constructing the hash table
MD-1980	DRILL-5130: UNION ALL difference in results
MD-1970	ArrayIndexOutOfBoundException occurs when querying MapR Database JSON with Drill
MD-1946	DRILL-5140: Fix CompileException in run-time generated code when record batch has large number of fields
MD-1945	DRILL-4139: Fix parquet partition pruning for BIT, INTERVAL and DECIMAL types
MD-1922	Use indexDesc instead of indexFid in getIndexTable API to support HashPartitioned Indexes
MD-1890	Drill Explorer on the mac does not connect when started as an app
MD-1848	Drill not able to read MapR Database when FQDN is longer than 64 characters
MD-1828	DRILL-5507 Millions of "Failure finding Drillbit running on host" info messages in foreman logs
MD-1776	Issue with column alias for nested table calculated column when using sum ()
MD-1671	Have to restart drillbits whenever hive metastore is restarted.
MD-1661	A query with limit 0 on MapR Database binary table not allowing filter to push past project

MD-1657	Random Error : Flatten does not support inputs of non-list values
MD-1640	Drill fails to compare multi-byte characters from hive table(DRILL-3250)
MD-1633	Query remains in CANCELLATION_REQUESTED state after "java.io.IOException: Broken pipe"
MD-1618	Moving a directory which contains a cache file causes subsequent queries to fail
MD-1607	Reverse a String: IndexOutOfBoundsException
MD-1597	simba and mapr jdbc driver differ in implementation
MD-1586	Pam authentication with Centrify doesn't work by JPam restriction
MD-1576	JBOSS Drill Connectivity - NullPointerException
MD-1524	linux ODBC is ALWAYS returning 64K as the length of a varchar or char
MD-1515	Drill planner/optimizer pushes down the Filter columns into Scan even after Filter is eliminated
MD-1473	Query failed with error Statement "break OrOP2" is not enclosed by a breakable statement with label "OrOP2"
MD-1455	Hash aggregate does not support schema changes
MD-1382	Use simple optimizer for low latency operational queries
MD-1361	Out of heap running CTAS against text delimited
MD-1359	Wrong results after set planner.enable_nljoin_for_scalar_only=false
MD-1339	Query text show as empty string in profile when queries are executed using PreparedStatements
MD-1256	unset MAPR_TICKETFILE_LOCATION for sqlline
MD-1238	Wrong result with Drill displaying date fields with parquet files generated by spark
MD-1237	Cancelled query stuck into CANCELLATION_REQUESTED forever
MD-1205	Simple query with only one join condition failed with "Hash join does not support schema changes"
MD-1168	Drill Web UI Page Source Has Links To External Sites
MD-1115	DRILL-5464: HashAgg throws a SchemaChangeException when atleast 1 scan fragment reads 0 rows
MD-965	UnsupportedOperationException: Unable to get holder type for minor type [LATE] and mode [OPTIONAL].
MD-853	Query on TPC-H SF100 dataset fails with "Hash join does not support schema changes" [MapR Database JSON Tables]
MD-752	Remove mapr jars from Drill packages and source them from $MAPR_HOME/lib

Drill 1.10.0-1808 Release Notes

This section provides reference information for fixes in Drill 1.10.0-1808.

The following release notes apply to the 1.10.0-1808 version of the Drill component:

Version	1.10.0-1808
Release Date	September 2018
MapR Version Interoperability	Component Versions for Released EEPs

last-updated: Apr 11, 2022

Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names, for example: • mapr-drill-1.10.0.201805300932-1.noarch.rpm • mapr-drill-yarn-1.10.0.201805300932-1.noarch.rpm

Fixes

Drill 1.10.0-1808 provides the following fixes:

Fix Tracking Numbers	Description
MD-3943	The io.dropwizard.metrics have been removed from Drill.
MD-3716	Queries on complex nested data in Parquet files no longer fail when selecting on all columns.
MD-3715	A Drillbit no longer hangs or loses connectivity.
MD-3544	Queries on complex nested data in Parquet files no longer fail.
MD-3471	Queries on views no longer intermittently return permission denied errors.
MD-3409	Drill returns the correct row count for queries on MapR Database JSON tables.
MD-3214	Drill reruns queries on Parquet data sources if they fail due to Parquet input stream read errors.
MD-3196	Queries with multiple concatenated items no longer hang.
MD-2758	Performance improvements for queries with the LIKE operator.
MD-2528	Drill can read Parquet columns with INT64 types.
DRILL-6140	Drill correctly lists operators in query profiles.
DRILL-6028	Drill splits the generated code in ChainedHashTable into blocks to avoid "code too large" error.
DRILL-5978	Hive libraries are updated to version 2.1.2-mapr-1710.
DRILL-5972	Improved performance for queries with filters on INFORMATION_SCHEMA.
DRILL-5971	Drill can read Parquet logical types INT64 and INT32.
DRILL-5906	The Hive version in Drill is upgraded to 1.2.0-mapr-1707 hive.version to avoid NullPointerException when querying Hive ORC tables.
DRILL-5857	Hive unit tests include a fix for NumberFormatException.
DRILL-5757	Queries with the CONVERT_TO_JSON function no longer fail when using fields that do not exist as a parameter.
DRILL-5697	Improved performance for queries with the LIKE operator to filter on pattern matching.
DRILL-5420	The ParquetAsyncPgReader no longer goes into an infinite loop during cleanup.
DRILL-5083	To avoid an infinite loop in the Merge Join, the status.getOutcome() method returns a failure if one of the batches has a “stop” status.
DRILL-4120	The Avro storage format allows implicit columns.
DRILL-4039	Queries no longer fail when non-ascii characters are used in string literals.

last-updated: Apr 11, 2022

Fix Tracking Numbers	Description
DRILL-3250	Drill no longer fails when comparing multi-byte characters from a Hive table.

Drill 1.10.0-1707 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Drill 1.10.0-1707.

The following release notes apply to the 1.10.0 version of the Apache Drill component included in the MapR Converged Data Platform.

Version	1.10.0
Release Date	August 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536, Interoperability Matrix, and Drill Support Matrix.
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in this Release

No new features in this release.

Fixes

This release of Drill on MapR includes the following fixes. For complete details, refer to the commit log for this project in GitHub.

MapR Fix	Description
MD-1970	Querying MapR Database JSON with Drill no longer produces an ArrayIndexOutOfBoundException.
MD-2273	Drill no longer returns IndexOutofBound errors when reading CSV files.
DRILL-5590	Drill no longer returns an IndexOutOfBoundsException when a text file contains more than 4,096 rows.
DRILL-5498	The CSV reader can handle duplicate column header names.
DRILL-5599	Queries no longer stay in the CANCELLATION_REQUESTED state after a connection with the client is interrupted.
MD-2050	Drill correctly calculates the number of rows and creates an optimal query plan.
DRILL-5496	You do not have to restart drillbits when a secure Hive metastore is restarted.
DRILL-5399	Running multiple queries concurrently no longer produces an UNSUPPORTED_OPERATION ERROR.
DRILL-5516	Drill optimizes memory usage for the HBase reader.
DRILL-5419	Drill now calculates the return string length for literals and certain string functions.
MD-1671, MD-1631	Drill no longer hangs or reports errors when retrieving schemas as the Hive server restarts.

last-updated: Apr 11, 2022

MapR Fix	Description
DRILL-5424	Queries that apply reverse functions on tables with long varchars no longer fail with IndexOutOfBoundsException errors.
DRILL-5413, DRILL-4678, DRILL-4347	Drill is updated with version of 1.4.0-drill-r21 Calcite.

Known Issues and Limitations

No known issues or limitations in this release.

Resolved Issues

No resolved issues in this release.

Drill 1.10.0-1703 Release Notes

The following release notes apply to the 1.10.0 version of the Apache Drill component included in the MapR Converged Data Platform.

Version	1.10.0
Release Date	April 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536, Interoperability Matrix, and Drill Support Matrix.
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in this Release

This Apache Drill release includes the following behavior changes that are specific to MapR:

- Support for authentication through MapR-SASL tickets.
- Support for installation on SUSE operating systems with Open JDK 1.8 and Oracle JDK 1.7 or 1.8.

This release also includes the following bug fixes and improvements:

- Tableau native connectivity.
- Support for the CREATE TEMPORARY TABLE AS (CTTAS) command.
- Support for Kerberos authentication between the client and drillbit.
- Support for authentication through MapR-SASL tickets.
- New JDBC connection option that improves fault tolerance when connecting directly to a Drill node from a client.
- Improved diagnostics in Drill Web Console with version and other query profile statistics.
- Improved Parquet compatibility with Hive/Spark generated Parquet files with support for INT96 timestamp datatype
- Configurable list of profiles in Drill web UI

Additional bug fixes and enhancements are listed in the Apache Drill 1.10.0 Release Notes

Fixes

MapR Fix	Description
Drill-5395	Region to fragment assignment logic no longer causes queries against MapR Database tables to fail with a null pointer exception.
MD-1457	The ODBC driver's DNS Configuration dialog box now supports Cluster IDs that include dots(.).
MD-1433	The planning stage now has better performance for queries against MapR Database binary tables.
MD-1259	The Apache Drill documentation now includes implicit column name parameters and details.
MD-1244	Using store.parquet.reader.int96_as_timestamp no longer throws IndexOutOfBoundsException.
MD-832	Drill verifies that all the running drillbits are in the same version. A warning displays in the Web UI whenever there is a mismatch.
MD-924	Drill compatibility issues with Parquet Date and Timestamp data types have been resolved.

Known Issues and Limitations

- MD-1588: When Drill is run under Warden on RedHat/CentOS operating systems, drillbits fail to start automatically after an upgrade to Drill 1.10.0.

  Workaround: Run the following command to manually start the drillbits:

  ```
  maprcli node services -name drill-bits -action start -nodes <node host
  names separated by a space>
  ```

- MD-1340: Select * query returns incorrect format for nested Date fields.

Drill 1.9.0-1703 Release Notes

The following release notes apply to the 1.9.0 version of the Apache Drill component included in the MapR Converged Data Platform.

Version	1.9.0
Release Date	April 2017
MapR Version Interoperability	MapR Drill 1.9.0 is certified on the MapR v5.1.0 and v 5.2.0 converged data platform. See EEP Components and OS Support on page 5536, Interoperability Matrix, and Drill Support Matrix.
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

None

Fixes

This MapR release includes the following fixes on the base release:

last-updated: Apr 11, 2022

MapR Fix Number	Description
MD-1433	The planning stage now has better performance for queries against MapR Database binary tables.
MD-1375	FileNotFoundException no longer displays when reading a parquet file.
MD-1314	The same query no longer produces different results when using the flatten operator.
Drill-5395	Region to fragment assignment logic no longer causes queries against MapR Database tables to fail with a null pointer exception.
Drill-5159	Drill MergeProjectRule now operates on Calcite logical convention.
Drill-5117	Queries on json files with 1000+ columns no longer fail with a compile error.
Drill-5056	UserException now writes more information to the log.

Known Issues

The following table lists the known issues in Drill 1.9.0:

Issue	Description
MD-1256	The MAPR_TICKETFILE_LOCATION variable in drill-env.sh needs to be unset so that a user other than the mapr user can connect to Drill through SQLLine and MapR-SASL.
MD-1229	Hive table partition pruning does not prune out all unnecessary partitions in some scenarios.
MD-1226	MapR-SASL plain authentication goes through Kerberos, by default, requiring regular users to have a valid Kerberos ticket when connecting to Drill using SQLLine through ZooKeeper.
MD-1208	Parquet filter pushdown does not prune enough partitions in queries where the predicate contains the TIME data type.

Drill 1.9.0 Release Notes

The following release notes apply to the 1.9.0 version of the Apache Drill component included in the MapR Converged Data Platform.

Version	1.9.0
Release Date	December 9, 2016
MapR Version Interoperability	MapR Drill 1.9.0 is certified on the MapR v5.1.0 and v5.2.0 Converged Data Platform. See Interoperability Matrices on page 5519 and Drill Support Matrix on page 5629.
Packages	See Package Names for MapR Ecosystem Packs (EEPs)

Noteworthy New Features in the MapR Distribution of Drill

This release provides enhanced query improvements with the following bug fixes and improvements:

- The Asynchronous Parquet Reader improves the performance of the Parquet Scan operator by increasing the speed at which the Parquet reader scans, decompresses, and decodes data. This feature is disabled by default.
- Parquet Filter Pushdown optimizes the performance by pruning extraneous data from a Parquet file to reduce the amount of data that Drill scans and reads when a query on a Parquet file contains a filter expression.
- Dynamic UDFs enable users to register and unregister UDFs in a muti-tenant environment using the new CREATE FUNCTION USING JAR and DROP FUNCTION USING JAR commands.
- Support for a variety of JOIN syntax generated by Tableau and other BI tools, including joins between tables with NULL column values.
- HTTPD Format Plugin adds the capability to query HTTP web server logs natively and also includes parse_url() and parse_query() UDFs that return maps of the URL and the query string.

Additional bug fixes and enhancements listed in the Apache Drill 1.9.0 Release Notes.

Default Configuration Changes

The default value for the `store.parquet.block-size` parameter is now 268435456 (256MB), the same size as MapR filesystem chunk sizes. Prior to this release, the default value was 536870912 (512 MB).

The `planner.enable_limit0_optimization` parameter is now enabled by default to optimize limit0 queries. Prior to this release, the option was disabled by default.

You can modify parameter values using the ALTER SYSTEM|SESSION command.

Resolved Issues

The following table lists resolved issues in Drill 1.9.0:

Issue	Description
MD-1217	When you use the JDBC URL format for a direct Drillbit connection, the driver can now shuffle between the Drillbits listed in the connection string instead of always selecting the first Drillbit listed.
MD-1163	The avgwidth stat for the timestamp data type (int64) is now 8 instead of 4 bytes.
MD-540	Query profiles now include the session options used for a query.

Known Issues

The following table lists the known issues in Drill 1.9.0:

Issue	Description
MD-1256	The MAPR_TICKETFILE_LOCATION variable in drill-env.sh needs to be unset so that a user other than the mapr user can connect to Drill through SQLLine and MapR-SASL.
MD-1229	Hive table partition pruning does not prune out all unnecessary partitions in some scenarios.
MD-1226	MapR-SASL plain authentication goes through Kerberos, by default, requiring regular users to have a valid Kerberos ticket when connecting to Drill using SQLLine through ZooKeeper.

last-updated: Apr 11, 2022

Issue	Description
MD-1208	Parquet filter pushdown does not prune enough partitions in queries where the predicate contains the TIME data type.

Limitations

See Drill-on-YARN Limitations.

Drill 1.8.0-1703 Release Notes

The following release notes apply to the 1.8.0 version of the Apache Drill component included in the MapR Converged Data Platform.

Version	1.8.0
Release Date	April 2017
MapR Version Interoperability	MapR Drill 1.8.0 is certified on the MapR v5.1.0 converged data platform. See the Interoperability Matrix and Drill Support Matrix.
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

No new features.

Fixes

This MapR release includes the following fixes on the base release:

MapR Fix Number	Description
MD-1314	The same query no longer produces different results when using the flatten operator.
MD-1217	The JDBC client randomly chooses a Drillbit in the cluster instead of selecting the first one in the list.
MD-1142	A simple query with a where condition on a particular column is no longer getting an illegalstateexception.
MD-1127	Drill 1.8.0 sandbox is now able to start even when loopback address is detected.
MD-1117	Queries for a single row group and single parquet file no longer take 6 seconds.
MD-1109	Queries that include the flatten function are no longer cancelled.
Drill-5159	Drill MergeProjectRule now operates on Calcite logical convention.
Drill-5094	Comparator guarantees transitive attribute.
Drill-5032	Drill query on hive parquet table no longer fails with OutOfMemoryError: Java heap space
Drill-4927	Drill now supports null equality joins.
Drill-4911	SimpleParallelizer now avoids plan serialization for logging purpose when debug logging is not enabled.

Known Issues and Limitations

For known issues, see the Drill 1.8.0 release notes.

For limitations, see Drill-on-YARN Limitations on page 3237

Resolved Issues

None.

Drill 1.8.0 - 1609 Release Notes

The following release notes apply to the 1.8.0-1609 version of the Apache Drill component included in the MapR Converged Data Platform.

Drill Version	1.8.0-1609	
Release Date	September 30, 2016	
MapR Version Interoperability	MapR Drill 1.8.0-1609 is certified on the MapR v5.1.0 and v5.2.0 converged data platform. See the Interoperability Matrix and Drill Support Matrix.	
Packages	**Drill (Warden)**	RedHat/CentOS: mapr-drill-1.8.0.20160912151 7-1.noarch.rpm Ubuntu: mapr-drill_1.8.0.20160912 1537_all.deb
	Drill (YARN)	RedHat/CentOS: mapr-drill-yarn-1.8.0.20160 9121517-1.noarch.rpm Ubuntu: mapr-drill-yarn_1.8.0.2016 09121537_all.deb

Known Issues

See the Drill 1.8.0 release notes.

Resolved Issues

Drill 1.8.0-1609 provides the following fixes:

- Drill 1.8.0 is available as part of the MapR Ecosystem Pack (EEP) version 1.1.0.
- You can use the MapR Installer to install or upgrade to Drill 1.8.0.
- You can now install Drill 1.8.0 on MapR v5.2.0, as well as MapR v5.1.0. When you install, you have the option of installing Drill under the MapR Warden service or under YARN.

Limitations

See Drill-on-YARN Limitations.

Drill 1.8.0 Release Notes

The following release notes apply to the 1.8.0 version of the Apache Drill component included in the MapR Converged Data Platform.

Version	1.8.0

last-updated: Apr 11, 2022

Release Date	September 12, 2016
MapR Version Interoperability	MapR Drill 1.8.0 is certified on the MapR v5.1.0 converged data platform. See Interoperability Matrix and Drill Support Matrix.
Packages	**Drill (Warden)** RedHat/CentOS: mapr-drill-1.8.0.201609121517-1.noarch.rpm Ubuntu: mapr-drill_1.8.0.201609121537_all.deb **Drill (YARN)** RedHat/CentOS: mapr-drill-yarn-1.8.0.201609121517-1.noarch.rpm Ubuntu: mapr-drill-yarn_1.8.0.201609121537_all.deb

Noteworthy New Features in the MapR Distribution of Drill

This release provides enhanced query improvements with the following bug fixes and improvements:

- Drill-on-YARN (mapr-drill-yarn package)
- Metadata cache pruning - See Partition Pruning Introduction and Optimizing Parquet Metadata Reading.
- IF EXISTS parameter with the DROP TABLE and DROP VIEW commands - See DROP TABLE and DROP VIEW.
- DESCRIBE SCHEMA command - See DESCRIBE.
- Multi-byte delimiter support - See List of Attributes and Definitions.
- New parameters for filter selectivity estimates - See System Options.

Additional bug fixes and enhancements listed in the Apache Drill 1.7.0 Release Notes and the Apache Drill 1.8.0 Release Notes.

Note: Currently, you cannot use the MapR Installer to install or upgrade to Drill 1.8. Support for this feature will be available soon.

Resolved Issues

The following table lists resolved issues in Drill 1.8.0:

Issue	Description
MD-1047	Query time is improved for simple queries on INFORMATION_SCHEMA.TABLES when there are many views.
MD-1044	Queries no longer get stuck in the CANCELLATION_REQUESTED state.
MD-1042	CTAS queries that contain an INNER JOIN and UNION no longer produce an inefficient query plan and resource errors.

MD-1030	Partition pruning now works correctly when optimization is applied to queries that contain an IN list with more multiple lines.
MD-949	Flatten on CONVERT_FROM no longer fails with a ClassCastException.
MD-904	Drill no longer throws an array index out of bound exception when reading a parquet file written by a mapreduce program.
MD-901	Drill no longer produces a Java compilation error for large queries with many CASE and CAST expressions.
MD-895	Drill queries no longer produce memory leaks.
MD-888	Queries no longer fail with a "Connection timed out" error.
MD-878	The RANK() window function no longer returns the wrong results for large datasets.
MD-746	Queries no longer hang when fragment 0 is running.
MD-727	Drill returns null values for non-existent columns in MapR Database JSON tables.
MD-603	Drill now returns the directory structure associated with a workspace.
MD-509	Drill no longer returns a NumberFormatException error when casting an empty string to int in HBase/MapR Database.

Known Issues

Issue	Description
MD-1076	Drill drops NULL fields when writing to a JSON file.
MD-1073	Queries on ORC tables no longer take an unusually long time when Drill runs for multiple days without restart.
MD-1061	Queries on Hive ORC tables fail with a "does not have access to maprfs:///" error.
MD-892	The drill-env.sh script does not set the file client read ahead throttle environment variable.
MD-833	Queries submitted from Tableau produce IndexOutOfBoundsException errors.

Limitations

See Drill-on-YARN Limitations.

Drill 1.7.0 Release Notes (Developer Preview)

The following release notes are for the 1.7.0 developer preview of the MapR distribution of Apache Drill. Release notes for prior releases are posted on the Apache Drill web site.

Version	1.7.0 (developer preview)
Release Date	July 5, 2016
MapR Version Interoperability	See Interoperability Matrices on page 5519 and Drill Support Matrix on page 5629.

last-updated: Apr 11, 2022

Noteworthy New Features

This release introduces the following enhancements:

- Improvements to MaxDir/MinDir functions. See Query Directory Functions.
- Access to Drill logs in the Web UI. See Log and Debug Introduction.
- Addition of JDBC/ODBC client IP in Drill audit logs. See Query Audit Logging.
- Monitoring via JMX. See Monitoring Metrics.
- Hive CHAR data type support. See Hive-to-Drill Data Type Mapping.
- Partition pruning enhancements.
- Improved metadata query performance on Hive tables.
- Ability to return file names as part of queries.

See Apache Drill 1.7.0 Release Notes for a complete list of JIRAs resolved in the 1.7.0 release.

Drill 1.6.0 - 1606 Release Notes

The following release notes apply to the 1.6.0 - 1606 version of the Apache Drill component included in the MapR Converged Data Platform.

Drill Version	1.6.0
Release Date	June 30, 2016
MapR Version Interoperability	See Interoperability Matrices on page 5519 and Drill Support Matrix on page 5629.
Packages	**Redhat** mapr-drill-1.6.0.201606241408-1.noarch.rpm **SuSe** mapr-drill-1.6.0.201606241408-1.noarch.rpm **Ubuntu** mapr-drill_1.6.0.201606241350_all.deb

Resolved Issues

The following table lists resolved issues in Drill 1.6.0 - 1606:

Issue	Description
DRILL-4715	Queries with multiple, non-trivial expressions no longer cause compilation errors at runtime due to memory limits imposed by the JVM.
DRILL-4694	Drill no longer adds extraneous NULL fields to JSON output files when you issue the CTAS statement to create JSON output from an existing JSON file. A new option, `store.json.writer.skip_null_fields`, controls this behavior. By default, `store.json.writer.skip_null_fields = true` and skips NULL fields. Setting `store.json.writer.skip_null_fields = false` reverts the behavior.

DRILL-4693	Columns now appear in the correct order when you issue a SELECT list expression with the COVERT_FROM function and the format type is JSON.
DRILL-4679	Queries that contain the CONVERT_FROM function with the JSON format type no longer fail with the following error when the input data has 0 rows due to a filter condition: `Error: SYSTEM ERROR: IllegalStateException: next() returned NONE without first returning OK_NEW_SCHEMA [#16, ProjectRecordBatch]Fragment 0:0`
DRILL-4676	Canceling a query during planning no longer blocks the drillbit from accepting new queries.
DRILL-4657	Running a query with the RANK function no longer produces incorrect results when a frame does not fit into two batches of data.
DRILL-4654	Drill now uses JMX (Java Management Extensions) to monitor queries at runtime and provide the following new metrics through the Drill Web Console and JConsole: • pending/running/completed queries • current memory usage (root allocator)
DRILL-4647	In instances when a drillbit is not running or query submission fails, the C++ client correctly propagates failed connection errors back to the application.
DRILL-4476	The UNION ALL operator now supports scenarios where the query expression on either side of the UNION ALL operator references an empty source.
DRILL-3714	Queries no longer remain in a CANCELLATION_REQUESTED state until the drillbits are restarted.
MD-850	Queries that contain two numbers CAST as decimals and added together no longer result in a data type mismatch.
MD-849	Queries on JSON files that contain two index levels no longer fail with an IndexOutOfBoundsException error.

MD-815	Queries on INFORMATION_SCHEMA (hive metadata) are significantly faster due to the use of a bulk load algorithm when the exec.enable_bulk_load_table_list system option is set to true:
	`ALTER SYSTEM SET`
	`` `exec.enable_bulk_load_table_list` = ``
	`true`

Drill 1.6.0 Release Notes

The following release notes are for the 1.6.0 version of the Apache Drill component included in the MapR distribution for Apache Hadoop. Release notes for prior releases are posted on the Apache Drill web site.

Version	1.6.0
Release Date	April 4, 2016

last-updated: Apr 11, 2022

MapR Version Interoperability	MapR Drill 1.6.0 is certified on the MapR v5.1.0 converged data platform. See Interoperability Matrices on page 5519 and Drill Support Matrix on page 5629.
Download Location	See Installing Drill.

Noteworthy New Features in the MapR Distribution of Drill

This release provides enhanced query improvements with the following bug fixes and improvements:

- JDK 1.8 support.
- JSON and binary table support through the MapR Database format plugin.
- Inbound impersonation which is useful in a multi-tier architecture where queries must run as the end user instead of the application user.
- More custom frames in the window function frame clause.
- Row count based pruning for Limit N queries
- Lazy reading of parquet metadata caching
- Early application of partition pruning in query planning

Additional bug fixes and enhancements listed in the Apache Drill 1.6.0 Release Notes.

Resolved Issues

The following table lists resolved issues in Drill 1.6.0:

Issue	Description
MD-803	LIMIT 0 optimization applies to the ranking and value window functions.
MD-784	LIMIT 0 optimization applies to the Hive UDF xpath_double.
MD-781	LIMIT 0 optimization applies to the TRIM function.
MD-608	Complex data functions FLATTEN and KVGEN no longer return null data when you issue queries through the ODBC driver.
MD-604	EXTRACT function with seconds no longer returns incorrect data when you issue queries through the ODBC driver.
MD-361	Drill honors the Hive skip.header.line.count and skip.footer.line.count table properties and skips header and footer lines when querying Hive tables created with these properties.

Known Issues

The following table lists known issues in Drill 1.6.0:

Issue	Description
MD-851	Pushdown is disabled for columns with TIMESTAMP values which makes these columns case-insensitive. See Limitations.
MD-845	LIMIT 0 optimization does not apply when used in a query with a FULL OUTER JOIN and the `planner.enable_mergejoin` option is set to true.

MD-817	LIMIT 0 optimization does not apply when a query contains math operations between an interval type and a constant.
MD-802	LIMIT 0 optimization applies to the AVG window function.
MD-794	LIMIT 0 optimization does not apply when the return type is VARBINARY.
MD-787	LIMIT 0 optimization does not apply to the Hive BIN function.
MD-786	LIMIT 0 does not apply to the Hive HEX function.
MD-747	SUM function queries with a decimal argument and LIMIT 0 optimization returns the wrong data types and nullability.
MD-727	If you query a column that does not exist in the records of the queried table, Drill does not return any rows when it should return rows with null values. See Limitations.

Limitations

(MD-851) Pushdown filtering is disabled on columns with TIMESTAMP values, which results in these columns being case-insensitive. For example, if a JSON table is created with the following JSON records where the “access” columns are not of a particular case:

```
{"_id":"ID1","name":"Alice","age":"25","access":"2013-10-23 11:11:11","DOB":
{"$dateDay":"1988-01-01"},"salary":"50.00"}
{"_id":"ID2","name":"Bob","age":"21","access":"2013-10-23 11:11:11","DOB":
{"$dateDay":"1988-01-01"},"salary":"150.00"}
{"_id":"ID3","name":"Frank","age":"32","ACCESS":"2013-10-23 11:11:11","dob":
{"$dateDay":"1988-01-01"},"salary":"250.00"}
```

Running a query on the “access” column returns all values because Drill does not push the filtering process down to MapR Database to get the subset of columns from records where the name is "access" in lowercase. Instead, MapR Database returns every value from each record that has a column named "access", regardless of the case. If some records include a column named "ACCESS" or "Access," these values are also returned in the result set.

```
SELECT * FROM dfs.`/tmp/data/student_db` where access = timestamp
'2013-10-23 11:11:11';
+------+----------------------+------+-------------+-----------+---------+
| _id  |        access        | age  |     dob     |   name    | salary  |
+------+----------------------+------+-------------+-----------+---------+
| ID1  | 2013-10-23 11:11:11  | 25   | 1988-01-01  | Alice  | 50.00   |
| ID2  | 2013-10-23 11:11:11  | 21   | 1980-12-12  | Bob     | 150.00  |
| ID3  | 2013-10-23 11:11:11  | 32   | 1970-10-12  | Frank      | 250.00  |
+------+----------------------+------+-------------+-----------+---------+
3 rows selected (0.223 seconds)
```

In comparison, you can see that a query on the “dob” column in the same table produces only one result because the column does not contain TIMESTAMP values and is therefore case-sensitive:

```
SELECT * FROM dfs.`/tmp/data/student_db` where dob = date '1988-01-01';
+------+----------------------+------+-------------+-------+---------+
| _id  |        ACCESS        | age  |     dob     | name  | salary  |
+------+----------------------+------+-------------+-------+---------+
| ID3  | 2013-10-23 11:11:11  | 32   | 1988-01-01  | Frank  | 250.00  |
+------+----------------------+------+-------------+-------+---------+
1 row selected (0.359 seconds)
```

Although there are three records with “dob” columns, there is only one record with the column name in lowercase.

last-updated: Apr 11, 2022

(MD-727) If you query one column that is partially or completely non-existent in the records of the queried table, Drill does not return any rows for the records when it should return the rows with null values. For example, if a JSON table is created from the following JSON records where only one record has a “time” column with a value and the other two records do not have time columns:

```
{"_id":"student1", "name":"Alice", "street":"123 Ballmer Av",
"zipcode":12345, "state":"CA"},
{"_id":"student2", "name":"Bob", "time":"2016-03-08", "street":"1 Infinite
Loop", "zipcode":12345, "state":"CA"},
{"_id":"student3", "name":"Frank", "street":"435 Walker Ct",
"zipcode":12345, "state":"CA"}
```

The query returns only one row for the record in which the time column exists with a value instead of returning three rows with two rows having null values in the column.

```
SELECT `time` FROM dfs.`/user/root/students`;
+------------+
| time       |
+------------+
| 2016-03-08|
+------------+
```

In comparison, if you query all of the columns in the table, you can see that the query returns all records and includes null values for records missing a time column.

```
SELECT * FROM dfs.`/user/root/students`;
+-----------+--------+--------+------------------+----------+-------------+
|    _id    |  name  | state  |      street      | zipcode  |    time     |
+-----------+--------+--------+------------------+----------+-------------+
| student1  | Alice  | CA     | 123 Ballmer Av   | 12345.0  | null        |
| student2  | Bob    | CA     | 1 Infinite Loop  | 12345.0  | 2016-03-08  |
| student3  | Frank  | CA     | 435 Walker Ct    | 12345.0  | null        |
+-----------+--------+--------+------------------+----------+-------------+
3 rows selected (0.314 seconds)
```

Drill 1.5.0 Release Notes (Developer Preview)

The following release notes are for the 1.5.0 developer preview of the MapR distribution of Apache Drill. Release notes for prior releases are posted on the Apache Drill web site.

Version	1.5.0 (Developer preview)
Release Date	February 23, 2016
MapR Version Interoperability	See Interoperability Matrices on page 5519 and Drill Support Matrix on page 5629

Noteworthy New Features

This release introduces the following enhancements:

- Improved Memory Allocator - Drill uses a new allocator that improves an operator’s use of direct memory and tracks the memory use more accurately. See Configuring Drill Memory.
- Configurable Caching of Hive Metadata - You can now configure the TTL for the Hive metadata client cache depending on how frequently the Hive metadata is updated. See Hive Metadata Caching.

See Apache Drill 1.5.0 Release Notes for a complete list of JIRAs resolved in the 1.5.0 release.

Drill 1.4.0 Release Notes

The following release notes are for the 1.4.0 version of the Apache Drill component included in the MapR distribution for Apache Hadoop.

Version	1.4.0
Release Date	January 12, 2016
MapR Version Interoperability	See Interoperability Matrix

Noteworthy New Features in the MapR Distribution of Drill

This release introduces a number of enhancements, including the following ones:

- Improved Tableau experience with faster Limit 0 queries (MD-602)
- Metadata (INFORMATION_SCHEMA) query performance improvements on Hive schemas/tables (MD-530, MD-504, MD-507, DRILL-4126, DRILL-4127, DRILL-4161)
- Optimized Parquet metadata caching, faster queries on a large number of files

  Metadata caching is new feature introduced in Drill 1.2 to speed up Drill queries involving large number of files. In 1.4, the cache size is considerably reduced using a variety of techniques including efficient storage of information as well reducing the redundant metadata. This significantly boosts performance of the queries. Note however that the depending on the size of data, the cache construction might take longer.
- Partition pruning improvements

  Partition pruning is significantly enhanced in 1.4. The enhancements including support for more datatypes, working better in conjunction with metadata caching as well as a moving the partition pruning evaluation from Hep to Volcano planner. (DRILL-4126, DRILL-3765)
- Improved Window functions resource usage and performance
- Updated 1.2.1 Windows ODBC driver (updates to Mac and Linux versions to follow)
- New and improved Drill JDBC driver.

  This driver is recommended by MapR for production deployments deployments and partner certifications. The new driver offers much more compatibility to the JDBC standard. This driver compliments the currently available open source version of JDBC driver.
- Stability improvements
- 90 bug fixes and additional enhancements listed in Apache Drill 1.4 Release Notes and Apache Drill 1.3 Release Notes

Known Issues

- Drill was upgraded to use a new Parquet reader in the Drill 1.3 dev preview release. Customers moving from Drill 1.2 and prior versions to 1.4 must migrate their data or regenerate their parquet in order to continue working with Drill in optimized way. For more information on how to upgrade the parquet, refer to the migration tool documentation.
- Merge join in 1.4 has known issues related to performance and data correctness in 1.4 (DRILL-3578). Hence, upgrading to 1.4 is not recommended for performant queries that use merge joins. If you do upgrade to 1.4, you must set the planner.enable_mergejoin option to false (that is, use hash join instead); otherwise, merge join queries will not run and/or produce incorrect results.
- Extract function with seconds returns incorrect data when you issue queries from the ODBC driver. (MD-604)

last-updated: Apr 11, 2022

- Complex data functions FLATTEN and KVGEN return null data when you issue queries from the ODBC driver. (MD-608)
- After upgrading to Release 1.4 or later, any custom function you wrote in Release 1.3 or earlier that is in a package that Drill scans by default will run. However, if you created *your own package* for the custom function, it will *not* run in Release 1.4 and later. You can use one of the following workarounds. The first workaround is quick and easy. The second is recommended for new UDFs you create in Release 1.4 and later.

  Quick Workaround

  1. Create a drill-module.conf file in $DRILL_HOME/conf/ if one does not already exist.
  2. Add the following code to the drill-module.conf, replacing com.yourgroupidentifier.udf with the package name(s) of your UDFs.

     ```
     drill.classpath.scanning.packages += "com.yourgroupidentifier.udf"
     ```

  3. Copy the drill-module.conf file to all Drillbits.
  4. Ensure that DRILL_HOME/conf/drill-override.conf does not contain any information regarding UDF packages.
  5. Restart Drill on all the Drillbits.

  Alternative Workaround

  1. Add the following code to the drill-module.conf in your UDF project (src/main/resources/drill-module.conf). Replace com.yourgroupidentifier.udf with the package name(s) of your UDFs.

     ```
     drill.classpath.scanning.packages += "com.yourgroupidentifier.udf"
     ```

  2. Recompile the UDFs and place the 2 jar files (source + binary) in Drill's classpath on all the Drillbits.
  3. Recompile the UDFs and place the 2 jar files (source + binary) in Drill's classpath on all the Drillbits.
  4. Restart drill on all the Drillbits.
- If you upgrade from Drill 1.2 or earlier, update Parquet data.

  Drill 1.4 is the first production ready release after the Drill upgrade to use the latest version of the Apache Parquet library, which was released in 1.3 dev preview. If you generated Parquet using Drill 1.2 and earlier, take one of the following actions:

  - Update data using the Drill migration utility.
  - Regenerate data using the new version of Drill.

  One of these actions is mandatory to continue to get accurate functionality and to optimize performance.

Resolved Issues

- Regardless of whether impersonation is enabled or not, this release caches data in the HiveMetaStore to improve performance. (DRILL-4126)

Drill 1.3.0 Release Notes (Developer Preview)

The following release notes are for the 1.3.0 developer preview of the MapR distribution of Apache Drill. Release notes for prior releases are posted on the Apache Drill web site.

Version	1.3.0 Developer Preview
Release Date	November 23, 2015
MapR Version Interoperability	See Interoperability Matrices on page 5519 and Drill Support Matrix on page 5629

Noteworthy New Features

This release introduces the following enhancements:

- Upgrade to the latest version of the Apache Parquet library (See DRILL-4070 and http://drill.apache.org/docs/partition-by-clause/)
- Performance improvements for Limit 0 and Limit 1 queries
- Hive table query planning improvements
- Partition pruning enhancements
- Support for CSV header parsing
- Improved query/operators profiles
- Capability to reset the SYSTEM and SESSION settings to their defaults
- Merge Join improvements
- Additional bug fixes and enhancements

Known Issue

The MapR 1.3 developer preview of Drill has the following known issue:

- DRILL-4070 - You must change the metadata of Parquet files generated by Drill 1.1 and 1.2 for the files to work optimally with Drill 1.3.0. See http://drill.apache.org/docs/partition-by-clause/.

Drill 1.2.0 Release Notes

Version	1.2.0
Release Date	November 7, 2015
Works with MapR version	See Drill Support Matrix on page 5629

Noteworthy New Features in MapR Distribution of Drill

This release introduces a number of enhancements, including the following ones:

- Partition pruning improvements
- Optimizes reads of Parquet-backed, Hive tables.
- Improved Hive support

  Supports Hive 1.2.1 and using multiple Hive versions (same as corresponding MapR platform version).
- Analytical/Window functions support

  Adds Lead, Lag, First_Value, Last_Value, and Ntile support and allows multiple window functions in the same query.
- DROP TABLE Command

- Parquet metadata caching

  Improves query performance on a large number of files.

- Security support for the Web Console
- Row key filter pushdown

  Improves performance when querying MapR Database and HBase.

- Improvements in the Drill JDBC driver.
- Read support for the Parquet INT96 data type support

  A new TIMESTAMP_IMPALA type used with the CONVERT_FROM function decodes a timestamp from Hive or Impala.

- Improved correlated subqueries
- Support for UNION DISTINCT syntax
- Improved LIMIT processing

Known Issues

The MapR 1.2 release of Drill does not include fixes for the following issues that were fixed in the Apache Drill 1.2 release:

Drill JIRA	Description
DRILL-1065	You cannot issue the ALTER SESSION RESET command to reset an option to its default system value.
DRILL-2361	Drill returns the following error message when column aliases for aggregate or group by fields include a dot (.): Field references must be singular names.
DRILL-2879	Drill cannot parse JSON files with the BSON OID data type, a MongoDB extension to the JSON format. Drill may also have issues parsing JSON files with the BSON Binary data type.
DRILL-3791	Testing of the JDBC storage plugin with MySQL is incomplete.
DRILL-3869	Queries submitted through the Drill Web Console with a trailing semicolon (;) fail, and Drill returns the following error message: org.apache.drill.common.exceptions.UserRemoteException: PARSE ERROR: Encountered ";" at line 1, column 42. Was expecting one of: "OFFSET" ... "FETCH" ...
DRILL-3888	The test JAR file configurations exist in each module's POM file instead of the root POM file.
DRILL-3916	Drill-bits cannot load the JDBC storage plugin when starting because the plugin does not exist.

Drill 1.1.0 Release Notes

This version of Drill is no longer supported.

Flume Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The release notes for Flume contain notes specific to MapR only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Flume 1.9.0.0 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Flume 1.9.0.0.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following release notes for the Flume 1.9.0.0 component are included in the MapR distribution for Apache Hadoop:

Flume 1.9.0.200-2110 (EEP 8.0.0) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Flume. You may also be interested in the Apache Flume 1.9.0 changelog or the Apache Flume homepage.

For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.9.0.200
Release Date	October 2021
HPE Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/flume
GitHub Release Tag	1.9.0.200-eep-800
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP (MEP) and OS to view the list of package names

New in This Release

No new features were introduced in this release.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	Fix Number and Description
cb11414	2021-09-07	MAPR-FLUME-82 Update maven artifact version strings to replace 'mep/mapr' with 'eep'
420b96f	2021-09-03	MAPR-FLUME-81 mapr-security-web jar is taken from the cluster

last-updated: Apr 11, 2022

eb25c7b	2021-09-03	MAPR- FLUME-79 kafka-eventstreams*.jar was added to classpath
d9a47e3	2021-07-23	MAPR- FLUME-71 Kafka version was updated to 2.6.1.0-mapr

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Flume 1.9.0.100-2101 (EEP 7.0.1) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Flume. You may also be interested in the Apache Flume 1.9.0 changelog or the Apache Flume homepage.

Version	1.9.0.100
Release Date	January 2021
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/flume
GitHub Release Tag	1.9.0.100-mapr-701
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Flume 1.9.0.100-2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
43bfe16	2020-11-12	MAPR-FLUME-70 Thrift version was updated to v0.13.0

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Flume 1.9.0.0-2009 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Flume 1.9.0.0 for EEP 7.0.0.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You may also be interested in the Apache Flume 1.9.0 changelog or the Apache Flume homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.0
Release Date	September 2020
MapR Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/flume
GitHub Release Tag	1.9.0.0-mapr-700
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
c9df319	2019-03-15	MAPR-FLUME-27: Flume uses 2.1.1.0-mapr version of Kafka
5343bb7	2019-03-20	MAPR-FLUME-26: KafkaChannel does not avoid duplicates while consumer rebalancing processing. KafkaChannel was extended to consume from multiple topics(listed explicitly or via specifying regex pattern). Sink and Source runner stop were fixed.
5fb966b	2019-06-05	MAPR-FLUME-43: Flume 1.9 uses hadoop 2.7.4.0-mapr-1908
6e39679	2020-01-21	MAPR-FLUME-44: Protobuf version was updated to 3.11.1 for Flume 1.9.
dc9d3dd	2020-03-31	MAPR-FLUME-1: Flume uses mapr version of mapr specific components.
27e0ce2	2020-04-07	MAPR-FLUME-50: Mapr-flume is built using java 11.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
f71421a	2020-04-16	MAPR-FLUME-55: XercesImpl version was upgraded to 2.12.0.
3af3507	2020-04-30	MAPR-FLUME-1 hive version was changed to 2.3.7-mapr
623e77a	2020-04-30	MAPR-FLUME-61: Zookeeper version was changed to 3.5.6-mapr.
7056d16	2020-05-05	MAPR-FLUME-58: libthrift version was upgraded to 0.12.0. Derby version was upgraded to 10.14.2.0.
f06f6b9	2020-05-14	MAPR-FLUME-53: jackson-databind version was upgraded to 2.10.0. Avro version was updated to 1.9.2. jackson-mapr-asl was excluded.
46a7704	2020-07-15	MAPR-FLUME-66: guava version was upgraded to 28.2-jre
2da21f7	2020-08-11	MAPR-FLUME-68: HBase version was changed to 1.4.12.0-mapr

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Flume 1.8.0 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following release notes for the Flume 1.8.0 component are included in the MapR distribution for Apache Hadoop.

Flume 1.8.0-2201 (EEP 6.3.6) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Flume. You may also be interested in the Apache Flume 1.8.0 changelog or the Apache Flume homepage.

Version	1.8.0
Release Date	January 2022
MapR Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/flume
GitHub Release Tag	1.8.0-mapr-2201
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Flume 1.8.0-2201 is a defect-repair release.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
5d7b40a	2022-01-25	MAPR-FLUME-85 log4j v1 was updated to the 1.3.1-mapr
e9b7914	2021-12-16	MAPR-FLUME-84 Log4j version was updated to 1.3.0-mapr
0229b54	2021-01-23	MAPR-FLUME-77 Upgrade version of rat-plugin

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Flume 1.8.0-2104 (EEP 6.3.4) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Flume. You may also be interested in the Apache Flume 1.8.0 changelog or the Apache Flume homepage.

Version	1.8.0
Release Date	April 2021
MapR Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/flume
GitHub Release Tag	1.8.0-mapr-2104
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Flume 1.8.0-2104 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

last-updated: Apr 11, 2022

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
e2c9356	2021-03-16	MAPR-FLUME-75 Hadoop jars were updated to 2.7.0-mapr-1808 version
35cee9b	2021-03-16	MAPR- FLUME-76 xml-apis version should be 1.4.01

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Flume 1.8.0-2101 (EEP 6.3.2) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Flume. You may also be interested in the Apache Flume 1.8.0 changelog or the Apache Flume homepage.

Version	1.8.0
Release Date	January 2021
MapR Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/flume
GitHub Release Tag	1.8.0-mapr-2101
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Flume 1.8.0-2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
2951e22	2020-10-20	MAPR-FLUME-69 CVE fixes were moved from Mapr Flume 1.9 to Mapr Flume 1.8
a0bd4c2	2020-11-12	MAPR-FLUME-70 Thrift version was updated to v0.13.0

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Flume 1.8.0-1904 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Flume 1.8.0 changelog or the Apache Flume homepage .

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.8.0-1904
Release Date	April 2019
MapR Version Interoperability	EEP Components and OS Support
Source on GitHub	MEP 5.x: https://github.com/mapr/flume/tree/1.8.0-mapr-mep-5.x-1904 MEP 6.x: https://github.com/mapr/flume/tree/1.8.0-mapr-mep-6.x-1904
GitHub Release Tag	1.8.0-mapr-1904
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

None.

Fixes

This MapR release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
5346fa1	2019-03-09	FLUME-33: FLUME agent stops abruptly when using HDFS SINK

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Flume 1.8.0-1901 Release Notes

last-updated: Apr 11, 2022

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Flume 1.8.0 changelog or the Apache Flume homepage .

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.8.0-1901
Release Date	February 2019
MapR Version Interoperability	EEP Components and OS Support
Source on GitHub	https://github.com/mapr/flume/tree/1.8.0-mapr-1901
GitHub Release Tag	1.8.0-mapr-1901
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Added encryption by default to secure clusters for Avro source and sink.

Fixes

This MapR release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
728b520	2018-10-19	MAPR-FLUME-14: Flume Avro source uses SSL by default for secure clusters

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Flume 1.8.0-1808 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Flume 1.8.0 changelog or the Apache Flume homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.8.0-1808

Release Date	September 2018
MapR Version Interoperability	EEP Components and OS Support
Source on GitHub	https://github.com/mapr/flume/tree/1.8.0-mapr-1808
GitHub Release Tag	1.8.0-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

None.

Fixes

This MapR release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
96334a1	2018-07-06	MAPR-FLUME-15: Flume Thrift source should use SSL by default for secure clusters
567f64d	2018-07-10	MAPR-FLUME-16: Updated kafka client version

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Flume 1.8.0-1803 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Flume 1.8.0 changelog or the Apache Flume homepage

Version	1.8.0-1803
Release Date	March 2018
MapR Version Interoperability	EEP Components and OS Support
Source on GitHub	https://github.com/mapr/flume
GitHub Release Tag	1.8.0-mapr-1803
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Flume 1.8.0-1803 introduces the following enhancement or MapR platform-specific behavior changes:

- Support for MapR-Streams v1.0.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
ae218cb 779d6f4	2018-01-31 2018-02-26	FLUME-3197: Upgrade Kafka client to 1.0.x

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None

Resolved Issues

- None

Flume 1.7.0 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following release notes for the Flume 1.7.0 component are included in the MapR distribution for Apache Hadoop.

Flume 1.7.0-1703 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Flume 1.7.0 changelog or the Apache Flume homepage

Version	1.7.0-1703
Release Date	April 2017
MapR Version Interoperability	Pre-MapR 5.2: Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 MapR 5.2 and later: EEP Components and OS Support
Source on GitHub	https://github.com/mapr/flume
GitHub Release Tag	1.7.0-mapr-1703
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Flume 1.7.0-1703 introduces the following enhancement or MapR platform-specific behavior changes:

- Support for authentication through MapR-SASL tickets.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
a374d23	2017.02.06	MAPR-26017 Option kafka.bootstrap.servers is required for reading MapR Streams
563106b	2016.12.19	MAPR-24511 Implement MAPR-SASL security in Flume
abca456	2016.12.13	MAPR-25508 Flume got java.io.IOException: No FileSystem

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None

Resolved Issues

- None

Flume 1.6.0 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following release notes for the Flume 1.6.0 component are included in the MapR distribution for Apache Hadoop.

Flume 1.6.0-1602 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Flume Version	1.6.0
Release Date	February 29, 2016
Source on GitHub	https://github.com/mapr/flume/tree/1.6.0-mapr-1602
MapR Version Interoperability	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release if you are running 5.0 or 5.1 and install Flume from the repository, or if you are running 5.2 and install EEP 1.0 or 1.1.0: • mapr-flume-1.6.0.201602111316-1.noarch.rpm • mapr-flume_1.6.0.201602111316_all.deb If you are running 5.2 and install EEP 1.1.1 or 2.0, see Package Names for MapR Ecosystem Packs (EEPs).

last-updated: Apr 11, 2022

New in this Release

This release of Apache Flume includes the following behavior change that is specific to MapR:

Support for MapR Event Store For Apache Kafka through the Kafka 0.9 API.	To read MapR Event Store For Apache Kafka topics, you can use a Kafka source or a Kafka channel. To write to a MapR Event Store For Apache Kafka topic, you can use a Kafka channel or a Kafka sink.

For details on the features provided in open source version of Flume 1.6, see the Apache Flume release notes.

Fixes

GitHub Commit	Date (YYYY-MM-DD)	Comment
3e11b9a	2015-12-25	MAPR-21957: Flume now includes support for MapR Event Store For Apache Kafka through the Kafka 0.9 API.
61f6ac3	2016-01-14	MAPR-22162: `flume-kafka-channel` now has a junit dependency.
0536731	2016-01-14	MAPR-22168: Flume channel now catches `UnknownTopicOrPartitionException` for the committed() method.
e71cfff	2016-01-19	MAPR-22205: Flume no longer throws `NoSuchMethodError` when an agent includes an Hbase 1.1 sink and source.

Flume 1.6.0-1509 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Flume Version	1.6.0
Release Date	Oct 5, 2015
Source on GitHub	https://github.com/mapr/asynchbase
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-flume-1.6.0.201509301714-1.noarch.rpm • mapr-flume_1.6.0.201509301714_all.deb

New in this release

For details on the features provided in Flume 1.6, see the Apache Flume release notes.

Flume 1.5.0 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following release notes for the Flume 1.5.0 component are included in the MapR Converged Data Platform:

Flume 1.5.0-1503 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You may also be interested in the Apache Flume 1.5.0 changelog or the Apache Flume homepage.

Flume Version	1.5.0
Release Date	March 27, 2015
MapR Version Compatibility	See the Ecosystem Support Matrix.
Source on GitHub	https://github.com/mapr/flume
GitHub Release Tag	1.5.0-mapr-1503
Maven Artifacts	https://repository.mapr.com/maven/

Fixes

This release from MapR includes the following fixes on the base Apache release. These fixes were back-ported from Flume Version 1.5.2. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
1d7ee18	2014-11-12	FLUME-2549: Enable SSLv2Hello for HttpSource.
2ed1a43	2014-11-12	FLUME-2548: Enable SSLv2Hello for Avro Source and NettyAvroRpcClient.
3d78e0b	2014-11-12	FLUME-2551: Add dev-support directory to source tarball.
c82f7b5	2014-11-05	FLUME-2533: HTTPS tests fail on Java 6.
ccadd4d	2014-08-05	FLUME-2441: HTTP Source Unit tests failed with IBM JDK 7.
429cf96	2014-10-22	FLUME-2511: Allow configuration of enabled protocols in Avro source and RpcClient.
b33bd46	2014-10-27	FLUME-2520: HTTP Source should be able to block a prefixed set of protocols.

Updating Flume for Compatibility with HBase 0.98

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Flume 1.5 is compatible with HBase 0.94 and 0.98. If you are using MapR Version 4.0.x and have upgraded to HBase 0.98, you need to update the Flume 1.5 jar files as follows:

1. Go to the directory where the Flume scripts are stored:

```
cd /opt/mapr/flume/flume-<version>/bin/
```


last-updated: Apr 11, 2022

2. Execute the update script:

```
bash flume-jars.sh
```

During the script execution, you should see the following log messages:

```
POST_YARN=1, HBASE_VERSION=0.98.4 : installing flume*-hbase.98-h2 jars
```

You can now use Flume 1.5 with HBase 0.98.

Flume 1.5.0-1501 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You may also be interested in the Apache Flume 1.5.0 changelog or the Apache Flume homepage.

Flume Version	1.5.0
Release Date	January 21, 2015
MapR Version Compatibility	See the Ecosystem Support Matrix.
Source on GitHub	https://github.com/mapr/flume
GitHub Release Tag	1.5.0-mapr-1501
Maven Artifacts	https://repository.mapr.com/maven/

Fixes

Commit	Date (YYYY-MM-DD)	Comment
7096e0f	2014-12-11	Added tmp extensions to prevent the JVM from using jar files from other versions of HBase.
e544e94	2014-12-04	Added a script to configure Flume jars after an upgrade to HBase 0.98.
334a2bc	2014-12-04	mapr-flume-1.5.0 was built with support for HBase 0.94 and 0.98.

Updating Flume for Compatibility with HBase 0.98

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Flume 1.5 is compatible with HBase 0.94 and 0.98. If you are using MapR Version 4.0.x and have upgraded to HBase 0.98, you need to update the Flume 1.5 jar files as follows:

1. Go to the directory where the Flume scripts are stored:

```
cd /opt/mapr/flume/flume-<version>/bin/
```

2. Execute the update script:

```
bash flume-jars.sh
```

During the script execution, you should see the following log messages:

```
POST_YARN=1, HBASE_VERSION=0.98.4 : installing flume*-hbase.98-h2 jars
```

You can now use Flume 1.5 with HBase 0.98.

Flume 1.5.0-1408 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You may also be interested in the Apache Flume 1.5.0 changelog or the Apache Flume homepage.

Flume Version	1.5.0
Release Date	August 30, 2014
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/flume.git
GitHub Release Tag	1.5.0-mapr-1408
Maven Artifacts	https://repository.mapr.com/maven/

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
e8792b4	2014-08-27	[Bug 14959] Fixed java.lang.NoSuchMethodException exception post Flume 1.5 upgrade.
545068f	2014-08-27	[Bug 14838] The bc command is removed from install and uninstall scripts, since it is not part of a standard Linux installation.

Flume 1.5.0-1407 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You may also be interested in the Apache Flume 1.5.0 changelog or the Apache Flume homepage.

Flume Version	1.5.0
Release Date	August 4, 2014
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/flume
GitHub Release Tag	mapr-1.5.0-release1407
Maven Artifacts	https://repository.mapr.com/maven/

last-updated: Apr 11, 2022

New in This Release

This is the initial release of Flume 1.5.0 on the MapR Distribution for Hadoop. This release supports MapReduce version 1 and MapReduce verion 2 and includes all patches made to Flume version 1.4.0.

Hadoop Release Notes

The release notes for the Hadoop and YARN components included in the MapR Data Platform contain notes specific to data-fabric only.

Hadoop 2.7.6.200 - 2201 (EEP 8.1.0) Release Notes

The notes below relate to the MapR Data Platform distribution of Apache Hadoop.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	2.7.6.200
Release Date	January 2022
Version Interoperability	See EEP Components and OS Support on page 5536.
GitHub Source	https://github.com/mapr/hadoop-common/
GitHub Release Tag	2.7.6.200-eep-810
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

Hadoop 2.7.6.100 - 2110 introduces the following enhancements:

- Added FIPS support.
- Added sharding for RM volume across multiple volumes to support stability.

Fixes

This HPE release includes the following fixes on the base Apache release:

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description
974986fae21	2022-01-25	MAPRHADOOP-213: updated log4j to 1.3.1-mapr version
65cf470fefa	2022-01-21	MAPRYARN-349: verify ownership/permissions for cluster directory when RM/HS starts
e587b904ab1	2022-01-13	MAPRHADOOP-211: handle case when volume mapr.resourcemanager.volume is already mounted with incorrect mount point
7ac1f8da988	2021-12-14	MAPRYARN-348: Updated Netty4 to 4.1.71.Final version
41bd3d1a720	2021-12-09	MAPRHADOOP-209: Fixed backward compatibility for client
ce8692d77c4	2021-12-06	MAPRYARN-337 - Create Spark spill local volumes by default
0cc00d6dd33	2021-12-01	MAPRYARN-345: Added FIPS support to AWS client

513f2c49362	2021-11-24	MAPRYARN-312: fix MapRTicketLocalizer to be able to operate with nm-local-dirs as with list of directories
a48cf251a22	2021-11-18	MAPRYARN-342: Removed Netty3 dependency
d25811160d6	2021-11-18	Backport HADOOP-11245. Update NFS gateway to use Netty4
ed9b5fa1f04 2b0e116df0b	2021-11-18	Backport HDFS-5570. Deprecate hftp / hsftp and replace them with webhdfs / swebhdfs. Contributed by Haohui Mai
c6245b1186e	2021-11-18	MAPRYARN-343: updated gson to 2.8.9 version
7c9a5632a37	2021-11-18	MAPRHADOOP-207: Jetty updated to 9.4.44.v20210927 version
330748ae5f0	2021-11-02	Backport YARN-4925. ContainerRequest in AMRMClient, application should be able to specify nodes/racks together with nodeLabelExpression. Contributed by Bibin A Chundatt
ddad84644f1	2021-10-25	MAPRYARN-333: Fixed logging ContainerLocalizer to local file system while yarn.use-central-logging-for-mapreduce-only is enabled
336907aa52c a6d6f29f6c1 0e5d6028056	2021-10-22	MAPRYARN-326: Added SCRAM-SASL to Hadoop
782d4b83149	2021-10-13	MAPRYARN-317: add ability to enable case in-sensitive groups/usernames in fair-scheduler file
59737fbb1dd	2021-10-04	Backport YARN-7157. Add admin configuration to filter per-user's apps in secure cluster. Contributed by Sunil G.
d1c8ea0dd4a	2021-09-03	MAPRYARN-296: YARN RM UI or commands do not show allocated containers for finished applications
1579432cb0d	2021-09-03	Backport YARN-4417. Make RM and Timeline-server REST APIs more consistent. Contributed by Wangda Tan
db2081bd96d	2021-09-03	Backport YARN-5440. Use AHSClient in YarnClient when TimelineServer is running (Xuan Gong via gtcarrera9)
078ab8abb3f	2021-09-03	Backport YARN-5767. Fix the order that resources are cleaned up from the local Public/Private caches. Contributed by Chris Trezzo
085526639b5	2021-08-20	MAPRHADOOP-195: Create symlinks to verify script for Hadoop services
56a48da51ec	2021-08-17	MAPRYARN-274: Shard RM volume across multiple volumes to support scalability
f45e2898856 42a6a35138c	2021-08-12	MAPRHADOOP-187: Add property to preserve link for "hadoop cp" and distCp operations
07887e3f7c5 8c752b00fcc	2021-07-21	MAPRHADOOP-185: Use alternate java.security for yarn scripts

Known Issues and Limitations

- None.

last-updated: Apr 11, 2022

Resolved Issues

- None.

Hadoop 2.7.6.100 - 2110 (EEP 8.0.0) Release Notes

The notes below relate to the MapR Data Platform distribution of Apache Hadoop.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	2.7.6.100
Release Date	October 2021
Version Interoperability	See EEP Components and OS Support on page 5536.
GitHub Source	Not applicable
GitHub Release Tag	2.7.6.100-eep-800
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

Hadoop 2.7.6.100 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

- All fixes from Apache Hadoop 2.7.6 have been backported.
- Support for file exclusion list in DistCp.
- Added a `yarn top` tool.
- Support for BCFKS keystores for Hadoop Credential Provider.

Fixes

This HPE release includes the following fixes on the base Apache release:

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description
4d102084877	2021-09-13	MAPRHADOOP-200: Load ssl configuration after core-site.
48960d6b6e5	2021-09-01	MAPRYARN-323-325: Added BC provider to Hadoop services
b808c225390	2021-09-01	Backport HADOOP-17699. Remove hardcoded SunX509 usage from SSLFactory. (#3016)
ca635e0db98	2021-09-01	MAPRHADOOP-192: Exclude bc-fips jars from Hadoop build
0dc49e773b8	2021-09-01	Backport HADOOP-13011 - Clearly Document the Password Details for Keystore-based Credential Providers
5400bc53028	2021-09-01	Backport HADOOP-17284. Support BCFKS keystores for Hadoop Credential Provider. (#2334)
c4ef4f0f162	2021-09-01	Backport HADOOP-12942. hadoop credential commands non-obviously use password of "none" (Mike Yoder via lmccay)
b93bcdf2592	2021-08-18	MAPRHADOOP-194: Update commons-compress to 1.21

4f1c4ec9828	2021-08-12	MAPRHADOOP-190: update htrace to 4.2.0-mapr-incubating
dc9fce5dd8f	2021-08-04	MAPRMR-21: ControlledJob#toString failed with NPE when job status is not successfully updated
f267f098be2	2021-08-04	Backport MAPREDUCE-6762. ControlledJob#toString failed with NPE when job status is not successfully updated (Weiwei Yang via Varun Saxena)
e0732a00771	2021-07-28	MAPRHADOOP-186: Update Jetty to 9.4.43.v20210629
a30951764be	2021-07-27	MAPRMR-20: add retry support for all Job actions
a42503a9153	2021-07-26	Update gradle-wrapper.properties
1d04de14c09	2021-07-26	Backport MAPRYARN-321: Upgraded Avro version to 1.10.1
2b2518f33a8	2021-07-23	Backport HADOOP-17341. Upgrade commons-codec to 1.15 (#2428)
b6988af1234	2021-07-20	Backport HADOOP-10075. addendum to fix compilation on Windows
7d0183ee3bd	2021-06-21	MAPRYARN-320: Kill child process from container-executor
ad8e8d08c0b	2021-06-18	Backport YARN-10490. yarn top command not quitting completely with ctrl+c. Contributed by Agshin Kazimli
1c737ee2f90	2021-06-18	MAPRYARN-316: Added disks to "yarn top" command
a35adb43f87	2021-06-18	Backport YARN-3348. Add a 'yarn top' tool to help understand cluster usage. Contributed by Varun Vasudev
9f5418a1de5	2021-06-07	Backport HADOOP-17602. Upgrade JUnit to 4.13.1. Contributed by Ahmed Hussein.
f3b647eea98	2021-06-04	Fixed LocatedFileStatusFetcher for work with symlinks
9b16bf0084b	2021-06-03	MAPRHADOOP-177: Fixed mkdir command for non existing parent dir
e96a8d7e003	2021-06-02	MAPRHADOOP-174: Fixed issue with copying multiple files to directory
740035b02f1	2021-05-31	Backport HADOOP-16245. Restrict the effect of LdapGroupsMapping SSL configurations to avoid interfering with other SSL connections. Contributed by Erik Krogen.
27e38cbbb28	2021-05-31	Backport HADOOP-12862. LDAP Group Mapping over SSL can not specify trust store. Contributed by Wei-Chiu Chuang and Konstantin Shvachko.
290a908be21	2021-05-31	Backport HADOOP-15345. Backport HADOOP-12185 to branch-2.7: NetworkTopology is not efficient adding/getting/removing nodes. Contributed by He Xiaoqiao.
532934140ad	2021-05-31	Backport HDFS-13195. DataNode conf page cannot display the current value after reconfig. Contributed by maobaolong.
f9918ab5550	2021-05-31	Backport HDFS-12884. BlockUnderConstructionFeature.truncateBlock should be of type BlockInfo. Contributed by chencan.
1c3f997eb9b	2021-05-31	Backport HADOOP-13105. Support timeouts in LDAP queries in LdapGroupsMapping. Contributed by Mingliang Liu.
209c93fbe66	2021-05-31	Backport HADOOP-12001. Fixed LdapGroupsMapping to include configurable Posix UID and GID attributes during the search. Contributed by Patrick White.

last-updated: Apr 11, 2022

fdb6bf9b675	2021-05-31	Backport HADOOP-15279. increase maven heap size recommendations
444cef90d29	2021-05-31	Backport HADOOP-15283. Upgrade from findbugs 3.0.1 to spotbugs 3.1.2 in branch-2 to fix docker image build.
a156fd13c95	2021-05-31	Backport HDFS-4210. Throw helpful exception when DNS entry for JournalNode cannot be resolved. Contributed by Charles Lamb and John Zhuge.
79158f841e8	2021-05-31	Backport HADOOP-15206. BZip2 drops and duplicates records when input split size is small. Contributed by Aki Tanaka
93547c2cb3b	2021-05-31	Backport HADOOP-12568. Update core-default.xml to describe posixGroups support. Contributed by Wei-Chiu Chuang.
28283d0740c	2021-05-31	Backport HADOOP-12793. Write a new group mapping service guide (Wei-Chiu Chuang via iwasakims)
a6aef9fa3fa	2021-05-31	Backport HADOOP-9477. Add posixGroups support for LDAP groups mapping service. (Dapeng Sun via Yongjun Zhang)
2b0f6773337	2021-05-31	Backport MAPREDUCE-7048. Uber AM can crash due to unknown task in statusUpdate. Contr ibuted by Peter Bacsko
ac06e483c05	2021-05-31	Backport HDFS-10453. ReplicationMonitor thread could stuck for long time due to the race between replication and delete of same file in a large cluster.. Contributed by He Xiaoqiao.
384340b76ed	2021-05-31	Backport HDFS-7959. WebHdfs logging is missing on Datanode (Kihwal Lee via sjlee)
953f75abed0	2021-05-31	Backport HDFS-13120. Snapshot diff could be corrupted after concat. Contributed by Xiaoyu Yao.
8bc1c2ecbaa	2021-05-31	Backport HADOOP-15212. Add independent secret manager method for logging expired tokens. Contributed by Daryn Sharp.
f73da507697	2021-05-31	Backport MAPREDUCE-7020. Task timeout in uber mode can crash AM. Contributed by Peter Bacsko
ef3f06d320b	2021-05-31	Backport HDFS-12371. BlockVerificationFailures and BlocksVerified show up as 0 in Datanode JMX. Contributed by Hanisha Koneru.
7570e0e491d	2021-05-31	Backport HADOOP-13508. FsPermission string constructor does not recognize sticky bit. Contributed by Atul Sikaria.
27e4c5fe92e	2021-05-31	Backport HDFS-11003. Expose XmitsInProgress through DataNodeMXBean. Contributed By Brahma Reddy Battula
a537ad3ea2d	2021-05-31	Backport HADOOP-13263. Reload cached groups in background after expiry. (Contributed bt Stephen O'Donnell)
358d2fd6795	2021-05-31	Backport HADOOP-14246. Authentication Tokens should use SecureRandom instead of Random and 256 bit secrets (Conttributed by Robert Kanter via Daniel Templeton)
fb29d47034b	2021-05-31	Backport HDFS-11384. Balancer disperses getBlocks calls to avoid NameNode's rpc queue saturation. Contributed by Konstantin V Shvachko.
442c573b4e6	2021-05-31	Backport MAPREDUCE-7028. Concurrent task progress updates causing NPE in Application Master. Contributed by Gergo Repas
ead4809bbc1	2021-05-31	Backport HDFS-12881. Output streams closed with IOUtils suppressing write errors. Contributed by Ajay Kumar

f7842437eff	2021-05-31	Backport MAPREDUCE-5124. AM lacks flow control for task events. Contributed by Peter Bacsko
9316e6866df	2021-05-31	Backport YARN-4167. NPE on RMActiveServices#serviceStop when store is null. (Bibin A Chundatt via rohithsharmaks) Backport YARN-6633 by Inigo Goiri.
573618f058e	2021-05-31	Backport YARN-3425. NPE from RMNodeLabelsManager.serviceStop when NodeLabelsManager.serviceInit failed. (Bibin A Chundatt via wangda) Backport YARN-6632 by Inigo Goiri.
38565378f4f	2021-05-31	Backport YARN-7661. NodeManager metrics return wrong value after update node resource. Contributed by Yang Wang
cb027e006c4	2021-05-28	Backport MAPRHADOOP-174: Added wildcard usage for path with symlinks
ffc31a97594	2021-05-26	Backport YARN-4348. ZKRMStateStore.syncInternal shouldn't wait for sync completion for avoiding blocking ZK's event thread. (ozawa)
226dfdeca76	2021-05-26	Backport YARN-3798. ZKRMStateStore shouldn't create new session without occurrance of SESSIONEXPIED. (ozawa and Varun Saxena)
4f4bf8a1446	2021-05-20	MAPRHADOOP-173: Moving cluster from secure to unsecure breaks yarn-site.xml
6a83c9d9918	2021-05-19	COMSECURE-384: Add Bouncy Castle JARs to Hadoop class path - unitTested
fa833c7d5bb	2021-05-17	MAPRHADOOP-171: Distcp to S3A does not work
ff9e411bb5f	2021-05-13	MAPRHADOOP-172: configure ssl-client/server.xml as part of hadoop-util configuration
da5ef438a37	2021-05-11	Backport HADOOP-15970. Upgrade plexus-utils from 2.0.5 to 3.1.0.
4e64e475366	2021-05-06	MAPRYARN-299: change service name identification approach
6d453dc22df	2021-04-28	MAPRHADOOP-169: Updated JQuery to 3.5.1 for SLS
2ac48dc37c5	2021-04-28	Backport HADOOP-17302. Upgrade to jQuery 3.5.1 in hadoop-sls. (#2379)
32e551d6ae2	2021-04-28	Backport HADOOP-14040. Use shaded aws-sdk uber-JAR 1.11.86. Contributed by Steve Loughran and Sean Mackrory
8394c1d02c1	2021-04-28	Backport HADOOP-12537 S3A to support Amazon STS temporary credentials. Contributed by Sean Mackrory.
1c36296b9e8	2021-04-28	Backport HADOOP-12723 S3A: Add ability to plug in any AWSCredentialsProvider. Contributed by Steven Wong.
eed76986ecd	2021-04-26	Backport HADOOP-1540. Support file exclusion list in distcp. Contributed by Rich Haase
912a88cdc84	2021-04-20	MAPRHADOOP-167: jackson-mapper-asl updated to 1.9.13-atlassian-5

Known Issues and Limitations

- None.

Resolved Issues

- None.

last-updated: Apr 11, 2022

Hadoop 2.7.5.0 - 2104 (EEP 7.1.0) Release Notes

The notes below relate to the MapR Data Platform distribution of Apache Hadoop.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	2.7.5.0
Release Date	April 2021
Version Interoperability	See EEP Components and OS Support on page 5536.
GitHub Source	https://github.com/mapr/hadoop-common/
GitHub Release Tag	2.7.5.0-mapr-mep-710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

Hadoop 2.7.5.0 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- All the fixes from Apache Hadoop 2.7.5 have been backported.
- Symbolic links are supported for all Hadoop operations.
- Hadoop Key Management Server (hadoop-kms) has been removed.
- Log4j APIs have moved to slf4j.
- Jetty version updated to 9.4.39.v2021032.
- Service verifier.

Fixes

This data-fabric release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
48b8b44e	2021-04-13	MAPRHADOOP-165: Jetty updated to 9.4.39.v20210325 due CVE-2021-28165
62625863	2021-04-06	MAPRHADOOP-163:fix mapred-site.xml HS_IP new value change
7df21655	2021-03-25	MAPRHADOOP-161: ssl-client/server.xml should be updated after Hadoop upgrade
2c4da9f4	2021-03-22	MAPRYARN-307: Job history server does not delete logs on new nodes with central logging
4406c09b	2021-03-17	MAPRYARN-299: Add service verifier to hadoop packages that contains a startable servicee
961269d8	2021-03-15	MAPRHADOOP-159: Fixed mfs and conf commands for Windows client
6b28f579	2021-03-09	MAPRHADOOP-155: CVE-2016-2402 vulnerability in okhttp

ceb85f35	2021-03-05	MAPRYARN-306: fix ConcurrentModificationException
705769a6	2021-03-03	MAPRHADOOP-152: Updated Jetty to 9.4.38.v20210224 due CVEs
d72447ab	2021-03-01	Backport HADOOP-17223. Update org.apache.httpcomponents:httpclient to 4.5.13 and httpcore to 4.4.13
258214e1	2021-02-26	Backport YARN-7590. Improve container-executor validation check. Contributed by Eric Yang.
e189bdff	2021-02-26	MAPRYARN-294: Added missed API for ContainerLocalizer logs
e0e5e7ff	2021-02-26	Backport YARN-8039. Clean up log dir configuration in TestLinuxContainerExecutorWithMocks.testStartLocalizer. Contributed by Miklos Szegedi.
1cac41e8	2021-02-26	Backport YARN-5277. When localizers fail due to resource timestamps being out, provide more diagnostics. Contributed by Siddharth Ahuja
9df6eea0	2021-02-26	Backport YARN-7705. Create the container log directory with correct sticky bit in C code. Contributed by Yufei Gu.
989909e7	2021-02-26	YARN-7363. ContainerLocalizer don't have a valid log4j config in case of Linux container executor. (Contributed by Yufei Gu)
1ace3db4	2021-02-26	Backport YARN-7261. Add debug message for better download latency monitoring. (Yufei Gu)
7125273e	2021-02-24	MAPRHADOOP-147: Fixed __HS_IP__ address at mapred-site.xml after Hadoop update
7430b2fc	2021-02-23	MAPRHADOOP-146: Added information about ACE to cp copy usages
1f3ff1a9	2021-02-22	MAPRYARN-125: fix typo in deprecated key replacement
c2be584e	2021-02-16	MAPRYARN-283: Implement maxContainer queue element at the queue level
8c75d4dd	2021-02-12	Backported YARN-6752. Display reserved resources in web UI per application
eaf1f314	2021-02-10	MAPRYARN-292: Nodemanager does not delete the appcache application directories
c5ede6c4	2021-02-10	MAPRHADOOP-120: Fixed symlink path for LocatedFileStatus
897dca05	2021-02-08	MAPRHADOOP-143: CVE-2020-17527 Tomcat 9.0.39 vulnerability. Updated to 9.0.43
98fe6480	2021-02-03	MAPRYARN-295: NodeManager shutsdown with NPE in AsyncDispatcher Thread
5cf60bb9	2021-02-02	MAPRHADOOP-142: MapTask checks if output dir exist in MFS ignoring the path schema
21585148	2021-01-06	MAPRHADOOP-88: Remove hadoop-kms
e5a0ca54	2020-12-29	MAPRYARN-288: Scheduler queue UI doesn't work
a1deb43c	2020-12-24	MAPRHADOOP-135: Moved ShuffleHandler to Netty 4
fa85c3c7	2020-12-07	MAPRHADOOP-121: Added symlink support for LS and Delete operations
77b364a2	2020-12-07	MAPRYARN-280: Support for symlinks as input/output for YARN application

last-updated: Apr 11, 2022

7dfc54f2	2020-12-07	MAPRHADOOP-121: Support symlinks for CLI
23ed8d45	2020-11-27	MAPRHADOOP-116 Move log4j APIs over to slf4j
df184cf8	2020-11-02	MAPRHADOOP-79: Update jackson to 2.11.1
a3119a41	2020-10-08	Backport some changes from hadoop-13597
641791bf	2020-10-06	Backport MFS-11328: Link activation jar under lib to find the class DataSource
3960fa46	2020-10-05	MAPRHADOOP-114: Update jetty dependency to version 9

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Hadoop 2.7.4.100 - 2101 (EEP 7.0.1) Release Notes

The notes below relate to the MapR Data Platform distribution of Apache Hadoop.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	2.7.4.100
Release Date	January 2021
Version Interoperability	See EEP Components and OS Support on page 5536.
GitHub Source	https://github.com/mapr/hadoop-common/
GitHub Release Tag	2.7.4.100-mapr-mep-701
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

This release includes bug fixes and updates but no significant new features.

Fixes

This data-fabric release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
4c7ba30	2020-10-09	Backported YARN-4411.RMAppAttemptImpl#createApplicationAttemptReport throws IllegalArgumentException. Contributed by Bibin A Chundatt and yarntime
e98c109	2020-10-21	Backported HADOOP-15054. upgrade hadoop dependency on commons-codec to 1.11

029cd64	2020-10-21	Backported HADOOP-12552. Fix undeclared/unused dependency to httpclient
7d3e416	2020-10-21	Backported HADOOP-13382. Remove unneeded commons-httpclient dependencies from POM files in Hadoop and sub-projects, while adding it in to the only place it is still needed, hadoop-openstack
469c37d	2020-10-21	Backported HADOOP-11614. Remove httpclient dependency from hadoop-openstack
ac3d2a2	2020-10-21	MAPRHADOOP-123: Updated netty versions
bc275b9	2020-10-21	Backported HADOOP-10296. Incorrect null check in SwiftRestClient#buildException
97807cb	2020-10-21	Backported HADOOP-11613. Remove commons-httpclient dependency from hadoop-azure
2a19cc5	2020-10-21	Backported HDFS-12221. Replace xcerces in XmlEditsVisitor
2809c52	2020-10-21	MAPRHADOOP-123: Updated commons-io to 2.7 version
0e080d9	2020-12-07	Backported HADOOP-9613. [JDK8] Update jersey version to latest 1.x release
511cb80	2020-12-22	MAPRMR-19: Add retry support to waitForCompletion() method
fbff7cd	2020-12-29	MAPRYARN-284: YARN kills container but hanging task process is not getting killed
f9a19df	2021-01-05	Backported YARN-5053. More informative diagnostics when applications killed by a user
107da45	2021-01-11	MAPRHADOOP-139: Fixed CVE-2020-13934, CVE-2020-9484 vulnerabilities
2f13f87	2021-01-11	MAPRHADOOP-136: Updated commons-compress
84e3d19	2021-01-13	MAPRYARN-286: RM hangs after calling System.exit() from SchedulerEventDispatcher
55ee369	2021-01-13	Backported YARN-6948. Invalid event: ATTEMPT_ADDED at FINAL_SAVING
08e7c27	2021-01-14	Backported YARN-3344. Fix warning - procfs stat file is not in the expected format
404473d	2021-01-14	Backported HADOOP-17283. Hadoop - Upgrade to jQuery 3.5.1
c91054b	2021-01-14	MAPRYARN-282: Inconsistent resource management between Warden and NodeManager

Known Issues and Limitations

- None.

Resolved Issues

- None.

Hadoop 2.7.4.0-2009 (EEP 7.0.0) Release Notes

The notes below relate to the MapR Data Platform distribution of Apache Hadoop.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	2.7.4.0
Release Date	September 2020
Version Interoperability	See EEP Components and OS Support on page 5536.
GitHub Source	https://github.com/mapr/hadoop-common/
GitHub Release Tag	2.7.4.0-mapr-700
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

This release:

- Decouples Hadoop from core and splits Hadoop into the following packages:
  - `mapr-hadoop-util`
  - `mapr-hadoop-client`
  - `mapr-hadoop-core`

  The `mapr-mapreduce2` package has been removed.
- Decouples the following YARN resources from core and moves them to the MEP repository:
  - `mapr-nodemanager`
  - `mapr-resourcemanager`
  - `mapr-historyserver`
  - `mapr-timelineserver`
- Provides support for YARN CGroups.
- Adds a CLI feature to manage the `node.labels` file that updates and autogenerates the labels file.
- Provides support for JDK 11.
- Adds saving ACE for Hadoop FS commands.

Fixes

This data-fabric release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
e70d0068212	2020-09-04	MAPRHADOOP-110: Added java8_support option to run distributed shell on jdk8
27e5feefb53	2020-08-28	MAPRYARN-232: fix bug with label expression.

82608b1e9d7	2020-08-27	MAPRHADOOP-99: Added support for jmxagent at JMX_REMOTEHOST
22cd170dc77	2020-08-25	MAPRHADOOP-109: create output directory for MapReduce tasks
8d49274d29e	2020-08-25	CORE-235: 'yarn.nodemanager.linux-container-executor.group' settings are not configured properly for non mapr admin user
9af92b94658	2020-08-25	MAPRHADOOP-110: Distributedshell application fix issues MAPRHADOOP-84 and MAPRHADOOP-107
7ff579baeb1	2020-08-14	MAPRHADOOP-108: Hadoop configure.sh doesn't work on Mac OS
0097c56aeda	2020-08-13	MAPRHADOOP-84: Fixed illegal reflective access warning on Windows node
2d51e2ff1c2	2020-08-13	MAPRHADOOP-107: Change GC to ParallelGC for YARN services and containers
670e6816a71	2020-08-13	MAPRHADOOP-110: Fix OOM issue in DistributedShell.
b95185e9953	2020-08-12	MAPRHADOOP-87: added retry property to get token authentication exception
63cabe9cf05	2020-08-12	MAPRYARN-267: Doesn't work link from RM to AM and HS at the non-secure cluster
9acc2c994af	2020-08-10	Backported YARN-8844. TestNMProxy unit test is failing. (Eric Yang via wangda)
40d5ea49fd6	2020-08-10	Backported YARN-4916. TestNMProxy.tesNMProxyRPCRetry fails. Contributed by Tibor Kiss. (cherry picked from commit 00058167431475c6e63c80207424f1d365569e3a)
7f396a21a1c	2020-08-10	Backported HADOOP-11875. [JDK9] Adding a second copy of Hamlet without _ as a one-character identifier.
cc73db526b4	2020-08-10	Backported HADOOP-11875. [JDK9] Adding a second copy of Hamlet without _ as a one-character identifier.
cb153bc81ed	2020-08-10	Backported Fixed import at LdapGroupsMapping
5063d1f7170	2020-08-10	Backported HADOOP-15936. [JDK 11] MiniDFSClusterManager & MiniHadoopClusterManager compilation fails due to the usage of '_' as identifier. Contributed by Zsolt Venczel.
55cc8d04ed0	2020-08-10	Backported HADOOP-16299. [JDK 11] Build fails without specifying -Djavac.version=11
f93adaec25a	2020-08-10	Set java 11 as default for compile
68e65b80365	2020-08-10	Backported HADOOP-15764. Addendum patch: Fix NPE in SecurityUtil.
cf292164bc4	2020-08-10	Backported HADOOP-15764. [JDK10] Migrate from sun.net.dns.ResolverConfiguration to the replacement. Contributed by Akira Ajisaka.
cee2b9a3dbd	2020-08-10	Backported HADOOP-15756. [JDK10] Migrate from sun.net.util.IPAddressUtil to the replacement. Contributed by Akira Ajisaka.
3d1f99ab0b2	2020-08-10	MAPRYARN-265: Blkio CGroups - blkio.weight replaced by new blkio.bfq.weight start from centos8
ec7a13d362a	2020-08-06	MAPRDB-2274: Fixed hadoop_symlinks.sh script for Mac
83186016c2d	2020-08-05	MAPRYARN-241: handle JMX options

d51c8f5d9cd	2020-08-03	MAPRHADOOP-106: Added configure-hadoop.bat to Hadoop project
f36240f896c	2020-07-31	MAPRHADOOP-103: Renamed ext-conf/get-jars-list.sh into ext-conf/mapr-eco-config.sh
cbe7c9d10d6	2020-07-31	MAPRYARN-266: Jobs fails when yarn.use-central-logging-for-mapreduce-only = true
b66ca7bff1c	2020-07-30	MAPRHADOOP-105: Hadoop build fail to generate headers on Windows
a96c5e9fd78	2020-07-29	Backported HADOOP-14586. StringIndexOutOfBoundsException breaks org.apache.hadoop.util.Shell on 2.7.x with Java 9. Contributed by Uwe Schindler.
fb9c3aaef18	2020-07-24	MAPRHADOOP-103: Added get-jars-list.sh for use by Core mapr-config.sh (CORE-221)
143c1ec6a9f	2020-07-24	MFS-2465: added aces prototypes to hdfs.h unitTested skipBuild review @ Rajesh
9cc3222e9c8	2020-07-23	MAPRHADOOP-102: error -1 while copying aces from MapR-FS to local FS
a6f621fa884	2020-07-23	MAPRHADOOP-101: Add ext-conf to the hadoop tar file
f8b702588e6	2020-07-21	Backported HDFS-14729. Upgrade Bootstrap and jQuery versions used in HDFS UIs. Contributed by Vivek Ratnavel Subramanian. This closes #1297
5ba299d1b8a	2020-07-21	Backported YARN-8426:Upgrade jquery-ui to 1.12.1 in YARN. Contributed by Sunil Govindan
0e206b9fd7d	2020-07-21	Backported HADOOP-15483. Upgrade jquery to version 3.3.1. Contributed by Lokesh Jain, Mukul Kumar Singh and Sunil Govindan.
c952b19130d	2020-07-16	MAPRHADOOP-97: Hadoop applications throw warnings
fc6d22f86eb	2020-07-16	MAPRHADOOP-96: Add support for cp/mv commands if symlink was created for directory
0d41507075b	2020-07-13	MAPRYARN-257: Sls tests finish work when run from command line
ad69d2a86cd	2020-07-09	MFS-6696: hadoop fs -mv should preserve source file ACEs to destination file ACEs within cross volumes
8c6fb1dae6e	2020-07-08	MAPRYARN-261: yarn admin users not able to see logs for running jobs
591c85a5ea2	2020-07-08	MAPRHADOOP-84: Illegal reflective access of ShimLoader when using jdk11 for containers
5fda517dbd7	2020-07-08	MAPRYARN-261: yarn admin users not able to see logs for running jobs
ed860bbd99c	2020-07-07	CORE-443: Fixed UMask to 022 for YARN services
14ae8ff7ea7	2020-07-07	MAPRYARN-223: Set max idle time for http and https connection
cb1f7ad20cb	2020-07-06	MAPRYARN-261: Yarn admin users not able to see logs for running jobs
b84ceb51267	2020-07-01	MAPRHADOOP-83: Updated tomcat version to 9.0.36

33e2f935f28	2020-06-19	Backported YARN-8147. TestClientRMService#testGetApplications sporadically fails. Contributed by Jason Lowe
847b75e0b08	2020-06-19	Backported HADOOP-13220. Follow on fixups after upgraded mini-kdc using Kerby. Contributed by Jiajia Li
13312bb98af	2020-06-19	Backported HADOOP-16935. Backport HADOOP-10848. Cleanup calling of sun.security.krb5.Config. (#1912)
82c9d989086	2020-06-19	Backported HADOOP-12911. Upgrade Hadoop MiniKDC with Kerby. Contributed by Jiajia Li
9cef4b0168e	2020-06-17	CORE-434: Linking more Hadoop jar files under Core, for Zookeeper to use
fa1eefe0ef6	2020-06-11	MAPRYARN-258: Publish system metrics to ATS in batches
f70d35cc38d	2020-06-10	MAPRYARN-254: Webproxy failed to validate certificate
067edee9c63	2020-06-05	MAPRHADOOP-90: Removed hadoop-pipes from Hadoop project
1e675edcebb	2020-06-03	MAPRHADOOP-87: added retry property for renew token authentication exception
50898cdb44e	2020-05-28	MAPRHADOOP-83: Upgrade Tomcat to 9.0.34 at the hadoop-kms.
139f9229675	2020-05-28	Backported HADOOP-16439. Upgrade bundled Tomcat in branch-2.
f25fa6db3bb	2020-05-26	MAPRYARN-232: fixed docs and refactored cli feature to manage the node.lables file
2ef67211a1f	2020-05-22	MAPRMR-16: Remove symlink creation to mapr libraries from the hadoop-util post install
a4e4914a409	2020-05-22	MAPRHADOOP-84: Illegal reflective access of ShimLoader when using Java11
740ecbf7c45	2020-05-21	MAPRMR-16: Added script for symlinks creating
9a87ce6c498	2020-05-14	MAPRHADOOP-81: Update htrace-core to 4.2.0-incubating
e4c36d6e350	2020-05-14	Backported HDFS-9187. Fix null pointer error in Globber when FS was not constructed via FileSystem#createFileSystem (cmccabe)
f4367bb816d	2020-05-14	Backported HDFS-9080. Update htrace version to 4.0.1 (cmccabe)
53341245fe5	2020-05-14	Backported HDFS-8008. Support client-side back off when the datanodes are congested. Contributed by Haohui Mai.
1f54dc59c12	2020-05-14	Backported HDFS-8026. Trace FSOutputSummer#writeChecksumChunks rather than DFSOutputStream#writeChunk (cmccabe)
65fbf4e65ae	2020-05-14	Backported HDFS-8100. Refactor DFSClient.Conf to a standalone class and separates short-circuit related conf to ShortCircuitConf.
33b4376755f	2020-05-14	Backported HDFS-7854. Separate class DataStreamer out of DFSOutputStream. Contributed by Li Bo.
20076f7d9ad	2020-05-08	MAPRYARN-250: Added timeout to rescan intermediate_done dirs
094f99dc84f	2020-05-07	SPARK-616: Removed MapRTicketGenerator

last-updated: Apr 11, 2022

3f8150da966	2020-05-07	MAPRHADOOP-82: Fixed exception after guava upgrade
eea73281d19	2020-05-07	SPARK-616: Move to use MapRTicketGenerator from core by default
0622a9745dd	2020-05-07	MAPRHADOOP-82: Update Guava version to 28.2-jre
9cce8949947	2020-05-07	Backported HADOOP-16210. Update guava to 27.0-jre in hadoop-project trunk. Contributed by Gabor Bota.
f48d8562069	2020-05-07	Backported HADOOP-14386. Rewind trunk from Guava 21.0 back to Guava 11.0.2.
ae7dd42e74e	2020-05-07	Backported HADOOP-14386. Rewind trunk from Guava 21.0 back to Guava 11.0.2.
ad083f5de6e	2020-05-07	Backported HADOOP-14382 Remove usages of MoreObjects.toStringHelper. Contributed by Andrew Wang
47e9d711513	2020-05-04	MAPRYARN-249: Clear disks after preemption
92c3887fb7f	2020-04-30	Backported YARN-6175. Negative vcore for resource needed to preempt
a527fe7fe56	2020-04-24	MAPRHADOOP-76: Review and clean up unnecessary files from Hadoop 2.7.4
9b138200dec	2020-04-24	Backported YARN-9783. Remove low-level zookeeper test to be able to build Hadoop against zookeeper 3.5.5. Contributed by Mate Szalay-Beko.
2b38194bc49	2020-04-24	Backported HADOOP-16579. Upgrade to Curator 4.2.0 and ZooKeeper 3.5.5 (#1656). Contributed by Norbert Kalmár, Mate Szalay-Beko
015f61e3b7e	2020-04-15	MAPRHADOOP-72: Fixed hadoop for work with java 1.8
33ac369d359	2020-04-15	Backported HADOOP-15767. [JDK10] Building native package on JDK10 fails due to missing javah. Contributed by Takanobu Asanuma.
8a870750f7d	2020-04-15	Backported HADOOP-14056. Update maven-javadoc-plugin to 2.10.4.
a901c178b9f	2020-04-15	Backported HDFS-11610. sun.net.spi.nameservice.NameService has moved to a new location. Contributed by Akira Ajisaka.
6e97361b594	2020-04-15	Backported HADOOP-12760. sun.misc.Cleaner has moved to a new location in OpenJDK 9. Contributed by Akira Ajisaka.
c75f864bcf3	2020-04-15	HADOOP-15304. [JDK10] Migrate from com.sun.tools.doclets to the replacement. Contributed by Akira Ajisaka.
8ab08d9a70b	2020-04-10	MAPRHADOOP-73: Add ability to manage node labels through RM Rest API
16fe1cfc914	2020-03-30	MAPRHADOOP-27: Configure hadoop on the edge node
e66229651ca	2020-03-24	MAPRYARN-246: RM Hangs with a FATAL error when space is provided in the queue name
622decf8b00	2020-03-24	Backported YARN-3400. [JDK 8] Build Failure due to unreported exceptions in RPCUtil (rkanter)
a1947649fe9	2020-03-24	MAPRYARN-232: Add CLI / maprcli feature to manage the node.lables file updates or autogenerate the labels file
e4d6f999430	2020-03-23	Backported HADOOP-13773. Wrong HADOOP_CLIENT_OPTS in hadoop-env on branch-2. Contributed by Fei Hui

7066457622e	2020-03-19	MAPRYARM-241: Added JMX configuration to YARN RM/NM
c50bc6d60ce	2020-03-18	MAPRYARN-154: Cleanup extra init mapreduce directory
cbbb05dc11b	2020-03-11	MAPRYARN-244: RM hangs after calling System.exit()
85ab8d513de	2020-03-10	MFS-4656: added hdfs api to set ticket
bbd3cd4deee	2020-03-04	Backported YARN-8382. cgroup file leak in NM. Contributed by Hu Ziqian.
ebb6b4b11bb	2020-02-27	MAPRHADOOP-28: Hadoop - Distribute Notice.txt across components starting with MEP 7.0
8657ed9fae9	2020-02-27	Backported YARN-4643. Container recovery is broken with delegating container runtime. Contributed by Sidharta Seethana
58746fe320e	2020-02-21	MAPRHADOOP-70: hadoop fs -cp -p and distCp should copy source file aces to destination file aces
fff79ee878b	2020-02-19	MAPRMR-14: DistCp error - duplicate files in input path
39e3b919f1e	2020-02-13	MAPRMR-13: Distcp link file copy doesn't work
825a1af3f29	2020-02-05	MAPRYARN-183: Nodemanager log level change for message: Can not find metadata for a job. Returning service metadata
e08862759b1	2020-01-30	MAPRYARN-224: Added logic to clean up Spill files directory
12b429601c3	2020-01-23	MAPRHADOOP-69: Disable TLSv1.1 by default and uses TLSv1.2 by default
2edb37aede0	2020-01-23	MAPRMR-12: Added support blocksperchunk for large files in distcp
18d193a7509	2020-01-17	MAPRYARN-211: Fix problem that NodeManager don't start when CPU lower than 4 and MFSCPU=4
9d1b3450a79	2020-01-17	MAPRYARN-185: Added CGroup support. Backported YARN-3365, YARN-3443, YARN-3366, YARN-2619, YARN-1912, YARN-3684, YARN-160, YARN-2194, YARN-2194, YARN-3982, YARN-3853, YARN-4253, YARN-1856, YARN-3542, YARN-4578, YARN-5849, YARN-6500, YARN-5301, YARN-6757, YARN-4744, YARN-6433, YARN-7818
0b68b6052f9	2020-01-17	MAPRYARN-224: NM should always remove spill directories or completed applications
28bd7d86327	2020-01-15	MAPRHADOOP-43: Evaluate impact of Protobuf upgrade for 6.2 to Protobuf 3.11.1 on Hadoop
ce5ba4702bd	2020-01-13	MAPRYARN-227: Infinite authentication loop when try to access AM UI from RM UI
6a34523c0cf	2019-12-23	MAPRYARN-223: Make jetty connection max idle time configurable
499840501ef	2019-12-18	MAPRHADOOP-64: upgraded tomcat to 6.0.53 version
2bbef34188b	2019-12-16	MAPRHADOOP-63: Removed jackson-databind dependency because it has security vulnerability
adb89d7b1c2	2019-12-10	MAPRHADOOP-61: Setting custom ticket location makes Kerberos fail for services
c8641f01c44	2019-11-25	MAPRYARN-221: ContainerLocalizer process hangs
c9f7560dd8e	2019-11-25	MAPRYARN-168: NM/RM process do not start with XX:ErrorFile option (fix for Ubuntu and SLES)

last-updated: Apr 11, 2022

631e2c09e64	2019-11-14	MAPRYARN-218: Log aggregation has not completed or is not enabled MAPRYARN-219: AggregatedLogDeletionService: Logs of all application containers are not removed when logs of single container are not accessible
51b197c0ef3	2019-11-11	MFS-3295: Fix build breakage due to an earlier commit. Including some header files needed for uid_t.
9778fb86988	2019-11-09	MAPRYARN-217: No metadata found for application by other users
36ef2eb3677	2019-11-08	MFS-3295: Added a new API hdfsConnectAsUid() for connecting to mapr cluster using a uid.
18288f2b6ed	2019-11-08	MAPRYARN-216: Node local log aggregation does not support hostname:port name format in command "yarn logs ... -nodeAddress"
0bc3a053882	2019-11-06	MAPRYARN-215: NodeLocalMetadataWriter init failed when restart NodeManager and mfs simultaneously
719246c1b94	2019-11-06	MAPRYARN-214: HistoryServer UI doesn't show local aggregated Logs
67608235160	2019-10-31	MAPRYARN-210: Node-local log aggregation
b9e18c50841	2019-10-09	CORE-297: Added security headers for RM, NM, JHS, TLS
2f41fd0ec1f	2019-09-23	MAPRYARN-208: Update Dockerfile, hadoop_env_checks and start-build-env files
f41206b97f8	2019-09-23	Backported HADOOP-11843. Make setting up the build environment easier. Contributed by Niels Basjes.
6aed14c8703	2019-09-18	MAPRHADOOP-59: Added debug logs to aggregated log deletion logic.
4200186ea25	2019-09-18	MAPRHADOOP-58: Changed version of commons-configuration because of CVE-2014-0114.
614d51ece3d	2019-09-04	MAPRHADOOP-54: Fixed getLabelsOnNode REST API method
3137436bea3	2019-08-14	MAPRHADOOP-57: Updated commons-net and commons-io versions
28a970b6c25	2019-08-02	Backported HADOOP-15865. ConcurrentModificationException in Configuration.overlay() method. Contributed by Oleksandr Shevchenko.
935de182482	2019-07-30	MAPRHADOOP-36: adding path to libjvm.so native lib to LD_LIBRARY_PATH env variable
b9514d38e04	2019-07-19	MAPRYARN-202: FairScheduler improvements. Contains partly or fully. YARN-1287. Consolidate MockClocks. YARN-4090. Make Collections.sort() more efficient by caching resource usage YARN-7382. NoSuchElementException in FairScheduler after failover causes RM crash
f2bdb27a4cd	2019-07-19	MAPRYARN-162: added preemptedDisks and allocatedDisks fields
1a29a9bca14	2019-07-12	MAPRHADOOP-37: add outDir null check
e94706b9266	2019-07-09	MAPRHADOOP-54: Add ability to fetch node labels through RM Rest API
bb7d41d794c	2019-07-04	MAPRYARN-200: Backport YARN-6797: TimelineWriter does not fully consume the POST response

1b02dfa19d0	2019-06-27	MAPRHADOOP-52: update dependency (upgrade version commons-compress and netty)
d0315775926	2019-06-27	Backported HADOOP-14597. Native compilation broken with OpenSSL-1.1.0. Contributed by Ravi Prakash.
bbfad1bdc9f	2019-06-26	MAPRYARN-191: Fixed some ClientRMService node/labels related methods.
28e0de41fc3	2019-06-21	MAPRHADOOP-49: Hadoop configuration after installation incomplete
f8560fdd218	2019-06-20	MAPRYARN-195: Fixed compare method at DominantResourceFairnessPolicy, change weight calculation for CPU, MEMORY, DISK.
ecc0d96cba4	2019-06-20	Backported YARN-8436. FSParentQueue: Comparison method violates its general contract. (Wilfred Spiegelenburg via Haibo Chen)
30ab0e70ef3	2019-06-18	MAPRMR-8: Added property for the configuration max number of blocks in the split for CombineFileInputFormat
dd01c093203	2019-06-14	MAPRHADOOP-37: Add copyAce logic on running Map Tasks
428acd4d377	2019-06-07	MAPRYARN-192: Refactored label wrapper to handle edge conditions
632659bfa7f	2019-06-07	MAPRYARN-191: Edited node labels page to work with MapR specific.
6bdf170a840	2019-06-07	MAPRYARN-191: Fixed yarn cluster -lnl to properly show label information.
8d95c10257a	2019-06-06	MAPRYARN-192: Added AND support to label expression
51919781c70	2019-06-04	MAPRYARN-154: Added creation of nm-staging volume to LocalVolumeAuxService.
97028f090e8	2019-06-03	MAPRYARN-168: Added -XX:ErrorFile default location to YARN_OPTS.
a44b8b3c195	2019-05-31	MAPRYARN-182: Added AM/PM flag to last heartbeat report time.
b43cb6a34a8	2019-05-30	MAPRHADOOP-37: added copyAce method to AbstractMapRFileSystem.
1cd2861e7d2	2019-05-29	MAPRYARN-187: Repair SLS to work with hadoop-2.7.4-mapr.
ab38d390906	2019-05-28	Backported YARN-4579. Allow DefaultContainerExecutor container log directory permissions to be configurable (rchiang via rkanter)
a283450c281	2019-05-24	MAPRHADOOP-42: Excluded mapr specific properties from map-reduce unit tests
7d3f70347e3	2019-05-23	Backported YARN-6078. Containers stuck in Localizing state. Adapted to 2.7 hadoop. (cherry picked from commit bd1a53c)
2e3f1b93ab3	2019-05-23	Backported YARN-6004. Refactor TestResourceLocalizationService#testDownloadingResources OnContainer so that it is less than 150 lines. (Chris Trezzo via mingma) (cherry picked from commit 61424da)
9c9a8a89d49	2019-05-22	MAPRHADOOP-42: Excluded mapr specific properties from unit tests
a043d169aeb	2019-05-13	MAPRYARN-183: changed the log levels to WARN from ERROR at LocalVolumeAusService.java

last-updated: Apr 11, 2022

e18c6c7bf33	2019-05-13	MAPRHADOOP-40: Set default mapr_home in ssl-client/server conf
a7489d421c6	2019-05-13	MAPRHADOOP-34: Property mapreduce.shuffle.ssl.enabled was add to the mapred-site.xml for secure clusters
12909327cc6	2019-05-07	MAPRHADOOP-39: Option -R doesn't clean up yarn-site.xml
81af4616c5a	2019-05-03	MAPRYARN-173: Fixed wrong constant usage in TestCapacityScheduler. Fixed using maxVcores instead of maxDisks in refreshMinimumAllocation in AbstractYarnScheduler. Created testRefreshMaximumResourceAllocationShouldReturnSameResourcesAsInArgument.
7ffee8be41e	2019-04-26	MAPRYARN-179: Fixed FairSharePolicy comparator.
0de1dad6d74	2019-04-24	MAPRHADOOP-36: Add condition for getting MaprFS instead of LocalFS for writing logs in case of enabled DFS logging.
bebab711a51	2019-04-24	MAPRMR-7: added new variable to FileSystem.Cache.Key
9c32b7a4296	2019-04-23	Backported HDFS-9612. DistCp worker threads are not terminated after jobs are done. (Wei-Chiu Chuang via Yongjun Zhang)
86199b5232f	2019-04-23	Backported HDFS-9347. Invariant assumption in TestQuorumJournalManager.shutdown() is wrong. Contributed by Wei-Chiu Chuang.
26dbab3311b	2019-04-23	MAPRYARN-175: yarn job fails to initialize when output directory is set to target cluster
32ae1a69d69	2019-04-18	Backported YARN-5969. FairShareComparator: Cache value of getResourceUsage for better performance. Fix using existing Resource.java API in 2.7.0 hadoop.
0627a2c1ec6	2019-04-18	Backported YARN-6769. Make schedulables without demand less needy in FairSharePolicy#compare. (Yunfan Zhou via Yufei Gu)
e9cb861304a	2019-04-18	Backported YARN-4690. Skip object allocation in FSAppAttempt#getResourceUsage when possible (Ming Ma via sjlee)
61cd9d9efdf	2019-04-10	MAPRYARN-171: jobs comparison(with disk usage) was fixed.
934258896f5	2019-03-22	MAPRYARN-161: History server Deletion thread stops once it finds an invalid application directory
1b4b61f19b5	2019-03-22	MAPRYARN-167: Timeline had incorrect configuration
00384e786b6	2019-03-21	MAPRYARN-169: Moved hadoop-maprfs package to hadoop-common
fae981d7127	2019-03-20	MAPRYARN-165: Options for setting oom_score_adj for NM child containers
239f2753da5	2019-03-15	MAPRHADOOP-32: Configure yarn-site before change permission
ef6d389c39f	2019-03-13	MFS-2179: Add few more C++ apis to hdfs header
11414518543	2019-03-13	Backported YARN-4000. RM crashes with NPE if leaf queue becomes parent queue during restart. Contributed by Varun Saxena
efb21906cd0	2019-03-13	Backported YARN-4087. Followup fixes after YARN-2019 regarding RM behavior when state-store error occurs. Contributed by Jian He

9e7298ee372	2019-03-13	Backported YARN-2019. Retrospect on decision of making RM crashed if any exception throw in ZKRMStateStore. Contributed by Jian He.
bfa630eba18	2019-03-13	Backported YARN-4459. container-executor should only kill process groups. Contributed by Jun Gong (cherry picked from commit 1ba31fe9e906dbd093afd4b254216601967a4a7b)
6087ca36843	2019-03-07	MAPRYARN-155. changed "dot" in dfs logging property to "underscore".
d0daa13f506	2019-02-04	COMSECURE-117: Cleanup external token only for finished application
feb85e70e82	2018-11-07	Backported MAPREDUCE-6741. Add MR support to redact job conf properties. Contributed by Haibo Chen (cherry picked from commit f1b74a3d9ff71bc014dbfd29a6996071b81d14c5)
b5120c88b17	2018-11-07	Backported MAPREDUCE-6259. IllegalArgumentException due to missing job submit time. Contributed by zhihai xu (cherry picked from commit bf70c5ae2824a9139c1aa9d7c14020018881cec2)
6014bac6e00	2018-10-23	Backported YARN-4153. TestAsyncDispatcher failed at branch-2.7. Contributed by Zhihai Xu
fc37a3014ce	2018-10-23	Backported YARN-3999. RM hangs on draing events. Contributed by Jian He
b26e91baae3	2018-10-23	Backported YARN-3978. Configurably turn off the saving of container info in Generic AHS (Eric Payne via jeagles)
c944bad4679	2018-10-16	Backported MAPREDUCE-6817. The format of job start time in JHS is different from those of submit and finish time. (Haibo Chen via kasha)
ad4fbefa295	2018-10-16	Backported MAPREDUCE-6620. Jobs that did not start are shown as starting in 1969 in the JHS web UI (haibochen via rkanter)
767c4bc75c2	2018-10-11	MAPR-32181: Added max delete size for ATS deletion thread wake up
88f57119dfc	2018-09-29	Backported YARN-4398. Remove unnecessary synchronization in RMStateStore. Contributed by Ning Ding
7fe347bd5a5	2018-09-25	MAPR-32319: Containers fail during creating a symlink which started with hyphen for a resource file
9c8adf15343	2018-09-17	Backported MAPREDUCE-7101. Add config parameter to allow JHS to alway scan user dir irrespective of modTime. (Thomas Marquardt via wangda)
5ebae0609bd	2018-09-17	Backported MAPREDUCE-6680. JHS UserLogDir scan algorithm sometime could skip directory with update in CloudFS (Azure FileSystem, S3, etc. Contributed by Junping Du
18fb150b10a	2018-09-17	MAPREDUCE-6662. Clear ASF Warnings on test data files. Contributed by Vinayakumar B
6ffc9e2dd6f	2018-09-17	MAPREDUCE-6577. MR AM unable to load native library without MR_AM_ADMIN_USER_ENV set (sjlee)
ce37c0d2d17	2018-09-17	MAPR-YARN-124. NullPointerException in IFile$Reader for Apache's shuffle implementation
c561fd8b999	2018-09-17	Backported MAPREDUCE-5807. Print usage for TeraSort job. Contributed by Rohith.

2c25b288632	2018-09-17	MAPRYARN-123: Job with label runs on node without proper label.
5d4137018e6	2018-09-17	MAPRYARN-125: Possibility to enable/disable multi split locations
858f75b226f	2018-09-17	Backported MAPREDUCE-7137. MRAppBenchmark.benchmark1() fails with NullPointerException
dfd4d41340f	2018-09-17	Backported MAPREDUCE-7139. TestShuffleProvider#testShuffleProviders and TestTaskAttemptContainerRequest#testAttemptContainerRequest fail since a static common container launch context does't recreate
530f3db816a	2018-09-17	Backported MAPREDUCE-7138. ThrottledContainerAllocator in MRAppBenchmark should implement RMHeartbeatHandler
9ac645828f1	2018-09-17	MAPRYARN-15: Set default value for 'java.security.auth.login.config' if unset before. Needed for unit tests. On a cluster this property set by JVMProperties.java.
603c3195d44	2018-09-17	Backported MAPREDUCE-5982. Task attempts that fail from the ASSIGNED state can disappear. Contributed by Chang Li Additional commit from 2.8.0. (cherry picked from commit ee4ee6af6a5a6299d27462adb6944206039bbbae)
a03346d409f	2018-09-17	Backported MAPREDUCE-6513. MR job got hanged forever when one NM unstable for some time. (Varun Saxena via wangda)
03d1f08536b	2018-09-17	Backported MAPREDUCE-5465. Tasks are often killed before they exit on their own. Contributed by Ming Ma
07b71529e9f	2018-09-17	Backported MAPREDUCE-7133. History Server task attempts REST API returns invalid data. Contributed by Oleksandr Shevchenko
81a931b6c00	2018-09-12	Backported YARN-8197. Fixed AM IP Filter and Webapp proxy to redirect app tracking-URLs correctly when UI is secure. Contributed by Sunil Govindan.
467c77e8096	2018-09-12	Backported YARN-4204. ConcurrentModificationException in FairSchedulerQueueInfo. (adhoot)
c70287ef4dd	2018-09-12	MAPRYARN-114: DistCp job fails during symlink copying The cause of the problem is comparing a size of a target file with a size of a symlink to ensure that copy was successful. They are not equal. We should compare the size of the source of symlink and target file which will be copied from a source of a symlink.
7cdb92251d7	2018-09-12	Backported HADOOP-12963 Allow using path style addressing for accessing the s3 endpoint. (Stephen Montgomery via stevel)
c2ec6367fec	2018-09-12	Backported HADOOP-12851. S3AFileSystem Uptake of ProviderUtils.excludeIncompatibleCredentialProviders. Contributed by Larry McCay.
a583bb41624	2018-09-12	Backported HADOOP-12548. Read s3a creds from a Credential Provider. Contributed by Larry McCay.
74a66284ec8	2018-09-12	Backported HADOOP-12292. Make use of DeleteObjects optional. (Thomas Demoor via stevel)
a6b4c204373	2018-09-12	Backported HADOOP-12269. Update aws-sdk dependency to 1.10.6 (Thomas Demoor via Lei (Eddy) Xu)

Known Issues and Limitations

- None.

Resolved Issues

- None.

HBase Release Notes

The release notes for HBase component included in the MapR Converged Data Platform contains notes specific to MapR only.

More details are available on the Apache website under the Release Notes for Apache HBase page and the Apache HBase Project page.

HBase 1.4.13.200 - 2201 (EEP 8.1.0) Release Notes

The notes below relate to the MapR Data Platform distribution of HBase.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. You may also be interested in the Ecosystem Component Release Notes on page 5658 and the Apache HBase homepage.

Version	1.4.13.200
Release Date	January 2022
Version Interoperability	See Interoperability Matrix, Ecosystem Support Matrix, and HBase Support Matrix.
Source on GitHub	https://github.com/mapr/hbase
GitHub Release Tag	1.4.13.200-eep-810
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP (MEP) and OS to view the list of package names.

New in this Release

HBase 1.4.13.200 - 2201 introduces the following enhancemtns or HPE platform-specific behavior changes:

- Federal Information Processing Standards (FIPS) support
- CVE fixes
- Bug fixes

Fixes

This HPE release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub:

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description
12c2a75dff9	2022-02-09	EEP-HBASE-280: FIPS mode check needs to be done over ssl-client instead of Zookeeper
af61cf17008	2022-01-25	EEP-HBASE-275: Upgrade Log4J version to '1.3.1-mapr'

last-updated: Apr 11, 2022

ddba4184c7	2022-01-13	EEP-HBASE-274: HBase REST authentication doesn't work for /jmx and /conf
45e13fbfe7	2022-01-05	EEP-HBASE-271: Update log4j v2 to the latest available (to 2.17+)
f33379b62c	2021-12-20	EEP-HBASE-263: Log4j Vulnerabilities: CVE-2019-17571 -- Upgrading to 1.3.0-mapr
04229b0d9b	2021-12-14	EEP-HBASE-263: Log4j Vulnerabilities: CVE-2019-17571
8ce02cfa42	2021-12-14	EEP-HBASE-269: CVE-2021-44228 - Log4j vulnerability in HBase
65769ae7c3	2021-11-25	EEP-HBASE-268: Upgrade Jetty to 9.4.44.v20210927 to sync with Hadoop
c98b48fd3c	2021-11-25	EEP-HBASE-264: Nimbus-jose-jwt Vulnerabilities: CVE-2019-17195, CVE-2017-12974 and CVE-2017-12972
f4c5bc5e5b	2021-11-24	EEP-HBASE-267: Update Spark version to 3.2.0.0-eep-SNAPSHOT
44c6284197	2021-11-23	EEP-HBASE-266: Exclude bcprov-jdk15on from HBase dependencies
cbee9488cf	2021-11-18	EEP-HBASE-260: Netty Vulnerabilities: WS-2020-0408, CVE-2021-37137 and CVE-2021-37136
c030c1b1af	2021-11-15	EEP-HBASE-257: HBase mapreduce jobs failed on cluster with enabled FIPS
51bc6be0cb	2021-11-09	EEP-HBASE-258: Update hadoop version to 2.7.6.200-eep-810-SNAPSHOT
74a9e15252	2021-10-25	MAPR-HBASE-252: [Hbase] CVE-2021-42340 Apache Tomcat DoS
748a79f5ab	2021-10-21	MAPR-HBASE-254: Upgrade slf4j dependencies from 1.7.5 to 1.7.25 to sync with Hadoop
21108dde8d	2021-10-18	MAPR-HBASE-251: HBase services can't start on cluster with enabled FIPS

Known Issues and Limitations

This release contains the following known issues and limitations:

- For a FIPS-enabled configuration, mixed mode support is not available in this release. For example, a non-FIPS client node cannot communicate with a FIPS server node.

HBase 1.4.13.100 - 2110 (EEP 8.0.0) Release Notes

The notes below relate to the MapR Data Platform distribution of HBase.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. You may also be interested in the Ecosystem Component Release Notes on page 5658 and the Apache HBase homepage.

Version	1.4.13.100
Release Date	October 2021
Version Interoperability	See Interoperability Matrix, Ecosystem Support Matrix, and HBase Support Matrix.
Source on GitHub	https://github.com/mapr/hbase
GitHub Release Tag	1.4.13.100-eep-800
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP (MEP) and OS to view the list of package names.

New in this Release

HBase 1.4.13.100 - 2110 introduces mainly bug fixes and fixes to common vulnerabilities and exposures (CVEs).

Fixes

This HPE release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub:

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description
dd0fda2904	2021-09-16	MAPR-HBASE-249: ThriftServer and RESTServer cannot start on core 7.0.0.
36bba5050e	2021-09-06	MAPR-HBASE-247: mapr-security-web jar should be taken from the cluster.
e9c5d738d8 912685bd11	2021-09-03 2021-09-03	MAPR-HBASE-246: Update the maven artifact version strings to eep.
c9e1ff499b	2021-08-11	MAPR-HBASE-245: Update Spark version to 3.1.2.0-mapr-SNAPSHOT.
0195fe8ea1	2021-08-09	MAPR-HBASE-244: Upgrade Avro version to 1.10.1.
447f5b6f36	2021-08-09	MAPR-HBASE-243: Update Hadoop version to 2.7.6.0-mapr-720-SNAPSHOT.
eda8dcfeea ae90385dbe	2021-07-27 2021-08-12	MAPR-HBASE-240: CVE-2012-5783 vulnerability in commons-httpclient. (HBASE-16267 Remove commons-httpclient dependency from hbase-rest module).
9f4c884c4d	2021-07-22	MAPR-HBASE-242: Too large error message/uninformative when running HBase shell from user without ticket.
0e453c03b9	2021-06-19	MAPR-HBASE-239: WS-2019-0379: commons-codec vulnerability.

last-updated: Apr 11, 2022

bb7013e7ab	2021-06-18	MAPR-HBASE-237: Sync Jackson version with hadoop-2.7.5.0.
4924431fad	2021-06-18	MapR [SPARK-889] HBase examples running fails with "org.apache.spark.unsafe.types.UTF8String is not a valid external type for schema of string".
ca6787138f	2021-06-04	MAPR-HBASE-236: CVE-2020-15250 vulnerability in JUnit.
861cf40791	2021-06-04	MAPR-HBASE-208: HBase build should use only internal repositories / mirrors.

Known Issues and Limitations

This release contains the following known issues and limitations:

- None

HBase 1.4.13.0 - 2104 (EEP 7.1.0) Release Notes

The notes below relate to the MapR Data Platform distribution of HBase.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. You may also be interested in the Ecosystem Component Release Notes on page 5658 and the Apache HBase homepage.

Version	1.4.13.0
Release Date	April 2021
MapR Version Interoperability	See Interoperability Matrix, Ecosystem Support Matrix, and HBase Support Matrix.
Source on GitHub	https://github.com/mapr/hbase
GitHub Release Tag	1.4.13.0-mapr-710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

HBase 1.4.13.0 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- CVEs fixes.
- Extra loggings.
- Service verifier.
- Bug fixes.

Fixes

This HPE release includes the following fixes on the base Apache release:

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description

451ca18c25	2021-04-27	MAPR-HBASE-233: Couldn't get into HBase shell on client node.
05892ab377	2021-04-13	MAPR-HBASE-232: Impersonation doesn't work for hbase-rest. (cherry picking HBASE-21960 Ensure RESTServletContainer used by RESTServer)
c517c848b5	2021-03-30	MAPR-HBASE-218: Add service verifier to HBase package (Additional checks on services).
229e1d69ca	2021-03-29	MAPR-HBASE-217: HBase data loss due to RS crash, extra-logging added for investigation.
3cfa6a2055	2021-03-22	MAPR-HBASE-218: Add service verifier to HBase package.
9b736d6a70	2021-03-22	MAPR-HBASE-167: Upgrade Jackson-Databind (transitive from HTrace) packaged with MapR-HBase.
3aed5b975a	2021-03-19	MAPR-HBASE-230: Update HBase dependencies to the latest artifacts for MEP7.1.0 release.
e2c24bb816	2021-03-17	MAPR-HBASE-225: Opentsdb daemon process does not start after setting noexec to /tmp dir in MapR 6.1/MEP-6.2.
cd81f567e3	2021-03-12	MAPR-HBASE-223: CVE: Security vulnerabilities in bootstrap.js (cherry picking HBASE-24971: updating JQuery).
4a3138f81f	2021-03-04	MAPR-HBASE-216: HBase process should start with ErrorFile option.
0fe43d8588	2021-03-02	MAPR-HBASE-223: CVE: security vulnerabilities in bootstrap.js (cherry picking HBASE-25079 + HBASE-25261).
a0d4312e03	2021-02-23	MAPR-HBASE-220: Dependency error for 3.0.1-b06-SNAPSHOT: PKIX path validation failed.
0b0be18f3b	2021-02-11	MAPR-HBASE-215: CVE-2014-3488,CVE-2019-16869 etc. Netty vulnerabilities.
911a5d63db	2021-02-09	MAPR-HBASE-214: Vulnerabilities in Tomcat.
8d43b042c4	2021-02-05	MAPR-HBASE-207: HBase Web UIs display incorrectly when encryption is disabled on secure cluster.

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

This release contains the following known issues and limitations:

- None

HBase 1.4.12.100 - 2101 (EEP 7.0.1) Release Notes

The notes below relate to the MapR Data Platform distribution of HBase.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.4.12.100
Release Date	January 2021
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hbase
GitHub Release Tag	1.4.12.0-mapr-701
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

This release contains CVE fixes.

Fixes

This HPE release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
fd7abcd11e	2021-01-08	MAPR-HBASE-206 Excessive logging from DynamicClassLoader
4b306ec007	2021-01-06	MAPR-HBASE-205 create command fails if {BULKLOAD => 'true'} specified
de33733588	2017-07-19	MAPR-HBASE-204 Fix shell noninteractive launch
8df472ddb6	2020-12-30	MAPR-HBASE-201 CVE-2016-6796: Tomcat / Jasper library vulnerability
41ec48aa4c	2020-12-23	MAPR-HBASE-199: Command 'processlist' fails with ERROR: Unexpected end of file from server on secure cluster
81d8aa72f7	2020-12-11	MAPR-HBASE-200: Netty vulnerabilities
9446669d5c	2020-11-24	MAPR-HBASE-195: upgrading thrift from 12 to 13. jetty version is updated based on mapr-security-web dependencies
67c8bce37b	2020-10-12	MAPR-HBASE-194 Error 500 when using HBase Rest to create namespace
3ccd2f9894	2020-09-15	MAPR-HBASE-191 Retrieve max version records using spark hbase binary connector

75e88be952	2020-09-07	MAPR-HBASE-184: Illegal reflective access of sun.net.www.protocol.http.HttpURLConnection.userAgent

Known Issues and Limitations

This release contains the following known issues and limitations:

- None

HBase 1.4.12.0-2009 (EEP 7.0.0) Release Notes

The notes below relate to the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.4.12.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hbase
GitHub Release Tag	1.4.12.0-mapr-700
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

- Java 11 Support
- CVEs fixes

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
63a98f35	2020-03-05	MAPR-HBASE-145: Update protobuf version to 3.11.1
2074692b	2020-03-30	MAPR-HBASE-141: Disable TLSv1 and TLSv1.1 protocol for Hbase thrift server
ceeefb42	2020-04-02	MAPR-HBASE-152: Duplicates of some jars in the hbase lib directory
b49b19fa	2020-04-08	MAPR-HBASE-151: Bad exception message
10e834fc	2020-04-20	MAPR-HBASE-164: Unable to modify a column family in a maprdb table using hbase shell

177800e6	2020-04-23	MAPR-HBASE-157: Fix io.netty vulnerability
4a364d2b	2020-04-03	MAPR-HBASE-163: Unable to create buffered mutator for maprdb table
985f6ae1	2020-04-24	MAPR-HBASE-154: Unable to start hbase shell with MapR ticket in case Kerberos is enables in the cluster
65e21d86	2020-04-27	MAPR-HBASE-165: ZK version updates to v3.5.6 for MEP 7.0.0
ddde8d1b	2020-04-30	MAPR-HBASE-166: hbase hbck command incorrectly uses HBASE_REGIONSERVER_OPTS environment variable
041ca362	2020-05-07	MAPR-HBASE-156: Fix org.codehaus.jackson vulnerability
520ebadd	2020-05-12	MAPR-HBASE-172: Throw UnsupportedOperationException for unsupported maprdb methods
11ccd452	2020-05-19	MAPR-HBASE-170: Update Google Guava library version to `28.2-jre` in HBase 1.4.12
2c67a1eb	2020-05-22	MAPR-HBASE-171: 'list_perm' and 'set_perm' commands are not available
9600575c	2020-05-26	MAPR-HBASE-168: HBase web pages don't look right
2216faab	2020-06-05	MAPR-HBASE-173: HBase doesn't start on core 6.2.0 with Java11
58287ba0	2020-06-17	MAPR-HBASE-158: Build and verify HBase and Java 11
37be28c7	2020-07-03	MAPR-HBASE-180: 404 ERROR for all requests to the hbase-rest
f67c7c47	2020-07-10	MAPR-HBASE-179: Warnings in the hbase shell
a466d4af	2020-07-20	MAPR-HBASE-181: HBase Rest Throws NoClassDefFoundError: org/apache/logging/log4j/core/Layout
c9da5fd0	2020-07-20	CORE-262 and CORE-263 Warden hbase memory/svc management add hbmaster and hbregionserver warden conf with the previously used memory settings
c726132a	2020-07-22	MAPR-HBASE-182: Could not finish disable_all/enable_all commands - 'enter' key does not work correctly in the hbase shell
4abed490	2020-07-29	MAPR-HBASE-183: Could not create region server group in the hbase shell
706c9461	2020-08-10	MAPR-HBASE-185: Unnecessary string in the output of command 'show_filters'

77a95708	2020-08-19	MAPR-HBASE-175: Hbase thrift Java JMX Server Insecure Configuration Remote Code Execution Vulnerability
796757af	2020-08-19	MAPR-HBASE-190: Update disruptor dependency to 3.4.2
e94abc9b	2020-08-21	MAPRDB-2275: [hbase-1.4.12] NPE in BufferedMutatorImpl.setRpcTimeout
70068f7a	2020-08-28	MAPR-HBASE-189: hbase.rest.client.RemoteAdmin.getRestVersion return 500 error
b092745b	2020-08-28	CORE-468: fix issue causing hbase classpath to break
049614b2	2020-09-01	MAPR-HBASE-174: Hbase services fails with NPE when Credential Provider is configured.

Known Issues and Limitations

This release contains the following known issues and limitations:

- None

HBase 1.1.13.500-2201 (EEP 6.3.6) Release Notes

The notes below relate to the MapR Data Platform distribution of Apache HBase.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. You may also be interested in Ecosystem Component Release Notes on page 5658 and the Apache HBase homepage.

Version	1.1.13.500
Release Date	January 2022
Version Interoperability	See Interoperability Matrix, Ecosystem Support Matrix, and HBase Support Matrix
Source on GitHub	https://github.com/mapr/hbase
GitHub Release Tag	1.1.13.500-mapr-636
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP (MEP) and OS to view the list of package names.

New in this Release

HBase 1.1.13.500 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

- CVE fixes
- Bug fixes

Fixes

This HPE release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description
3f220a0c74	2022-01-25	EEP-HBASE-275: Upgrade Log4J version to '1.3.1-mapr'
7fdeb174ed	2022-01-13	EEP-HBASE-274: HBase REST authentication doesn't work for /jmx and /conf
a5f4e12215	2021-12-20	EEP-HBASE-270: Update disruptor dependency in EEP-6.3.5
ec24def3de	2021-12-20	EEP-HBASE-263: Log4j Vulnerabilities: CVE-2019-17571 -- Upgrading to 1.3.0-mapr
69fc348174	2021-12-14	EEP-HBASE-263: Log4j Vulnerabilities: CVE-2019-17571
90584ea4d5	2021-11-18	EEP-HBASE-260: Netty Vulnerabilities: WS-2020-0408, CVE-2021-37137 and CVE-2021-37136
42b703d124	2021-10-25	MAPR-HBASE-252: [Hbase] CVE-2021-42340 Apache Tomcat DoS

Known Issues and Limitations

This release contains the following known issues and limitations:

- ASYNC-5: Data-fabric-SASL security is not implemented in AsyncHBase. As a result, an AsyncHBase client connection fails on a secure data-fabric cluster. **Workaround:** Implement AsyncHBase security or disable HBase security.
- Data-fabric client nodes cannot be configured for HBase by using the `configure.sh` script. **Workaround:** To configure an HBase client node, you can use the `configure_client.sh` script. For more information, see Configuring HBase on a Client Node.
- MD-5925: Drill versions 1.11.0 through 1.16.0.x do not support querying HBase tables. **Workaround:** None.
- ASYNC-9: An error is generated when AsyncHBase is used to put data to a cluster with HBase and Kerberos configured. The error occurs when `hbase.rpc.protection` is set to privacy or integrity in the AsyncHbase client configuration. **Workaround:** To avoid the error, set `hbase.rpc.protection` to `authentication`.

HBase 1.1.13.400-2110 (EEP 6.3.5) Release Notes

The notes below relate to the MapR Data Platform distribution of Apache HBase.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. You may also be interested in Ecosystem Component Release Notes on page 5658 and the Apache HBase homepage.

Version	1.1.13.400
Release Date	October 2021
Version Interoperability	See Interoperability Matrix, Ecosystem Support Matrix, and HBase Support Matrix
Source on GitHub	https://github.com/mapr/hbase

GitHub Release Tag	1.1.13.400-mapr-635
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP (MEP) and OS to view the list of package names.

New in this Release

HBase 1.1.13.400 - 2110 introduces mainly bug fixes and fixes to common vulnerabilities and exposures (CVEs).

Fixes

This HPE release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description
6676521667 a85d668dba	2021-08-12 2021-07-27	MAPR-HBASE-240: CVE-2012-5783 vulnerability in commons-httpclient. (HBASE-16267 Remove commons-httpclient dependency from hbase-rest module).
3082076dae	2021-07-19	MAPR-HBASE-239: WS-2019-0379: commons-codec vulnerability.
66514954a1	2021-06-04	MAPR-HBASE-236: CVE-2020-15250 vulnerability in JUnit.
04854c357b	2021-06-04	MAPR-HBASE-208: HBase build should use only internal repositories / mirrors.

Known Issues and Limitations

This release contains the following known issues and limitations:

- ASYNC-5: Data-fabric-SASL security is not implemented in AsyncHBase. As a result, an AsyncHBase client connection fails on a secure data-fabric cluster. **Workaround:** Implement AsyncHBase security or disable HBase security.
- Data-fabric client nodes cannot be configured for HBase by using the `configure.sh` script. **Workaround:** To configure an HBase client node, you can use the `configure_client.sh` script. For more information, see Configuring HBase on a Client Node.
- MD-5925: Drill versions 1.11.0 through 1.16.0.x do not support querying HBase tables. **Workaround:** None.
- ASYNC-9: An error is generated when AsyncHBase is used to put data to a cluster with HBase and Kerberos configured. The error occurs when `hbase.rpc.protection` is set to privacy or integrity in the AsyncHbase client configuration. **Workaround:** To avoid the error, set `hbase.rpc.protection` to `authentication`.

HBase 1.1.13.300-2104 (EEP 6.3.4) Release Notes

The notes below relate to the MapR Data Platform distribution of Apache HBase.

last-updated: Apr 11, 2022

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. You may also be interested in Ecosystem Component Release Notes on page 5658 and the Apache HBase homepage.

Version	1.1.13.300
Release Date	April 2021
MapR Version Interoperability	See Interoperability Matrix, Ecosystem Support Matrix, and HBase Support Matrix
Source on GitHub	https://github.com/mapr/hbase
GitHub Release Tag	1.1.13.300-mapr-634
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

HBase 1.1.13.300 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- CVEs fixes.
- Extra loggings.
- Bug fixes.

Fixes

This HPE release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description
442bf075d7	2021-04-27	MAPR-HBASE-233: Couldn't get into HBase shell on client node.
c4b124d200	2021-03-29	MAPR-HBASE-217: HBase data loss due to RS crash, extra-logging added for investigation.
af92293fe5	2021-03-22	MAPR-HBASE-167: Upgrade Jackson-Databind (transitive from HTrace) packaged with MapR-HBase.
2b694bbdd2	2021-03-17	MAPR-HBASE-225: Opentsdb daemon process does not start after setting noexec to /tmp dir in MapR 6.1/MEP-6.2.
6539a26268	2021-03-16	MAPR-HBASE-228: HBase mapreduce jobs failed with exitCode=1 due to shaded Guava (partially reverting MAPR-HBASE-192).
00b6a7cdfc	2021-03-12	MAPR-HBASE-223: CVE: Security vulnerabilities in bootstrap.js (cherry picking HBASE-24971: updating JQuery).

0290b06632	2021-03-10	MAPR-HBASE-224: HBase Rest threads go in loop while processing TLSv1.3 requests.
a832859220	2021-03-04	MAPR-HBASE-216: HBase process should start with ErrorFile option.
b356b53bc2	2021-03-02	MAPR-HBASE-223: CVE: security vulnerabilities in bootstrap.js (cherry picking HBASE-25079 + HBASE-25261).
c03d5db9b5	2021-02-11	MAPR-HBASE-215: CVE-2014-3488,CVE-2019-16869 etc. Netty vulnerabilities.
e0ad1f0ea1	2021-02-09	MAPR-HBASE-214: Vulnerabilities in Tomcat.
2960be4671	2021-02-05	MAPR-HBASE-207: HBase Web UIs display incorrectly when encryption is disabled on secure cluster.

Known Issues and Limitations

This release contains the following known issues and limitations:

- ASYNC-5: Data-fabric-SASL security is not implemented in AsyncHBase. As a result, an AsyncHBase client connection fails on a secure data-fabric cluster. Workaround: Implement AsyncHBase security or disable HBase security.
- Data-fabric client nodes cannot be configured for HBase by using the `configure.sh` script. Workaround: To configure an HBase client node, you can use the `configure_client.sh` script. For more information, see Configuring HBase on a Client Node.
- MD-5925: Drill versions 1.11.0 through 1.16.0.x do not support querying HBase tables. Workaround: None.
- ASYNC-9: An error is generated when AsyncHBase is used to put data to a cluster with HBase and Kerberos configured. The error occurs when `hbase.rpc.protection` is set to privacy or integrity in the AsyncHbase client configuration. Workaround: To avoid the error, set `hbase.rpc.protection` to authentication.

HBase 1.1.13.200-2101 (EEP 6.3.2) Release Notes

The notes below relate to the MapR Data Platform distribution of Apache HBase.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.1.13.200
Release Date	January 2021
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hbase
GitHub Release Tag	1.1.13.100-mapr-632
Maven Artifacts	https://repository.mapr.com/maven/

Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

This release contains CVE fixes.

Fixes

This HPE release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
d96811df1c	2021-01-08	MAPR-HBASE-206 Excessive logging from DynamicClassLoader
6061cf247f	2020-12-15	MAPR-HBASE-201 CVE-2016-6796: Tomcat / Jasper library vulnerability
35a6b58080	2020-12-11	MAPR-HBASE-200: Netty vulnerabilities
c590ff5446	2020-11-30	MAPR-HBASE-195 updating thrift version; from 12 to 13
c080f6b2e0	2020-10-02	MAPR-HBASE-192: Update Guava version to 28.2-jre with shading older version
259668771f	2020-09-28	MAPR-HBASE-192: Enable hbase-shaded packages
7cce08beaf	2020-09-22	MAPR-HBASE-192 Replacing guava Stopwatch with hadoop.util.StopWatch. it is deprecated guava version > v16
aa51877ced	2020-09-15	MAPR-HBASE-191 Retrieve max version records using spark hbase binary connector

Known Issues and Limitations

This release contains the following known issues and limitations:

- ASYNC-5: Data-fabric-SASL security is not implemented in AsyncHBase. As a result, an AsyncHBase client connection fails on a secure data-fabric cluster. Workaround: Implement AsyncHBase security or disable HBase security.
- Data-faberic client nodes cannot be configured for HBase by using the `configure.sh` script. Workaround: To configure an HBase client node, you can use the `configure_client.sh` script. For more information, see Configuring HBase on a Client Node.
- MD-5925: Drill versions 1.11.0 through 1.16.0.x do not support querying HBase tables. Workaround: None.
- ASYNC-9: An error is generated when AsyncHBase is used to put data to a cluster with HBase and Kerberos configured. The error occurs when `hbase.rpc.protection` is set to privacy or integrity in the AsyncHbase client configuration. Workaround: To avoid the error, set `hbase.rpc.protection` to authentication.

HBase 1.1.13.100-2009 (EEP 6.3.1) Release Notes

last-updated: Apr 11, 2022

The notes below relate to the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.1.13.100
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hbase
GitHub Release Tag	1.1.13.100-mapr-631
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

- CVEs fixes

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
57ced46e	2020-03-13	MAPR-HBASE-147: HBase configure.sh is overwriting properties set by users
b9a5f219	2020-03-17	MAPR-HBASE-141: Disable TLSv1 and TLSv1.1 protocol for Hbase thrift server
a00b2126	2020-04-03	MAPR-HBASE-151: Bad exception message
989f01d5	2020-04-13	MAPR-HBASE-161: Disable TRACE HTTP method for thrift http server
aae417bf	2020-04-23	MAPR-HBASE-157: Fix io.netty vulnerability
0386e825	2020-04-24	MAPR-HBASE-154: Unable to start hbase shell with MapR ticket in case Kerberos is enables in the cluster
44ad7460	2020-04-30	MAPR-HBASE-166: hbase hbck command incorrectly uses HBASE_REGIONSERVER_OPTS environment variable
9b25a4c4	2020-05-07	MAPR-HBASE-156: Fix org.codehaus.jackson vulnerability
07dca2fd	2020-05-26	MAPR-HBASE-168: HBase web pages don't look right
7e9bf8d0	2020-06-01	MAPR-HBASE-150 add JMX option handling

last-updated: Apr 11, 2022

dd838c9b	2020-07-08	MAPR-HBASE-176: ResourceConfig could not be instantiated
e85c9857	2020-09-01	MAPR-HBASE-174: Hbase services fails with NPE when Credential Provider is configured

Known Issues and Limitations

This release contains the following known issues and limitations:

- ASYNC-5: MapR-SASL security is not implemented in AsyncHBase. As a result, an AsyncHBase client connection fails on a secure MapR cluster. Workaround: Implement AsyncHBase security or disable HBase security.
- MapR client nodes cannot be configured for HBase by using the configure.sh script. Workaround: To configure an HBase client node, you can use the configure_client.sh script. For more information, see Configuring HBase on a Client Node.
- MD-5925: Drill versions 1.11.0 through 1.16.0.x do not support querying HBase tables. Workaround: None.
- ASYNC-9: An error is generated when AsyncHBase is used to put data to a cluster with HBase and Kerberos configured. The error occurs when hbase.rpc.protection is set to privacy or integrity in the AsyncHbase client configuration. Workaround: To avoid the error, set hbase.rpc.protection to authentication.

HBase 1.1.8-2101 (EEP 5.0.6) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache HBase.

Version	1.1.8
Release Date	January 2021
HPE Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hbase
GitHub Release Tag	1.1.8-mapr-2101
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

HBase 1.1.8-2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
e276401226	2021-01-12	MAPR-HBASE-197: port locking of hbase rest and thrift services UI

last-updated: Apr 11, 2022

312b45d089	2020-09-15	MAPR-HBASE-191 Retrieve max version records using spark hbase binary connector

For details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

HBase 1.1.8-2009 (EEP 5.0.5) Release Notes

The notes below relate to the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.1.8
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hbase
GitHub Release Tag	1.1.8-mapr-2009
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

- None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
6194878c	2020-03-17	MAPR-HBASE-141: Disable TLSv1 and TLSv1.1 protocol for Hbase thrift server
aaee6032	2020-04-15	MAPR-HBASE-161: Disable TRACE HTTP method for thrift http server
57f5fa3d	2020-05-28	MAPR-HBASE-150: add JMX option handling

Known Issues and Limitations

None.

HBase 1.1.13.0-1912 (EEP 6.3.0) Release Notes

The notes below relate to the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.1.13.0
Release Date	December 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hbase
GitHub Release Tag	1.1.13.0-mapr-630
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

- Added HBase Master and RegionServer services.
- Added MaprSasl support for HBase Master, RegionServer, REST, and Thrift.
- Added configurable HTTP Security Headers.
- Added the use of HTTP principal for Kerberos authentication for HBase Thrift.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
ac9f771e	2019-01-11	HBASE-24: The key store and trust store password for hbase-rest and hbase-thrift should be taken from ssl-client.xml
39be41fc	2019-01-23	HBASE-36: Implement configure.sh script for HBase. Warden files were added, existing were changed. Configure.sh logic was changed.
f2ad71cd	2018-12-13	HBASE-47: Enable authentication for HBase
8b541603	2018-11-18	[SPARK-343] Mapr-db binary connector for Spark does not match predicate with binary keys
43a979f1	2018-12-13	MAPR-15021: HBase package shouldn't contain maprfs jar
713b2026	2019-03-18	MAPR-HBASE-48: HBase: Encrypt RPC connection

ff05a227	2019-01-04	MAPR-HBASE-72: Add opportunity to use security parameters to hbase-rest-1.1.13-mapr-1710-sources.jar!/org/apache/hadoop/hbase/rest/client/Client
8018fae0	2019-01-04	MAPR-HBASE-68: HBase: Enable security for HBase Thrift
a6c11486	2019-01-18	MAPR-HBASE-78: Make HBase Info Web UIs secure
b66dc1d4	2019-01-23	MAPR-HBASE-45: Monitoring: HBase should be monitored by Spyglass
c345baa4	2019-01-25	MAPR-HBASE-88: Endpoint /status/cluster returns list tables instead cluster status
e751182c	2019-02-25	MAPR-HBASE-75: Hbase importtsv mapreduce job failed with 'Try to get a MapR table object from a non-MapR connection'
3a5d84f5	2019-02-25	MAPR-HBASE-97: HBase not working on kerberos cluster
0f9393f1	2019-02-26	MAPR-HBASE-80: CVE-2018-1320 vulnerability in Apache Thrift
bda6a8db	2019-04-04	MAPR-HBASE-107: Change default zookeeper port for hbase
11686313	2019-06-06	MAPR-HBASE-115: mapr.hbase.default.db is changed to default after each run of configure.sh
b11dccd7	2019-06-12	MAPR-HBASE-116: Problem accessing /version/cluster at REST server
c1cb35ec	2019-07-10	MAPR-HBASE-117: Endpoint /status/cluster returns error 500
40fc9b8f	2019-07-17	MAPR-HBASE-119: There are no secure properties in the hbase-site.xml after upgrade from 1.1.8 to 1.1.13
e5c8f29b	2019-08-15	MAPR-HBASE-124: Skip delegation token generation for MaprDB connections
9fc34e70	2019-08-21	MAPR-HBASE-125: Incorrect parsing of kerberosEnable parameter in mapr-clusters.conf leads to errors while starting services in kerberos mode
8ab15ce0	2019-09-13	MAPR-HBASE-126: There is no PAM for HBase Web UIs on kerberos cluster
c0844912	2019-11-06	MAPR-HBASE-127: Add security headers for HBase webservices
606e4555	2019-10-31	MAPR-SPARK-632: Change spark version to 2.4.4.0 for MEP-6.3.0 in pom.xml

last-updated: Apr 11, 2022

7e59aa10	2019-11-07	MAPR-HBASE-130: Uninformative error when impersonation user does not have permissions
53052a5f 89b27e57	2019-11-15	MAPR-HBASE-132: '401 Client Error' when try to open HBase Browser in Hue on kerberos cluster
438db6a8	2019-11-29	CORE-347: fix handling of defaultdb
6f864c85	2019-11-28	HBASE-134: fix usage of not_configure_yet and allow multiple configure.sh runs

Known Issues and Limitations

This release contains the following known issues and limitations:

- ASYNC-5: MapR-SASL security is not implemented in AsyncHBase. As a result, an AsyncHBase client connection fails on a secure MapR cluster. **Workaround:** Implement AsyncHBase security, or disable HBase security.
- MapR client nodes cannot be configured for HBase by using the `configure.sh` script. **Workaround:** To configure an HBase client node, you can use the `configure_client.sh` script. For more information, see Configuring HBase on a Client Node on page 185.
- MD-5925: Drill versions 1.11.0 through 1.16.0.x do not support querying HBase tables. **Workaround:** None.
- ASYNC-9: An error is generated when AsyncHBase is used to put data to a cluster with HBase and Kerberos configured. The error occurs when `hbase.rpc.protection` is set to `privacy` or `integrity` in the AsyncHbase client configuration. **Workaround:** To avoid the error, set `hbase.rpc.protection` to `authentication`.

Resolved Issues

- None.

HBase 1.1.8-1904 Release Notes

The notes below relate to the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.1.8
Release Date	April 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hbase/tree/rel/1.1.8
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

- None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
a54c8b0	2019-02-28	HBASE-80: CVE-2018-1320 vulnerability in Apache Thrift

Known Issues and Limitations

- None.

Resolved Issues

- None.

HBase 1.1.8-1901 Release Notes

The notes below relate to the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.1.8
Release Date	February 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hbase/tree/rel/1.1.8
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

- None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
d3f2bff	2019-01-11	MAPR-HBASE-73: Throws java.lang.NoClassDefFoundError at creating table from Hive-2.1.1
1912dc2	2018-11-18	MAPR-SPARK-343: Mapr-db binary connector for Spark does not match predicate with binary keys

Known Issues and Limitations

None.

last-updated: Apr 11, 2022

Resolved Issues

None.

HBase 1.1.8-1808 Release Notes

The notes below relate to the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.1.8
Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hbase/tree/rel/1.1.8
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

- HBase Master and HBase Regionserver are removed for MapR 6.0.0 and EEP 4.0.0. The Thrift and REST server packages are still available to support MapR Database binary tables.
- MapR-SASL authentication, encryption, and impersonation for HBase Thrift Gateway are enabled by default on secure clusters.
- On a secure cluster, by default, HBase REST and HBase Thrift read the `ssl-client.xml` file and configure SSL using this file.
- PAM authentication, encryption, and impersonation for HBase REST are enabled by default on secure clusters.

Note: HBase Master and HBase Regionserver are still available for previous EEP releases.

Resolved Issues

This MapR release includes the following bug fixes:

Bug	Description
Bug 28640	Updated Jersey version to 1.19.4 in the MapR Spark connector. In addition, fixed the build failure for licenses and added a Jersey servlet dependency in the hbase-server/pom.xml file
MAPR-HBASE-20	Changed permissions to hbase-site.xml
MapR [Spark-245]	Fixed Spark integration with MapR Database binary on MapR-SASL
MAPR-HBASE-22	Adding Impersonation configuration for HBase REST (enabled by default on secure clusters)
MAPR-HBASE-24	The key store and trust store password for hbase-rest and hbase-thrift are taken from ssl-client.xml
MAPR-HBASE-25	Updated zookeeper version in HBase from 3.4.6 to 3.4.11

Bug	Description
MAPR-HBASE-31	Restart of HBase Rest and HBase Thrift services during configure.sh
MAPR-HBASE-2	Enabled secure by default for HBase Rest/Thrift Gateway

HBase 1.1.8-1710 Release Notes

The notes below relate to the MapR Converged Data Platform.

Version	1.1.8
Release Date	November 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hbase
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

HBase Master and HBase Regionserver are removed for MapR 6.0/MEP 4.0.0. The Thrift and REST server packages are still available to support MapR Database binary tables.

Note: HBase Master and HBase Regionserver are still available for previous MapR/MEP releases.

Resolved Issues

This MapR release includes the following bug fixes:

Bug	Description
Bug 28640	Updated Jersey version to 1.19.4 in the MapR Spark connector. In addition, fixed the build failure for licenses and added a Jersey servlet dependency in the hbase-server/pom.xml file.

HBase 1.1.8-1703 Release Notes

The notes below relate to the MapR Converged Data Platform.

Version	1.1.8
Release Date	April 2017
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/hbase
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

Resolved Issues

This MapR release includes the following bug fix.

Bug	Description
25241	In MapR Database, when using the HBase Java FuzzyRowFilter filter, the wrong result is returned due to incorrect mask preprocessing.

HBase 1.1-1602 Release Notes

The notes below relate to the MapR Converged Data Platform. You may also be interested in the Apache HBase release notes for HBase 1.1.0.

Version	1.1
Release Date	February 29, 2016
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/hbase
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

This is the initial MapR release of HBase 1.1. In addition to HBase 1.1 features, this release includes the following feature that is specific to MapR:

- **Default Database Setting.**

  When you run configure.sh, a default database for HBase clients is automatically set based on the packages available on the cluster. When mapr-hbase-regionserver or mapr-hbase-master packages are installed on the cluster, HBase is the default database for all HBase client connections. You can use the configure.sh -defaultdb option to change the default database setting for the cluster or you specify a value to override the default in the HBase job configuration or the hbase-site.xml. For more information, see MapR's HBase and confgure.sh documentation.

- **Default Ports**

  The following ports for HBase 1.1 are changed:

Table

Service	Port
HBase Master	16000
HBase Master (for GUI)	16010
HBase RegionServer	16020
HBase RegionServer (for GUI)	16030
HBase Status MultiCast	16100

Note: If you need to use the previous default ports from HBase 0.98.x, change the `hbase-site.xml` by adding the following properties:

```
<property>
        <name>hbase.master.port</name>
        <value>60000</value>
</property>
<property>
        <name>hbase.master.info.port</name>
        <value>60010</value>
</property>
<property>
        <name>hbase.regionserver.port</name>
        <value>60020</value>
</property>
<property>
        <name>hbase.regionserver.info.port</name>
        <value>60030</value>
</property>
```

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
8cc858f	2016-01-28	Backported HBASE-14799 to fix potential security vulnerability in the commons-collections library when you accept and process Java object serialization data.
ed43aca	2015-11-11	**MAPR 13750:** fs.mapr.readbuffering and fs.mapr.aggregate.writes are now disabled for HBase RegionServers.

HBase 0.98.12.1-1605 Release Notes

You might also be interested in the Apache HBase release notes for v0.98.12.1.

Version	0.98.12.1
Release Date	June 6, 2016
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and the HBase Support Matrix on page 5630.
Source on GitHub	https://github.com/mapr/hbase
Maven Artifacts	https://repository.mapr.com/maven/

	The following packages are associated with this release: • mapr-hbase-0.98.12.201606022139-1.noarch.rpm • mapr-hbase_0.98.12.201606022122_all.deb • mapr-hbase-internal-0.98.12.201606022139-1.noarch.rpm • mapr-hbase-internal_0.98.12.201606022122_all.deb • mapr-hbase-master-0.98.12.201606022139-1.noarch.rpm • mapr-hbase-master_0.98.12.201606022122_all.deb • mapr-hbase-regionserver-0.98.12.201606022139-1.noarch.rpm • mapr-hbase-regionserver_0.98.12.201606022122_all.deb • mapr-hbase-rest-0.98.12.201606022139-1.noarch.rpm • mapr-hbase-rest_0.98.12.201606022122_all.deb • mapr-hbasethrift-0.98.12.201606022139-1.noarch.rpm • mapr-hbasethrift_0.98.12.201606022122_all.deb

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
df28a4d	2016-02-18	**MAPR 22664:** The script `hbase-daemon.sh` is now fixed to initialize the variable `loglog`, allowing the volume `mapr.hbase` to be created and preventing performance problems due to tables being created in the root volume.
036ae59	2016-05-19	**MAPR 23035:** REST clients sending scan requests with accept types containing "application/x-protobuf" or "application/protobuf" no longer receive IllegalArgumentException errors from the HBase server.

HBase 0.98.12.1-1602 Release Notes

You may also be interested in the Apache HBase release notes for v0.98.12.1.

Version	0.98.12.1
Release Date	March 2, 2016
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and the HBase Support Matrix on page 5630.

last-updated: Apr 11, 2022

Source on GitHub	https://github.com/mapr/hbase
Maven Artifacts	https://repository.mapr.com/maven/
	The following packages are associated with this release: • mapr-hbase-0.98.12.201603011130-1.noarch.rpm • mapr-hbase_0.98.12.201603011120_all.deb • mapr-hbase-internal-0.98.12.201603011130-1.noarch.rpm • mapr-hbase-internal_0.98.12.201603011120_all.deb • mapr-hbase-master-0.98.12.201603011130-1.noarch.rpm • mapr-hbase-master_0.98.12.201603011120_all.deb • mapr-hbase-regionserver-0.98.12.201603011130-1.noarch.rpm • mapr-hbase-regionserver_0.98.12.201603011120_all.deb • mapr-hbase-rest-0.98.12.201603011130-1.noarch.rpm • mapr-hbase-rest_0.98.12.201603011120_all.deb • mapr-hbasethrift-0.98.12.201603011130-1.noarch.rpm • mapr-hbasethrift_0.98.12.201603011120_all.deb

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
8d901d8	2016-01-28	Backported HBASE-14799 to fix potential security vulnerability in the commons-collections library when you accept and process Java object serialization data.

HBase 0.98.12.1-1506 Release Notes

You may also be interested in the Apache HBase release notes for v0.98.12.1.

Version	0.98.12.1
Release Date	July 10, 2015
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/hbase
GitHub Release Tag	0.98.12-mapr-1506
Maven Artifacts	https://repository.mapr.com/maven/

last-updated: Apr 11, 2022

New in This Release

This is the initial release of HBase 0.98.12.1.

When HBase 0.98.12.1 runs on MapR version 5.0, it includes the following updates for MapR Database:

- HTable.checkAndMutate() API is supported by MapR Database
- Truncate API will retain ACEs for MapR Database binary tables

For more information about MapR Database API compatibility, refer to MapR Database documentation for *Creating MapR Database Applications with Java*.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
f977386	2015-06-19	**MAPR-19053:** Bulk importing HFiles from within the hbase volume now moves the file instead of making a copy during the incremental load phase.

HBase 0.98.9-1503 Release Notes

You may also be interested in the Apache HBase release notes for v0.98.9.

Version	0.98.9-mapr-1503
Release Date	March 27, 2015
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/hbase
GitHub Release Tag	0.98.9-mapr-1503
Maven Artifacts	https://repository.mapr.com/maven/

New in This Release

- C APIs for HBase are supported by MapR Database in MapR version 4.1.
- HBase Rest Gateway Server can be deployed as a pluggable service on MapR version 4.0.1 or higher.
- HBase 0.98.9 introduces the HBaseAdmin.truncateTable() API. In this MapR release of HBase, the HBaseAdmin.truncateTable() API is also supported for MapR Database tables with MapR version 4.0.1 or higher.
- The performance test utility for libhbase includes support for Zipfian and uniform random key generation. It also runs a performance test for scans.

Note: The HTable.checkAndMutate() API is not supported by MapR Database.

Apply the latest 4.x EBF patch if you want to use HBase replication. Contact Customer Support for details.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
7f86d82	2015-01-28	MAPR-16493: ImportFiles command no longer fails with the following error: `java.lang.RuntimeException: java.lang.ClassNotFoundException: Class org.apache.hadoop.hbase.mapreduce.HFileOutputFormat`
3ecea99	2015-01-29	MAPR-16948: To avoid data loss, speculative execution is now disabled for CopyTable jobs.
a1d4d58	2015-02-03	MAPR-17043: To increase performance, PutCombiner is disabled for ImportTsv jobs.
cff5b9a	2015-02-04	MAPR-17057: list_perm command no longer fails with the following error: `ERROR: Security features for MapR tables are not available in this version.`
51a3973	2015-02-10	MAPR-17122: CopyTable does not fail for MapR Database tables.
dd61dc4	2015-02-10	MAPR-16750: CopyTable supports bulkload mode for MapR Database tables. However, the user must reset the the bulkload attribute on the MapR Database table after running the CopyTable utility.
e00abd0	2015-02-10	HBASE-13009: The HBase REST UI is accessible.

Hive Release Notes

The release notes for the Hive component included in the MapR Data Platform contain notes specific to data-fabric only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536 or EEP Support and Lifecycle Status on page 5531. To view release notes for prior data-fabric releases, see Previous Versions on page 6578.

Hive 2.3.9 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.3.9.

The following release notes for the Hive 2.3.9 component are included in the MapR Data Platform distribution for Apache Hadoop:

Hive 2.3.9.0 - 2201 (EEP 8.1.0) Release Notes

The following notes relate specifically to the MapR Data Platform Distribution for Apache Hive. You may also be interested in the Apache Hive-2.3.9 Release Notes and the Apache Hive homepage.

Hive Version	2.3.9.0

last-updated: Apr 11, 2022

Release Date	January 2022
HPE Version Interoperability	See Hive and HCatalog Support Matrix and Ecosystem Support Matrix and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.3.9.0-eep-810
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to http://package.mapr.hpe.com/releases/MEP/, and select your EEP(MEP) and OS to view the list of package names.

ODBC/JDBC Drivers	Hive 2.3.9 works with the following MapR Hive drivers:
	• ODBC Drivers
	  • Mac OS X
	  • Linux
	  • 32-bit
	  • 64-bit
	  • Windows
	  • 32-bit
	  • 64-bit
	For additional driver information, see Connecting to HiveServer2.

Feature support

The following list describes support of various components and functionality with Hive 2.3.9 - 2201:

- Supports Hive-2.3.9 on Tez-0.9.2 For more information, see Tez 0.9.2 - 2201 (EEP 8.1.0) Release Notes on page 6481.
- Does not support Hive on Spark. You cannot use Spark as a query engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- Does not support HDFS encryption in Hive tables.
- Does not support LLAP with Hive-2.3.9 because Apache Slider is not an MapR supported ecosystem component.
- Starting from Hive 2.1, Hive must run the `schematool` command as an initialization step.

Changes in default security configuration

The following list describes changes in default security for Hive 2.3.9 - 2201:

- Starting from EEP-8.1.0, Hive supports SCRAM token and SCRAM-SHA-256 authentication in MapR Data Platform.

Table

#	Property	Data Type	Default value	Description

Table (Continued)

1	hive.delegation.token.authentication	String	DIGEST	Delegation token authentication method. Possible values are DIGEST, SCRAM

To configure SCRAM token and SCRAM-SHA-256 authentication, set the following property on `HIVE_HOME/conf/hive-site.xml` file:

```
<property>
    <name>hive.delegation.token.authentication</name>
    <value>SCRAM</value>
</property>
```

Execute `MAPR_HOME/server/configure.sh -R` script on a newly installed and Data-Fabric SASL or KERBEROS secured cluster to automatically configure the following authentications:

1. For a FIPS enabled cluster, Hive configures `hive.delegation.token.authentication=SCRAM` authentication.
2. For a non-FIPS cluster if you configure Hadoop with `hadoop.security.token.authentication.method=SCRAM` authentication, Hive configures the SCRAM authentication.
3. For other clusters, Hive configures `hive.delegation.token.authentication=DIGEST` authentication.

For non-secure clusters, Hive configures `hive.delegation.token.authentication=DIGEST` authentication.

When you upgrade Hive, the upgrade does not update the value of the set `hive.delegation.token.authentication` property.

Manually set the value of `hive.delegation.token.authentication` property when you change the cluster settings from FIPS to non-FIPS or from non-FIPS to FIPS.

New in This Release

Hive 2.3.9.0 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

- Starting from EEP-8.1.0, Hive supports FIPS and SCRAM SASL.
- Beginning with EEP 8.1.0, JAR artifacts for Hive use four digits instead of three digits. For example:

```
hive-service-rpc-2.3.9.0-mapr-SNAPSHOT.jar
hive-llap-ext-client-2.3.9.0-mapr-SNAPSHOT.jar
hive-exec-2.3.9.0-mapr-SNAPSHOT.jar
hive-beeline-2.3.9.0-mapr-SNAPSHOT.jar
```

If your application includes a Hive dependency in the `pom.xml` file, you must update the JAR artifact before using Hive 2.3.9 - 2201.

The Hive package name on package.mapr.hpe.com continues to use two digits, and the Hive root folder continues to use three digits.

last-updated: Apr 11, 2022

- Improved `Describe table` operator in terms of fetching statistics of partitions. Starting from EEP 8.0.0, you can fetch the partition information using the `describe` command with `formatted` or `extended` statements.

  Configure the `hive.describe.partitionedtable.ignore.stats` property to change the behaviour of fetching statistics of partitions. It is set to the default value of `false`.

```
<property>
    <name>hive.describe.partitionedtable.ignore.stats</name>
    <value>false</value>
    <description>Enables partitioned table stats collection for 'DESCRIBE
FORMATTED' or 'DESCRIBE EXTENDED' commands</description>
</property>
```

```
<property>
    <name>hive.describe.partitionedtable.ignore.stats</name>
    <value>true</value>
    <description>Disables partitioned table stats collection for
'DESCRIBE FORMATTED' or 'DESCRIBE EXTENDED' commands</description>
 </property
```

- Hive supports symbolic links on MapR File System.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
a6d17888e2	2022-01-31	EEP-HIVE-1153 : FAILURE! - in org.apache.hive.conftool.SslDefaultTest
5d4dcca0d9	2022-01-31	EEP-HIVE-1151 : Build Hive ECO EEP 8.1.0 components with DF v 6.2.0
be5e7ce3a7	2022-01-27	EEP-HIVE-1148 : Cannot run HiveMetastore service on non-fips cluster for EEP-8.1.0
bc913e56b1	2022-01-27	EEP-HIVE-1147 : JDBC connection failed if working with different connections on FIPS enabled clusters
9dc2d25f98	2022-01-24	EEP-HIVE-1141 : Hive services cannot communicate between FIPS / non-FIPS nodes
e527acd2dd	2022-01-18	EEP-HIVE-1135: HiveServer2 will fail if default provider will not support FIPS
3498c725a6	2022-01-06	EEP-HIVE-1065: CVE-2021-37136, CVE-2021-37137, WS-2020-0408, CVE-2021-21290: netty-*-4.1.55.Final.jar
e82cca2f60	2022-01-05	EEP-HIVE-1062: CVE-2016-5007, CVE-2016-9878 ,CVE-2018-1271, CVE-2018-1272, CVE-2020-5421: spring-*-3.2.16.RELEASE.jar

e2ae1f021e	2022-01-05	EEP-HIVE-1068: CVE-2020-9480: spark-network-common_2.11-2.3.0.jar, CVE-2018-17190: spark-core_2.11-2.3.0.jar
110f14340f	2022-01-04	EEP-HIVE-1117: Update log4j v2 to the latest available (to 2.17+)
5cd9fcc2d9	2022-01-04	EEP-HIVE-1116: Hive returns an incorrect number of columns
ae3e3d49e8	2022-01-04	Revert "MAPR-HIVE-930: Cannot run join with Order by and Limit clause specified at the same time"
8b01f203da	2021-12-30	EEP-HIVE-1119: com.fasterxml.jackson.annotation.JsonFormat.empty()Lcom/fasterxml/jackson/annotation/JsonFormat
250a524cbe	2021-12-27	EEP-HIVE-1099 : [FIPS] HS2 connection PAM + SSL doesn't work with SBD configuration when FIPS enabled
da9feb7041	2021-12-24	EEP-HIVE-1064: CVE-2021-30639 ; CVE-2021-33037: tomcat-coyote-10.0.4.jar
3288fbbea0	2021-12-24	EEP-HIVE-1059: CVE-2019-10172, CVE-2019-10202: jackson-mapper-asl-1.9.13.jar, jackson-mapper-asl-1.9.2.jar
70be9aeb2c	2021-12-24	EEP-HIVE-1056: CVE-2021-35515, CVE-2021-35516, CVE-2021-35517, CVE-2021-36090: commons-compress-1.20.jar
2a7a346e3e	2021-12-24	EEP-HIVE-1055: CVE fixes of bcprov-jdk15on-1.52.jar
2e9596dc8a	2021-12-24	EEP-HIVE-1054: WS-2021-0419: gson-2.2.4.jar
63a06a47d7	2021-12-22	EEP-HIVE-1115 : Unrecognized VM option UseGCLogFileRotation
3dd5c4f710	2021-12-21	EEP-HIVE-1088: [Hive-2.3.9] Hive on Tez engine + native S3/OPAL Unable to load AWS credentials from any provider in the chain
6e06d411a3	2021-12-21	EEP-HIVE-1097: CVE-2021-44228 - Log4j vulnerability
facbb602ee	2021-12-21	EEP-HIVE-1091 : Add SCRAM-SASL to Hive
158d663d48	2021-12-07	EEP-HIVE-1087: CAST gives NULL values during insert when vectorization enabled.
1e74e4cccb	2021-12-02	MAPR-HIVE-1090 : TLSv1.2 SSLContext not available
904758aef6	2021-11-25	MAPR-HIVE-1086 : Upgrade Jetty to 9.4.44.v20210927

last-updated: Apr 11, 2022

04c60792f4	2021-11-24	MAPR-HIVE-1074 : Beeline fails to connect to HiveServer2 with java.lang.NoSuchFieldError: BCFKS error on Core 6.2.0/EEP 8.1.0
2851a75533	2021-11-22	MAPR-HIVE-1071 : Fix SslDefaultTest
c6e21ea94d	2021-11-22	MAPR-HIVE-1069 Update hbase version to 1.4.13.0-eep-810-SNAPSHOT
62c6ca2825	2021-11-18	MAPR-HIVE-1053 : Relative path in absolute URI: slider reads hdfs-site.xml from hadoop-hdfs.jar
35f8178503	2021-11-18	MAPR-HIVE-1036 : java.lang.NoClassDefFoundError: org/apache/commons/digester/Digester
e7b42cb8b1	2021-11-16	MAPR-HIVE-1026 : Migrate to python 3 in LLAP package.py file
b925b5c99d	2021-11-16	MAPR-HIVE-1025 : LLAP server expects tez.tar.gz archive in MapR FS
7fa0949cde	2021-11-16	MAPR-HIVE-1033 : Add MapR slider dependency to Hive
b7cf317ce4	2021-11-11	MAPR-HIVE-1031 : logError: command not found if any error happens during configuring Hive
c2c8453683	2021-11-11	MAPR-HIVE-1024 : Replace deprecated AuthMethod.DIGEST with AuthMethod.TOKEN in HadoopThriftAuthBridge25Sasl
806f29ca93	2021-11-10	DFDEVOPS-2081 : Configure Jenkins job for hive-2.3.9 mep-8.1.0
4364d7d1c4	2021-11-10	MAPR-HIVE-1022 : Update Hive version to 2.3.9.0-eep-810-SNAPSHOT
3212a89673	2021-11-09	MAPR-HIVE-1021 : Upgrade mapr-core version to 7.0.0.0-mapr-SNAPSHOT
d300f64a22	2021-11-04	MAPR-HIVE-831 : Move Hive to 4 digits in jar artefacts
bca47dbbea	2021-11-02	MAPR-HIVE-975: Customer request to investigate temporary hive session files cleanup improvements
304849fc9a	2021-11-01	MAPR-HIVE-1018 : Update hadoop version to 2.7.6.200-eep-810-SNAPSHOT
7c8fe3cc80	2021-11-01	MAPR-HIVE-1012 : Hiveserver2 could not start on cluster with enabled FIPS
1ccc9d7b5a	2021-10-13	MAPR-HIVE-1002 : Hive-2.3 does not remove old compressed logs

c26ff7e273	2021-10-11	MAPR-HIVE-1016 : Update Conjars repository URL to secure
4bf0c7aadc	2021-10-11	MAPR-HIVE-1015 : Configure repositories for Jenkins job
69d55d6809	2021-10-11	MAPR-HIVE-997 : ConfigureShInsecureTest hangs up
92a25a685c	2021-10-11	MAPR-HIVE-1013 : Update calcite version to 1.10.0-eep
16c0ba4efd	2021-10-11	MAPR-HIVE-1014 : Could not transfer artifact org.pentaho:pentaho-aggdesigner-algorithm:jar:5.1.5-jhyde

This release from HPE also includes the following back-ported issues. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
95ae2bebc9	2021-12-21	HIVE-17774: compaction may start with 0 splits and fail
ed4b16c0fa	2021-10-08	HIVE-16820 : TezTask may not shut down correctly before submit (Sergey Shelukhin, reviewed by Siddharth Seth)
794c971152	2021-10-28	HIVE-20072 : Write access being requested when performing select on a table

Known Issues and Limitations

- HIVE-1089: Hive on MapReduce engine does not support data insertion into Versioned buckets.You must use Unversioned buckets on MaprReduce engine in S3 file system.

  MapR Technologies recommends using Hive on Tez engine for full S3 file system support.

- HIVE-19502: Unable to insert values into table stored by JdbcStorageHandler
- HIVE-19286: NPE in MERGE operator on MR mode

- HIVE-760: [Hive-2.3] Could not start hive-metastore on Centos 8 MetaException(message:Version information not found in metastore)

  Starting in EEP 7.0.0, use the MySQL driver with MariaDB.

```
<property>
    <name>javax.jdo.option.ConnectionURL</name>
    <value>jdbc:mysql://localhost:3306/hive?
createDatabaseIfNotExist=true</value>
 </property>
 <property>
    <name>javax.jdo.option.ConnectionDriverName</name>
    <value>com.mysql.jdbc.Driver</value>
    <description>Driver class name for a JDBC metastore</description>
 </property>
```

  Some SELECT queries can be converted to a single FETCH task minimizing latency. Currently, the query should be single sourced and should not have a subquery or any aggregations or distincts (which incurs RS), lateral views and joins:

```
none : disable hive.fetch.task.conversion
minimal : SELECT star, filter on partition columns, LIMIT only
more :  SELECT, filter, LIMIT only (support TABLESAMPLE and virtual
columns)
```

Resolved Issues

- MAPR-TEZ-172 fixes a HIVE-789 known issue from Hive 2.3.8 - 2104 (EEP 7.1.0) Release Notes on page 5897 in this release.

Hive 2.3.9 - 2110 (EEP 8.0.0) Release Notes

The following notes relate specifically to the MapR Data Platform Distribution for Apache Hive. You may also be interested in the Apache Hive-2.3.9 Release Notes and the Apache Hive homepage.

Hive Version	2.3.9
Release Date	October 2021
HPE Version Interoperability	See Hive and HCatalog Support Matrix and Ecosystem Support Matrix and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.3.9-eep-2110
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to http://package.mapr.hpe.com/releases/MEP/, and select your EEP(MEP) and OS to view the list of package names.

ODBC/JDBC Drivers	Hive 2.3.9 works with the following MapR Hive drivers:
	• ODBC Drivers
	  • Mac OS X
	  • Linux
	  • 32-bit
	  • 64-bit
	  • Windows
	  • 32-bit
	  • 64-bit
	For additional driver information, see Connecting to HiveServer2.

Feature support

The following list describes support of various components and functionality with Hive 2.3.9 - 2110:

- Supports Hive-2.3.9 on Tez-0.9.2 For more information, see Tez 0.9.2 - 2110 (EEP 8.0.0) Release Notes on page 6482.
- Does not support Hive on Spark. You cannot use Spark as a query engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- Does not support HDFS encryption in Hive tables.
- Does not support LLAP with Hive-2.3.9 because Apache Slider is not an MapR supported ecosystem component.
- Starting from Hive 2.1, Hive must run the `schematool` command as an initialization step.

Changes in default security configuration

The following list describes changes in default security for Hive 2.3.9 - 2110:

- None.

New in This Release

Hive 2.3.9 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

last-updated: Apr 11, 2022

- Improved `Describe table` operator in terms of fetching statistics of partitions. Starting from EEP 8.0.0, you can fetch the partition information using the `describe` command with `formatted` or `extended` statements.

  Configure the `hive.describe.partitionedtable.ignore.stats` property to change the behaviour of fetching statistics of partitions. It is set to the default value of `false`.

  ```
  <property>
      <name>hive.describe.partitionedtable.ignore.stats</name>
      <value>false</value>
      <description>Enables partitioned table stats collection for 'DESCRIBE
  FORMATTED' or 'DESCRIBE EXTENDED' commands</description>
  </property>
  ```

  ```
  <property>
      <name>hive.describe.partitionedtable.ignore.stats</name>
      <value>true</value>
      <description>Disables partitioned table stats collection for
  'DESCRIBE FORMATTED' or 'DESCRIBE EXTENDED' commands</description>
   </property
  ```

- Hive supports symbolic links on MapR File System.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
2549e5350a	2021-09-17	MAPR-HIVE-994: Non mapr user unable to read SSL configuration from XML files on Core 7.0
e28a1edd63	2021-09-14	MAPR-HIVE-1007: Permission denied to hbase temp files while running hcat jobs from other user
c2864e3d06	2021-09-08	MAPR-HIVE-999 : Make mapr-db jar with provided scope
705bab5a12	2021-09-06	MAPR-HIVE-998 : Update htrace version to 4.2.0-eep-incubating
46862cf932	2021-09-03	MAPR-HIVE-990 : mapr-security-web jar should be taken from the cluster
c4aed1b675	2021-09-03	MAPR-HIVE-995 : Upadate pig vesion to 0.17.0.0-eep-SNAPSHOT
7e012fa650	2021-09-03	MAPR-HIVE-993 : Update hbase version to 1.4.13.0-eep-SNAPSHOT
1a886bbe78	2021-09-03	MAPR-HIVE-992 : Update tez version to 0.9.2.0-eep-SNAPSHOT
1439299a5c	2021-09-03	MAPR-HIVE-991 : Update hadoop version to 2.7.6.0-eep-800-SNAPSHOT
09724c6192	2021-09-03	MAPR-HIVE-987 : Update the maven artifact version strings to eep

6dbd5a3ac8	2021-08-25	MAPR-HIVE-981: [symlink functionality] Implement LOAD DATA INPATH functionality from symlinks with relative path
29c2c6f3ab	2021-08-18	MAPR-HIVE-979: [symlink functionality] cannot insert in the external table based on symlinked directory
1073cecb48	2021-08-17	MAPR-HIVE-977 : Downgrade jackson to v2.11.1 or to 2.11.3 to be consistent with core version
13ec2228e5	2021-08-17	MAPR-HIVE-976 : Update tez version from 0.9.2-mapr-SNAPSHOT to 0.9.2.0-mapr-SNAPSHOT for development artifacts
bfc754f1ca	2021-08-11	MAPR-HIVE-973 : FAILURE! - in org.apache.hadoop.hive.ql.lockmgr.TestDbTxnManager2
7add0d42af	2021-08-09	MAPR-HIVE-972 : Replace Apache htrace-4.2.0-incubating with 4.2.0-mapr-incubating dependency
ad1eeb8b20	2021-08-09	MAPR-HIVE-971 : Exclude htrace-3.1
bcddd3cc00	2021-08-09	MAPR-HIVE-969: Add possibility to run MR jobs against source files that are symlinks to original data
74302e7f0d	2021-08-09	MAPR-HIVE-970 : Update hadoop version to 2.7.6.0-mapr-720-SNAPSHOT
f6953ff0a8	2021-08-05	MAPR-HIVE-968: Add possibility to run TEZ jobs against source files that are symlinks to original data
8186a55978	2021-07-29	MAPR-HIVE-880: Add possibility to distinguish file/dir links during Hive DML/DDL operations
10779d1b56	2021-07-29	MAPR-HIVE-960 : CVE-2012-5783 vulnerability in commons-httpclient
0be9010773	2021-07-29	MAPR-HIVE-959 : Update derbyclient and derbynet to most feasible version
d0825bf854	2021-07-29	MAPR-HIVE-963 : CVE-2020-13956,WS-2017-3734 vulnerabilities in httpclient
7982b4be51	2021-07-29	MAPR-HIVE-962 : WS-2019-0379: commons-codec vulnerability
d70a411074	2021-07-29	MAPR-HIVE-953 : FAILURE! - in org.apache.hive.hcatalog.templeton.TestCustomHeadersE2e
88056bb162	2021-07-29	MAPR-HIVE-952 : FAILURE! - in org.apache.hadoop.hive.ql.io.parquet.TestVectorizedColumnReader
054a0b73b2	2021-07-29	MAPR-HIVE-896 : CVE-2020-17521 vulnerability in Groovy

last-updated: Apr 11, 2022

5b44866042	2021-07-29	MAPR-HIVE-927 : NPE thrown from XmlUtil by Hive Client
2e446568b2	2021-07-29	MAPR-HIVE-858 : WARNING: Illegal reflective access org.apache.hive.com.esotericsoftware.kryo.serializers.FieldSerializer
b3520d3c85	2021-07-29	MAPR-HIVE-950 : HiveVersionInfo.getShortVersion returns wrong version
9b4c3ad944	2021-06-08	MAPR-HIVE-949 : Make SchemaEvolution class behavior the same as in Orc-1.5.12
9b75385d22	2021-06-06	MAPR-HIVE-894 : CVE-2020-13955 vulnerability in Calcite
2a7526fdea	2021-05-25	MAPR-HIVE-947 : org.apache.hadoop.hive.ql.exec.tez.TezTask at creating session
a64ce8195c	2021-05-23	MAPR-HIVE-945 : FAILURE! - in org.apache.hadoop.hive.maprdb.json.MapRDbJsonFetchByIdOptimizerPositiveTest
4215a9ab82	2021-05-23	MAPR-HIVE-944 : FAILURE! - in org.apache.hadoop.hive.hbase.TestHBaseSerDe
30655745a5	2021-05-23	MAPR-HIVE-943 : Fix org.apache.hadoop.hive.cli.TestCliDriverMethods
bfda9cc9a8	2021-05-23	MAPR-HIVE-942 : FAILURE! - in org.apache.hadoop.hive.accumulo.predicate.TestAccumuloPredicateHandler

This release from HPE also includes the following back-ported issues. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
3a7b6db040	2021-09-15	HIVE-24965: Describe table partition stats fetch should be configurable
1d1bb4c2fd	2021-09-15	HIVE-22453: Describe table unnecessarily fetches partitions
96b37ab69c	2021-09-14	HIVE-23756 : Added more constraints to the package.jdo file
40cdec9dd2	2021-09-12	HIVE-24177 : hive mapjoin throws udf class not found
606c3c240b	2021-09-08	HIVE-17659 : get_token thrift call fails for DBTokenStore in remote HMS mode (Vihang Karajgaonkar, reviewed by Aihua Xu)
a2416a115c	2021-09-06	HIVE-25054: Upgrade `jodd-core` dependency to get rid of CVE-2018-21234 (Abhay Chennagiri, reviewed by Jesus Camacho Rodriguez)

e593c8cb4e	2021-08-18	HIVE-17824 : msck repair table should drop the missing partitions from metastore (Janaki Lahorani, reviewed by Peter Vary, Alexander Kolbasov and Vihang Karajgaonkar)
3591ea65fa	2021-08-18	HIVE-16143: Improve msck repair batching (Vihang Karajgaonkar, reviewed by Sahil Takiar & Aihua Xu)
fab9a7603a	2021-07-29	HIVE-19228: Remove commons-httpclient 3.x usage (Janaki Lahorani reviewed by Aihua Xu)
0ffeae33b1	2021-06-13	HIVE-21200: Vectorization: date column throwing java.lang.UnsupportedOperationException for parquet (#2276)
c6300400bd	2021-06-13	HIVE-24608: Switch back to get_table in HMS client for Hive 2.3.x (#2080)
0518323174	2021-06-13	HIVE-18147 : Tests can fail with java.net.BindException: Address already in use (Janaki Lahorani, reviewed by Andrew Sherman and Vihang Karajgaonkar)
d6766f34fb	2021-06-13	HIVE-21563 : Improve Table#getEmptyTable performance by disable registerAllFunctionsOnce
a3477edb7f	2021-06-13	HIVE-24797: Disable validate default values when parsing Avro schemas (#1994)
1fc7585a2e	2021-06-08	ORC-437: Make acid schema checks case insensitive
9120da5c4f	2021-05-31	HIVE-21075 : Metastore: Drop partition performance downgrade with Postgres DB
39d42ddf12	2021-05-31	HIVE-9447: Metastore: inefficient Oracle query for removing unused column descriptors when add/drop table/partition (Selina Zhang reviewed by Ashutosh Chauhan, Adam Szita)
1ccb218119	2021-05-22	HIVE-21085: Materialized views registry starts non-external tez session (Jesus Camacho Rodriguez, reviewed by Ashutosh Chauhan)
dea7190511	2021-05-22	HIVE-19691: Start SessionState in materialized views registry (Jesus Camacho Rodriguez, reviewed by Ashutosh Chauhan)
09b4ca437f	2021-05-22	HIVE-17853: RetryingMetaStoreClient loses UGI impersonation-context when reconnecting after timeout (Chris Drome, reviewed by Mithun Radhakrishnan)

b8902a7bb8	2021-05-22	HIVE-23534: NPE in RetryingMetaStoreClient#invoke when catching MetaException with no message (Stamatis Zampetakis, reviewed by Jesus Camacho Rodriguez)
11db00d681	2021-05-22	HIVE-18494: Regression: from HIVE-18069, the metastore directsql is getting disabled (Jesus Camacho Rodriguez, reviewed by Gopal V)
1920988b66	2021-05-22	HIVE-18069: MetaStoreDirectSql to get tables has misplaced comma (Jesus Camacho Rodriguez, reviewed by Aihua Xu) (addendum)
e5ed2cb9ed	2021-05-22	HIVE-18069: MetaStoreDirectSql to get tables has misplaced comma (Jesus Camacho Rodriguez, reviewed by Aihua Xu)
9b506546a4	2021-05-22	HIVE-15436: Enhancing metastore APIs to retrieve only materialized views (Jesus Camacho Rodriguez, reviewed by Ashutosh Chauhan)
051002d23a	2021-05-22	HIVE-6990 : Direct SQL fails when the explicit schema setting is different from the default on (Bing Li, Sergey Shelukhin via Ashutosh Chauhan)

Known Issues and Limitations

- HIVE-19502 Unable to insert values into table stored by JdbcStorageHandler
- HIVE-19286 NPE in MERGE operator on MR mode
- HIVE-760 [Hive-2.3] Could not start hive-metastore on Centos 8 MetaException(message:Version information not found in metastore)

  Starting in MEP 7.0.0, use the MySQL driver with MariaDB.

```
<property>
    <name>javax.jdo.option.ConnectionURL</name>
    <value>jdbc:mysql://localhost:3306/hive?
createDatabaseIfNotExist=true</value>
  </property>
 <property>
    <name>javax.jdo.option.ConnectionDriverName</name>
    <value>com.mysql.jdbc.Driver</value>
    <description>Driver class name for a JDBC metastore</description>
 </property>
```

  Some SELECT queries can be converted to a single FETCH task minimizing latency. Currently, the query should be single sourced and should not have a subquery or any aggregations or distincts (which incurs RS), lateral views and joins:

```
none : disable hive.fetch.task.conversion
minimal : SELECT star, filter on partition columns, LIMIT only
more :  SELECT, filter, LIMIT only (support TABLESAMPLE and virtual
columns)
```

Resolved Issues

- MAPR-TEZ-172 fixes a HIVE-789 known issue from Hive 2.3.8 - 2104 (EEP 7.1.0) Release Notes on page 5897 in this release.

Hive 2.3.8 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.3.8.

The following release notes for the Hive 2.3.8 component are included in the MapR Data Platform distribution for Apache Hadoop:

Hive 2.3.8 - 2104 (EEP 7.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hive. You may also be interested in the Apache Hive-2.3.8 Release Notes and the Apache Hive homepage.

Hive Version	2.3.8
Release Date	April 2021
HPE Version Interoperability	See Hive and HCatalog Support Matrix andEcosystem Support Matrix and MEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.3.8-mapr-2104
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate tohttp://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

ODBC/JDBC Drivers	Hive 2.3.8 works with the following MapR Hive drivers:
	• ODBC Drivers
	  • Mac OS X
	  • Linux
	  • 32-bit
	  • 64-bit
	  • Windows
	  • 32-bit
	  • 64-bit
	For additional driver information, see Connecting to HiveServer2.

Feature support

The following list describes support of various components and functionality with Hive 2.3.8 - 2104:

- Supports Hive-2.3.8 on Tez-0.9.2 For more information, see Tez 0.9.2-2104 (MEP 7.1.0) Release Notes.
- Does not support Hive on Spark. You cannot use Spark as an query engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- Does not support HDFS encryption in Hive tables.
- Does not support LLAP with Hive-2.3.8 because Apache Slider is not an MapR supported ecosystem component.

last-updated: Apr 11, 2022

- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

Changes in default security configuration

The following list describes changes in default security for Hive 2.3.8 - 2104:

- None.

New in This Release

Hive 2.3.8 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- Added configuration to view audit logs for connected, disconnected, and total connected users in HiveServer2.
- Added Service verifier.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
9704751	2021-05-14	MAPR-HIVE-930: Cannot run join with Order by and Limit clause specified at the same time
7c2414	2021-05-14	Revert "ZOO-36: Upgrade Hive log jars to match the Zookeeper client"
097fa53	2021-05-14	Revert "MAPR-HIVE-725 : Avatica's unshaded Jackson (2.6) conflicts with Jackson 2.7+ through hive-exec dependency
ceaa1302	2021-05-14	Revert "HIVE-15708 : Upgrade calcite version to 1.12 (Remus Rusanu via Ashutosh Chauhan)"
8489e7db9	2021-04-19	MAPR-HIVE-924 : FAILURE! - in org.apache.hive.hcatalog.api.TestHCatClient
5a058e051	2021-04-19	MAPR-HIVE-925 : FAILURE! - in TestHiveRemote>TestHive.testAutoPurgeTablesAndPartitions
9659809	2021-04-19	MAPR-HIVE-877 : Add service verifier to hive package that contains a startable service (Part II)
fd887ffb	2021-04-12	MAPR-HIVE-919: FAILURE! - in org.apache.hadoop.hive.ql.metadata.TestHive
b984382e	2021-04-08	MAPR-HIVE-907 : java.lang.NoClassDefFoundError: org/xerial/snappy/Snappy while reading parquet data
50bf5f	2021-04-08	MAPR-HIVE-917 : FAILURE! - in org.apache.hadoop.hive.ql.io.parquet.TestHiveSchemaConverter
8e3d4bd	2021-03-26	MAPR-HIVE-857 : Class path contains multiple SLF4J bindings

b08237d	2021-03-25	MAPR-HIVE-912 : Update to hbase version to 1.4.13.0-mapr-SNAPSHOT
a84e3f6	2021-03-22	MAPR-HIVE-911 : Update to Hadoop to 2.7.5.0-mapr-710-SNAPSHOT
bebe9227	2021-03-16	MAPR-HIVE-908 : FAILURE! - in org.apache.hadoop.hive.metastore.TestHiveMetaStoreTimeout
e9ac75	2021-03-14	MAPR-HIVE-897 : CVE-2021-25329 Tomcat vulnerability
33f3104	2021-03-14	MAPR-HIVE-898 : Unable to connect hs2 through jdbc HiveConnection
78222e2	2021-03-09	MAPR-HIVE-877 : Add service verifier to hive package that contains a startable service
a1f0262	2021-03-03	MAPR-HIVE-893 : Can't insert values into Hive table and simple selecting data from table: java.lang.NoClassDefFoundError: org/apache/hive/com/google/common/util/concurrent/internal/InternalFutureFailureAccess
f2e586d	2021-03-22	MAPR-HIVE-882 : /opt/mapr/hive/hive-2.3/conf.backup.*_* : No such file or directory
ccaa8b2	2021-02-22	MAPR-HIVE-883 : Make parquet versions consistent in hive/drill/sqoop
cb576a8	2021-02-18	MAPR-HIVE-873 : Add audit logs for connected, disconnected and total users
1134475	2021-02-16	MAPR-HIVE-872 : Update Hadoop version to 2.7.4.0-mapr-710-SNAPSHOT
88cdd7	2021-02-16	MAPR-HIVE-879 : Update tomcat version to latest 10.0.2
881acb6	2021-02-16	MAPR-HIVE-878 : Refactor tomcat dependency management
fa6f8d4	2021-02-08	MAPR-HIVE-874 : CVE-2020-25649 Jackson databind vulnerability
f276ad01	2021-02-03	MAPR-HIVE-871 : Unable to create table in database created with specified location when storage based authorization is enabled
91db8cf	2021-02-02	MAPR-HIVE-869: missing Hive Queries page configuration while installing hive on tez via UI Installer
66f1f3	2021-01-28	MAPR-HIVE-856: hive-site.xml contains many empty lines between properties
012e21b	2021-01-25	MAPR-HIVE-812 : The query does not work when hive.vectorized.execution.enabled is set to true

last-updated: Apr 11, 2022

618dbc4e	2021-01-22	MAPR-HIVE-864 : FAILURE! - in org.apache.hadoop.hive.serde2.avro.TestAvroSerializer
3a5d3d4f	2021-01-21	MAPR-HIVE-860: Multiple Vulnerabilities in Hive jars found

This release from HPE also includes the following back-ported issues. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
c067c097	2021-04-08	HIVE-22648: Upgrade Parquet to 1.11.0 (Marta Kuczora, reviewed by Adam Szita)
6782db81	2021-04-08	HIVE-19464: Upgrade Parquet to 1.10.0 (Jesus Camacho Rodriguez, reviewed by Prasanth Jayachandran)
7bd9eff	2021-03-23	HIVE-20204: Type conversion during IN () comparisons is using different rules from other comparison operations (Jason Dere, reviewed by Ashutosh Chauhan)
18727a	2021-03-22	HIVE-21455: Too verbose logging in AvroGenericRecordReader (Miklos Szurap, reviewed by David Mollitor and Peter Vary)
a174ab74	2021-03-22	HIVE-22891: Skip PartitionDesc Extraction In CombineHiveRecord For Non-LLAP Execution Mode (Syed Shameerur Rahman, reviewed by Ádám Szita)
cf798d	2021-03-22	HIVE-24436: Fix Avro NULL_DEFAULT_VALUE compatibility issue (#1722)
6f91c4	2021-03-22	HIVE-23980: Shade Guava from hive-exec in Hive 2.3 (#1356)
0004ff7	2021-03-22	HIVE-24324: Remove deprecated API usage from Avro (#1621) (#1672)
43875c1	2021-03-22	HIVE-23323: Add qsplits profile (Zoltan Haindrich reviewed by Miklos Gergely)
89e5f1e0	2021-02-18	HIVE-21943 : Add audit log in HiveServer2
732b650	2021-02-05	HIVE-15160: Can't order by an unselected column (Pengcheng Xiong, reviewed by Ashutosh Chauhan)
05524d04	2021-02-05	HIVE-16117: SortProjectTransposeRule should check for monotonicity preserving CAST (Jesus Camacho Rodriguez, reviewed by Ashutosh Chauhan)
225d30c	2021-02-22	HIVE-24559: Fix some spelling issues (Ricky Ma reviewed by Vihang Karajgaonkar and Miklos Gergely)

last-updated: Apr 11, 2022

ec0c8ac	2021-02-22	HIVE-24553. Exclude calcite from test-jar dependency of hive-exec (#1794)
3c0fe12	2021-02-22	HIVE-24551: Hive should include transitive dependencies from calcite after shading it (#1792)
765be22	2021-02-22	HIVE-22708: Fix for HttpTransport to replace String.equals (Naveen Gangam, reviewed by Peter Vary)
7288caf4	2021-02-22	HIVE-24512: Exclude calcite in packaging. (#1760)

Known Issues and Limitations

- HIVE-947: If you add `tez.history.logging.service.class` and `tez.tez-ui.history-url.base` properties to `tez-site.xml` file, Hive applications will fail in EEP 7.1.0. To fix this issue, remove these properties from the `tez-site.xml` configuration file.
- HIVE-19502 Unable to insert values into table stored by JdbcStorageHandler
- HIVE-19286 NPE in MERGE operator on MR mode
- HIVE-789 [Hive-Hbase integration] Unable to run queries against hive-hbase tables. ClassNotFoundException: HiveHBaseTableInputFormat [MEP-7.0.0]

If you run an HBase + Hive + Tez integration in MEP 7.1.0, you may encounter the following exception:

```
Caused by: java.lang.ClassNotFoundException:
org.apache.hadoop.hbase.client.mapr.BaseTableMappingRules
    at java.base/
jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:581
)
    at java.base/
jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(ClassLoaders.java:
178)
    at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:522)
    ... 39 more
```

This exception can occur due to the new Tez classloader implemented in the Tez project. To resolve this issue, put the following additional JAR files in the `/apps/tez/tez-0.9` folder.

Issue the following commands before you run HBase + Hive + Tez integration in MEP 7.1.0:

```
hadoop fs -mkdir /apps/tez/tez-0.9/hbase
hadoop fs -put /opt/mapr/hbase/hbase-1.4.12/lib/* /apps/tez/tez-0.9/hbase/
```

Add the following property to `/opt/mapr/tez/tez-0.9/conf/tez-site.xml`:

```
<property>
<name>tez.lib.uris</name>
<value>${fs.defaultFS}/apps/tez/tez-0.9,${fs.defaultFS}/apps/tez/tez-0.9/
lib,${fs.defaultFS}/apps/tez/tez-0.9/hbase/</value>
 </property>
```

It is assumed that the Hive version is 2.3, Hbase version is 1.4.12, Tez version is 0.9, Hadoop version is 2.7.4, Zookeeper version is 3.5.6.0, and ecosystem release is 2009.

HIVE-760 [Hive-2.3] Could not start hive-metastore on Centos 8 MetaException(message:Version information not found in metastore)

last-updated: Apr 11, 2022

Starting in MEP 7.0.0, use the MySQL driver with MariaDB.

```
<property>
    <name>javax.jdo.option.ConnectionURL</name>
    <value>jdbc:mysql://localhost:3306/hive?createDatabaseIfNotExist=true</
value>
  </property>
 <property>
    <name>javax.jdo.option.ConnectionDriverName</name>
    <value>com.mysql.jdbc.Driver</value>
    <description>Driver class name for a JDBC metastore</description>
 </property>
```

Some SELECT queries can be converted to a single FETCH task minimizing latency. Currently, the query should be single sourced and should not have a subquery or any aggregations or distincts (which incurs RS), lateral views and joins:

```
none : disable hive.fetch.task.conversion
minimal : SELECT star, filter on partition columns, LIMIT only
more :  SELECT, filter, LIMIT only (support TABLESAMPLE and virtual columns)
```

Resolved Issues

- None.

Hive 2.3.7 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.3.7.

The following release notes for the Hive 2.3.7 component are included in the MapR Data Platform distribution for Apache Hadoop:

Hive 2.3.7 - 2101 (EEP 7.0.1) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hive. You may also be interested in the Apache Hive-2.3.7 Release Notes and the Apache Hive homepage.

Hive Version	2.3.7
Release Date	January 2021
HPE Version Interoperability	See Hive and HCatalog Support Matrix andEcosystem Support Matrix andMEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.3.7-mapr-2101
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate tohttp://package.mapr.hpe.com/releases/MEP/, and select your MEP and OS to view the list of package names.

ODBC/JDBC Drivers	Hive 2.3.7 works with the following HPE Hive drivers:
	• ODBC Drivers
	  • Mac OS X
	  • Linux
	  • 32-bit
	  • 64-bit
	  • Windows
	  • 32-bit
	  • 64-bit
	For additional driver information, see Connecting to HiveServer2.

Feature support

The following list describes support of various components and functionality with Hive 2.3.7 - 2101:

- Supports Hive-2.3.7 on Tez-0.9.2 For more information, see Tez 0.9.2-2101 (MEP 7.0.1) Release Notes.
- Does not support Hive on Spark. You cannot use Spark as an query engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- Does not support HDFS encryption in Hive tables.
- Does not support LLAP with Hive-2.3.7 because Apache Slider is not an HPE supported ecosystem component.
- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

Changes in default security configuration

The following list describes changes in default security for Hive 2.3.7 - 2101:

- None.

New in This Release

Hive 2.3.7 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
7365788	2021-01-06	MAPR-HIVE-859 : ERROR Unable to invoke factory method in class class org.apache.logging.log4j.core.appender.RandomAccessFileAppender for element RandomAccessFile. java.lang.reflect.InvocationTargetException

last-updated: Apr 11, 2022

6793d50	2021-01-06	MAPR-HIVE-738 : Set hive.exec.submit.local.task.via.child to true by default
20b0c86	2020-12-28	MAPR-HIVE-854 : Update Hadoop version to 2.7.4.0-mapr-700
42ee1f8	2020-12-28	MAPR-HIVE-853 : Compilation error: cannot find symbol class HBaseCommonTestingUtility
9d5412f	2020-12-14	MAPR-HIVE-852: Fix Netty vulnerability in Hive
7b844f9	2020-12-09	MAPR-HIVE-850 : CLONE - CVE-2018-11771, CVE-2019-12402: commons-compress vulnerabilities
a859e66	2020-12-09	MAPR-HIVE-848 : CLONE - WS-2019-0490: jcommander vulnerability
efc4c99	2020-12-07	MAPR-HIVE-844 : CVE-2020-9484 etc.: Tomcat vulnerabilities
cce8662	2020-12-04	MAPR-HIVE-845 : CVE-2009-2625,CVE-2012-0881 etc. vulnerabilities in xerces
6dd7f7b	2020-12-03	MAPR-HIVE-843 : CVE-2020-13692: PostreSQL JDBC driver vulnerability
b48b161	2020-12-03	MAPR-HIVE-700 : CVE-2014-0114: Vulnerability with commons-beanutils
8f08881	2020-12-03	MAPR-HIVE-757 : Do not send server (jetty) details
3e8cbe6	2020-12-03	MAPR-HIVE-841 : CVE-2020-27216: jetty vulnerability
ccca1c3	2020-11-27	MAPR-HIVE-839 : IF NOT EXISTS statement does not work for external tables when StorageBasedAuthorizationProvider is used
9cefaee	2020-11-05	MAPR-HIVE-836 : Validate MapR Ticket (expiry date) before connecting from Beeline to HiveServer2
4769494	2020-10-21	MAPR-HIVE-835 : PAM authentication requires MapR SASL ticket on secure cluster
204958b	2020-10-27	MAPR-HIVE-733 : CLONE - configure.sh backup config files indefinitely without any cleanup
60a2313	2020-10-28	MAPR-HIVE-791 : Unnecessary WARNINGs during hive CLI start
2748a6b	2020-09-29	MAPR-HIVE-746: Upgrade/Remove tomcat jasper library dependencies
ffc0475	2020-09-18	MAPR-HIVE-689: Hive date/ timestamp miscalculation while crossing daylight savings time

last-updated: Apr 11, 2022

This release from HPE also includes the following back-ported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
ef91ccf	2020-12-23	HIVE-15444 : tez.queue.name is invalid after tez job running on CLI
018dfec	2020-12-10	HIVE-21489 : EXPLAIN command throws ClassCastException in Hive
2b2bb7f	2020-12-18	HIVE-22144 : HiveServer Web UI: Adding secure flag to the cookies options
147d8d6	2020-12-16	HIVE-23583 : Upgrade to ant 1.10.9 due to CVEs

Known Issues and Limitations

- HIVE-19502 Unable to insert values into table stored by JdbcStorageHandler
- HIVE-19286 NPE in MERGE operator on MR mode
- HIVE-789 [Hive-Hbase integration] Unable to run queries against hive-hbase tables. ClassNotFoundException: HiveHBaseTableInputFormat [MEP-7.0.0]

If you run an HBase + Hive + Tez integration in MEP 7.0.1, you may encounter the following exception:

```
Caused by: java.lang.ClassNotFoundException:
org.apache.hadoop.hbase.client.mapr.BaseTableMappingRules
    at java.base/
jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:581
)
    at java.base/
jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(ClassLoaders.java:
178)
    at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:522)
    ... 39 more
```

This exception can occur due to the new Tez classloader implemented in the Tez project. To resolve this issue, put the following additional JAR files in the `/apps/tez/tez-0.9` folder.

Issue the following commands before you run HBase + Hive + Tez integration in MEP 7.0.1:

```
hadoop fs -mkdir /apps/tez/tez-0.9/hbase
hadoop fs -put /opt/mapr/hbase/hbase-1.4.12/lib/* /apps/tez/tez-0.9/hbase/
```

Add the following property to `/opt/mapr/tez/tez-0.9/conf/tez-site.xml`:

```
<property>
<name>tez.lib.uris</name>
<value>${fs.defaultFS}/apps/tez/tez-0.9,${fs.defaultFS}/apps/tez/tez-0.9/
lib,${fs.defaultFS}/apps/tez/tez-0.9/hbase/</value>
 </property>
```

It is assumed that the Hive version is 2.3, Hbase version is 1.4.12, Tez version is 0.9, Hadoop version is 2.7.4, Zookeeper version is 3.5.6.0, and ecosystem release is 2009.

HIVE-760 [Hive-2.3] Could not start hive-metastore on Centos 8 MetaException(message:Version information not found in metastore)

last-updated: Apr 11, 2022

Starting in MEP 7.0.1, use the MySQL driver with MariaDB.

```
<property>
    <name>javax.jdo.option.ConnectionURL</name>
    <value>jdbc:mysql://localhost:3306/hive?createDatabaseIfNotExist=true</
value>
  </property>
 <property>
    <name>javax.jdo.option.ConnectionDriverName</name>
    <value>com.mysql.jdbc.Driver</value>
    <description>Driver class name for a JDBC metastore</description>
 </property>
```

Some SELECT queries can be converted to a single FETCH task minimizing latency. Currently, the query should be single sourced and should not have a subquery or any aggregations or distincts (which incurs RS), lateral views and joins:

```
none : disable hive.fetch.task.conversion
minimal : SELECT star, filter on partition columns, LIMIT only
more :  SELECT, filter, LIMIT only (support TABLESAMPLE and virtual columns)
```

Resolved Issues

- None.

Hive 2.3.7-2009 (EEP 7.0.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.3.7-2009 in EEP 7.0.0.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive homepage and the following Apache Hive release notes:

- Apache Hive-2.3.7 Release Notes
- Apache Hive-2.3.6 Release Notes
- Apache Hive-2.3.5 Release Notes
- Apache Hive-2.3.4 Release Notes
- Apache Hive-2.3.3 Release Notes
- Apache Hive-2.3.2 Release Notes
- Apache Hive-2.3.1 Release Notes
- Apache Hive-2.3.0 Release Notes

These release notes contain MapR-specific information only and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Hive Version	2.3.7
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.3.7-mapr-2009

Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

ODBC / JDBC Drivers	Hive 2.3.7 works with the following MapR Hive ODBC drivers:
	• Mac OS X
	• Linux
	  • 32-bit
	  • 64-bit
	• Windows
	  • 32-bit
	  • 64-bit
	For additional driver information, see Connecting to HiveServer2 on page 3515.

Feature Support

- Hive-2.3.7 works with Tez-0.9.2. For more information, see Tez 0.9.2-2009 (MEP 7.0.0) Release Notes.
- Hive on Spark is not supported; you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- HDFS encryption is not supported in Hive tables.
- LLAP is not supported with Hive-2.3.7, as Apache Slider is not a supported ecosystem component.
- In Hive 2.1 and later, Hive must run the `schematool` command as an initialization step.
- In EEPs 6.3.1 and 7.0.0, the default protocol version for TLS (SSL) is `TLSv1.2`, but you can modify the protocol version. For more information, see Configure the TLS (SSL) Protocol Version in Hive on page 3449.

Default Security Configuration Change

By default, the `hive-site.xml` file now contains the following property on secured clusters:

Property	Default Value
hive.metastore.pre.event.listeners	org.apache.hadoop.hive.ql.security.authorization.AuthorizationPreEventListener

This property is added to `hive-site.xml` during Hive updates (no version change) and upgrades (with version change).

New Features

- None.

Known Issues

- Some SELECT queries can be converted to a single FETCH task to minimize latency. Currently, a query should be single sourced, without a subquery, and should not have any aggregations or distincts (which incurs RS), lateral views, or joins:
  - none: disable `hive.fetch.task.conversion`
  - minimal: SELECT *, filter on partition columns, LIMIT only
  - more: SELECT, filter, LIMIT only (support TABLESAMPLE and virtual columns)
- Use MySQL driver with MariaDB. In `hive-site.xml`, set the following properties:

```
<property>
    <name>javax.jdo.option.ConnectionURL</name>
    <value>jdbc:mysql://localhost:3306/hive?
createDatabaseIfNotExist=true</value>
</property>
<property>
    <name>javax.jdo.option.ConnectionDriverName</name>
    <value>com.mysql.jdbc.Driver</value>
    <description>Driver class name for a JDBC metastore</description>
</property>
```

- HIVE-19502 - Unable to insert values into table stored by JdbcStorageHandler.
- HIVE-19286 - NPE in MERGE operator in MR mode.
- HIVE-760 - [Hive-2.3] Could not start hive-metastore on Centos 8 MetaException(message:Version information not found in metastore)
- HIVE-789 - [Hive-Hbase integration] Unable to run queries against hive-hbase tables.

  When using HBase with Hive-on-Tez, some queries, such as SELECT COUNT(*), fail because the new Tez classloader cannot load the MapR-specific classes. When this occurs, the system returns the following exception:

```
org.apache.hadoop.hbase.client.mapr.BaseTableMappingRules
    at java.base/
jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:5
81)
    at java.base/
jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(ClassLoaders.jav
a:178)
    at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:522)
    ... 39 more
```

  Workaround

  This workaround resolves the issue when running HBase 1.4.12 with Hive 2.3 on Tez 0.9 in an environment with Hadoop 2.7.4 and ZooKeeper 3.5.6.0 in ecosystem release 2009.

  To resolve the issue, complete the following steps before you run HBase with Hive-on-Tez:

  1. Create the `/apps/tez/tez-0.9/hbase` directory:

```
hadoop fs -mkdir /apps/tez/tez-0.9/hbase
```


last-updated: Apr 11, 2022

2. Add the HBase JAR files to the `/apps/tez/tez-0.9/hbase` directory:

```
hadoop fs -put /opt/mapr/hbase/hbase-1.4.12/lib/* /apps/tez/tez-0.9/
hbase/
```

3. Update `/opt/mapr/tez/tez-0.9/conf/tez-site.xml` with the following property:

```
<property>
    <name>tez.lib.uris</name>
    <value>${fs.defaultFS}/apps/tez/tez-0.9,${fs.defaultFS}/apps/tez/
tez-0.9/lib,${fs.defaultFS}/apps/tez/tez-0.9/hbase/</value>
</property>
```

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
f50bb9f	2020-09-04	MAPR-HIVE-791 : Unnecessary WARNINGs during hive CLI start
8755a63	2020-09-03	MAPR-HIVE-816 : Update ZK version to 3.5.6.0-mapr-SNAPSHOT
8060cf9	2020-08-28	MAPR-HIVE-790 Added jmxagent jar into classpath
6cbb679	2020-08-26	MAPR-HIVE-741: Exculude org.mortbay.jetty from maven dependencies (Part 2)
822aab1	2020-08-27	MAPR-HIVE-814 : Replace mapr.release.version with mapr.core.version
06ce5fe	2020-08-24	MAPR-HIVE-790 Added support for jmxagent at MAPR_JMXREMOTEHOST
ab62da2	2020-08-17	MAPR-HIVE-808 : Upgrade com.lmax.disruptor to version 3.4.2
1804443	2020-08-09	MAPR-HIVE-805 : FAILURE! - in org.apache.hive.hcatalog.streaming.mutate.client.lock.TestLock
818e7f2	2020-08-09	MAPR-HIVE-804 : FAILURE! - in org.apache.hive.hcatalog.pig.TestAvroHCatLoader
288030d	2020-08-08	MAPR-HIVE-802 : FAILURE! - in org.apache.hadoop.hive.hbase.TestPutResultWritable
69bad5f	2020-08-08	MAPR-HIVE-801 : Fix java.lang.NoClassDefFoundError: org/eclipse/jetty/util/ClassVisibilityChecker
2612916	2020-08-08	MAPR-HIVE-800 : FAILURE! - in org.apache.hadoop.hive.ql.session.TestClearScratchDirUtil

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
4dfb9b2	2020-08-08	MAPR-HIVE-798 : FAILURE! - in org.apache.hadoop.hive.ql.exec.TestMapRDbJsonFetchByIdOperator
8ff1d1a	2020-08-07	MAPR-HIVE-755 : Illegal reflective access of ShimLoader when using Java 11
37fad15	2020-08-01	MAPR-HIVE-795 : Hive script / serviceCheckScript.sql has world writable permissions
285f1cd	2020-07-30	MAPR-HIVE-762 : Select from MapRDB JSON table with >= AND NOT LIKE returns empty result
06d1643	2020-07-30	MAPR-HIVE-782 : Failed to start Hive services due to Password file not found: jmxremote.password [non-secure cluster]
d87ce2b	2020-07-16	MAPR-HIVE-709 : Too verbose MapRDBSerDe
86872aa	2020-07-10	CORE-458 : ERROR [HiveServer2-Background-Pool: Thread-44] zookeeper.ZKDataRetrieval: Most likely SessionExpirationException. Need to reset ZK and call myself again
e1fa18a	2020-07-08	MAPR-HIVE-783 : Exclude slf4j-log4j12 from mapr-encryption tool
e65f666	2020-07-03	MAPR-HIVE-777 : no WebHcat process on port 50111 after service started. MEP-7, CentOS 8.1
b0c82be	2020-07-01	MAPR-HIVE-778 : Update pig version to 0.17.0.0-mapr-SNAPSHOT
cc098ac	2020-06-30	MAPR-HIVE-749 : Hive queries on tables with hdfs URI Scheme failing in compilation phase in 2.3
41d5dc8	2020-06-25	MAPR-HIVE-772 : WebHCat service failed to start on CentOS 8.1 MEP-7.0.0 with IllegalArgumentException
06f7625	2020-06-22	ZOO-36: Upgrade Hive log jars to match the Zookeeper client
0e92269	2020-06-19	MAPR-HIVE-771 : Update datanucleus to latest version
4cdb8f2	2020-06-19	MAPR-HIVE-770 : Add logging for datanucleus maven plugin
e4dab46	2020-06-20	MAPR-HIVE-769 : Fix compilation error on centos 8: MockUriInfo is not abstract and does not override abstract method relativize(java.net.URI)

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
3673257	2020-06-18	MAPR-HIVE-768 : Fix compilation error on centos 8: hiveserver2_jsp is not abstract and does not override abstract method getDependants()
0179594	2020-06-18	MAPR-HIVE-767 : Change mapr-core dependency to 4 digit 6.2.0.0-mapr-SNAPSHOT
1cb0bd0	2020-06-18	MAPR-HIVE-764 : Fix testSetFullFileStatusFailInheritGroup
7a8f79b	2020-06-18	MAPR-HIVE-761 : Remove jdo-api dependency and use datanucleus-jdo instead
90b46dd	2020-06-15	MAPR-HIVE-753 : Can't find com.sun.common.util.logging.LogStrings bundle
2991010	2020-06-11	MAPR-HIVE-756 : ERROR Failed to execute goal org.apache.maven.plugins:maven-shade-plugin
47596d0	2020-06-10	MAPR-HIVE-752 : Set Java 11 for maven-compiler-plugin
8cdf3af	2020-06-10	MAPR-HIVE-751 : Fix: cannot find symbol class Cleaner, location: package sun.misc
6d85bb3	2020-06-10	MAPR-HIVE-750 : Fix java.lang.ClassNotFoundException: javax.xml.bind.JAXBContext
e49a866	2020-06-03	MAPR-HIVE-747 : Remove PrintGCTimeStamps option since it is not supported in Java 11
4f1dd19	2020-05-22	MAPR-HIVE-687 : Hiveserver2 Java JMX Server Insecure Configuration Remote Code Execution Vulnerability
516117c	2020-05-18	MAPR-HIVE-737 : Failed Hive Druid Handler tests
fe5b7af	2020-05-13	MAPR-HIVE-734 : Fix org.apache.hadoop.hive.llap.registry.impl.TestSlotZnode
8d15ca8	2020-05-13	MAPR-HIVE-733 : Update curator version to 4.2.0
6f2ab30	2020-05-12	MAPR-HIVE-730 : Update MapR Maven repository URL to use https
291b8d2	2020-05-12	MAPR-HIVE-729 : Fix setMetaStorePreEventListenerCustomNotRemoveProperty(org.apache.hive.conftool.ConfToolTest)
3b56612	2020-05-11	MAPR-HIVE-726 : Update Guava version to 28.2-jre

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
154f6fe	2020-05-07	MAPR-HIVE-725 : Avatica's unshaded Jackson (2.6) conflicts with Jackson 2.7+ through hive-exec dependency
c7d9a1e	2020-05-05	MAPR-HIVE-714 : ZK updates to v3.5.6 at MEP7.0.0
e6e0676	2020-05-04	MAPR-HIVE-724 : Change Tez version in Hive-2.3.7 MEP-7.0.0 to 0.9.2-mapr-SNAPSHOT
ec32584	2020-04-30	MAPR-HIVE-723 : [Hive-Hbase integration] failed to select from Hive-Hbase table. HBase-1.4.12.0 [MEP-7.0.0]
5cdf6b9	2020-04-29	MAPR-HIVE-708 : Update thrift to ver 0.13.0 or most feasible newer version
741ca67	2020-04-28	MAPR-HIVE-705 : Update jasper-compiler to most feasible version with fix
0b51959	2020-04-27	MAPR-HIVE-704 : Update scala-compiler to ver 2.11.12/ 2.12.4 or most feasible newer version
e62a0c9	2020-04-27	MAPR-HIVE-701 : CVE-2017-18640: snakeyaml vulnerability at Hive
ae366a8	2020-04-27	MAPR-HIVE-700 : CVE-2014-0114: Vulnerability with commons-beanutils
1f73d9d	2020-04-27	MAPR-HIVE-703 : Update plexus-utils to ver 3.0.16 or most feasible upper version
8c538fc	2020-04-27	MAPR-HIVE-706 : Update derby to ver 10.12.1.1 or most feasible newer/ older version
e3f7723	2020-04-26	MAPR-HIVE-699 : Fix io.netty vulnerability
9479d49	2020-05-18	MAPR-HIVE-698 : Fix org.codehaus.jackson vulnerability
7312b11	2020-04-14	MAPR-HIVE-589 Hive jobs fail with "Unable to rename output"
bd9b9b1	2020-04-18	MAPR-HIVE-712 : Move Hive-2.3.6 MEP-7.x to two digit versioning in package name
8491cb9	2020-03-23	MAPR-HIVE-690 : Enable SSL protocol version TLSv1.2 as default
e8ee9eb	2020-03-16	MAPR-HIVE-561: Add hive configure.sh logging
8b0a614	2020-03-16	MAPR-HIVE-685: Enable AuthorizationPreEventListener in hive-site.xml for Security-By-Default

Commit	Date (YYYY-MM-DD)	Comment
0176bad	2020-02-24	MAPR-HIVE-683 : Add JUnit test for StorageBasedAuthorizationProvider and update of hive.metastore.warehouse.dir
9d814cf	2020-02-20	MAPR-HIVE-681 : StorageBasedAuthorizationProvider does not allow create databases after changing hive.metastore.warehouse.dir
6724858	2020-02-19	MAPR-HIVE-674: Failed to launch Hive WebHCat job zookeeper-3.4.11-mapr-1808.jar does not exist.
04b6072	2020-01-28	MAPR-HIVE-660 : Create root scratch dir with 733 instead of 777 perms [HIVE-8143] (Part II)
a0bfa31	2020-01-14	MAPR-HIVE-615: Add Junit tests for MapR Db Json optimization when projection push down is used
d53a0ed	2020-01-13	MAPR-HIVE-660 : Create root scratch dir with 733 instead of 777 perms [HIVE-8143]
62e55cf	2020-01-15	MAPR-HIVE-572 : Evaluate impact of Protobuf upgrade for 6.2 to Protobuf 3.5 or higher on Hive
94736a6	2020-01-08	MAPR-HIVE-656: Hive on Tez : Job of merging small files will be submitted into another queue (default queue)
741e989	2019-12-06	MAPR-HIVE-652: SQL Standard Based Authorization does not allow to create a function over HiveServer2 even to Admin user

This release also includes the following back-ported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
cc603a0	2020-08-29	HIVE-10296: Cast exception observed when hive runs a multi join query on metastore (postgres), since postgres pushes the filter into the join, and ignores the condition before applying cast (Karthik Manamcheri, reviewed by Sergey Shelukhin
21d923c	2020-08-17	HIVE-22278: Upgrade log4j to 2.12.1 (David Lavati via Zoltan Haindrich)
437a8c5	2020-08-17	HIVE-18384: ConcurrentModificationException in log4j2.x library (Prasanth Jayachandran reviewed by Sergey Shelukhin)

Commit	Date (YYYY-MM-DD)	Comment
5a3db2b	2020-08-08	HIVE-21777: Maven jar goal is producing warning due to missing dependency (Aron Hamvas via Peter Vary)
d830346	2020-08-08	HIVE-21836: Update apache directory server version to 1.5.7 (Zoltan Haindrich reviewed by Laszlo Bodor)
f605ca8	2020-07-13	HIVE-18699: Check for duplicate partitions in HiveMetastore.exchange_partitions (Marta Kuczora, reviewed by Adam Szita, Peter Vary)
51de45d	2020-06-04	HIVE-19700: Workaround for JLine issue with UnsupportedTerminal (Naveen Gangam, reviewed by Yongzhi Chen)
549c374	2020-06-11	HIVE-17879 : Upgrade Datanucleus Maven Plugin
f68fea9	2020-06-11	HIVE-23363 : Upgrade DataNucleus dependency to 5.2
1aab82c	2020-03-29	HIVE-21512: Upgrade jms-api to 2.0.2 (Zoltan Haindrich reviewed by Peter Vary)
f6606ce	2020-08-08	HIVE-22066: Upgrade Apache parent POM to version 21 (David Mollitor, reviewed by Ashutosh Chauhan)
f535560	2020-08-08	HIVE-16281: Upgrade master branch to JDK8 (Aihua Xu, reviewed by Thejas M Nair, Sergio Peña, Sean Busbey)
1ceac6f	2020-08-08	HIVE-22097: Incompatible java.util.ArrayList for java 11 (Attila Magyar via Laszlo Bodor, Prasanth Jayachandran)
f5e8e4b	2020-08-08	HIVE-19383 : Add ArrayList$SubList kryo serializer (Ashutosh Chauhan via Prasanth J)
4ffaf82	2020-08-08	HIVE-21584: Java 11 preparation: system class loader is not URLClassLoader (Zoltan Matyus, reviewed by Adam Szita, Zoltan Haindrich)
84774e2	2020-08-08	HIVE-23209: ptest2 compilation failure after HIVE-21603 - upgrade mockito-core in testutils/ptest2
2f25d15	2020-08-08	HIVE-21603 : Java 11 preparation: update powermock version (Panos G via Ashutosh Chauhan)
a1eebe8	2020-06-15	HIVE-21670: Replacing mockito-all with mockito-core dependency (Ivan Suller via Zoltan Haindrich)

Commit	Date (YYYY-MM-DD)	Comment
14c4d1a	2020-06-11	HIVE-19433: HiveJoinPushTransitivePredicatesRule hangs (Vineet Garg,reviewed by Jesus Camacho Rodriguez)
6e53218	2020-06-09	HIVE-19799: remove jasper dependency (Prasanth Jayachandran reviewed by Thejas M Nair)
741e268	2020-06-03	HIVE-16911: Upgrade groovy version to 2.4.11 (Aihua Xu, reviewed by Yongzhi Chen)
e266fea	2020-06-03	HIVE-16371: Add bitmap selection strategy for druid storage handler (Slim Bouguerra, reviewed by Jesus Camacho Rodriguez)
468ba91	2020-06-03	HIVE-16474: Upgrade Druid version to 0.10 (Nishant Bangarwa, reviewed by Jesus Camacho Rodriguez)
cacff9a	2020-06-03	HIVE-14069 : update curator version to 2.12.0 (Jason Dere via Ashutosh Chauhan)
c3ecf30	2020-06-03	HIVE-15708 : Upgrade calcite version to 1.12 (Remus Rusanu via Ashutosh Chauhan)
3719911	2020-05-18	HIVE-15393 : Update Guava version (Slim Bouguerra via Ashutosh Chauhan)
46ce644	2020-05-18	HIVE-18586: Upgrade Derby to 10.14.1.0 (Janaki Lahorani, reviewed by Aihua Xu)
53d3977	2020-05-18	HIVE-23073 : Shade netty and upgrade to netty 4.1.48.Final (Laszlo Bodor via Ashutosh Chauhan)
414a5d5	2020-05-18	HIVE-18436: Upgrade to Spark 2.3.0 (Sahil Takiar, reviewed by Rui Li)
a1fa8ab	2020-05-11	HIVE-18433: Upgrade version of com.fasterxml.jackson (Janaki Lahorani, reviewed by Aihua Xu)
76e2357	2020-04-23	HIVE-23086 Two tests fail on branch-2.3 (gates, reviewed by jcamacho)
302cd39	2020-04-23	HIVE-22249: Support Parquet through HCatalog (Jay Green-Stevens via Peter Vary)
839a94a	2020-04-23	HIVE-21508: ClassCastException when initializing HiveMetaStoreClient on JDK10 or newer (Ana Jalba, via Peter Vary)
71ea54a	2020-03-03	HIVE-16839 : Unbalanced calls to openTransaction/commitTransaction when alter the same partition concurrently (Guang Yang, reviewed by Karthik Manamcheri and Vihang Karajgaonkar)

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
9bc66b7	2020-04-14	HIVE-19199: ACID: DbTxnManager heartbeat-service needs static sync init (Gopal V, reviewed by Eugene Koifman)
a7a0efa	2020-02-05	HIVE-7089 : StorageBasedAuthorizationProvider fails to allow non-admin users to create databases in writable directories
ce28628	2020-01-30	HIVE-11741: Add a new hook to run before query parse/compile
8c8322c	2019-12-17	HIVE-22151 : Turn off hybrid grace hash join by default (Ashutosh Chauhan via Vineet Garg)

Hive 2.3.6 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.3.6.

The following release notes for the Hive 2.3.6 component are included in the MapR distribution for Apache Hadoop:

Hive 2.3.6 - 2201 (EEP 6.3.6) Release Notes

The notes below relate specifically to the MapR Data Platform Distribution for Apache Hive. You may also be interested in the Apache Hive-2.3.6 Release Notes and the Apache Hive homepage.

Hive Version	2.3.6
Release Date	January 2022
HPE Version Interoperability	See Hive and HCatalog Support Matrix and Ecosystem Support Matrix and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.3.6-mapr-2201
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to http://package.mapr.hpe.com/releases/MEP/, and select your MEP and OS to view the list of package names.

ODBC/JDBC Drivers	Hive 2.3.6 works with the following MapR Hive drivers:
	• ODBC Drivers
	  • Mac OS X
	  • Linux
	  • 32-bit
	  • 64-bit HIVE-20204
	  • Windows
	  • 32-bit
	  • 64-bit
	For additional driver information, see Connecting to HiveServer2.

Feature support

The following list describes support of various components and functionality with Hive 2.3.6 - 2201:

- Supports Hive-2.3.6 on Tez-0.9.1 For more information, see Tez 0.9.1 - 2201 (EEP 6.3.6) Release Notes on page 6489.
- Does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- Does not support HDFS encryption in Hive tables.
- Does not support LLAP with Hive-2.3.6 because Apache Slider is not a supported HPE ecosystem component.
- Starting from Hive 2.1, Hive must run the `schematool` command as an initialization step.

Changes in default security configuration

The following list describes changes in default security for Hive 2.3.6 - 2201:

- None.

New in This Release

Hive 2.3.6 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

last-updated: Apr 11, 2022

- Improved `Describe table` operator in terms of fetching statistics of partitions. Starting from EEP 6.3.5, you can fetch the partition information using the `describe` command with `formatted` or `extended` statements.

  Configure the `hive.describe.partitionedtable.ignore.stats` property to change the behaviour of fetching statistics of partitions. It is set to the default value of `false`.

```
<property>
    <name>hive.describe.partitionedtable.ignore.stats</name>
    <value>false</value>
    <description>Enables partitioned table stats collection for 'DESCRIBE
FORMATTED' or 'DESCRIBE EXTENDED' commands</description>
</property>
```

```
<property>
    <name>hive.describe.partitionedtable.ignore.stats</name>
    <value>true</value>
    <description>Disables partitioned table stats collection for
'DESCRIBE FORMATTED' or 'DESCRIBE EXTENDED' commands</description>
 </property
```

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
21f6a46d26	2022-01-25	EEP-HIVE-1149: Upgrade Log4J version to '1.3.1-mapr'
db3c3fe39d	2022-01-19	EEP-HIVE-1145 : Add only necessary jars to Hive class path from Slider lib folder
c358c77ba0	2022-01-19	MAPR-HIVE-857 : Class path contains multiple SLF4J bindings
5310a60e7c	2022-01-19	EEP-HIVE-1138 : Add some org.apache.curator classes to container classpath
485c2dddb3	2022-01-19	EEP-HIVE-1137 : Add tez jars to containers folder app/install//lib/tez/
5c6a759068	2022-01-19	EEP-HIVE-1136 : Apply existing LLAP daemon security for kerberos only
8eb9d621b8	2022-01-10	EEP-HIVE-1129 : java.lang.ClassNotFoundException: org.apache.commons.logging.LogFactory
2bdd87c769	2022-01-06	EEP-HIVE-1065: CVE-2021-37136, CVE-2021-37137, WS-2020-0408, CVE-2021-21290: netty-*-4.1.55.Final.jar
8c97c3dad4	2022-01-05	EEP-HIVE-1062: CVE-2016-5007, CVE-2016-9878 ,CVE-2018-1271, CVE-2018-1272, CVE-2020-5421: spring-*-3.2.16.RELEASE.jar

e8824ff58b	2022-01-05	EEP-HIVE-1068: CVE-2020-9480: spark-network-common_2.11-2.3.0.jar, CVE-2018-17190: spark-core_2.11-2.3.0.jar
4b59b03b22	2022-01-05	EEP-HIVE-1123 : UnsupportedClassVersionError: com/beust/jcommander/ParameterException
0f3c07c5db	2022-01-04	EEP-HIVE-1117: Update log4j v2 to the latest available (to 2.17+)
62cde6b526	2022-01-04	EEP-HIVE-1116: Hive returns an incorrect number of columns
f3eb784483	2022-01-04	Revert "MAPR-HIVE-930: Cannot run join with Order by and Limit clause specified at the same time"
b7e98bc780	2021-12-30	EEP-HIVE-1119: com.fasterxml.jackson.annotation.JsonFormat.empty()Lcom/fasterxml/jackson/annotation/JsonFormat
b6c3e29c75	2021-12-24	EEP-HIVE-1064: CVE-2021-30639 ; CVE-2021-33037: tomcat-coyote-10.0.4.jar
bd35d3d8b0	2021-12-24	EEP-HIVE-1059: CVE-2019-10172, CVE-2019-10202: jackson-mapper-asl-1.9.13.jar, jackson-mapper-asl-1.9.2.jar
a7414780f7	2021-12-24	EEP-HIVE-1056: CVE-2021-35515, CVE-2021-35516, CVE-2021-35517, CVE-2021-36090: commons-compress-1.20.jar
b9b59431d8	2021-12-24	EEP-HIVE-1055: CVE fixes of bcprov-jdk15on-1.52.jar
c72f9e03b2	2021-12-24	EEP-HIVE-1054: WS-2021-0419: gson-2.2.4.jar
3b9477aa97	2021-12-22	MAPR-HIVE-1053 : Relative path in absolute URI: slider reads hdfs-site.xml from hadoop-hdfs.jar
1bcc87122e	2021-12-22	MAPR-HIVE-1036 : java.lang.NoClassDefFoundError: org/apache/commons/digester/Digester
f385e38ed5	2021-12-22	MAPR-HIVE-1025 : LLAP server expects tez.tar.gz archive in MapR FS
ce47408acc	2021-12-22	MAPR-HIVE-1033 : Add MapR slider dependency to Hive
307f5634f4	2021-12-15	MAPR-HIVE-919: FAILURE! - in org.apache.hadoop.hive.ql.metadata.TestHive
6a6adc1605	2021-12-14	EEP-HIVE-1097: CVE-2021-44228 - Log4j vulnerability

38f79750e5	2021-12-03	EEP-HIVE-1087: CAST gives NULL values during insert when vectorization enabled.
0c49ac7aa6	2021-11-11	MAPR-HIVE-1031 : logError: command not found if any error happens during configuring Hive
86fce467db	2021-11-11	MAPR-HIVE-1024 : Replace deprecated AuthMethod.DIGEST with AuthMethod.TOKEN in HadoopThriftAuthBridge25Sasl
5aa83575a3	2021-11-02	MAPR-HIVE-975: Customer request to investigate temporary hive session files cleanup improvements
5d17652c59	2021-10-13	MAPR-HIVE-1002 : Hive-2.3 does not remove old compressed logs
949a43e679	2021-10-11	MAPR-HIVE-1015 : Configure repositories for Jenkins job

This release also includes the following back-ported issues. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
d1c4ee1ed3	2021-12-08	HIVE-17774: compaction may start with 0 splits and fail
e2f6d07cbf	2021-10-28	HIVE-20072 : Write access being requested when performing select on a table
be8520b3dc	2021-10-08	HIVE-16820 : TezTask may not shut down correctly before submit (Sergey Shelukhin, reviewed by Siddharth Seth)

Known Issues and Limitations

- HIVE-19502 Unable to insert values into table stored by JdbcStorageHandler
- HIVE-19286 NPE in MERGE operator on MR mode

Some SELECT queries can be converted to a single FETCH task minimizing latency. Currently, the query should be single sourced, not having any subquery, and should not have any aggregations or distincts (which incurs RS), lateral views or joins:

```
none : disable hive.fetch.task.conversion
minimal : SELECT star, filter on partition columns, LIMIT only
more :  SELECT, filter, LIMIT only (support TABLESAMPLE and virtual columns)
```

Resolved Issues

- None.

Hive 2.3.6 - 2110 (EEP 6.3.5) Release Notes

The notes below relate specifically to the MapR Data Platform Distribution for Apache Hive.

Hive Version	2.3.6
Release Date	October 2021

HPE Version Interoperability	See Hive and HCatalog Support Matrix and Ecosystem Support Matrix and MEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.3.6-mapr-2110
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to http://package.mapr.hpe.com/releases/MEP/, and select your MEP and OS to view the list of package names.

ODBC/JDBC Drivers	Hive 2.3.6 works with the following MapR Hive drivers:
	• ODBC Drivers
	  • Mac OS X
	  • Linux
	  • 32-bit
	  • 64-bit HIVE-20204
	  • Windows
	  • 32-bit
	  • 64-bit
	For additional driver information, see Connecting to HiveServer2.

Feature support

The following list describes support of various components and functionality with Hive 2.3.6 - 2110:

- Supports Hive-2.3.6 on Tez-0.9.1 For more information, see Tez 0.9.1-2104(MEP 6.3.4) Release Notes.
- Does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- Does not support HDFS encryption in Hive tables.
- Does not support LLAP with Hive-2.3.6 because Apache Slider is not a supported HPE ecosystem component.
- Starting from Hive 2.1, Hive must run the `schematool` command as an initialization step.

Changes in default security configuration

The following list describes changes in default security for Hive 2.3.6 - 2110:

- None.

New in This Release

Hive 2.3.6 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

last-updated: Apr 11, 2022

- Improved `Describe table` operator in terms of fetching statistics of partitions. Starting from EEP 6.3.5, you can fetch the partition information using the `describe` command with `formatted` or `extended` statements.

  Configure the `hive.describe.partitionedtable.ignore.stats` property to change the behaviour of fetching statistics of partitions. It is set to the default value of `false`.

```
<property>
    <name>hive.describe.partitionedtable.ignore.stats</name>
    <value>false</value>
    <description>Enables partitioned table stats collection for 'DESCRIBE
FORMATTED' or 'DESCRIBE EXTENDED' commands</description>
</property>
```

```
<property>
    <name>hive.describe.partitionedtable.ignore.stats</name>
    <value>true</value>
    <description>Disables partitioned table stats collection for
'DESCRIBE FORMATTED' or 'DESCRIBE EXTENDED' commands</description>
 </property
```

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
5cf874c69e	2021-09-14	MAPR-HIVE-1007: Permission denied to hbase temp files while running hcat jobs from other user
d40b435ebc	2021-08-17	MAPR-HIVE-977 : Downgrade jackson to v2.11.1 or to 2.11.3 to be consistent with core version
55471aefc0	2021-07-29	MAPR-HIVE-960 : CVE-2012-5783 vulnerability in commons-httpclient
96739404d6	2021-07-23	MAPR-HIVE-965 : Throws exception at INSERT statement with Avro table on MapReduce engine
864aa58888	2021-07-21	MAPR-HIVE-959 : Update derbyclient and derbynet to most feasible version
dbc1444cce	2021-07-21	MAPR-HIVE-963 : CVE-2020-13956,WS-2017-3734 vulnerabilities in httpclient
f1b65a14f4	2021-07-21	MAPR-HIVE-962 : WS-2019-0379: commons-codec vulnerability
de8eff00d4	2021-06-28	MAPR-HIVE-896 : CVE-2020-17521 vulnerability in Groovy
d6cc638433	2021-06-22	MAPR-HIVE-927 : NPE thrown from XmlUtil by Hive Client
cf0ce2e550	2021-06-13	MAPR-HIVE-950 : HiveVersionInfo.getShortVersion returns wrong version
4ba9150835	2021-06-08	MAPR-HIVE-949 : Make SchemaEvolution class behavior the same as in Orc-1.5.12

d6a0cd99ae	2021-06-05	MAPR-HIVE-894 : CVE-2020-13955 vulnerability in Calcite

This release also includes the following back-ported issues. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
2e00b3ab7e	2021-09-15	HIVE-24965: Describe table partition stats fetch should be configurable
33c59e5bc1	2021-09-15	HIVE-22453: Describe table unnecessarily fetches partitions
59d9e532b1	2021-09-14	HIVE-23756 : Added more constraints to the package.jdo file
8f93305198	2021-09-12	HIVE-24177 : hive mapjoin throws udf class not found
d086e9f44e	2021-09-06	HIVE-25054: Upgrade `jodd-core` dependency to get rid of CVE-2018-21234 (Abhay Chennagiri, reviewed by Jesus Camacho Rodriguez)
f5c8373511	2021-08-18	HIVE-17659 : get_token thrift call fails for DBTokenStore in remote HMS mode (Vihang Karajgaonkar, reviewed by Aihua Xu)
2f963af3df	2021-08-18	HIVE-17824 : msck repair table should drop the missing partitions from metastore (Janaki Lahorani, reviewed by Peter Vary, Alexander Kolbasov and Vihang Karajgaonkar)
f219115a01	2021-08-18	HIVE-16143: Improve msck repair batching (Vihang Karajgaonkar, reviewed by Sahil Takiar & Aihua Xu)
b3197062e6	2021-07-28	HIVE-19228: Remove commons-httpclient 3.x usage (Janaki Lahorani reviewed by Aihua Xu)
9078072775	2021-07-23	HIVE-24324: Remove deprecated API usage from Avro (#1621)
b8d972a30a	2021-06-08	ORC-437: Make acid schema checks case insensitive
4a139e039f	2021-05-31	HIVE-21075 : Metastore: Drop partition performance downgrade with Postgres DB
6d952ad320	2021-05-31	HIVE-9447: Metastore: inefficient Oracle query for removing unused column descriptors when add/drop table/partition (Selina Zhang reviewed by Ashutosh Chauhan, Adam Szita)
79986d8ae1	2021-05-21	HIVE-21085: Materialized views registry starts non-external tez session (Jesus Camacho Rodriguez, reviewed by Ashutosh Chauhan)

22a4f78a98	2021-05-21	HIVE-19691: Start SessionState in materialized views registry (Jesus Camacho Rodriguez, reviewed by Ashutosh Chauhan)
46a18aa72b	2021-05-21	HIVE-17853: RetryingMetaStoreClient loses UGI impersonation-context when reconnecting after timeout (Chris Drome, reviewed by Mithun Radhakrishnan)
3b72d6b9c3	2021-05-21	HIVE-23534: NPE in RetryingMetaStoreClient#invoke when catching MetaException with no message (Stamatis Zampetakis, reviewed by Jesus Camacho Rodriguez)
85b0e4b2dd	2021-05-21	HIVE-18494: Regression: from HIVE-18069, the metastore directsql is getting disabled (Jesus Camacho Rodriguez, reviewed by Gopal V)
d91f984cf6	2021-05-21	HIVE-18069: MetaStoreDirectSql to get tables has misplaced comma (Jesus Camacho Rodriguez, reviewed by Aihua Xu) (addendum)
5e06c5859e	2021-05-21	HIVE-18069: MetaStoreDirectSql to get tables has misplaced comma (Jesus Camacho Rodriguez, reviewed by Aihua Xu)
efa1df1148	2021-05-21	HIVE-15436: Enhancing metastore APIs to retrieve only materialized views (Jesus Camacho Rodriguez, reviewed by Ashutosh Chauhan)
e642981bf1	2021-05-21	HIVE-6990 : Direct SQL fails when the explicit schema setting is different from the default on (Bing Li, Sergey Shelukhin via Ashutosh Chauhan)

Known Issues and Limitations

- HIVE-19502 Unable to insert values into table stored by JdbcStorageHandler
- HIVE-19286 NPE in MERGE operator on MR mode

Some SELECT queries can be converted to a single FETCH task minimizing latency. Currently, the query should be single sourced, not having any subquery, and should not have any aggregations or distincts (which incurs RS), lateral views or joins:

```
none : disable hive.fetch.task.conversion
minimal : SELECT star, filter on partition columns, LIMIT only
more :  SELECT, filter, LIMIT only (support TABLESAMPLE and virtual columns)
```

Resolved Issues

- None.

Hive 2.3.6 - 2104 (EEP 6.3.4) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hive.

Hive Version	2.3.6
Release Date	April 2021
HPE Version Interoperability	SeeHive and HCatalog Support Matrix andEcosystem Support Matrix and MEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.3.6-mapr-2104
Maven Artifacts	See Maven Artifacts for MapR.
Package Names	Navigate tohttp://package.mapr.hpe.com/releases/MEP/, and select your MEP and OS to view the list of package names.

ODBC/JDBC Drivers	Hive 2.3.6 works with the following MapR Hive drivers:
	• ODBC Drivers
	  • Mac OS X
	  • Linux
	  • 32-bit
	  • 64-bit HIVE-20204
	  • Windows
	  • 32-bit
	  • 64-bit
	For additional driver information, seeConnecting to HiveServer2.

Feature support

The following list describes support of various components and functionality with Hive 2.3.6 - 2104:

- Supports Hive-2.3.6 on Tez-0.9.1 For more information, see Tez 0.9.1-2104(MEP 6.3.4) Release Notes.
- Does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- Does not support HDFS encryption in Hive tables.
- Does not support LLAP with Hive-2.3.6 because Apache Slider is not a supported HPE ecosystem component.
- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

Changes in default security configuration

The following list describes changes in default security for Hive 2.3.6 - 2104:

- None.

New in This Release

Hive 2.3.6 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

last-updated: Apr 11, 2022

- Added configuration to view audit logs for connected, disconnected, and total connected users in HiveServer2.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
a4e067b	2021-05-14	MAPR-HIVE-930: Cannot run join with Order by and Limit clause specified at the same time.
c108592	2021-05-14	Revert "MAPR-HIVE-725 : Avatica's unshaded Jackson (2.6) conflicts with Jackson 2.7+ through hive-exec dependency
d5141b22	2021-05-14	Revert "HIVE-15708 : Upgrade calcite version to 1.12 (Remus Rusanu via Ashutosh Chauhan)"
5fe63174	2021-04-19	MAPR-HIVE-920 : FAILURE! - in org.apache.hadoop.hive.maprdb.json.MapRDbJsonFetchByIdOptimizerPositiveTest
39751b9f	2021-04-13	MAPR-HIVE-923 : Incorrect timestamp result of SELECT with WHERE clause
942bc9be	2021-04-08	MAPR-HIVE-917 : FAILURE! - in org.apache.hadoop.hive.ql.io.parquet.TestHiveSchemaConverter
520908f73	2021-03-16	MAPR-HIVE-908 : FAILURE! - in org.apache.hadoop.hive.metastore.TestHiveMetaStoreTimeout
0f23428d1	2021-03-14	MAPR-HIVE-897 : CVE-2021-25329 Tomcat vulnerability
08b7f601c	2021-03-14	MAPR-HIVE-898 : Unable to connect hs2 through jdbc HiveConnection
29c7ef020	2021-02-25	MAPR-HIVE-882 : /opt/mapr/hive/hive-2.3/conf.backup.*_* : No such file or directory
633a5d14	2021-02-22	MAPR-HIVE-883 : Make parquet versions consistent in hive/drill/sqoop
cee60ee4	2021-02-18	MAPR-HIVE-873 : Add audit logs for connected, disconnected and total users
04c84b40	2021-02-16	MAPR-HIVE-879 : Update tomcat version to latest 10.0.2
d93a9f2a0	2021-02-16	MAPR-HIVE-878 : Refactor tomcat dependency management
621e47a3	2021-02-08	MAPR-HIVE-874 : CVE-2020-25649 Jackson databind vulnerability
53a0d475	2021-02-03	MAPR-HIVE-871 : Unable to create table in database created with specified location when storage based authorization is enabled

48d69f56	2021-02-02	MAPR-HIVE-869: missing Hive Queries page configuration while installing hive on tez via UI Installer
ee9ff0e5	2021-01-28	MAPR-HIVE-863 : Hive query executed in java code failing with ClassNotFoundException
3ad9582	2021-01-25	MAPR-HIVE-812 : The query does not work when hive.vectorized.execution.enabled is set to true
7ace643d	2021-01-22	MAPR-HIVE-864 : FAILURE! - in org.apache.hadoop.hive.serde2.avro.TestAvroSerializer
8542d3fd	2021-01-15	MAPR-HIVE-860: Multiple Vulnerabilities in Hive jars found

This release also includes the following back-ported issues. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
e920e2340	2021-04-08	HIVE-22648: Upgrade Parquet to 1.11.0 (Marta Kuczora, reviewed by Adam Szita)
833e688d	2021-04-08	HIVE-19464: Upgrade Parquet to 1.10.0 (Jesus Camacho Rodriguez, reviewed by Prasanth Jayachandran)
89140ba6	2021-03-25	HIVE-22708: Fix for HttpTransport to replace String.equals (Naveen Gangam, reviewed by Peter Vary)
57e56458	2021-03-23	HIVE-20204: Type conversion during IN () comparisons is using different rules from other comparison operations (Jason Dere, reviewed by Ashutosh Chauhan)
653c1185	2021-03-17	HIVE-21455: Too verbose logging in AvroGenericRecordReader (Miklos Szurap, reviewed by David Mollitor and Peter Vary)
0a219a9d0	2021-03-11	HIVE-22891: Skip PartitionDesc Extraction In CombineHiveRecord For Non-LLAP Execution Mode (Syed Shameerur Rahman, reviewed by Ádám Szita)
6618d2190	2021-02-18	HIVE-21943 : Add audit log in HiveServer2
2f56ebff	2021-02-05	HIVE-15160: Can't order by an unselected column (Pengcheng Xiong, reviewed by Ashutosh Chauhan)
5956d7c47	2021-02-05	HIVE-16117: SortProjectTransposeRule should check for monotonicity preserving CAST (Jesus Camacho Rodriguez, reviewed by Ashutosh Chauhan)

last-updated: Apr 11, 2022

Known Issues and Limitations

- HIVE-19502 Unable to insert values into table stored by JdbcStorageHandler
- HIVE-19286 NPE in MERGE operator on MR mode

Some SELECT queries can be converted to a single FETCH task minimizing latency. Currently, the query should be single sourced, not having any subquery, and should not have any aggregations or distincts (which incurs RS), lateral views or joins:

```
none : disable hive.fetch.task.conversion
minimal : SELECT star, filter on partition columns, LIMIT only
more :  SELECT, filter, LIMIT only (support TABLESAMPLE and virtual columns)
```

Resolved Issues

- None.

Hive 2.3.6 - 2101 (EEP 6.3.2) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hive.

Hive Version	2.3.6
Release Date	January 2021
HPE Version Interoperability	SeeHive and HCatalog Support Matrix andEcosystem Support Matrix andMEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.3.6-mapr-2101
Maven Artifacts	See Maven Artifacts for MapR.
Package Names	Navigate tohttp://package.mapr.hpe.com/releases/MEP/, and select your MEP and OS to view the list of package names.

ODBC/JDBC Drivers	Hive 2.3.6 works with the following HPE Hive drivers:
	• ODBC Drivers
	  • Mac OS X
	  • Linux
	  • 32-bit
	  • 64-bit
	  • Windows
	  • 32-bit
	  • 64-bit
	For additional driver information, seeConnecting to HiveServer2.

Feature support

The following list describes support of various components and functionality with Hive 2.3.6 - 2101:

- Supports Hive-2.3.6 on Tez-0.9.1 For more information, see Tez 0.9.1-2101 (MEP 6.3.2) Release Notes.

- Does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- Does not support HDFS encryption in Hive tables.
- Does not support LLAP with Hive-2.1.1 because Apache Slider is not a supported HPE ecosystem component.
- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

Changes in default security configuration

The following list describes changes in default security for Hive 2.3.6 - 2101:

- None.

New in This Release

Hive 2.3.6 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
1fb62f0	2021-01-06	MAPR-HIVE-859 : ERROR Unable to invoke factory method in class class org.apache.logging.log4j.core.appender.RandomAccessFileAppender for element RandomAccessFile. java.lang.reflect.InvocationTargetException
504b21a	2021-01-06	MAPR-HIVE-738 : Set hive.exec.submit.local.task.via.child to true by default
5148c12	2020-12-28	MAPR-HIVE-853 : Compilation error: cannot find symbol class HBaseCommonTestingUtility
d0d6bdd	2020-12-14	MAPR-HIVE-852: Fix Netty vulnerability in Hive
24d0020	2020-12-10	HIVE-21489 : EXPLAIN command throws ClassCastException in Hive
a9e23f7	2020-12-09	MAPR-HIVE-850 : CLONE - CVE-2018-11771, CVE-2019-12402: commons-compress vulnerability
3e62bc6	2020-12-09	MAPR-HIVE-848 : CLONE - WS-2019-0490: jcommander vulnerability
33b4049	2020-12-07	MAPR-HIVE-844 : CVE-2020-9484 etc.: Tomcat vulnerabilities
c290ef5	2020-12-04	MAPR-HIVE-845 : CVE-2009-2625,CVE-2012-0881 etc. vulnerabilities in xerces

last-updated: Apr 11, 2022

e8f906a	2020-12-03	MAPR-HIVE-843 : CVE-2020-13692: PostreSQL JDBC driver vulnerability
4feb830	2020-12-03	MAPR-HIVE-700 : CVE-2014-0114: Vulnerability with commons-beanutils
25fa3ca	2020-12-03	MAPR-HIVE-757 : Do not send server (jetty) details
7f615a8	2020-12-03	MAPR-HIVE-841 : CVE-2020-27216: jetty vulnerability
40e7e79	2020-11-27	MAPR-HIVE-839 : IF NOT EXISTS statement does not work for external tables when StorageBasedAuthorizationProvider is used
1b5e8ec	2020-11-05	MAPR-HIVE-836 : Validate MapR Ticket (expiry date) before connecting from Beeline to HiveServer2
e3c6553	2020-11-21	MAPR-HIVE-835 : PAM authentication requires MapR SASL ticket on secure cluster
d098f35	2020-10-27	MAPR-HIVE-733 : CLONE - configure.sh backup config files indefinitely without any cleanup
35403d5	2020-10-01	MAPR-HIVE-746: Upgrade/Remove tomcat jasper library dependencies
b84e678	2020-10-01	MAPR-HIVE-823: Fix TestHS2HttpServerPam and TestHS2HttpServer
0860bb3	2020-09-29	MAPR-HIVE-822: Fixed test in TestDruidStorageHandler
424faa0	2020-09-18	MAPR-HIVE-689: Hive date/ timestamp miscalculation while crossing daylight savings time

This release also includes the following back-ported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
c8785fa	2020-12-23	HIVE-15444 : tez.queue.name is invalid after tez job running on CLI
5b1359c	2020-11-18	HIVE-22144 : HiveServer Web UI: Adding secure flag to the cookies options
c1ec2a3	2020-11-16	HIVE-23583 : Upgrade to ant 1.10.9 due to CVEs

Known Issues and Limitations

- HIVE-19502 Unable to insert values into table stored by JdbcStorageHandler
- HIVE-19286 NPE in MERGE operator on MR mode

Some SELECT queries can be converted to a single FETCH task minimizing latency. Currently, the query should be single sourced, not having any subquery, and should not have any aggregations or distincts (which incurs RS), lateral views or joins:

```
none : disable hive.fetch.task.conversion
minimal : SELECT star, filter on partition columns, LIMIT only
more :  SELECT, filter, LIMIT only (support TABLESAMPLE and virtual columns)
```

Resolved Issues

- None.

Hive 2.3.6-2009 (EEP 6.3.1) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.3.6-2009 in EEP 6.3.1.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive homepage and the following Apache Hive release notes:

- Apache Hive-2.3.7 Release Notes
- Apache Hive-2.3.6 Release Notes
- Apache Hive-2.3.5 Release Notes
- Apache Hive-2.3.4 Release Notes
- Apache Hive-2.3.3 Release Notes
- Apache Hive-2.3.2 Release Notes
- Apache Hive-2.3.1 Release Notes
- Apache Hive-2.3.0 Release Notes

These release notes contain MapR-specific information only and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Hive Version	2.3.6
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.3.6-mapr-2009
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

last-updated: Apr 11, 2022

ODBC / JDBC Drivers	Hive 2.3.6 works with the following MapR Hive ODBC drivers:
	• Mac OS X
	• Linux
	  • 32-bit
	  • 64-bit
	• Windows
	  • 32-bit
	  • 64-bit
	For additional driver information, see Connecting to HiveServer2 on page 3515.

Feature Support

- MapR supports Hive-2.3.6 on Tez-0.9.1 For more information, see Tez 0.9.1-2009 (MEP 6.3.1) Release Notes.
- Hive on Spark is not supported; you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- HDFS encryption is not supported in Hive tables.
- LLAP is not supported with Hive-2.3.6, as Apache Slider is not a supported ecosystem component.
- In Hive 2.1 and later, Hive must run the `schematool` command as an initialization step.
- In EEPs 6.3.1 and 7.0.0, the default protocol version for TLS (SSL) is `TLSv1.2`, but you can modify the protocol version. For more information, see Configure the TLS (SSL) Protocol Version in Hive on page 3449.

Default Security Configuration Change

By default, the `hive-site.xml` file now contains the following property on secured clusters:

Property	Default Value
hive.metastore.pre.event.listeners	org.apache.hadoop.hive.ql.security.authorization.AuthorizationPreEventListener

This property is added to `hive-site.xml` during Hive updates (no version change) and upgrades (with version change).

New Features

- None.

Known Issues

Some SELECT queries can be converted to a single FETCH task to minimize latency. Currently, a query should be single sourced, without a subquery, and should not have any aggregations or distincts (which incurs RS), lateral views, or joins:

- none: disable `hive.fetch.task.conversion`

- minimal: SELECT *, filter on partition columns, LIMIT only
- more: SELECT, filter, LIMIT only (support TABLESAMPLE and virtual columns)
- HIVE-19502 - Unable to insert values into table stored by JdbcStorageHandler
- HIVE-19286 - NPE in MERGE operator on MR mode

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
3ba69c4	2020-08-28	MAPR-HIVE-741: Exclude org.mortbay.jetty from maven dependencies (Part 2)
018ef4b	2020-08-26	MAPR-HIVE-813 : Update mapr.core.version to 6.1.0-mapr
f4e3caa	2020-07-15	MAPR-HIVE-753 : Can't find com.sun.common.util.logging.LogStrings bundle
d214837	2020-08-16	MAPR-HIVE-741: Exclude org.mortbay.jetty from maven dependenices
51b1a49	2020-08-01	MAPR-HIVE-795 : Hive script / serviceCheckScript.sql has world writable permissions
b69b888	2020-07-30	MAPR-HIVE-762 : Select from MapRDB JSON table with >= AND NOT LIKE returns empty result
a190732	2020-07-30	MAPR-HIVE-782 : Failed to start Hive services due to Password file not found: jmxremote.password [non-secure cluster]
aa51497	2020-07-16	MAPR-HIVE-709 : Too verbose MapRDBSerDe
f487a46	2020-07-08	MAPR-HIVE-783 : Exclude slf4j-log4j12 from mapr-encryption tool
ffa0c77	2020-07-01	MAPR-HIVE-779 : Update pig version to 0.16.0-mapr-SNAPSHOT
7778bbb	2020-06-30	MAPR-HIVE-749 : Hive queries on tables with hdfs URI Scheme failing in compilation phase in 2.3
95303ba	2020-05-22	MAPR-HIVE-687 : Hiveserver2 Java JMX Server Insecure Configuration Remote Code Execution Vulnerability
6887cc7	2020-05-12	MAPR-HIVE-730 : Update MapR Maven repository URL to use https
7c5f9c1	2020-05-12	MAPR-HIVE-729 : Fix setMetaStorePreEventListenerCustomNotRemoveProperty(org.apache.hive.conftool.ConfToolTest)

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
7adc138	2020-05-07	MAPR-HIVE-725 : Avatica's unshaded Jackson (2.6) conflicts with Jackson 2.7+ through hive-exec dependency
79b7a0f	2020-04-28	MAPR-HIVE-708 : Update thrift to ver 0.13.0 or most feasible newer version
333b60e	2020-04-28	MAPR-HIVE-705 : Update jasper-compiler to most feasible version with fix
b60006c	2020-04-27	MAPR-HIVE-704 : Update scala-compiler to ver 2.11.12/ 2.12.4 or most feasible newer version
ba2deb1	2020-04-27	MAPR-HIVE-701 : CVE-2017-18640: snakeyaml vulnerability at Hive
1a6aced	2020-04-27	MAPR-HIVE-700 : CVE-2014-0114: Vulnerability with commons-beanutils
9a711e1	2020-04-27	MAPR-HIVE-703 : Update plexus-utils to ver 3.0.16 or most feasible upper version
17a92cd	2020-04-26	MAPR-HIVE-706 : Update derby to ver 10.12.1.1 or most feasible newer/ older version
0868725	2020-04-26	MAPR-HIVE-699 : Fix io.netty vulnerability
2aed86c	2020-04-26	MAPR-HIVE-698 : Fix org.codehaus.jackson vulnerability
df50071	2020-04-14	MAPR-HIVE-589 Hive jobs fail with "Unable to rename output"
b222d1a	2020-04-23	MAPR-HIVE-719 : Configure Hive-2.3 MEP-6.3.1 to build from branch-2.3.6-mapr
6dd4643	2020-04-18	MAPR-HIVE-712 : Move Hive-2.3.6 MEP-7.x to two digit versioning in package name
8491cb9	2020-03-23	MAPR-HIVE-690 : Enable SSL protocol version TLSv1.2 as default
e8ee9eb	2020-03-16	MAPR-HIVE-561: Add hive configure.sh logging
8b0a614	2020-03-05	MAPR-HIVE-685: Enable AuthorizationPreEventListener in hive-site.xml for Security-By-Default
0176bad	2020-02-24	MAPR-HIVE-683 : Add JUnit test for StorageBasedAuthorizationProvider and update of hive.metastore.warehouse.dir
9d814cf	2020-02-20	MAPR-HIVE-681 : StorageBasedAuthorizationProvider does not allow create databases after changing hive.metastore.warehouse.dir

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
6724858	2020-02-17	MAPR-HIVE-674: Failed to launch Hive WebHCat job zookeeper-3.4.11-mapr-1808.jar does not exist.
04b6072	2020-01-28	MAPR-HIVE-660 : Create root scratch dir with 733 instead of 777 perms [HIVE-8143] (Part II)
a0bfa31	2020-01-14	MAPR-HIVE-615: Add Junit tests for MapR Db Json optimization when projection push down is used
d53a0ed	2020-01-13	MAPR-HIVE-660 : Create root scratch dir with 733 instead of 777 perms [HIVE-8143]
62e55cf	2020-01-15	MAPR-HIVE-572 : Evaluate impact of Protobuf upgrade for 6.2 to Protobuf 3.5 or higher on Hive
94736a6	2020-01-08	MAPR-HIVE-656: Hive on Tez : Job of merging small files will be submitted into another queue (default queue)
741e989	2019-12-06	MAPR-HIVE-652: SQL Standard Based Authorization does not allow to create a function over HiveServer2 even to Admin user

This release also includes the following back-ported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
64ebdfb	2020-08-29	HIVE-10296: Cast exception observed when hive runs a multi join query on metastore (postgres), since postgres pushes the filter into the join, and ignores the condition before applying cast (Karthik Manamcheri, reviewed by Sergey Shelukhin)
caebf8e	2020-07-13	HIVE-18699: Check for duplicate partitions in HiveMetastore.exchange_partitions (Marta Kuczora, reviewed by Adam Szita, Peter Vary)
b5b4cda	2020-07-13	HIVE-19700: Workaround for JLine issue with UnsupportedTerminal (Naveen Gangam, reviewed by Yongzhi Chen)
f58ae04	2020-06-11	HIVE-19433: HiveJoinPushTransitivePredicatesRule hangs (Vineet Garg,reviewed by Jesus Camacho Rodriguez)
df0aa3c	2020-06-10	HIVE-19799: remove jasper dependency (Prasanth Jayachandran reviewed by Thejas M Nair)

Commit	Date (YYYY-MM-DD)	Comment
4b9b258	2020-06-03	HIVE-16911: Upgrade groovy version to 2.4.11 (Aihua Xu, reviewed by Yongzhi Chen)
894033a	2020-05-18	HIVE-14069 : update curator version to 2.12.0 (Jason Dere via Ashutosh Chauhan)
bf12c85	2020-05-18	HIVE-15708 : Upgrade calcite version to 1.12 (Remus Rusanu via Ashutosh Chauhan)
9ddd43e	2020-05-18	HIVE-18586: Upgrade Derby to 10.14.1.0 (Janaki Lahorani, reviewed by Aihua Xu)
44eadc7	2020-05-18	HIVE-23073 : Shade netty and upgrade to netty 4.1.48.Final (Laszlo Bodor via Ashutosh Chauhan)
576c802	2020-05-18	HIVE-18436: Upgrade to Spark 2.3.0 (Sahil Takiar, reviewed by Rui Li)
d7f384d	2020-05-11	HIVE-18433: Upgrade version of com.fasterxml.jackson (Janaki Lahorani, reviewed by Aihua Xu)
71ea54a	2020-04-14	HIVE-16839 : Unbalanced calls to openTransaction/commitTransaction when alter the same partition concurrently (Guang Yang, reviewed by Karthik Manamcheri and Vihang Karajgaonkar)
9bc66b7	2020-04-14	HIVE-19199: ACID: DbTxnManager heartbeat-service needs static sync init (Gopal V, reviewed by Eugene Koifman)
a7a0efa	2020-02-05	HIVE-7089 : StorageBasedAuthorizationProvider fails to allow non-admin users to create databases in writable directories
ce28628	2020-01-17	HIVE-11741: Add a new hook to run before query parse/compile
8c8322c	2020-01-17	HIVE-22151 : Turn off hybrid grace hash join by default (Ashutosh Chauhan via Vineet Garg)

Hive 2.3.6-1912 (EEP 6.3.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.3.6-1912 for EEP 6.3.0.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the following:

- Apache Hive-2.3.6 Release Notes
- Apache Hive-2.3.5 Release Notes
- Apache Hive-2.3.4 Release Notes
- Apache Hive-2.3.3 Release Notes

- Apache Hive-2.3.2 Release Notes
- Apache Hive-2.3.1 Release Notes
- Apache Hive-2.3.0 Release Notes
- Apache Hive homepage

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Hive Version	2.3.6
Release Date	December 2019
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.3.6-mapr-1912
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

ODBC / JDBC Drivers	Hive 2.3.6 works with the following MapR Hive ODBC drivers:
	• Mac OS X
	• Linux
	  • 32-bit
	  • 64-bit
	• Windows
	  • 32-bit
	  • 64-bit
	For additional driver information, see Connecting to HiveServer2 on page 3515.

Feature Support

- MapR supports Hive-2.3.6 on Tez-0.9.1. For more information, see Tez 0.9.1-1912 (EEP 6.3.0) Release Notes on page 6494.
- MapR does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- MapR does not support HDFS encryption in Hive tables.
- MapR supports HBase using Hive 2.3.6 starting with MEP 6.3.0 and MapR Core 6.1.0.
- MapR does not support LLAP with Hive 2.3.6, as Apache Slider is not a MapR ecosystem component.

last-updated: Apr 11, 2022

- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

New Features

- None.

Known Issues

- HIVE-655 - Failed to launch Hive WebHCat job after update. See the workaround in Troubleshooting Hive and Tez on page 3598.
- HIVE-19502 Unable to insert values into table stored by `JdbcStorageHandler`
- HIVE-19286 NPE in `MERGE` operator on MR mode
- PIG-25 – Pig 0.16 failed to use ORC Storage. Pig - Hive dependency issue – MapR Pig does not support MapR Hive ORC integration.
- HIVE-522 – "INSERT INTO" overwrites old data when the destination table is encapsulated by a backquote.

  After HQL harmonization with requirements of the SQL standard, queries with table names or database names that contain a dot (`.`) can fail. For detailed information, see Troubleshooting Hive and Tez on page 3598.

Some select queries can be converted to single FETCH task minimizing latency. Currently, the query should be single sourced not having any subquery and should not have any aggregations or distincts (which incur RS), lateral views, and joins:

- `none:` disable hive.fetch.task.conversion
- `minimal: select` star, filter on partition columns, `LIMIT` only
- `more: SELECT`, filter, `LIMIT` only (support TABLESAMPLE and virtual columns)

Important:

Hive-2.3.6 MEP-6.3.0 requires Pig-0.16 MEP-6.3.0 and will not support Pig-0.16 from old MEPs. For details, see the Pig 0.16.0-1912 Release Notes on page 6318.

Changes in Security with Default Configuration

- Added properties to default `hive-site.xml` configuration on secured cluster. See the following table:

Table

No.	Property	Value
1	`hive.server2.webui.jetty.response.headers.file`	`/opt/mapr/hive/hive-2.3/conf/headers.xml`

- Added properties to default `webhcat-site.xml` configuration on secured cluster. See the following table.

Table

No.	Property	Value
1	`templeton.jetty.response.headers.file`	`/opt/mapr/hive/hive-2.3/conf/headers.xml`

Note that the property above is added both in a Hive update (no version change) and a Hive upgrade (with version change).

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
17fe3ee	2019-11-20	MAPR-HIVE-642: Joda Time Library update for Brazil to v2.10.3.jar
af42452	2019-11-19	MAPR-HIVE-641: CLONE - old jetty version at hive lib
3250b11	2019-07-01	MAPR-HIVE-376 : Jetty web server version for HS2 web UI should be updated
b923f16	2019-10-31	MAPR-HIVE-633: Update HBase dependency to 1.1.13 for MEP-6.3.0
3d3da90	2019-10-22	MAPR-HIVE-618 : Add security headers for WebHCat (Part II) (#845)
431b114	2019-10-18	MAPR-HIVE-617 : Add Junit tests for configure.sh script. Verification of hive-site.xml and webhcat-site.xml
ba53b45	2019-10-18	MAPR-HIVE-628 : Add utility method to represent xml properties as key-value map
74a7408	2019-10-18	MAPR-HIVE-627 : Remove unused option from configure.sh
4fcf1ea	2019-10-02	MAPR-HIVE-618 : Add security headers for WebHCat
df40ce4	2019-09-30	MAPR-HIVE-619 : Unnecessary excessive WebHCat security configuration on secure clusters after security refactoring
1a5bde7	2019-09-23	MAPR-HIVE-616 : Implement SBD behaviour in case of configuring Hive on KRB clusters after security refactoring
6ba42a4	2019-09-20	MAPR-HIVE-566: UDAF throws IllegalArgumentException for a complex input when column stats is not provided
5d464bb	2019-09-19	MAPR-HIVE-614 : Hive on MapR DB JSON table does not honour the column's comment
1e6abbf	2019-09-17	MAPR-HIVE-611: Hive - MapR-DB-json integration. NullPointerException in WHERE clause when no row found
ccdeb4a	2019-09-02	MAPR-HIVE-605 : Support of Avro 1.8

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
f255efd	2019-08-23	MAPR-HIVE-604 : Process kerberos authentication method in HiveConf object
ea4d923	2019-09-05	MAPR-HIVE-580: Hive on Mapr-db JSON table with nested schema showing wrong results
e3a107d	2019-08-02	MAPR-HIVE-593 : Spark thriftserver fails when work with hive-maprdb json table
0ef33d7	2019-07-31	MAPR-HIVE-586: hive.scratch.dir.permission doesn't work
d5455ce	2019-07-27	MAPR-HIVE-587 : Unable to retrieve Data through hcat/Pig to Hive
7417cb6	2019-07-12	MAPR-HIVE-573 : IndexOutOfBoundsException on parsing query
27463ef	2019-07-20	MAPR-HIVE-583 : Configure hive.metastore.authentication for KRB and MAPRSASL security cluster
fec9184	2019-07-17	MAPR-HIVE-559 : Hive configure.sh changes hive-site.xml permissions despite having .customSecure file presence
a48088c	2019-06-14	MAPR-HIVE-544 : Refactor Mapr SASL security implementation
5b5b8dd	2019-06-11	MAPR-HIVE-541 : Hive creates Kerberos-based thrift connection when there is no delegation token on MapR SASL cluster
65bc50a	2019-06-08	MAPR-HIVE-570 : ClassNotFoundException: HiveMapRDBJsonInputFormat in MaprDB-json integration
1e6a377	2019-06-24	MAPR-HIVE-556 : Error while starting Hive CLI on Tez: ERROR exec.Task: Failed to execute tez graph
1d86356	2019-06-17	MAPR-HIVE-520 : Use separate log file for each Hive service
2286789	2019-06-10	MAPR-HIVE-528 : UNION query with regular expressions works in Hive-2.1 and does not work in Hive-2.3
3d110189	2019-06-10	MAPR-HIVE-534 : Change hive-maprdb-json-common version to 2.3.5
dc6e806	2019-05-20	MAPR-HIVE-519 : Hive ORC read INT, BIGINT as NULL for Data created by spark
63b3829	2019-04-22	MAPR-HIVE-509 : MapRDBJsonSplit class should be available for Spark

This release by MapR also includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
f6f1ebb	2019-11-12	HIVE-19085: FastHiveDecimal abs(0) sets sign to +ve (Gopal V, reviewed by Matt McCline)
d5dd4e4	2019-09-06	HIVE-17088: HS2 WebUI throws a NullPointerException when opened (Sergio Pena, reviewed by Aihua Xu)
739b41c	2019-06-27	HIVE-16049: upgrade to jetty 9 (Sean Busbey & Aihua Xu, reviewed by Sergio Peña)
fced649	2019-10-25	HIVE-22107: Correlated subquery producing wrong schema
8bdc8158	2019-10-22	HIVE-8472: Add ALTER DATABASE SET LOCATION (Mithun Radhakrishnan, reviewed by Alan Gates) (#851)
3abf19c	2019-10-22	HIVE-20409: Hive ACID: Update/ delete/merge does not clean hdfs staging directory (Rajkumar Singh, reviewed by Vineet Garg) (#847)
626b50d	2019-10-14	HIVE-18553 : Support schema evolution in Parquet Vectorization reader (Ferdinand Xu, reviewed by Vihang Karajgaonkar)
94190e2	2019-10-14	HIVE-18323 : Vectorization: add the support of timestamp in VectorizedPrimitiveColumnReader for parquet (Vihang Karajgaonkar, reviewed by Aihua Xu and Ferdinand Xu)
cb2eb4d	2019-10-14	HIVE-17931: Implement Parquet vectorization reader for Array type
6c78cdf	2019-10-14	HIVE-17381: When we enable Parquet Writer Version V2, hive throws an exception: Unsupported encoding: DELTA_BYTE_ARRAY. (Colin Ma, reviewed by Ferdinand Xu)
622aab2	2019-10-11	HIVE-17085: ORC file merge/ concatenation should do full schema check (Prasanth Jaychandran reviewed by Zoltan Haindrich)
fece5c9	2019-07-16	HIVE-21980: Parsing time can be high in case of deeply nested subqueries (Zoltan Haindrich reviewed by Vineet Garg)
d6bd1b6	2019-08-13	HIVE-22096 Backport HIVE-21584 to branch-2.3 (Yuming Wang via Alan Gates)
fd97a4e	2019-08-30	HIVE-21859 Backport HIVE-17466 (get_partition_values) to branch-2.3 (Piotr Findeisen via Alan Gates)

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
6752b7b	2019-08-30	HIVE-21786 : Update repo URLs in poms branch 2.3 version
a3860a7	2019-07-31	HIVE-4605 : Hive job fails while closing reducer output - Unable to rename
6a73e80	2019-07-26	HIVE-14557: Nullpointer When both SkewJoin and Mapjoin Enabled
ef3a1f5	2019-07-25	HIVE-21540: Query with join condition having date literal throws SemanticException (Sankar Hariappan, reviewed by Zoltan Haindrich)
3d11872	2019-07-20	HIVE-19990: Query with interval literal in join condition fails(Vineet Garg, reviewed by Zoltan Haindrich)
f26f063	2019-07-23	HIVE-18786 : NPE in Hive windowing functions (Dongwook Kwon via Ashutosh Chauhan)
45b037e	2019-06-24	HIVE-21104: PTF with nested structure throws ClassCastException (Rajesh Balamohan reviewed by Gopal
dee97c4	2019-06-22	HIVE-20318 : NullPointerException when union with lateral view
7e53093	2019-06-22	HIVE-16334: Query lock contains the query string, which can cause OOM on ZooKeeper (Peter Vary via Chaoyu Tang)
1616d4d	2019-06-19	HIVE-16907: "INSERT INTO" overwrite old data when destination table encapsulated by backquote
9c26a52	2019-06-19	HIVE-6590 : Hive does not work properly with boolean partition columns (wrong results and inserts to incorrect HDFS path)
1e4cb68	2019-06-19	HIVE-18907: Create utility to fix acid key index issue from HIVE-18817 (Jason Dere, reviewed by Prasanth Jayachandran)
0ae141a	2019-06-19	HIVE-18817 - ArrayIndexOutOfBounds exception during read of ACID table. (Eugene Koifman, Jason Dere, Prasanth Jayachandran, reviewed by Jason Dere)
c7f7fd6	2019-05-03	HIVE-21680 Backport HIVE-17644 to branch-2.3 (Yuming Wang via Alan Gates)
e822e4f	2019-04-25	Backport HIVE-17764 to branch-2.3 (Yuming Wang via Alan Gates)
20e7e54	2019-04-24	HIVE-21639 Spark test failed since HIVE-10632 (Yuming Wang via Alan Gates)

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
8396169	2019-06-19	HIVE-18624: Parsing time is extremely high (~10 min) for queries with complex select expressions (Zoltan Haindrich reviewed by Ashutosh Chauhan)

Resolved Issues

- None.

Hive 2.3.3 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.3.3.

The following release notes for the Hive 2.3.3 component are included in the MapR distribution for Apache Hadoop:

Hive 2.3.3-1904 (EEP 6.2.0, EEP 6.1.1, and EEP 6.0.2) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.3.3-1904.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Hive Version	2.3.3
Release Date	April 2019
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.3.3-mapr-1904
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

ODBC/JDBC Drivers	Hive 2.3.3 works with the following MapR Hive drivers:
	• ODBC Drivers
	  • Mac OS X
	  • Linux
	  • 32-bit
	  • 64-bit
	  • Windows
	  • 32-bit
	  • 64-bit
	For additional driver information, see Connecting to HiveServer2.

Feature Support

- EEP 6.1.0 supports Hive-2.3.3 on Tez-0.9.

  For more information, see Tez 0.9.1-1904 (EEP 6.2.0, EEP 6.1.1, and EEP 6.0.2) Release Notes on page 6495.

- EEP 6.1.0 does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive.

  However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

- EEP 6.x does not support HDFS encryption in Hive tables.
- EEP 6.x does not support LLAP with Hive-2.3.3, because Apache Slider is not a MapR ecosystem component.
- Hive 2.1 and later needs to run the `schematool` command as an initialization step.

New Features

- MapR Database JSON projection pushdown.
- Metrics report file `/tmp/hive_report.json` is split: `/tmp/hiveserver2_report.json` and `/tmp/hivemetastore_report.json` for HiveServer2 and Hive Metastore, respectively.

Changes in Security with Default Configuration

- Added the following properties to the `hive-site.xml` configuration by default on a secured cluster:

Table

Property	Value
`hive.server2.metrics.file.location`	`/tmp/hiveserver2_report.json`
`hive.metastore.metrics.file.location`	`/tmp/hivemetastore_report.json`

- Removed the following property from the `hive-site.xml` configuration by default on a secured cluster:

Table

Property	Value
`hive.service.metrics.file.location`	`/tmp/hive_report.json`

API Changes

The following classes are moved from `hive-maprdb-json-handler-2.3.3-mapr-XXXX.jar` to `hive-exec-2.3.3-mapr-XXXX.jar`:

- `org.apache.hadoop.hive.maprdb.json.shims.DocumentWritable`.
- `org.apache.hadoop.hive.maprdb.json.shims.MapRDBJsonSplit`.
- `org.apache.hadoop.hive.maprdb.json.shims.MapRDBProxy`.
- `org.apache.hadoop.hive.maprdb.json.shims.RecordReaderWrapper`.
- `org.apache.hadoop.hive.maprdb.json.shims.RecordWriterWrapper`.

last-updated: Apr 11, 2022

Known Issues

- In HIVE-19502, you cannot insert values into a table stored by `JdbcStorageHandler`.
- In HIVE-19286, NPE in `MERGE` operator on MR mode.
- In Bug 32349, [6.1RC1] Simple fetch from MapR Database JSON tables does not work. Workaround: Set `hive.fetch.task.conversion=none` in the `hive-site.xml` file or in the Hive CLI.
- Some select queries can be converted to single FETCH task minimizing latency. Currently the query should be single sourced not having any sub query and should not have any aggregations or distincts (which incurs RS), lateral views and joins:
  - `none`: Disable `hive.fetch.task.conversion`
  - `minimal`: SELECT star, filter on partition columns, LIMIT only
  - `more`: SELECT, filter, LIMIT only (support TABLESAMPLE and virtual columns)
- The Hive vectorized execution feature has many bugs in Hive 2.x. It is recommended to turn off this feature at a system level and only use it for certain queries which work fine using it. You must evaluate the benefit of this feature against the potential stability issues on a case by case basis.
- Spark does not support SSL encryption for metastore when `hive.metastore.use.SSL = true` is used.
  - SPARK-533 Spark Thriftserver fails with LDAP+KERBEROS+SSL configuration.

Fixes

This release includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
4d38efc 787b53a	2019-05-17	MAPR-HIVE-499 : Most information is lost when hive log4j2 routing appender rotated logs
4a9bbca	2019-04-16	MAPR-HIVE-507 : hive.metastore.use.SSL should be set to false by default
e379881	2019-03-27	MAPR-HIVE-462: INSERT OVERWRITE LOCAL DIRECTORY fails with permission error in hive-2.3
715c0e7	2019-04-15	MAPR-HIVE-500 : CLONE - Hive timers metrics availability is not constant
e12fa76	2019-04-12	MAPR-HIVE-503 : Rename webhcat.pid to hive-mapr-webhcat.pid
68d2f5f	2019-04-08	MAPR-HIVE-476 : There is no encryption between client(HS2) and HMS server while working through maprsasl security
0f3fbbf	2019-03-29	MAPR-HIVE-485 : Disable vectorized execution in Hive by default

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
c6e95ae	2019-03-23	MAPR-HIVE-482 : HS2 takes time to start because of the 'get_all_databases'
c8a4882	2019-03-19	MAPR-HIVE-471 : Distribute Notice.txt across components starting with MEP 6.2
a062e71	2019-03-14	MAPR-HIVE-474 : Webhcat SSL doesn't have a valid keystore
b3e7287	2019-03-13	MAPR-HIVE-475 : Change tez version in Hive-2.3 to 0.9.1
f0c1053	2019-03-12	MAPR-HIVE-457 : Hive MR job fails with NullPointerException if we execute cleardanglingscratchdir
24c17f8	2019-03-06	MAPR-HIVE-465 : Investigate error logs in Hive Metastore after implementing Thrift v0.12.0
e58ca04	2019-03-06	MAPR-HIVE-464 : Backup hive-env.sh during backup files process
b0d4b45	2019-03-04	MAPR-HIVE-432 : CLONE - CVE-2018-1320 vulnerability in Apache Thrift
a19f0f9	2019-02-19	MAPR-HIVE-393 : Implement id / key projection pushdown for Hive - MapR-DB JSON Integration
250c986	2019-02-20	MAPR-HIVE-434 : 'Australia/Sydney' timezone conversion issues
cd2c5d5	2019-02-14	MAPR-HIVE-448 : Backup Hive configuration files only if they were changed
0449328	2019-02-14	MAPR-HIVE-449 : absence of webhcat pid file under $MAPR_PID_DIR
dfea394	2019-01-31	MAPR-HIVE-431 : Remove "hive-log4j.properties" file from $HIVE_CONF directory
0e5d4c5	2019-01-29	MAPR-HIVE-442 : SC2145: Argument mixes string and array. Use * or separate argument
4e63a15	2019-02-07	MAPR-HIVE-307 : Add -SNAPSHOT to Hive version

This release also includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date	Comment
636a671	2019-04-18	HIVE-19018: beeline -e now requires semicolon even when used with query from command line

Commit	Date	Comment
4cdb1ad	2019-03-26	HIVE-16958: Setting hive.merge.sparkfiles=true will return an error when generating parquet databases
12b1b39	2019-03-26	HIVE-20126: OrcInputFormat does not pass conf to orc reader options
d66b657	2019-03-26	HIVE-20091: Tez: Add security credentials for FileSinkOperator output

Related Links

Following are release notes for the Hive component included in the MapR Converged Data Platform. You might also be interested in the following documents:

- Apache Hive 2.3.3 Release Notes
- Apache Hive 2.3.2 Release Notes
- Apache Hive 2.3.1 Release Notes
- Apache Hive 2.3.0 Release Notes

You can also refer to the Apache Hive homepage.

Hive 2.3.3-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes
This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.3.3-1901.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Hive Version	2.3.3
Release Date	February 2019
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.3.3-mapr-1901
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

ODBC/JDBC Drivers	Hive 2.3.3 works with the following MapR Hive drivers:
	• ODBC Drivers
	  • Mac OS X
	  • Linux
	  • 32-bit
	  • 64-bit
	  • Windows
	  • 32-bit
	  • 64-bit
	For additional driver information, see Connecting to HiveServer2.

Feature Support

- EEP 6.1.0 supports Hive-2.3.3 on Tez-0.9.

  For more information, see Tez 0.9.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6495.

- EEP 6.1.0 does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive.

  However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

- EEP 6.1.0 does not support HDFS encryption in Hive tables.
- EEP 6.1.0 does not support HBase with Hive-2.3.3 starting from the 6.0.0 release.
- EEP 6.0.0 does not support LLAP with Hive-2.3.3, because Apache Slider is not a MapR ecosystem component.
- Hive 2.1 and later needs to run the `schematool` command as an initialization step.

New Features

- Implemented preserving warden configuration files during package update.
- Backported FallbackHiveAuthorizerFactory in scope of CVE-2018-11777. For more information, see Fallback Hive Authorizer on page 3458.

Changes in Security with Default Configuration

- The following properties are added to the `hive-site.xml` configuration by default on a secured cluster:

```
<property>
  <name>hive.users.in.admin.role</name>
  <value>mapr</value>
</property>

<property>
  <name>hive.conf.restricted.list</name>
  <value>hive.security.authenticator.manager,
    hive.security.authorization.manager,
    hive.security.metastore.authorization.manager,
    hive.security.metastore.authenticator.manager,
    hive.users.in.admin.role,
    hive.server2.xsrf.filter.enabled,
    hive.security.authorization.enabled,
    hive.server2.authentication.ldap.baseDN,
    hive.server2.authentication.ldap.url,
    hive.server2.authentication.ldap.Domain,
    hive.server2.authentication.ldap.groupDNPattern,
    hive.server2.authentication.ldap.groupFilter,
    hive.server2.authentication.ldap.userDNPattern,
    hive.server2.authentication.ldap.userFilter,
    hive.server2.authentication.ldap.groupMembershipKey,
    hive.server2.authentication.ldap.userMembershipKey,
    hive.server2.authentication.ldap.groupClassKey,
    hive.server2.authentication.ldap.customLDAPQuery,
    hive.exec.pre.hooks,
    hive.exec.post.hooks,
    hive.exec.failure.hooks,
    hive.exec.query.redactor.hooks,
    hive.semantic.analyzer.hook,
    hive.query.lifetime.hooks,
    hive.exec.driver.run.hooks,
    hive.server2.session.hook</value>
</property>

<property>
  <name>hive.security.authorization.enabled</name>
  <value>true</value>
</property>

<property>
  <name>hive.security.authorization.manager</name>

<value>org.apache.hadoop.hive.ql.security.authorization.plugin.fallback.Fa
llbackHiveAuthorizerFactory</value>
</property>

<property>
  <name>hive.server2.metrics.enabled</name>
  <value>true</value>
</property>

<property>
  <name>hive.service.metrics.reporter</name>
  <value>JSON_FILE,JMX</value>
</property>

<property>
```


last-updated: Apr 11, 2022

```
  <name>hive.service.metrics.file.location</name>
  <value>/tmp/hive_report.json</value>
</property>
```

Known Issues

- In HIVE-19502, you cannot insert values into a table stored by `JdbcStorageHandler`.
- In HIVE-19286, NPE in `MERGE` operator on MR mode.
- In Bug 32349, [6.1RC1] Simple fetch from MapR Database JSON tables does not work. Workaround: Set `hive.fetch.task.conversion=none` in the `hive-site.xml` file or in the Hive CLI.
- Some select queries can be converted to single FETCH task minimizing latency. Currently the query should be single sourced not having any sub query and should not have any aggregations or distincts (which incurs RS), lateral views and joins:
  - `none`: Disable `hive.fetch.task.conversion`
  - `minimal`: SELECT star, filter on partition columns, LIMIT only
  - `more`: SELECT, filter, LIMIT only (support TABLESAMPLE and virtual columns)
- Vectorized execution is a new Hive feature that can show performance improvements in some cases and cause stability issues with others. The Hive vectorized execution feature has many bugs in Hive 2.x. It is recommended to turn off this feature at a system level and only use it for certain queries which work fine using it. You must evaluate the benefit of this feature against the potential stability issues on a case by case basis.

Fixes

This release includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
adc7e1e	2018-08-23	MAPR-HIVE-286: Error in TestVectorStringExpressions.TestVectorStringExpressions while runnimg on Jenkins
c3e3f8b	2018-08-20	MAPR-HIVE-304 : Hive configure.sh ignores .customSecure file presence
b5b29bf	2018-08-21	MAPR-32194 : Hardcoded reference to hdfs in Hive
9d9f022	2018-08-24	MAPR-HIVE-31529: GenericUDFNamedStruct should constant fold at compile time
4b53647	2018-09-18	MAPR-HIVE-320 : Force Hive not to care about columns case when reading from MapR DB JSON
66b5639	2018-09-24	MAPR-HIVE-317: Enable StorageBasedAuthorizationProvider in scope of SBD
e1f2e8f	2018-09-26	MAPR-HIVE-322: Add the -Dfs.cache.lru.enable=true parameter in hive-env.sh by default in future versions

Commit	Date (YYYY-MM-DD)	Comment
dab0389	2018-10-01	MAPR-HIVE-294: Authorization issue while trying to obtain logs in HS2 Web UI
fe60205	2018-10-02	MAPR-HIVE-327 : Remove SNAPSHOT from mapr-core-6.1.0 dependency
0e68ee2	2018-10-03	MAPR-HIVE-329 : Incorrect verification of customSecure parameter during hive configure.sh
94a69a1	2018-10-03	MAPR-31413: add "explain ast"
fe60205	2018-10-04	MAPR-HIVE-330 : Remove SNAPSHOT from zookeeper dependency
6b12833	2018-10-04	MAPR-HIVE-311 : Set up hive.conf.restricted.list configuration for clusters with MapR SASL security in scope of SBD
9391140	2018-10-05	MAPR-HIVE-331 : Refactor conftool. Use enum instead of boolean for security variable
0c236c6	2018-10-15	MAPR-HIVE-340 : configure.sh should keep custom configs in warden.hs2/hivemeta.conf
7f12ec8	2018-11-15	MAPR-HIVE-379 : Warden services do not copy while running configure.sh -R after update. Cannot run Hive after update
0ab9b97	2018-10-31	MAPR-HIVE-367 : Extend hive.conf.restricted.list configuration for clusters with MapR SASL security in scope of SBD
5a00cb5	2018-10-23	MAPR-HIVE-346 : Use rename of files where it is possible instead of full copy during moving data from staging dirs
ca44eeb	2018-11-16	MAPR-HIVE-385 : CVE-2018-11777: Blocking local resource access in HiveServer2
ea74298	2018-11-28	MAPR-HIVE-391 : Configure Hive service for collecting metrics
a2de839	2018-12-11	MAPR-HIVE-409 : Hive Metastore schema upgrade from Hive 2.1 to Hive 2.3 fails with an exception "ORA-00911: invalid character" on Oracle Database
5731f91	2018-12-17	MAPR-HIVE-378 : Hive displays NULLS instead of Decimal values in external table (MabRDb Binary Table column has BigDecimal)
e0cb75e	2018-12-18	MAPR-HIVE-413 : Remove duplicates from ConfTool Junit tests

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
0587434	2018-12-18	MAPR-HIVE-406 : Configuration of Hive for creating json report with available metrics
61b75da	2018-12-19	MAPR-HIVE-404 : Unnecessary configure Hive for collecting metrics during minor version update
bdce35c	2018-12-18	MAPR-HIVE-389 Develop the logic for preserving custom configuration in Hive warden conf
0c67732	2018-12-28	MAPR-HIVE-323 : Hive Metastore failed to start on Derby DB after running fresh configure.sh -R
b0cb2b3	2019-01-08	MAPR-HIVE-416: unable to apply custom security. .customSecure is ignored by Hive configure.sh
6d10d94	2019-01-09	MAPR-HIVE-395 : hive.warehouse.subdir.inherit.perms and hive.optimize.insert.dest.volume don't work on CTAS query

This release also includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date	Comment
177bb3f	2018-08-31	HIVE-16788: ODBC call SQLForeignKeys leads to NPE if you use PK arguments rather than FK arguments
e1f2e8f	2018-09-26	HIVE-18767: Some alterPartitions invocations throw 'NumberFormatException: null'
fe60205	2018-10-23	HIVE-20420: Provide a fallback authorizer when no other authorizer is in use
ab23698	2018-10-25	HIVE-18778: Needs to capture input/output entities in explain
d50f28f	2018-10-01	HIVE-17333 Oracle does not allow change from VARCHAR2 to CLOB for upgrade
13d0605	2018-11-20	HIVE-17631 : upgrade orc to 1.4.1
d50f28f	2018-12-17	HIVE-18558: Upgrade orc version to 1.4.2
3436caa	2018-12-17	HIVE-18674 : update Hive to use ORC 1.4.3
2ad14f1	2018-12-17	HIVE-19465: Upgrade ORC to 1.5.0
c24a5e7	2018-12-17	HIVE-19226: Extend storage-api to print timestamp values in UTC
763aaf8	2018-12-17	HIVE-19669: Upgrade ORC to 1.5.1
c80b2f4	2018-12-17	HIVE-18007 : Address maven warnings

Commit	Date	Comment
d4b3ee6	2018-12-20	HIVE-17272: when hive.vectorized.execution.enabled is true, query on empty partitioned table fails with NPE

Related Links

Following are release notes for the Hive component included in the MapR Converged Data Platform. You might also be interested in the following documents:

- Apache Hive 2.3.3 Release Notes
- Apache Hive 2.3.2 Release Notes
- Apache Hive 2.3.1 Release Notes
- Apache Hive 2.3.0 Release Notes

You can also refer to the Apache Hive homepage.

Hive 2.3.3-1808 (EEP 6.0.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.3.3-1808 EEP 6.0.0.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Hive Version	2.3.3
Release Date	September 2018
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive/tree/2.3.3-mapr-1808
GitHub Release Tag	2.3.3-mapr-1808
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

Feature Support

- EEP 6.0.0 supports Hive-2.3.3 on Tez-0.9.

  For more information, see Tez 0.9.1-1808 (EEP 6.0.0) Release Notes on page 6496.

- EEP 6.0.0 does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive.

  However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

- EEP 6.0.0 does not support HDFS encryption in Hive tables.
- EEP 6.0.0 does not support HBase with Hive-2.3.3 starting from the 6.0.0 release.

last-updated: Apr 11, 2022

- EEP 6.0.0 does not support LLAP with Hive-2.3.3, because Apache Slider is not a MapR ecosystem component.
- Hive 2.1 and later needs to run the `schematool` command as an initialization step.

New Features

- `MERGE` operator for MapR Database JSON Tables in Tez mode

  For more information, see Understanding the MERGE Statement on page 3481.
- `hive.metastore.allow.new.db.in.existing.directory` property with default value as false

  You must set `hive.metastore.allow.new.db.in.existing.directory=true` in the `hive-site.xml` file to create a database if the directory already exists and then restart Hive Metastore and HiveServer2.

  CAUTION: You cannot set the `hive.metastore.allow.new.db.in.existing.directory` using the Hive CLI or Beeline.
- Ability to read default SSL passwords and keystore location from `/opt/mapr/conf/ssl-client.xml`

Changes in Security with Default Configuration

- The following properties are removed from the default `hive-site.xml` configuration on a secured cluster:

Property	Value
`hive.server2.webui.keystore.path`	`/opt/mapr/conf/ssl_keystore`
`hive.server2.webui.keystore.password`	`Default keystore password`

- The following property is added to the `hive-site.xml` configuration by default on a secured cluster:

Property	Value
`hive.server2.use.SSL`	`true`

- Because HiveServer2 is configured to use SSL encryption by default starting from Hive-2.3 MEP-6.0.0, you must add `ssl=true;` to a JDBC connection string when using PAM or MapR-SASL authentication.
- You can configure the JDBC connection string with SSL enabled or disabled.

  For more information, refer to Configuring JDBC Connection String with SSL Encryption Enabled or Disabled on page 3521.
- World-readable permissions enabled for Hive configuration files.

Known Issues

- In HIVE-19502, you cannot insert values into a table stored by `JdbcStorageHandler`.
- You cannot connect to a Spark Thrift Server on a Kerberos-secured cluster, because Kerberos and SSL are not compatible.

  However, you can modify the `hive.server2.use.SSL` to `false` in the `hive-site.xml` file.
- In MAPR-HIVE-302, you cannot interact with Hive tables on PostgreSQL after upgrade.

- In Bug 32349, Simple Fetch from MapR Database JSON tables does not work in the Hive 2.3 release. **Workaround**: Set `hive.fetch.task.conversion=none` in the `hive-site.xml` file or using the Hive CLI. The `hive.fetch.task.conversion` property is used for query conversion. Some select queries can be converted to a single FETCH task that minimizes the latency. Currently, the query should be single sourced and not have any subquery and also must not have any aggregations or distincts (which incurs RS - ReduceSinkOperator, requiring a MapReduce task), lateral views and joins:
  - `none`: disable hive.fetch.task.conversion
  - `minimal`: SELECT star, filter on partition columns, LIMIT only
  - `more`: SELECT, filter, LIMIT only (supports TABLESAMPLE and virtual columns)
- A non-administrative user can install hooks, which could represent a security vulnerability. See Preventing a Non-Administrative User from Installing Hooks on page 3429 for more information.
- Vectorized execution is a new Hive feature that can show performance improvements in some cases and cause stability issues with others. The Hive vectorized execution feature has many bugs in Hive 2.x. It is recommended to turn off this feature at a system level and only use it for certain queries which work fine using it. You must evaluate the benefit of this feature against the potential stability issues on a case by case basis.

Fixes

The Hive 2.3.3-1808 (EEP 6.0.0) release includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
5ea2352	2018-08-08	MAPR-HIVE-300: Hive HCat does not create a proper "client" on MapR-SASL cluster without Hive metastore
c50c43b	2018-07-24	MSEN-11: You can save a file with a newline at the end
d009cf0	2018-07-21	MAPR-HIVE-281: WebHCat issues an error during an attempt to run a Hive query
56ea17e	2018-07-17	MAPR-HIVE-285: The configure.sh fails when a custom security flag is set
34180bf	2018-07-09	MAPR-HIVE-272: The HiveMapRDBJsonOutputFormat class must implement the HiveOutputFormat interface
af94d85	2018-06-17	MAPR-31641: The failed queries with invalid table alias or column reference are deleted
9c572f7	2018-06-27	MAPR-31803: The fix for Bug 30031 results in users not being able to create databases on an existing location
1855442	2018-06-21	MAPR-HIVE-242: The configure.sh script adds two rows to the end of hive-site.xml after every launching

Commit	Date (YYYY-MM-DD)	Comment
cf0655e	2018-06-19	MAPR-HIVE-258: The Zookeeper version in Hive is updated from 3.4.6 to 3.4.11
61773b5	2018-06-19	MAPR-HIVE-244: The Hive configure.sh script overwrites properties by secure by default values and does not take into account the .not_configured_yet flag
836f751	2018-06-19	MAPR-HIVE-251: The default SSL password is read from the /opt/mapr/conf/ssl-server.xml
28025b1	2018-06-22	MAPR-HIVE-262: Only client passwords from ssl-client.xml are used
3c463a5	2018-06-01	MAPR-HIVE-223: NPE during CREATE ROLE uses SQL Standard-Based Hive Authorization
0a458b4	2018-05-22	MAPR-31380: The HeartBeat thread uses a cancelled delegation token while connecting to meta
bc7a6bd	2018-05-22	MAPR-HIVE-212: SSL is setup by default when PAM plus MapR-SASL is configured by default
39ae559	2018-05-12	MAPR-HIVE-184: SSL encryption with PAM authentication is configured on the MapR-SASL secured cluster
1f92945	2018-05-07	MAPR-HIVE-208: Import to Hive as a parquet data format failed and could not initialize class org.apache.derby.jdbc.AutoloadedDriver40
f371524	2018-04-17	MAPR-HIVE-193: The MERGE syntax for MapR Database JSON tables is implemented
dfc7cfd	2018-04-24	MAPR-31175: The hive.exec.tmp.maprfsvolume should be false when operating with Tez
19fd04f	2018-04-13	MAPR-HIVE-196: Running the configure.sh -R reconfigures the underlying MetastoreDB to derby in the hive-site.xml file
5cfebda	2018-04-12	MAPR-27663: PidFilePatternConverter does not append the PID to the log name
d842904	2018-04-11	MAPR-HIVE-190: The log writes in two Hive.log files instead of one
61735a1	2018-04-11	MAPR-HIVE-171: A bash script for MapR configuration tool is added
1f12b05	2018-04-11	MAPR-HIVE-167: A sub-module API for configuring hadoop.security.credential.provider.path is added into the MapR configuration

Commit	Date (YYYY-MM-DD)	Comment
72c75e7	2018-04-11	MAPR-HIVE-165: The HiveEncryption tool is moved to a separate sub-module and split into a CLI processing class and an Encryption processing class
bd611a2	2018-04-10	MAPR-HIVE-189: World-readable permissions for Hive configuration files are set
efd951f	2018-04-03	MAPR-HIVE-187: The Hive CLI does not start on a 6.0.1 cluster with Spark 2.2.1
4f23f26	2018-03-21	MAPR-30940: A Hive job fails by AccessControlException against files on the NM local disk
c9445f0	2018-03-20	MAPR-30895: Tez jobs are shown as KILLED in a RM UI

The Hive 2.3.3-1808 (EEP 6.0.0) release also includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date	Comment
5b67110	2018-08-07	HIVE-12408: SQLStdAuthorizer does not require an external table creator to be the owner of directory, in addition to read or write permissions
e57f603	2018-07-25	HIVE-17429: Hive JDBC does not return rows when querying Impala
d1c113d	2018-07-24	HIVE-18393: Error returned when some other type is read as string from parquet tables
224b05b	2018-06-25	HIVE-13000: Hive returns useless parsing error
3b0918d	2018-06-13	HIVE-17963: Fix for HIVE-17113 is improved for non-blobstore file systems
d313347	2018-06-12	HIVE-17113: Duplicate bucket files are written to table by runaway task
3eaba39	2018-05-31	HIVE-17155: findConfFile() in HiveConf.java has issues with the configuration path
800c6e5	2018-05-25	HIVE-19649: Clean up inputs in JDBC PreparedStatement. Add unit tests
24165d1	2018-04-05	HIVE-18710: Extend inheritPerms to ACID in Hive 2.X

Related Links

Following are release notes for the Hive component included in the MapR Converged Data Platform. You might also be interested in the following documents:

- Apache Hive 2.3.3 Release Notes

last-updated: Apr 11, 2022

- Apache Hive 2.3.2 Release Notes
- Apache Hive 2.3.1 Release Notes
- Apache Hive 2.3.0 Release Notes

You can also refer to the Apache Hive homepage.

Hive 2.1.1 Release Notes
This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.1.1.

The following release notes for the Hive 2.1.1 component are included in the MapR distribution for Apache Hadoop:

Hive 2.1.1-2009 (EEP 5.0.5) Release Notes
This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.1.1-2009 in EEP 5.0.5.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 2.1.1 Release Notes or the Apache Hive homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Hive Version	2.1.1
Release Date	September 2020
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.1.1-mapr-2009
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

Feature Support

- MapR supports Hive-2.1.1 on Tez-0.8.4. For more information, see Tez 0.8.4-2009 (MEP 5.0.5) Release Notes.
- MapR does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- MapR does not support HDFS encryption in Hive tables.
- MapR does not support HBase with Hive-2.1.1 starting from mapr-core-6.0.0.
- MapR does not support LLAP with Hive-2.1.1, as Apache Slider is not a MapR ecosystem component.
- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

New Features

- None.

Changes in Security with Default Configuration

- None.

Important:

Because `hive.users.in.admin.role` and `hive.security.authorization.manager` were added to the default Hive configuration, some actions in Hive are restricted according to security best practices and CVE-2018-11777

The new class `FallbackHiveAuthorizerFactory` does the following to mitigate the above-mentioned CVE-2018-11777:

- Disallows local file location in SQL statements for all except the administrator.
- Allows `set` only selected white list parameters.
- Disallows `dfs` commands for all except the administrator.
- Disallows `ADD JAR` statements for all except the administrator.
- Disallows `COMPILE` statements for all except the administrator.
- Disallows `TRANSFORM` statements.

For more information, refer to the documentation that describes `FallbackHiveAuthorizerFactory`.

Known Issues

- None.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
f0bb560a	2020-07-30	MAPR-HIVE-782 : Failed to start Hive services due to Password file not found: jmxremote.password [non-secure cluster]
27f9c9a8	2020-05-22	MAPR-HIVE-687 : Hiveserver2 Java JMX Server Insecure Configuration Remote Code Execution Vulnerability
a640d53	2020-05-12	MAPR-HIVE-730 : Update MapR Maven repository URL to use https
3cb01c9f	2019-12-06	MAPR-HIVE-652: SQL Standard Based Authorization does not allow to create a function over HiveServer2 even to Admin user

This release by MapR also includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
0f2eeb4b	2020-07-13	HIVE-18699: Check for duplicate partitions in HiveMetastore.exchange_partitions (Marta Kuczora, reviewed by Adam Szita, Peter Vary)
88661e19d76	2020-02-26	HIVE-16321 Possible deadlock in metastore with Acid enabled
4e60249	2019-12-17	HIVE-22151 : Turn off hybrid grace hash join by default (Ashutosh Chauhan via Vineet Garg)
40792d03	2019-12-17	HIVE-15338: Wrong result from non-vectorized DATEDIFF with scalar parameter of type DATE/TIMESTAMP (Matt McCline, reviewed by Jason Dere)

Hive 2.1.1-1912 (EEP 5.0.4) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.1.1-1912 for EEP 5.0.4.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 2.1.1 Release Notes or the Apache Hive homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Hive Version	2.1.1
Release Date	December 2019
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.1.1-mapr-1912
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

Feature Support

- MapR supports Hive-2.1.1 on Tez-0.8.4. For more information, see Tez 0.8.4-1912 (EEP 5.0.4) Release Notes on page 6498.
- MapR does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- MapR does not support HDFS encryption in Hive tables.
- MapR does not support HBase with Hive-2.1.1 starting from mapr-core-6.0.0.
- MapR does not support LLAP with Hive-2.1.1, as Apache Slider is not a MapR ecosystem component.

- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

New Features

- None.

Changes in Security with Default Configuration

- None.

Important:

Because `hive.users.in.admin.role` and `hive.security.authorization.manager` were added to the default Hive configuration, some actions in Hive are restricted according to security best practices and CVE-2018-11777

The new class `FallbackHiveAuthorizerFactory` does the following to mitigate the above-mentioned CVE-2018-11777:

- Disallows local file location in SQL statements for all except the administrator.
- Allows `set` only selected white list parameters.
- Disallows `dfs` commands for all except the administrator.
- Disallows `ADD JAR` statements for all except the administrator.
- Disallows `COMPILE` statements for all except the administrator.
- Disallows `TRANSFORM` statements.

For more information, see Action Restrictions with Fallback Hive Authorizer on page 3459.

Known Issues

PIG-25 – Pig 0.16 failed to use ORC Storage. Pig - Hive dependency issue – MapR Pig does not support MapR Hive ORC integration.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
6593151	2019-11-19	MAPR-HIVE-642: Joda Time Library update for Brazil to v2.10.3.jar
3ecfd64	2019-09-19	MAPR-HIVE-614: Hive on MapR DB JSON table does not honour the column's comment
4b0749d	2019-09-02	MAPR-HIVE-605: Support of Avro 1.8
ca70108	2019-09-05	MAPR-HIVE-580: Hive on Mapr-db JSON table with nested schema showing wrong results
74c6819	2019-07-17	MAPR-HIVE-559: Hive configure.sh changes hive-site.xml permissions despite having .customSecure file presence

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
824327a	2019-06-24	MAPR-HIVE-556: Error while starting Hive CLI on Tez: ERROR exec. Task: Failed to execute tez graph

This release by MapR also includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
308acbc	2019-11-04	HIVE-14817: Shutdown the SessionManager timeoutChecker thread properly upon shutdown. (Siddharth Seth, reviewed by Thejas Nair)
d147012	2019-06-22	HIVE-16334: Query lock contains the query string, which can cause OOM on ZooKeeper (Peter Vary via Chaoyu Tang)
ca75435	2019-06-07	HIVE-16907: "INSERT INTO" overwrite old data when destination table encapsulated by backquote
68864ac	2019-06-07	HIVE-14360: Starting BeeLine after using !save, there is an error logged: "Error setting configuration: conf"

Resolved Issues

- None.

Hive 2.1.1-1904 (EEP 5.0.3 and EEP 4.1.4) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.1.1-1904.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 2.1.1 Release Notes or the Apache Hive homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Hive Version	2.1.1
Release Date	April 2019
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.1.1-mapr-1904
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

Feature Support

- MapR supports Hive-2.1.1 on Tez-0.8.4. For more information, see Tez 0.8.4-1901 (EEP 4.1.3 and EEP 5.0.2) Release Notes on page 6499.
- MapR does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- MapR does not support HDFS encryption in Hive tables.
- MapR does not support HBase with Hive-2.1.1 starting from mapr-core-6.0.0.
- MapR does not support LLAP with Hive-2.1.1 as Apache Slider is not a MapR ecosystem component.
- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

New Features

- None.

Changes in Security with Default Configuration

- None.

Because `hive.users.in.admin.role` and `hive.security.authorization.manager` were added to default Hive configuration, some actions in Hive are restricted according to security best practices and CVE-2018-11777

The new class `FallbackHiveAuthorizerFactory` will do the following to mitigate the above-mentioned CVE-2018-11777:

- Disallow local file location in SQL statements for all except the administrator.
- Allow `set` only selected white list parameters.
- Disallow `dfs` commands for all except the administrator.
- Disallow `ADD JAR` statements for all except the administrator.
- Disallow `COMPILE` statements for all except the administrator.
- Disallow `TRANSFORM` statements.

For more information, see Action Restrictions with Fallback Hive Authorizer on page 3459.

Known Issues

- Vectorized execution is a new Hive feature that can show performance improvements in some cases and cause stability issues with others. The Hive vectorized execution feature has many bugs in Hive 2.x. It is recommended to turn off this feature at a system level and only use it for certain queries which work fine using it. You must evaluate the benefit of this feature against the potential stability issues on a case by case basis.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
274aaff 060fff6	2019-05-17	MAPR-HIVE-499 : Most information is lost when hive log4j2 routing appender rotated logs
15c7b31	2019-03-18	MAPR-HIVE-478 : Use mapr tez dependency in Hive-2.1
4da18a	2019-03-12	MAPR-HIVE-457 : Hive MR job fails with NullPointerException if we execute cleardanglingscratchdir
7a30356	2019-03-06	MAPR-HIVE-465 : Investigate error logs in Hive Metastore after implementing Thrift v0.12.0
c3abff2	2019-03-04	MAPR-HIVE-432 : CLONE - CVE-2018-1320 vulnerability in Apache Thrift
48b1e33	2019-02-25	MAPR-HIVE-449 : absence of webhcat pid file under $MAPR_PID_DIR
5365c0b	2019-02-15	MAPR-HIVE-434 : 'Australia/Sydney' timezone conversion issues
cc4a278	2019-02-07	MAPR-HIVE-307 : Add -SNAPSHOT to Hive version

This release by MapR also includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
5665541	2019-03-19	HIVE-16196 : UDFJson having thread-safety issues
958f74e	2019-02-25	HIVE-15291 : Comparison of timestamp fails if only date part is provided
acc60f8	2019-02-08	HIVE-17640: Comparison of date return null if time part is provided in string.

Resolved Issues

- None.

Hive 2.1.1-1901 (EEP 4.1.3 and EEP 5.0.2) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.1-1901.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 2.1.1 Release Notes or the Apache Hive homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Hive Version	2.1.1
Release Date	February 2019

MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.1.1-mapr-1901
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

Feature Support

- MapR supports Hive-2.1.1 on Tez-0.8.4. For more information, see Tez 0.8.4-1901 (EEP 4.1.3 and EEP 5.0.2) Release Notes on page 6499.
- MapR does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- MapR does not support HDFS encryption in Hive tables.
- MapR does not support HBase with Hive-2.1.1 starting from mapr-core-6.0.0.
- MapR does not support LLAP with Hive-2.1.1 as Apache Slider is not a MapR ecosystem component.
- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

New Features

- Implemented preserving warden configuration files during package update.
- Backported `FallbackHiveAuthorizerFactory` in the scope of CVE-2018-11777.

Changes in Security with Default Configuration

- Added the following properties to the `hive-site.xml` configuration by default on a secured cluster:

Table

Property	Value
`hive.users.in.admin.role`	`mapr`
`hive.conf.restricted.list`	`hive.security.authenticator.manager`, `hive.security.authorization.manager`, `hive.users.in.admin.role`,`hive.server2.xsrf.filter.enabled`, `hive.exec.pre.hooks`,`hive.exec.post.hooks`, `hive.exec.failure.hooks`,`hive.exec.query.redactor.hooks`, `hive.semantic.analyzer.hook`,`hive.exec.driver.run.hooks`, `hive.server2.session.hook`
`hive.security.authorization.enabled`	`true`
`hive.security.authorization.manager`	`org.apache.hadoop.hive.ql.security.authorization.plugin.fallback.FallbackHiveAuthorizerFactory`

last-updated: Apr 11, 2022

Because `hive.users.in.admin.role` and `hive.security.authorization.manager` were added to default Hive configuration, some actions in Hive are restricted according to security best practices and CVE-2018-11777

The new class `FallbackHiveAuthorizerFactory` will do the following to mitigate the above-mentioned CVE-2018-11777:

- Disallow local file location in SQL statements for all except the administrator.
- Allow `set` only selected white list parameters.
- Disallow `dfs` commands for all except the administrator.
- Disallow `ADD JAR` statements for all except the administrator.
- Disallow `COMPILE` statements for all except the administrator.
- Disallow `TRANSFORM` statements.

For more information, see Action Restrictions with Fallback Hive Authorizer on page 3459.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
3209ad7	2018-08-20	MAPR-HIVE-304 : Hive configure.sh ignores .customSecure file presence
41a6f32	2018-08-21	MAPR-32194 : Hardcoded reference to hdfs in Hive
8e228c8	2018-08-20	MAPR-31529: GenericUDFNamedStruct should constant fold at compile time
e4f3809	2018-09-06	MAPR-31413: add "explain ast"
7817ec2	2018-10-01	MAPR-HIVE-294: Authorization issue while trying to obtain logs in HS2 Web UI
bb5b17d	2018-10-03	MAPR-HIVE-329 : Incorrect verification of customSecure parameter during hive configure.sh
27aec85	2018-10-04	MAPR-HIVE-311 : Set up hive.conf.restricted.list configuration for clusters with MapR SASL security in scope of secure by default
e8748f62	2018-10-08	MAPR-HIVE-331 : Refactor conftool. Use enum instead of boolean for security variable
bc5dd15	2018-10-11	MAPR-HIVE-340 : configure.sh should keep custom configs in warden.hs2/hivemeta.conf
603b40f	2018-11-12	MAPR-HIVE-379 : Warden services do not copy while running configure.sh -R after update. Cannot run Hive after update

Commit	Date (YYYY-MM-DD)	Comment
934573e	2018-11-23	MAPR-HIVE-346 : Use rename of files where it is possible instead of full copy during moving data from staging dirs
50f7d8b	2018-11-31	MAPR-HIVE-367 : Extend hive.conf.restricted.list configuration for clusters with MapR SASL security in scope of secure by default
7f14674	2018-10-16	MAPR-HIVE-385 : CVE-2018-11777: Blocking local resource access in HiveServer2
d68fc4c	2018-08-23	MAPR-HIVE-286: Error in TestVectorStringExpressions.TestVectorStringExpressions while running on Jenkins
aaa387e	2018-12-14	MAPR-HIVE-411 : ConfToolTest.restrictedListSecurityOffTest is failed in Hive-2.1
2a9b793	2018-12-14	MAPR-HIVE-378 : Hive displays NULLS instead of Decimal values in external table (MapRDb Binary)
b87873d	2018-12-18	MAPR-HIVE-413 : Remove duplicates from ConfTool Junit tests
b680056	2018-12-20	MAPR-HIVE-389 : Develop the logic for preserving custom configuration in Hive warden.conf files
f2b21ca	2018-12-24	MAPR-HIVE-414 : Hive Metastore schema upgrade fails on MS SQL Database
3001d6d	2018-12-28	MAPR-HIVE-323 : Hive Metastore failed to start on Derby DB after running fresh configure.sh -R
1f22ff5	2019-01-08	MAPR-HIVE-416: unable to apply custom security. .customSecure is ignored by Hive configure.sh
d86903a	2019-01-09	MAPR-HIVE-395 : hive.warehouse.subdir.inherit.perms and hive.optimize.insert.dest.volume don't work on CTAS query

This release by MapR also includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
78665cef	2018-08-16	HIVE-15422: HiveInputFormat::pushProjectionsAndFilters paths comparison generates huge number of objects for partitioned dataset
652ad1c	2018-08-31	HIVE-16788: ODBC call SQLForeignKeys leads to NPE if you use PK arguments rather than FK arguments

Commit	Date (YYYY-MM-DD)	Comment
3931127	2018-11-23	HIVE-20420: Provide a fallback authorizer when no other authorizer is in use
7686e68	2018-11-30	HIVE-14007. Replace hive-orc module with ORC 1.3.1
b0f8fb6	2018-12-17	HIVE-15841: Upgrade Hive to ORC 1.3.3
cda4352	2018-12-17	HIVE-17631 : upgrade orc to 1.4.1
6c7dc72	2018-12-17	HIVE-18558: Upgrade orc version to 1.4.2
5e70406	2018-12-17	HIVE-18674 : update Hive to use ORC 1.4.3
82dffa1	2018-12-17	HIVE-19465: Upgrade ORC to 1.5.0
f8b0139	2018-12-17	HIVE-19226: Extend storage-api to print timestamp values in UTC
04d746b	2018-12-17	HIVE-19669: Upgrade ORC to 1.5.1
6f31bba	2018-10-08	HIVE-18007 : Address maven warnings
f920681	2018-10-25	HIVE-18778: Needs to capture input/output entities in explain
4c176fe	2018-12-20	HIVE-17272: when hive.vectorized.execution.enabled is true, query on empty partitioned table fails with NPE

Known Issues

- Vectorized execution is a new Hive feature that can show performance improvements in some cases and cause stability issues with others. The Hive vectorized execution feature has many bugs in Hive 2.x. It is recommended to turn off this feature at a system level and only use it for certain queries which work fine using it. You must evaluate the benefit of this feature against the potential stability issues on a case by case basis.

Resolved Issues

- None.

Hive 2.1.1-1901 (EEP 3.0.5) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.1-1901.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 2.1.1 Release Notes or the Apache Hive homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Hive Version	2.1.1
Release Date	February 2019

MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.1.1-mapr-1901
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

Feature Support

- MapR supports Hive-2.1.1 on Tez-0.8.4. For more information, see Tez 0.8.4-1901 (EEP 4.1.3 and EEP 5.0.2) Release Notes on page 6499.
- MapR does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- MapR does not support HDFS encryption in Hive tables.
- MapR does not support HBase with Hive-2.1.1 starting from mapr-core-6.0.0.
- MapR does not support LLAP with Hive-2.1.1 as Apache Slider is not a MapR ecosystem component.
- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

New Features

- Implemented preserving warden configuration files during package update.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
41a6f32	2018-08-13	MAPR-32194 : Hardcoded reference to hdfs in Hive
8e228c8	2018-08-20	MAPR-31529: GenericUDFNamedStruct should constant fold at compile time
e4f3809	2018-09-06	MAPR-31413: add "explain ast"
7817ec2	2018-10-01	MAPR-HIVE-294: Authorization issue while trying to obtain logs in HS2 Web UI
934573e	2018-11-23	MAPR-HIVE-346 : Use rename of files where it is possible instead of full copy during moving data from staging dirs
7f14674	2018-10-16	MAPR-HIVE-385 : CVE-2018-11777: Blocking local resource access in HiveServer2

Commit	Date (YYYY-MM-DD)	Comment
d68fc4c	2018-08-23	MAPR-HIVE-286: Error in TestVectorStringExpressions.TestVectorStringExpressions while running on Jenkins
2a9b793	2018-12-14	MAPR-HIVE-378 : Hive displays NULLS instead of Decimal values in external table (MapRDb Binary)
b680056	2018-12-20	MAPR-HIVE-389 : Develop the logic for preserving custom configuration in Hive warden.conf files
f2b21ca	2018-12-24	MAPR-HIVE-414 : Hive Metastore schema upgrade fails on MS SQL Database
d86903a	2019-01-09	MAPR-HIVE-395 : hive.warehouse.subdir.inherit.perms and hive.optimize.insert.dest.volume don't work on CTAS query

This release by MapR also includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
78665ce	2018-08-16	HIVE-15422: HiveInputFormat::pushProjectionsAndFilters paths comparison generates huge number of objects for partitioned dataset
652ad1c	2018-08-31	HIVE-16788: ODBC call SQLForeignKeys leads to NPE if you use PK arguments rather than FK arguments
3931127	2018-11-23	HIVE-20420: Provide a fallback authorizer when no other authorizer is in use
7686e68	2018-11-30	HIVE-14007. Replace hive-orc module with ORC 1.3.1
b0f8fb6	2018-12-17	HIVE-15841: Upgrade Hive to ORC 1.3.3
cda4352	2018-12-17	HIVE-17631 : upgrade orc to 1.4.1
6c7dc72	2018-12-17	HIVE-18558: Upgrade orc version to 1.4.2
5e70406	2018-12-17	HIVE-18674 : update Hive to use ORC 1.4.3
82dffa1	2018-12-17	HIVE-19465: Upgrade ORC to 1.5.0
f8b0139	2018-12-17	HIVE-19226: Extend storage-api to print timestamp values in UTC
04d746b	2018-12-17	HIVE-19669: Upgrade ORC to 1.5.1
6f31bba	2018-10-08	HIVE-18007 : Address maven warnings

Commit	Date (YYYY-MM-DD)	Comment
f920681	2018-10-25	HIVE-18778: Needs to capture input/output entities in explain
4c176fe	2018-12-20	HIVE-17272: when hive.vectorized.execution.enabled is true, query on empty partitioned table fails with NPE

Known Issues

- Vectorized execution is a new Hive feature that can show performance improvements in some cases and cause stability issues with others. The Hive vectorized execution feature has many bugs in Hive 2.x. It is recommended to turn off this feature at a system level and only use it for certain queries which work fine using it. You must evaluate the benefit of this feature against the potential stability issues on a case by case basis.

Resolved Issues

- None.

Hive 2.1.1-1808 (EEP 4.1.2 and EEP 5.0.1) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.1-1808.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 2.1.1 Release Notes or the Apache Hive homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Hive Version	2.1.1
Release Date	September 2018
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive/tree/2.1.1-mapr-1808
GitHub Release Tag	2.1.1-mapr-1808
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

Feature Support

- MapR supports Hive-2.1.1 on Tez-0.8.4. For more information, see Tez 0.8.4-1808 (EEP 4.1.2 and EEP 5.0.1) Release Notes on page 6500.
- MapR does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- MapR does not support HDFS encryption in Hive tables.
- MapR does not support HBase with Hive-2.1.1 starting from mapr-core-6.0.0.

last-updated: Apr 11, 2022

- MapR does not support LLAP with Hive-2.1.1 as Apache Slider is not a MapR ecosystem component.
- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

New Features

- Added the UPDATE operator for MapR Database JSON Tables in the Tez mode. For more information, see Understanding the UPDATE Statement on page 3469 .
- Added the `hive.metastore.allow.new.db.in.existing.directory` property with the default value as `false`. Set `hive.metastore.allow.new.db.in.existing.directory=true` in the `hive-site.xml` file to create a database if the directory already exists.

  CAUTION: Do not set the `hive.metastore.allow.new.db.in.existing.directory` using the Hive CLI or Beeline because it will not take effect. To change the value, set it only in the `hive-site.xml` file and then restart HMS and HS2.

- You can configure JDBC connection string with SSL enabled or disabled. For more information, refer to Configuring JDBC Connection String with SSL Encryption Enabled or Disabled on page 3521.
- Implemented preserving configuration during package update. For more information, see Pre-Upgrade Steps for Hive on page 339.

Known Issues

- In MAPR-HIVE-302, you cannot interact with Hive tables on PostgreSQL after upgrade.
- Vectorized execution is a new Hive feature that can show performance improvements in some cases and cause stability issues with others. The Hive vectorized execution feature has many bugs in Hive 2.x. It is recommended to turn off this feature at a system level and only use it for certain queries which work fine using it. You must evaluate the benefit of this feature against the potential stability issues on a case by case basis.

Changes in Security with Default Configuration

- Added the following property to the `hive-site.xml` configuration by default on a secured cluster:

Table

Property	Value
`hive.server2.use.SSL`	`true`

- Since HiveServer2 is configured to use SSL encryption by default starting from Hive-2.3 EEP 5.0.1, add `ssl=true;` to a JDBC connection string when PAM or MAPR-SASL authentication is used.
- Added world-readable permissions (644) for the `hive-site.xml` configuration file.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
288ce1a	2018-08-13	MAPR-HIVE-301: Avoid redundant creation of HiveConf object while creating transport object for connection

Commit	Date (YYYY-MM-DD)	Comment
287ebc2	2018-08-08	MAPR-HIVE-300: Hive HCat does not create a proper "client" on MapR-SASL cluster without Hive metastore
192353c	2018-07-25	MSEN-11: Save file with newline at the end
faa8f994	2018-07-21	MAPR-HIVE-281: WebHCat throws an error at trying to run a Hive query
222272d	2018-07-16	MAPR-HIVE-285: Hive configure.sh fails when custom security flag is set
60dd8cd	2018-07-10	MAPR-HIVE-272: HiveMapRDBJsonOutputFormat class must implement HiveOutputFormat interface
e946055	2018-07-02	MAPR-HIVE-268: Errors during configure.sh on a non-secure cluster
faad098	2018-06-25	MAPR-31641: Hive deletes failed queries with invalid table alias or column reference
977818f	2018-06-27	MAPR-31803: Fix for Bug 30031 results in users not being able to create databases on existing location
603e327	2018-06-25	MAPR-HIVE-257: Insert overwrite from empty table do not overwrite data (only on Tez)
42d06f3	2018-06-21	MAPR-HIVE-242: Configure.sh adds two rows to the end of hive-site.xml after every launching
8532a25	2018-06-05	MAPR-HIVE-244: Hive configure.sh overwrites properties by Secure-By-Default values and not take into account .not_configured_yet flag
97ca25d	2018-06-01	MAPR-HIVE-223: NPE during CREATE ROLE using SQL Standard Based Hive Authorization
85e38e8	2018-05-24	MAPR-31380: HeartBeat thread uses cancelled delegation token while connecting to meta
7fb2b20	2018-05-22	MAPR-HIVE-212: Setup SSL by default when PAM and MapR-SASL is on by default
64d6523	2018-05-21	MAPR-HIVE-228: Throw an exception while trying to update maprdb.column.id column
2c3c929	2018-04-20	MAPR-31175: hive.exec.tmp.maprfsvolume should be false on Tez mode
ea822ac	2018-04-23	MAPR-HIVE-194: Hive-2.3 and Hive-2.1, JSON artifacts are not updated

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
0737e03	2018-04-16	MAPR-HIVE-196: Running configure.sh -R reconfigures underlying MetastoreDB to derby in hive-site.xml
346c8cb	2018-04-12	MAPR-27663: PidFilePatternConverter does not append the pid to the log name
a9fb431	2018-04-11	MAPR-HIVE-190: Log writes in two hive.log files instead of one
e839526	2018-03-27	MAPR-HIVE-174: Implement UPDATE syntax for MapR Database JSON documents
f0eba10	2018-03-21	MAPR-30940: Hive job fails by AccessControlException against files on the NM local disk
4284798	2018-04-11	MAPR-30895: Tez jobs are shown as KILLED in RM UI
188cb42	2018-04-11	MAPR-HIVE-171: Add bash script for MapR configuration tool
39d3ebe	2018-04-10	MAPR-HIVE-167: Add into MapR configuration sub-module API for configuring hadoop.security.credential.provider.path
968bfa7	2018-04-02	MAPR-HIVE-189: Set world-readable permissions for hive conf files

This release by MapR also includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
83827db	2018-08-06	HIVE-14037: java.lang.ClassNotFoundException for the jar in hive.reloadable.aux.jars.path in mapreduce
e650456	2018-07-10	HIVE-16114: NullPointerException in TezSessionPoolManager when getting the session
054e9d4	2018-07-02	HIVE-18393: Error returned when some other type is read as string from parquet tables
93f18f3	2018-06-28	HIVE-16667: package.jdo changes to map DB CLOBs to JDO VARCHAR
ba76a6d	2018-06-21	HIVE-13000: Hive returns useless parsing error
77f9699	2018-06-13	HIVE-17963: Fix for HIVE-17113 can be improved for non-blobstore filesystems
2b8244b	2018-06-12	HIVE-17113: Duplicate bucket files can get written to table by runaway task

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
d1e3bf2	2018-05-31	HIVE-17155: findConfFile() in HiveConf.java has some issues with the conf path
ef66d89	2018-05-25	HIVE-15950: Make DbTxnManager use Metastore client consistently with callers
c5684fb	2018-05-25	HIVE-18879: Disallow embedded element in UDFXPathUtil needs to work if xercesImpl.jar in classpath
4eb174b	2018-05-25	HIVE-18815: Remove unused feature in HPL/SQL
160b723	2018-05-22	HIVE-18788: Clean up inputs in JDBC PreparedStatement
1937925	2018-04-11	HIVE-16133: Footer cache in Tez AM can consume too much memory

Hive 2.1.1-1808 (EEP 3.0.4) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.1-1808.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 2.1.1 Release Notes or the Apache Hive homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Hive Version	2.1.1
Release Date	September 2018
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive/tree/2.1.1-mapr-1808
GitHub Release Tag	2.1.1-mapr-1808
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

Feature Support

- MapR supports Hive-2.1.1 on Tez-0.8.4. For more information, see Tez 0.8.4-1808 (EEP 4.1.2 and EEP 5.0.1) Release Notes on page 6500.
- MapR does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- MapR does not support HDFS encryption in Hive tables.
- MapR does not support HBase with Hive-2.1.1 starting from mapr-core-6.0.0.

last-updated: Apr 11, 2022

- MapR does not support LLAP with Hive-2.1.1 as Apache Slider is not a MapR ecosystem component.
- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

New Features

- Added the UPDATE operator for MapR Database JSON tables in Tez mode. For more information, see Understanding the UPDATE Statement on page 3469.
- Added the `hive.metastore.allow.new.db.in.existing.directory` property with the default value as `false`. Set `hive.metastore.allow.new.db.in.existing.directory=true` in the `hive-site.xml` file to create a database if the directory already exists.

  CAUTION: Do not set the `hive.metastore.allow.new.db.in.existing.directory` using the Hive CLI or Beeline because it will not take effect. To change the value, set it only in the `hive-site.xml` file and then restart Hive Metastore and HiveServer2.

- You can configure JDBC connection string with SSL enabled or disabled. For more information, refer to Configuring JDBC Connection String with SSL Encryption Enabled or Disabled on page 3521.
- Implemented preserving configuration during package update. For more information, see Pre-Upgrade Steps for Hive on page 339.

Known Issues

- MAPR-HIVE-302 - Cannot interact with Hive tables on postgresql after upgrade.
- Vectorized execution is a new Hive feature that can show performance improvements in some cases and cause stability issues with others. The Hive vectorized execution features has many bugs in Hive 2.x so it has been disabled by default. It is recommended to turn off this feature at a system level and only use it for certain queries which work fine using it. You must evaluate the benefit of this feature against the potential stability issues on a case by case basis.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
288ce1a	2018-08-13	MAPR-HIVE-301: Avoided redundant creation of HiveConf object while creating transport object for connection
287ebc2	2018-08-08	MAPR-HIVE-300: Hive HCat does not create a proper "client" on MapR-SASL cluster without Hive metastore
192353c	2018-07-25	MSEN-11 Save file with newline at the end
60dd8cd	2018-07-10	MAPR-HIVE-272: HiveMapRDBJsonOutputFormat class must implement HiveOutputFormat interface
faad098	2018-06-25	MAPR-31641: Hive deletes failed queries with invalid table alias or column reference

Commit	Date (YYYY-MM-DD)	Comment
977818f	2018-06-27	MAPR-31803 : Fix for Bug 30031 results in users not being able to create databases on existing location
603e327	2018-06-25	MAPR-HIVE-257: Insert overwrite from empty table do not overwrite data (only on Tez)
97ca25d	2018-06-01	MAPR-HIVE-223: NPE during CREATE ROLE using SQL Standard Based Hive Authorization
85e38e8	2018-05-24	MAPR-31380 HeartBeat thread uses cancelled delegation token while connecting to meta
64d6523	2018-05-21	MAPR-HIVE-228 : Throw an exception while trying to update maprdb.column.id column
2c3c929	2018-04-20	MAPR-31175: hive.exec.tmp.maprfsvolume should be false on Tez mode
ea822ac	2018-04-23	MAPR-HIVE-194: Hive-2.3 and Hive-2.1, JSON artifacts are not updated
346c8cb	2018-04-12	MAPR-27663: PidFilePatternConverter does not append the pid to the log name
a9fb431	2018-04-11	MAPR-HIVE-190: Log writes in two hive.log files instead of one
e839526	2018-03-27	MAPR-HIVE-174 : Implement UPDATE syntax for MapR Database JSON documents
f0eba10	2018-03-21	MAPR-30940: Hive job fails by AccessControlException against files on the NM local disk
4284798	2018-04-11	MAPR-30895: Tez jobs are shown as KILLED in RM UI
968bfa7	2018-04-02	MAPR-HIVE-189 : Set world-readable permissions for hive conf files

This release by MapR also includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
e650456	2018-07-10	HIVE-16114 : NullPointerException in TezSessionPoolManager when getting the session
054e9d4	2018-07-02	HIVE-18393 : Error returned when some other type is read as string from parquet tables
93f18f3	2018-06-28	HIVE-16667 package.jdo changes to map DB CLOBs to JDO VARCHAR

Commit	Date (YYYY-MM-DD)	Comment
ba76a6d	2018-06-21	HIVE-13000 : Hive returns useless parsing error
77f9699	2018-06-13	HIVE-17963: Fix for HIVE-17113 can be improved for non-blobstore filesystems
2b8244b	2018-06-12	HIVE-17113: Duplicate bucket files can get written to table by runaway task
d1e3bf2	2018-05-31	HIVE-17155: findConfFile() in HiveConf.java has some issues with the conf path
ef66d89	2018-05-25	HIVE-15950 Make DbTxnManager use Metastore client consistently with callers
c5684fb	2018-05-25	HIVE-18879: Disallow embedded element in UDFXPathUtil needs to work if xercesImpl.jar in classpath
4eb174b	2018-05-25	HIVE-18815: Remove unused feature in HPL/SQL
160b723	2018-05-22	HIVE-18788: Clean up inputs in JDBC PreparedStatement
1937925	2018-04-11	HIVE-16133 : Footer cache in Tez AM can consume too much memory

Hive 2.1.1-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.1-1803 EEP 4.1.1 and EEP 5.0.0.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 2.1.1 Release Notes or the Apache Hive homepage.

Hive Version	2.1.1
Release Date	March 2018
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.1.1-mapr-1803
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

Feature Support

- MapR supports Hive-2.1.1 on Tez-0.8.4. For more information, see Tez 0.8.4-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes on page 6501.
- MapR does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

- MapR does not support HDFS encryption in Hive tables.
- MapR does not support HBase with Hive-2.1.1 starting from mapr-core-6.0.0.
- MapR does not support LLAP with Hive-2.1.1 as Apache Slider is not a MapR ecosystem component.
- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

New Features

- Added PAM authentication support for HiveServer2 Web UI.
- Added REST API WebHCat SSL encryption support.
- Added HiveMetastore password encryption.
- Added separate files for HiveServer2 and HiveMetastore logs.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
ccf82995	2017-11-01	MAPR-29528: Cannot change hive.exec.scratchdir at runtime.
15ce7e3f	2017-10-26	MAPR-HIVE-103: Verify flag before disable logging in ECO configure.sh.
25a6248	2017-11-06	MAPR-HIVE-107: Set 777 permission to /user/hive/warehouse because of impersonation enabled by default.
88f00ba	2017-11-10	MAPR-29571: Maprcli command start up more than one HiveMetastore process on Suse.
9b0f52f8	2017-11-17	MAPR-23955: Running configure.sh -R changes permissions of hive logs.
2da240d0	2017-11-15	MAPR-29591: Flooding of WARNING: org.apache.parquet.CorruptStatistics: Ignoring statistics because created_by could not be parsed' in application logs.
57742b60	2017-11-15	MAPR-29743: [HIVE-2.1] Some messages are missed in hive.log if HiveServer2 and HiveMetastore share in log4j config.
484eba4	2017-11-24	MAPR-30116: Create sub-folder structure if hive.exec.scratchdir is changed in runtime.
cf2eb327	2017-12-06	MAPR-30201: Cannot start Hive CLI in embedded mode after running configure.sh -R after fresh installation.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
620c897	2017-12-07	MAPR-HIVE-118: Init metastore Uri if Derby DB is used by default and metastore is installed.
60e2cbd4	2017-11-14	MAPR-29712: UpdateInputAccessTimeHook fails for non-current database.
72108e7	2017-12-14	MAPR-30224: Cannot create /user/hive/warehouse using Hive configure.sh on fresh cluster.
4004495c	2017-12-13	MAPR-30248 WebHcat logs does not work on hive-2.1.
1ac82db7	2017-12-18	MAPR-30045: [Hive-2.1] Hive WebHcat DLL does not work, Hive Job issue.
4ffe2c8	2017-12-20	MAPR-30310 Configure.sh cannot copy WebHcat required lib into hadoop common due to cannot find hadoop_version variable.
414ec0e41	2017-12-21	MAPR-HIVE-139: Remove hardcoded password property removing from installer hive-site.xml and implement logic for it in Hive.
d059322	2018-01-12	MAPR-HIVE-142 Add catch block in configure.sh WebHCat script. Warning trying to create link when the link is already created.
b63f1291	2018-01-22	MAPR-30556: Incorrect work with MapR Database JSON tables.
76d5a36d	2018-01-22	MAPR-HIVE-140: HiveServer2 Web UI remains unsecured in 6.0 secured by default clusters.
ee119950	2018-01-26	MAPR-HIVE-147: Configure PAM + SSL for HS2 web UI on MAPR SASL cluster using configure.sh script.
4c529b1	2018-02-01	MAPR-30631: Kryo exception when deserializing HiveNullValueSequenceFileOutputFormat.
b16765c4	2018-02-07	MAPR-30719: Hive configure.sh overwrites custom hive.metastore.uris property for the first run.
235dd03	2018-01-19	MAPR-HIVE-111: Review file permissions for Hive.
19700c6	2018-02-15	MAPR-HIVE-143: REST API-WebHCat missing SSL support.
64021d7	2018-02-16	MAPR-HIVE-158: Configure REST-API WebHCat with SSL on MAPR SASL cluster using configure.sh script.
4a78363a	2018-02-19	MAPR-HIVE-146: Wrong owner of hiveversion file after hive configure.sh executed.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
d31e666e	2018-02-21	MAPR-30738: Running configure.sh -R changes group/owner of hive logs.
ff54d883	2018-02-21	MAPR-30838: Errors loading org.apache.hive.conftool.ConfCli while running configure.sh -R.
827a99f	2018-02-21	MAPR-HIVE-160: Set hive.metastore.execute.setugi to false on secure cluster installation.
1d86d5c	2018-02-23	MAPR-30626: Starts up more that one process for hive services on CentOS 6.8.
d95fe41	2018-03-15	MAPR-30968: Hive throws NPE when writing STRUCT data type with NULL value into MapR Database table.
91bda9	2018-03-14	MAPR-HIVE-178: Unable to execute configure.sh -R for Hive if mapr user does not have a maprticket on a mapr-secure cluster.
538722a	2018-04-02	MAPR-HIVE-187: Hive 2.1 CLI did not start on 6.0.1 cluster with Spark 2.2.1.

This release by MapR also includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
63eae24	2017-11-07	HIVE-17664: Refactor and add new tests.
79cd8c5	2017-11-27	HIVE-14139: NPE dropping permanent function.
9d991a8e	2017-12-03	HIVE-16950: Dropping hive database/table which was created explicitly in default database location, deletes all databases data from default database location.
d128066	2017-12-19	HIVE-16050: Regression: Union of null with non-null.
c65b75d7	2018-01-22	HIVE-13864: Beeline ignores the command that follows a semicolon and comment.
0127924	2018-01-22	HIVE-16935: Hive should strip comments from input before choosing which CommandProcessor to run.
67c4d4ce	2018-01-22	HIVE-17050: Multiline queries that have comment in middle fail when executed via "beeline -e".
102ab8b3	2018-01-22	HIVE-18127: Do not strip '--' comments from shell commands issued from the CLIDriver.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
2fec4cea	2018-02-05	HIVE-17731: Add a backward compat option for external users to HIVE-11985.

Known Issues and Limitations

- The hive-site.xml file permissions is 640.
- For accessing and using Hive, administrators have the ability to manage permissions for "Other" users. There are at least two available options on how to enable the using of hive-site.xml:
  1. Add users who want to work with Hive to a mapr group.
  2. Set permission to 644 to make it world-readable.

Hive 2.1.1-1803 (EEP 3.0.3) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.1-1803.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 2.1.1 Release Notes or the Apache Hive homepage.

Hive Version	2.1.1
Release Date	March 2018
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	2.1.1-mapr-1803
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

Feature Support

- MapR supports Hive-2.1.1 on Tez-0.8.4. For more information, see Tez 0.8.4-1803 (EEP 3.0.3) Release Notes on page 6502.
- MapR does not support Hive on Spark, so you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- MapR does not support HDFS encryption in Hive tables.
- MapR does not support HBase with Hive-2.1.1 starting from mapr-core-6.0.0.
- MapR does not support LLAP with Hive-2.1.1 as Apache Slider is not a MapR ecosystem component.
- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

New Features

- Added PAM authentication support for HiveServer2 Web UI.

- Added REST API WebHCat SSL encryption support.
- Added separate files for HiveServer2 and HiveMetastore logs.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
ccf82995	2017-11-01	MAPR-29528: Cannot change hive.exec.scratchdir at runtime.
88f00ba	2017-11-10	MAPR-29571: Maprcli command start up more than one HiveMetastore process on Suse.
2da240d0	2017-11-15	MAPR-29591: Flooding of WARNING: org.apache.parquet.CorruptStatistics: Ignoring statistics because created_by could not be parsed' in application logs.
57742b60	2017-11-15	MAPR-29743: [HIVE-2.1] Some messages are missed in hive.log if HiveServer2 and HiveMetastore share in log4j config.
484eba4	2017-11-24	MAPR-30116: Create sub-folder structure if hive.exec.scratchdir is changed in runtime.
60e2cbd4	2017-11-14	MAPR-29712: UpdateInputAccessTimeHook fails for non-current database.
4004495c	2017-12-13	MAPR-30248 WebHcat logs does not work on hive-2.1.
1ac82db7	2017-12-18	MAPR-30045: [Hive-2.1] Hive WebHcat DLL does not work, Hive Job issue.
b63f1291	2018-01-22	MAPR-30556: Incorrect work with MapR Database JSON tables.
76d5a36d	2018-01-22	MAPR-HIVE-140: HiveServer2 Web UI remains unsecured in 6.0 secured by default clusters.
4c529b1	2018-02-01	MAPR-30631: Kryo exception when deserializing HiveNullValueSequenceFileOutputFormat.
235dd03	2018-01-19	MAPR-HIVE-111: Review file permissions for Hive.
19700c6	2018-02-15	MAPR-HIVE-143: REST API-WebHCat missing SSL support.
4a78363a	2018-02-19	MAPR-HIVE-146: Wrong owner of hiveversion file after hive configure.sh executed.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
1d86d5c	2018-02-23	MAPR-30626: Starts up more that one process for hive services on CentOS 6.8.

This release by MapR also includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
63eae24	2017-11-07	HIVE-17664: Refactor and add new tests.
79cd8c5	2017-11-27	HIVE-14139: NPE dropping permanent function.
9d991a8e	2017-12-03	HIVE-16950: Dropping hive database/table which was created explicitly in default database location, deletes all databases data from default database location.
d128066	2017-12-19	HIVE-16050: Regression: Union of null with non-null.
c65b75d7	2018-01-22	HIVE-13864: Beeline ignores the command that follows a semicolon and comment.
0127924	2018-01-22	HIVE-16935: Hive should strip comments from input before choosing which CommandProcessor to run.
67c4d4ce	2018-01-22	HIVE-17050: Multiline queries that have comment in middle fail when executed via "beeline -e".
102ab8b3	2018-01-22	HIVE-18127: Do not strip '--' comments from shell commands issued from the CLIDriver.
2fec4cea	2018-02-05	HIVE-17731: Add a backward compat option for external users to HIVE-11985.

Known Issues and Limitations

- The hive-site.xml file permissions is 640.
- For accessing and using Hive, administrators have the ability to manage permissions for "Other" users. There are at least two available options on how to enable the using of hive-site.xml:
  1. Add users who want to work with Hive to a mapr group.
  2. Set permission to 644 to make it world-readable.

Hive 2.1.1-1710 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.1-1710.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 2.1.1 Release Notes or the Apache Hive homepage.

Hive Version	2.1

last-updated: Apr 11, 2022

Release Date	November 2017
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/hive/tree/2.1.1-mapr-1710
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

Feature Support

- MapR supports Hive-2.1.1 on Tez-0.8.4.
- MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- MapR does not support HDFS encryption in Hive tables.
- MapR does not support Hbase-0.9X with Hive-2.1.1. Only Hbase-1.X is compatible with Hive-2.1.1.
- MapR does not support LLAP with Hive-2.1.1 since Apache Slider is not in MapR ecosystem.
- Starting from Hive 2.1, Hive needs to run the `schematool` command as an initialization step.

New Features

- Includes Hive support for MapR Database JSON tables.
- Includes MapR-SASL as default security for HiveServer 2 and Hive Metastore on secure cluster.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
7ea39f4	2017-10-25	MAPR-HIVE-104: The issue that caused TableClosedException when performing an operation on a closed table is now fixed.
ace4158	2017-09-19	MAPR-HIVE-70: The issue that caused TableClosedException when performing an operation on a closed table is now fixed.
e7314ca	2017-09-12	MAPR-29056: The issue that caused an exception when inserting data into table on AWS is now fixed.
2b313da	2017-09-11	MAPR-HIVE-21: Hive now includes support for MapR Database JSON tables from Hive. For more information, see Connecting Using Hive MapR Database JSON Connector on page 3466.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
f8f543c	2017-09-05	MAPR-28898: The issue that caused 'multiple SLF4J bindings' warnings is now fixed.
59ef497	2017-05-30	MAPR-HIVE-3: Hive is now secured by default. For more information, see Hive Security on page 3427.
d7f2dd1	2017-08-29	MAPR-HIVE-41: The issue that caused warnings on CentOS when trying to uninstall already removed Hbase JARs is now fixed.
6adf28d	2017-08-28	MAPR-28881: The `hive-site.xml` file now contains only needed properties because all unnecessary properties have been removed.
0133468	2017-07-21	MAPR-HIVE-42: The issue that caused HiveServer2 to not stop, but continue running when the cluster was shutdown (from the UI) is now fixed.
8a01d62	2017-08-24	MAPR-27403: Hive now uses Joda-time version 2.9.9 because some functionality were breaking in Joda-time version below 2.8.0.
0cca11a	2017-08-14	MAPR-28428: The issue that caused incorrect result for Hive join query with COALESCE in WHERE condition is now fixed.

This release by MapR also includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
2753757	2017-04-19	HIVE-16473: The issue that caused Hive-on-Tez to fail to write to an HBase table is now fixed.
60a5918	2017-09-26	HIVE-10378: The issue that caused Hive Update statement set keyword to work with lower case and not uppercase is now fixed. If incorrect column name is specified in the set clause, a warning is now returned.
51ed0ed	2016-11-15	HIVE-15208: The query string is now HTML encoded for Web UI.
f896f7d	2016-03-30	HIVE-13380: The decimal now has lower precedence than double in type hierarchy.
70af868	2016-07-22	HIVE-14251: Hive will now only perform implicit conversion within each type of group including string group, number group, or date group, and not across groups. So, in order to union a string type with a date type, an explicit cast from string to date or from date to string is needed.

Commit	Date (YYYY-MM-DD)	Comment
f9517f0	2017-09-06	HIVE-12274: The width of metastore text columns for general configuration storage has now been increased.
4c8bfb3	2017-08-17	HIVE-14564: The issue that caused an exception when SelectOperator did column pruning is now fixed.

Hive 2.1.1-1707 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 2.1-1703.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 2.1.1 Release Notes or the Apache Hive homepage.

Hive Version	2.1
Release Date	August 2017
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/hive/tree/2.1.1-mapr-1707
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in this Release

No new features in this release.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
6044467	2017-07-12	MAPR-28346: The issue that caused permission related errors when Hive CLI was executed by a user other than the mapr user is now fixed.
d16c24d	2017-07-07	MAPR-28251: The issue that caused errors while inserting data into partitioned table on Kerberos is now fixed.
0168f6c	2017-06-20	MAPR-27941: The issue that caused Hive on Tez mode to not launch when `hive.aux.jars.path` was set is now fixed.
9ffd14d	2017-06-19	MAPR-27477: The issue that caused an exception from MapR filesystem when `hive.warehouse.subdir.inherit.perms` was set to `true` and user, who was not a member of parent owner group, was creating the DB on Hive is now fixed.

Commit	Date (YYYY-MM-DD)	Comment
b6e34d6	2017-06-14	MAPR-27799: The issue that caused Hive `DROP DATABASE` query to throw exceptions in the logs is now fixed.
78cb4a6	2017-06-13	MAPR-27646: The issue that caused Hive to duplicate column comments while passing information about partitions to Partition object FieldSchema object is now fixed.
271f527	2017-06-08	MAPR-27613: The issue that caused an exception when querying table using HiveServer2 is now fixed.
31940a3	2017-05-24	MAPR-27072: The issue that caused RowContainer.tmp to be written to the local filesystem and not the local volume is now fixed.
0eb653e	2017-06-01	MAPR-22115: The issue that caused Hive to prompt for username and password when using MapR SASL and Kerberos for authentication is now fixed.
6aad8e3	2017-05-24	MAPR-27033: The issue that caused Datanucleus MSSQLServerAdapter to generate incorrect syntax for OFFSET-FETCH clause when working with MS SQL SERVER that is greater than or equal to 2012 is now fixed.
e44cf88	2017-05-25	MAPR-27427: The issue that caused Hive parser to ignore comments when the line started with two dashes (`--`) is now fixed.
a894d68	2017-05-15	MAPR-27308: The issue that caused some queries with multiple inserts to fail in Hive 2.1 is now fixed.
2136184	2017-05-18	MAPR-27175: The issue that caused write-permission error when users from group different from table owner tried to write to table is now fixed. With this fix, the default configuration for log destination has been changed to use the home directory allowing each user to have separate log file.
7abb218	2017-05-26	MAPR-27483: The mismatch in hive and hbase in the result returned when counting number of records when there were NULL fields is now fixed by a new property, `hive.read.all.hbase.column`, which when set to: • `true` will allow Hive to read all columns from HBase table. • `false` may cause Hive to ignore null fields when counting number of records.

Commit	Date (YYYY-MM-DD)	Comment
6f824be	2017-05-19	MAPR-27234: The issue that caused Hive on Tez queries to MapR Database to fail is now fixed.
b016f63	2017-04-20	MAPR-HIVE-10: HiveServer2 PAM configuration is now consistent with MapR PAM configuration.
d521948	2017-04-20	MAPR-HIVE-23: The issue that cause `hive.warehouse.subdir.inherit.perms` property to not work with CTAS query is now fixed.
87dc95d	2017-04-10	MAPR-23637: The issue that caused Hive Server Dynamic Service Discovery to not work in secure MapR SASL cluster is now fixed.
ea5c811	2017-03-29	MAPR-26678: Hive now includes `hplsql-site.xml` file for HPL/SQL.
944eafb	2017-03-30	MAPR-HIVE-6: Incorrect implementation of HiveServer2 Authentication is now fixed.

This release by MapR includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
9ffd14d	2017-07-19	HIVE-11166: The issue that caused an exception when executing an insert into SQL statement on a HBase table with HBaseStorageHandler in HDBCServer of Spark is now fixed.
67d3b3c	2017-05-25	HIVE-15627: The `hive.vectorized.adaptor.usage.mode` will now vectorize all UDFs and not just those in supportedGenericUDFs.
2874bed	2017-05-24	HIVE-15361: The issue that caused INSERT dynamic partition on S3 to fail with a MoveTask failure is now fixed.
901d5cf	2017-05-24	HIVE-16161: The `packaging.minimizeJar` is now disabled for JDBC build to allow the standalone JDBC jar to have the necessary classes.
e6eff5d	2017-05-24	HIVE-16160: The issue that caused OutOfMemoryError when GC overhead limit exceeded on HiveServer2 is now fixed.
d35291b	2017-05-24	HIVE-15866: The issue that cause LazySimpleDeserializeRead to not recognize lower case variant is now fixed.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
49e3fdb	2017-05-24	HIVE-15848: The issue that caused count or sum distinct to be incorrect when `hive.optimize.reduce deduplication` was set to `true` is now fixed.
ac2d7b4	2017-05-23	HIVE-15754: The issue that caused exchange partition event to not generate notifications in `notification_log` is now fixed.
7ab50ee	2017-05-22	HIVE-15731: The issue that caused interrupt status on thread to not return sessions to the sessionPool is now fixed.
0a28540	2017-05-22	HIVE-15684: Fixed the posBigTable used in VectorMapJoinOuterFilteredOperator to prevent IndexOutOfBoundsException error.
2f3120e	2017-05-22	HIVE-15519: The issue that caused Hive decimal type column precision to return zero even when column has precision set is now fixed.
108da0e	2017-05-22	HIVE-15493: The issue that caused wrong result for LEFT outer join in Tez using MapJoinOperator is now fixed.
9ca0a75	2017-05-22	HIVE-15488: The issue that caused native Vector MapJoin to fail when trying to serialize BigTable rows with (unreferenced) complex types is now fixed.
bb191d5	2017-05-22	HIVE-15327: The issue that caused outerjoin to produce wrong results depending on joinEmitInterval value is now fixed.
6e9051b	2017-05-22	HIVE-16043: The `TezConfiguration.TEZ_QUEUE_NAME` will now be used instead of `tez.queue.name`.
b0b3de3	2017-05-22	HIVE-15421: The exception handling in DagUtils.localizeResource is now fixed to fail early with different messages for different failures.
3bcd439	2017-05-22	HIVE-14060: The localhost is now removed from Hive splits to allow for the allocation of "*" containers instead of waiting for heartbeat.
0e663e7	2017-05-18	HIVE-15199: The issue that caused INSERT INTO data on S3 to replace the old rows with new ones is now fixed.
4a7cad3	2017-05-18	HIVE-15236: The timestamp and date comparison will now happen in timestamp instead of string.

Commit	Date (YYYY-MM-DD)	Comment
87c33cc	2017-05-18	HIVE-14804: The issue that caused HPLSQL with multiple DB connections to not switch back to Hive is now fixed.
cd962e7	2017-05-18	HIVE-14706: The issue that caused column level lineage information to be returned as null is now fixed.
0b92a71	2017-05-18	HIVE-14607: The issue that caused ORC split generation to fail with IndexOutOfBoundsException is now fixed.
f2e3348	2017-05-18	HIVE-14336: VectorUDFAdaptor can now be configured to specify whether to attempt vectorization using `hive.vectorized.adaptor.usage.mode` property (whose value can be none, chosen, all) in `HiveConf.java` file.
8d5844f	2017-05-18	HIVE-13966: The `HiveConf.java` file now contains the following new properties to prevent DbNotificationListener from losing DDL operation notifications: • `hive.metastore.transactional.event.listeners` • `hive.metastore.event.listeners`
70785cf	2017-05-18	HIVE-13403: The issue that caused StreamingAPI to create empty buckets is now fixed.
84d3193	2016-08-25	HIVE-15331: The issue that caused decimal multiplication with high precision/scale to often return NULL is now fixed.

Feature Support

- MapR supports Hive-2.1.1 on Tez-0.8.4.
- MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- MapR does not support HDFS encryption in Hive tables.
- MapR does not support Hbase-0.9X with Hive-2.1.1. Only Hbase-1.X is compatible with Hive-2.1.1.
- MapR does not support LLAP with Hive-2.1.1 since Apache Slider is not in MapR ecosystem.
- Starting from Hive 2.1, Hive needs to run the schematool command as an initialization step.

Resolved Issues

None.

Hive 2.1-1703 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 2.1.1 Release Notes or the Apache Hive homepage.

Hive Version	2.1
Release Date	April 2017
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/hive/tree/2.1.1-mapr-1703
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)
API for this Version	See Hive 2.1 API on page 3543

New in This Release

This version of Hive includes the following:

- Hive Hybrid Procedural SQL On Hadoop (HPL/SQL)

  Hive Hybrid Procedural SQL On Hadoop (HPL/SQL), which is available in Hive 2.1, is a tool that implements procedural SQL for Hive.

  HPL/SQL is an open source tool that implements procedural SQL language for Apache Hive, SparkSQL, Impala, as well as any other SQL-on-Hadoop implementation, any NoSQL, and any RDBMS.

  HPL/SQL is a hybrid and heterogeneous language that understands syntaxes and semantics of almost any existing procedural SQL dialect, and you can use with any database (for example, running existing Oracle PL/SQL code on Apache Hive and Microsoft SQL Server, or running Transact-SQL on Oracle, Cloudera Impala, or Amazon Redshift).

  Note: Create the `hplsql-site.xml` file to configure HPL/SQL feature. See http://www.hplsql.org/configuration for more information.

- Dynamically partitioned hash join for Tez.
- Support for aggregate push down through joins.
- DBTokenStore support to HS2 delegation token.
- Hive View Column Authorization.
- UDF substring_index

  Returns the substring from string `str` before count occurrences of the delimiter.

- Quarter UDF

  The quarter from a string / date / timestamp returned by the QUARTER(date) function may be useful for different domains like retail, finance etc.

- Support for limited integer type promotion in ORC.
- ORC file dump in JSON format

  ORC file dump uses custom format. Will be useful to dump ORC metadata in json format so that other tools can be built on top it.

- UDF `aes_encrypt` and `aes_decrypt` with AES (Advanced Encryption Standard) algorithm.

  Oracle JRE supports AES-128 out of the box AES-192 and AES-256 are supported if Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files are installed.

- Possibility for Hive Parser to support multi col in clause (x,y..) in ((..),..., ()).
- Support of special characters in quoted table names.
- Support for "show create database".
- Support escaping carriage return and new line for LazySimpleSerDe.
- Banker's rounding BROUND UDF

  With banker's rounding, the value is rounded to the nearest even number. Also known as "Gaussian rounding", and, in German, "mathematische Rundung".

- Command to kill an ACID transaction.

  This cleans up all state related to this transaction. The initiator of this (if still alive) will get an error trying to heartbeat/commit and will become aware that the transaction failed.

- Support for modifying the numRows and dataSize for a table/partition.
- Support vectorizing when the input format is TEXTFILE and other formats for better Map Vertex performance.
- Support for NULLS FIRST/NULLS LAST.

  The NULLS FIRST and NULLS LAST options can be used to determine whether nulls appear before or after non-null data values when the ORDER BY clause is used.

- Supports aggregate functions in over clause.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
3b83fea	2017-01-22	MAPR-26541: The variable $BASEMAPR will now be initialized by $HOME_MAPR from parent pid and if it cannot be defined, will be set to /opt/mapr by default.
6ff94bc	2017-02-28	MAPR-25720: When restarting HS2, the issue that caused Session manager to delete operation_logs folder a second time after a huge delay is now fixed.
e8a6f79	2017-02-23	MAPR-26193: The issue that caused the "Permission Denied" message when launching a hive shell is now fixed.

Commit	Date (YYYY-MM-DD)	Comment
f69e9ee	2017-02-17	MAPR-25698: The missing `log4j2.component.properties` file is now included with Hive and the `log4j2.disable.jmx` property value is set to `false` by default to fix the AccessControlExceptionImport error when importing from MySQL to Hive.
7d3b630	2017-02-07	MAPR-26169: The issue that caused the FileNotFoundException when there was no file with localPath (for example, no reduce work) is now fixed.
8d40378	2017-02-07	MAPR-25952: When starting Hive, the issue that caused the message about absence of hbase is now fixed.
7afb69c	2017-01-30	MAPR-25938: The conflicts in the versions of included Sentry libraries which caused insert queries to fail with exception is now fixed.
13f2e20	2017-01-25	MAPR-25880: The missing HiveOperation field is now included in HiveSemanticAnalyzerHookContext to allow StateStore to acces the current HiveOperation.
e1f5878	2017-01-26	MAPR-25822: The issue that caused INSERT INTO 'table' VALUES command to overwrite previously inserted data is now fixed.

Known Issues and Limitations

Known Issues

Sqoop import to Hive as parquet file fails when the entire cluster is configured to use Tez.	This is because of sqoop's incompatibility with Tez. **Workaround:** Do not configure the entire cluster to use Tez.
Percentage sampling is not supported in `org.apache.hadoop.hive.ql.io.HiveInputFormat`.	Hive uses `org.apache.hadoop.hive.ql.io.HiveInputFormat` by default and so queries like `'SELECT * FROM tablename TABLESAMPLE(20 percent);'` will not work for Hive on Tez. **Workaround:** Instead of `org.apache.hadoop.hive.ql.io.HiveInputFormat`, use `org.apache.hadoop.hive.ql.io.CombineHiveInputFormat`. To change input format, do one of the following:

- Set `hive.tez.input.format` in hive shell. For example:

```
hive> set
hive.tez.input.format=org.apache.ha
doop.hive.ql.io.CombineHiveInputFor
mat;
```

- Add `org.apache.hadoop.hive.ql.io.CombineHiveInputFormat` to `hive-site.xml` file. For example:

```
<property>
  <name>hive.tez.input.format</
name>

<value>org.apache.hadoop.hive.ql.io
.CombineHiveInputFormat</value>
</property>
```

Limitations

- MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.
- MapR does not support HDFS encryption in Hive tables.
- MapR does not support Hbase-0.9X with Hive-2.1.1. Only Hbase-1.X is compatible with Hive-2.1.1.
- MapR does not support LLAP with Hive-2.1.1 since Apache Slider is not in the MapR ecosystem
- MapR does not support Apache Knox and Apache Ranger. HiveServer2 HTTP mode is not available with X-Forwarded-Host header for authorization/audits.
- MapR does not support masking and filtering of rows/columns since Apache Ranger is not in the MapR ecosystem.

Resolved Issues

None.

Hive 1.2.1 Release Notes

The following release notes for the Hive 1.2.1 component are included in the MapR distribution for Apache Hadoop:

Hive 1.2.1-1710 Release Notes
This section provides reference information, including new features, fixes, known issues, and limitations for Hive 1.2.1-1710.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 1.2.1 changelog or the Apache Hive homepage.

Hive Version	1.2.1
Release Date	November 2017

last-updated: Apr 11, 2022

MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hive/tree/1.2.0-mapr-1710[1]
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/, and select your EEP and OS to view the list of package names.

[1] Although the version number in the GitHub source URL is 1.2.0, the content in GitHub applies to MapR Hive 1.2.1

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
5bafc9f	2017-09-20	HIVE-11592: The issue causing the exception when reading an ORC file whose metadata section exceeded protobuf message size limit is now fixed.

Hive 1.2.1-1707 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hive 1.2.1-1707.

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 1.2.1 changelog or the Apache Hive homepage.

Hive Version	1.2.1
Release Date	August 2017
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hive/tree/1.2.0-mapr-1707*
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

* Although the version number in the GitHub source URL is 1.2.0, the content in GitHub applies to MapR Hive 1.2.1

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
bca7e4a	2017-07-12	MAPR-27613: The issue that caused an exception when querying table using HiveServer2 is now fixed.
ce549fa	2017-06-27	MAPR-28068: The issue that caused record reader to throw an exception when trying to read an empty split is now fixed.
b3be3a1	2017-06-15	MAPR-27799: The issue that caused Hive "DROP DATABASE" query to throw exceptions in the logs is now fixed.
5ce6342	2017-06-01	MAPR-22115: The issue that caused Hive to prompt for username and password when using MapR SASL and Kerberos for authentication is now fixed.
12d7524	2017-05-29	MAPR-27427: The issue that caused Hive parser to ignore comments when the line started with two dashes (`--`) is now fixed.
a50cbae	2017-05-24	MAPR-27033: The issue that caused Datanucleus MSSQLServerAdapter to generate incorrect syntax for OFFSET-FETCH clause when working with MS SQL SERVER that is greater than or equal to 2012 is now fixed.
bd2e507	2017-05-24	MAPR-27483: The mismatch in hive and hbase in the result returned when counting number of records when there were NULL fields is now fixed by a new property, `hive.read.all.hbase.column`, which when set to: • `true` will allow Hive to read all columns from HBase table. • `false` may cause Hive to ignore null fields when counting number of records.
ef1fd04	2017-04-24	MAPR-HIVE-13: Added Hive Integration HCatalog Unit Tests to the process of Hive auto testing.

Commit	Date (YYYY-MM-DD)	Comment
b7cbc74	2017-05-23	MAPR-26938: The issue that caused Hive metastore JVM to encrypt more than 256M of data in one shot and crush itself as result is fixed.
d901210	2015-08-04	MAPR-26310: The issue that caused Hive CLI startup to take some time when there were large number of databases is now fixed.
8c1aab1	2017-03-23	MAPR-26388: Hive will now process the version of Spark to determine whether spark-assemby JAR is needed.
40fb934	2017-03-21	MAPR-26325: The issue that caused Hive insert record to have a different column order than the select statement column order is now fixed.
ca0a9c4	2017-03-16	MAPR-26237: The issue that caused some queries with multiple inserts to fail in Hive 1.2 is now fixed.
e5570c1	2017-03-13	MAPR-26406: Hive 1.2 did not work in classic mode because `xercesImpl-2.9.1.jar` was not in the Hive classpath. This is now fixed.
4a5ef75	2017-03-13	MAPR-26138: The outdated `commons-lang3` in Hive was breaking Spark. This is now fixed.
195ec67	2017-03-09	MAPR-26337: The issue that caused exceptions while using Hive delegation tokens is now fixed.
5527271	2017-03-07	MAPR-26371: Some tests were working incorrectly because of missing `mapr.login.conf` file. This is now fixed.

This release by MapR includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
2692767	2017-05-31	HIVE-12790: The Metastore connection was leaking in HiveServer2 as a result of multiple open connections. This is now fixed.
c0654ff	2015-10-16	HIVE-12200: The issue that caused INSERT INTO table using a select statement without a FROM clause to fail is now fixed.
98be5eb	2017-04-26	HIVE-13510: The issue that caused dynamic partitioning to fail when remote metastore was being used is now fixed.
a4fe01f	2017-03-22	HIVE-12988: Includes immprovements to dynamic partition loading.

Commit	Date (YYYY-MM-DD)	Comment
fa0e50e	2017-03-21	HIVE-12908: Includes immprovements to dynamic partition loading.
df7488e	2017-03-20	HIVE-12907: Includes immprovements to dynamic partition loading.
028dde3	2017-03-20	HIVE-12897: Includes immprovements to dynamic partition loading.
966f2ae	2015-07-02	HIVE-11157: The issue that caused `Hive.get(HiveConf)` to return the same Hive object to different user sessions is now fixed.
0503550	2017-03-27	HIVE-13954: Parquet logs will now go to STDERR (similar to Hive) instead of STDOUT (which Hive uses for query output only).
3166549	2017-04-03	HIVE-15331: The issue that caused decimal multiplication with high precision/scale to often return NULL is now fixed.

Hive 1.2.1-1703 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 1.2.1 changelog or the Apache Hive homepage.

Hive Version	1.2.1
Release Date	April 2017
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hive/tree/1.2.0-mapr-1703*
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

* Although the version number in the GitHub source URL is 1.2.0, the content in GitHub applies to MapR Hive 1.2.1

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
8552714	2017-03-03	MAPR-26326: The static variable didRegisterAllFuncs was used in static block before it was initialized, which returned a NPE. This is now fixed.
7f25cad	2017-02-13	MAPR-26108: The issue that caused Hive 1.2 to fail after installation and service restart is now fixed.
a09b18d	2017-02-08	MAPR-26059: The incorrect import in Hive junit tests is now fixed.
fe4d980	2017-02-08	MAPR-25227: Backported Apache Hive JUnit tests to Hive 1.2.
0628470	2017-01-25	MAPR-25845: Set scope is now 'provided' for Avro JAR files in Hive.

This release by MapR includes the following backported issues. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
ee73def	2016-07-15	HIVE-14004: The issue that caused minor compaction to return ArrayIndexOutOfBoundsException is now fixed.
c7632c6	2016-03-09	HIVE-13216: When opening a malformed ORC file, the issue with ORC Reader leaving file open until GC is now fixed.
84540a1	2015-07-02	HIVE-11102: The issue that caused ORC reader to not correctly estimate the size of ACID data files is now fixed.
f2ad544	2015-08-18	HIVE-11429: The default JDBC result set fetch size (# rows it fetches in one RPC call) has been increased to 1000 (from 50).
b7efd1a	2015-10-14	HIVE-11499: Datanucleus will no longer cache classloaders and cause UDFs to permgen leaks.
7a84fd2	2015-11-13	HIVE-11718: Hive JDBC ResultSet.setFetchSize(0) will now return the fetch size.
af87c41	2015-10-20	HIVE-11721: The issue that caused non-ascii characters to not work is now fixed.
0c99222	2015-10-23	HIVE-11901: Hive StorageBasedAuthorizationProvider will no longer require write permission on table for SELECT statements.
383e162	2015-09-26	HIVE-11950: WebHCat status file will now show UTF8 characters.

Commit	Date (YYYY-MM-DD)	Comment
335a188	2015-09-28	HIVE-11977: The issue that caused Hive queries on the table to fail when there was a zero length file with an external avro table in the top level directory is now fixed.
1eac0eb	2015-12-01	HIVE-12182: The ALTER TABLE PARTITION COLUMN command will no longer ignore partition column comments.
ba7a7f8	2015-11-12	HIVE-12365: The issue that caused resource paths, which were removed externally, to be sent to cluster as empty strings resulting in query failures is now fixed.
09c0a0e	2015-12-03	HIVE-12444: The issue that caused Global Limit optimization on ACID table (without base directory) to throw exception is now fixed.
09738e8	2015-12-04	HIVE-12505: The issue that caused insert overwrite command in same encrypted zone to fail to remove some existing files (resulting in incorrect results) is now fixed.
fb91540	2016-01-11	HIVE-12660: The issue that caused HS2 memory leak when used with .hiverc file is now fixed.
109b049	2016-01-05	HIVE-12664: The issue that caused the optimization check for reduce deduplication to only check the first child node for join (resulting in an exception) is now fixed.
1240255	2016-01-19	HIVE-12682: The issue that caused reducers in dynamic partitioning job to spend a lot of time running hadoop.conf.Configuration.getOverlay is now fixed.
a4de686	2016-03-10	HIVE-13144: The issue that caused HS2 to leak ZK ACL objects when the curator retried to create the persistent ephemeral node, which was deleted from ZK, is now fixed.
6d505ae	2016-02-26	HIVE-13160: When HMS was not ready, the issue that caused HS2 to not load UDFs on startup is now fixed.
f0cb6ee	2016-05-27	HIVE-13561: The issue that caused HiveServer2 to leak ClassLoaders when adding a JAR file is now fixed.
b824995	2016-07-13	HIVE-14210: ExecDriver will now call: • SSLFactory truststore reloader threads leaking in HiveServer2 • jobclient.close() function to trigger the clean up of resources after the submitted job completes.

Commit	Date (YYYY-MM-DD)	Comment
7263210	2016-07-21	HIVE-14282: The issue that caused HCatLoader ToDate() function to throw exception when filtering hive partition table by column of DATE datatype is now fixed.
f64f098	2016-07-28	HIVE-14349: UDFLike LIKE -> Regex conversion will now anchor the regexes making the vectorized LIKE use `matches()`.
75292a5	2016-11-01	HIVE-15099: The issue that caused `PTFOperator.PTFInvocation` to throw a NPE when the inputPart was not properly reset is now fixed.
f5ae95d	2016-12-13	HIVE-13452: StatsOptimizer will not return any rows on empty table with group by.
a7b6ab5	2016-11-19	HIVE-15247: The purge option for drop table will now be passed to storage handlers to give the storage handler more control on how to handle drop table.
71f90d7	2016-08-10	HIVE-14436: The Hive query run with mapReduce engine with `hive.optimize.skewjoin` set to `true` will no longer fail with an exception.
8f113da	2016-08-10	HIVE-13754: The issue that caused users reference count to go into negative values preventing tearDownIfUnused from closing the client connection when called and resulting in resource leak in HiveClientCache is now fixed.
7ff1638	2016-09-03	HIVE-10809: The issue that caused HCat FileOutputCommitterContainer to leave behind empty _SCRATCH directories when static partition was added through HCatStorer or HCatWriter is now fixed.
28140c9	2016-06-28	HIVE-14113: The issue that caused create function to fail, but show function to show the failed create function in the function list is now fixed.
9676bb0	2015-07-08	HIVE-11201: The issue that caused HCatalog to ignore user specified avro schema in the table definition, but generate its own avro based on Hive metastore is now fixed.
73736a4	2015-11-20	HIVE-12450: The issue that caused OrcFileMergeOperator to not use correct compression buffer size, but the default 256KB that the output file honored, is now fixed.

Commit	Date (YYYY-MM-DD)	Comment
9844138	2016-02-18	HIVE-13021: The issue that caused GenericUDAFEvaluator.isEstimable(agg) to always returns false is now fixed.

Hive 1.2.1-1611 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 1.2.1 changelog or the Apache Hive homepage.

Hive Version	1.2.1
Release Date	December 9, 2016
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/hive/tree/1.2.0-mapr-1611*
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

* Although the version number in the GitHub source URL is 1.2.0, the content in GitHub applies to MapR Hive 1.2.1

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
0dfe83b	2016-11-09	HIVE-12354: The issue that caused MapJoin with double keys to be slow on MR is now fixed.
ee401749	2016-11-07	HIVE-13632: The issue that caused Hive to fail when inserting empty array into parquet table is now fixed.
c636f923f	2016-11-03	MAPR-25105: The issue that caused data file and table directory to not inherit permissions from the parent directory is now fixed.
9405aa6	2016-10-26	MAPR-25039: The issue that caused load data to fail if the path from where data is loaded has asterisk is now fixed.
23a2e53	2016-10-10	MAPR-24542: The value for `datanucleus.schema.autoCreateAll` property is now `true` by default.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
14e848a	2016-10-07	HIVE-10956: The issue that caused HS2 to leak HMS connections is now fixed.
b90cb52	2016-09-30	MAPR-24696: The issue that caused Hive partition directory URI to be double-encoded is now fixed.
0779f7e	2016-09-22	HIVE-10685: The issue that caused `ALTER TABLE orders CONCATENATE` operation to create duplicate rows in table is now fixed.
eba492c	2016-09-20	MAPR-24425: The issue that caused `msck repair` command on partitioned table to return "Partitions not in metastore" error is now fixed.

Hive 1.2.1-1609 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 1.2.1 changelog or the Apache Hive homepage.

Hive Version	1.2.1
Release Date	September 30, 2016
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/hive/tree/1.2.0-mapr-1609*
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hive-1.2.201609261225-1.noarch.rpm • mapr-hive_1.2.201609261226_all.deb • mapr-hivemetastore-1.2.201609261225-1.noarch.rpm • mapr-hivemetastore_1.2.201609261226_all.deb • mapr-hiveserver2-1.2.201609261225-1.noarch.rpm • mapr-hiveserver2_1.2.201609261226_all.deb • mapr-hivewebhcat-1.2.201609261225-1.noarch.rpm • mapr-hivewebhcat_1.2.201609261226_all.deb

* Although the version number in the GitHub source URL is 1.2.0, the content in GitHub applies to MapR Hive 1.2.1

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
093203c4	2016-08-29	HIVE-5277: The issue causing HBaseStorageHandler to skip first rows with null values when only row key is selected is now fixed.

Hive 1.2.1-1608 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 1.2.1 changelog or the Apache Hive homepage.

Hive Version	1.2.1
Release Date	September 1, 2016
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/hive/tree/1.2.0-mapr-1608
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hive-1.2.201608311134-1.noarch.rpm • mapr-hive_1.2.201608311134_all.deb • mapr-hivemetastore-1.2.201608311134-1.noarch.rpm • mapr-hivemetastore_1.2.201608311134_all.deb • mapr-hiveserver2-1.2.201608311134-1.noarch.rpm • mapr-hiveserver2_1.2.201608311134_all.deb • mapr-hivewebhcat-1.2.201608311134-1.noarch.rpm • mapr-hivewebhcat_1.2.201608311134_all.deb

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
1fbbaa04	2016-08-16	HIVE-13120: The issue causing the ORC split generation in HiveServer2 to return permission errors is now fixed.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
ee26f8a0	2016-08-15	MAPR-24242: The incorrect timestamp format in the partition name when table was partitioned by timestamp and LOAD DATA was used is now fixed.
04034276	2016-08-12	MAPR-24248: The `isNonLocalScratchDirUsed` in `getLocalScratchDir()` method is now set to `false` to allow operations with scratch directories to use the local filesystem instead of the MapR filesystem.
cabc37a4f	2016-08-10	HIVE-11617: The issue causing explain job to (sometimes) not finish or be very slow when there are many lateral views is now fixed.
82de9cc3	2016-08-01	MAPR-24105: The issue causing Spark engine for Hive to fail with an execution error is now fixed.
5115eef0	2016-05-04	HIVE-13677: When folding CASE expressions, Hive no longer throws exception during Serialization.
79dc6054	2015-09-14	HIVE-11817: A query with OVER statement will no longer fail when table contains null values.

Hive 1.2.1-1607 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 1.2.1 changelog or the Apache Hive homepage.

Hive Version	1.2.1
Release Date	July 29, 2016
MapR Version Interoperability	See Hive and HCatalog Support Matrix and Ecosystem Support Matrix
Source on GitHub	https://github.com/mapr/hive/tree/1.2.0-mapr-1607
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hive-1.2.201607240717-1.noarch.rpm • mapr-hive_1.2.201607240717_all.deb • mapr-hivemetastore-1.2.201607240717-1.noarch.rpm • mapr-hivemetastore_1.2.201607240717_all.deb • mapr-hiveserver2-1.2.201607240717-1.noarch.rpm • mapr-hiveserver2_1.2.201607240717_all.deb • mapr-hivewebhcat-1.2.201607240717-1.noarch.rpm • mapr-hivewebhcat_1.2.201607240717_all.deb

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
12e7cce	2016-07-22	MAPR- 24026/HIVE-11817: A query with OVER statement will no longer fail when table contains null values.
b77136e	2016-07-13	MAPR-23970: When running a query, checks are now included to determine if current filesystem is non-local before running getScratchDir() method, which assumes the filesystem is not local.
c1175423	2016-06-06	MAPR-23560: Symbolic links to the PID files to track memory utilization of HiveServer2 and HiveMetaStore processes are now available in `/opt/mapr/pid`.
70a5bcc	2016-06-02	HIVE-11498: HIVE authorization will skip (and no longer check) permissions for dummy entities.
0304149f	2016-05-20	MAPR-22357: The issue causing -1 to be returned is now fixed: • The property, `hive.rpc.query.plan` in `hive-site.xml` file, can be set to `true` to specify whether to send the query plan via local resource or RPC. • The parameter, `hive.kryo.buffer.size` in `hive-site.xml` file, can be used to control Kryo buffer size. The default value for Kryo buffer size is now 10 * 4096.

The following issues were backported from Apache Hive-1.2.2. You may also refer to the Apache Hive 1.2.2 Release Notes.

Commit	Date (YYYY-MM-DD)	Comment
c6d08f102b	2016-05-19	HIVE-9013: When auth is enabled, the `set` command will no longer expose the DB metastore password.
0544a99b2	2016-05-19	HOVE-10308: The issue that caused the `java.lang.IllegalArgumentException` when vectorization was executed has been fixed.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
161b711c	2016-05-18	HIVE-11151: The calcite rule will no longer add `is not null` filters on constants.
923163f9	2016-05-18	HIVE-11171: The issue that caused the join reordering algorithm to introduce projects between joints, which did not trigger the multijoin optimization in SemanticAnalyzer, has been fixed.
f96969f8c	2016-05-18	HIVE-11216: The issue that caused UDF GenericUDFMapKeys to throw NPE when a null map value was passed in has been fixed.
a5889f7	2016-05-18	HIVE-11224: The issue that caused AggregateStatsCache to trigger `java.util.ConcurrentModificationException` has been fixed.
90cd1f42ad	2016-05-18	HIVE-11301: The issue in thrift metastore that resulted in disconnecting when getting stats has been fixed.
c6a8f75fc1	2016-05-18	HIUVE-11344: The issue that made PartInfo object unusable if `HCatSplit.write` was called has been fixed.
a8ed673a	2016-05-18	HIVE-11470: The issue that caused DynamicPartFileRecordWriterContainer to throw a NullPointerException when fetching a local file-writer instance with null part-keys has been fixed.
90a3d4c3c	2016-05-18	HIVE11606: The issue that caused bucket map joins to produce incorrect results in case of container reuse has been fixed.
08b07ca2d	2016-05-18	HIVE-11745: The issue that caused alter table exchange partition with multiple partitions to not work has been fixed.
1efbb81f9c	2016-05-18	HOVE-12021: The issue that caused HivePreFilteringRule to introduce incorrect common operands has been fixed.
87f2d1b0	2016-05-18	HIVE-12083: The issue that caused an empty AggrStats object to be returned if partNames or colNames were empty has been fixed.
135871325	2016-05-18	HIVE-12344: The issue that caused projectNonColummnEquiConditions method in HiveCalciteUtil to assign the wrong type for newly created conditions has been fixed.

Commit	Date (YYYY-MM-DD)	Comment
6bfa219	2016-05-18	HIVE-12345: The issue that caused hidden configuration variables to be visible through beeline when connecting to HS2 has been fixed.
d95345809	2016-05-18	HIVE-12937: The issue that caused users of DbNotificationListener to have a large number of old notification events that were not deleted has been fixed.
492afa2778	2016-05-18	HIVE-13115: The issue that caused MetaStore Direct SQL getPartitions() method to throw an exception when the column schemas for a partition were null has been fixed.
a77035ef	2016-05-18	HIVE-10996: The issue that caused multi-join inner query to produce incorrect results has been fixed.
39e4a9ea5	2016-05-18	HIVE-11172: The issue that caused vectorization to return wrong results for an aggregate query with a where clause and without a group by has been fixed.
04844ad38	2016-05-18	HIVE-11517: The issue that caused query on a Q file (that uses ORC with vectorization enabled) to produce incorrect results has been fixed.
373ba93f	2016-05-18	HIVE-11605: The issue that caused incorrect results when converting to a bucket map join has been fixed.
2336c38b6	2016-05-18	HIVE-11988: Users who do not have permission to create table in hive can no longer import data for a table.
2489ffec	2016-05-18	HIVE-12437: The issue that caused SMB join in Tez to fail when one of the tables was empty has been fixed.
bfa62787	2016-05-18	HIVE-12947: The issue that caused SMB join to fail when reconnecting, during container re-use, the (already connected) work items has been fixed.
4abf5672	2016-05-18	HIVE-12981: ThriftCLIService will no longer use `getShortName()`, which assumes a short name is always the part before "@" and "/", as it is no longer compatible.

Hive 1.2.1-1605 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 1.2.1 changelog or the Apache Hive homepage.

Hive Version	1.2.1
Release Date	June 6, 2016

MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/hive/tree/1.2.0-mapr-1605
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hive-1.2.201606020917-1.noarch.rpm • mapr-hive_1.2.201606020917_all.deb • mapr-hivemetastore-1.2.201606020917-1.noarch.rpm • mapr-hivemetastore_1.2.201606020917_all.deb • mapr-hiveserver2-1.2.201606020917-1.noarch.rpm • mapr-hiveserver2_1.2.201606020917_all.deb • mapr-hivewebhcat-1.2.201606020917-1.noarch.rpm • mapr-hivewebhcat_1.2.201606020917_all.deb

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
0304149f	2016-05-20	MAPR-22357: The buffer underflow (while performing Hive queries) caused by small Kryo buffer size is now fixed with a new parameter, `hive.kryo.buffer.size` in `hive-site.xml` file, which can be used to control Kryo buffer size. The default value for Kryo buffer size is now `10 * 4096`.
b3e811b	2016-05-12	MAPR-23302: ListBucketing tables with uppercase skew values can now be created.
5673a07	2016-05-11	MAPR-23221: The mismatch between the MapR native library included in the standalone Hive JDBC jar file and the native library in MapR is now fixed. The standalone Hive JDBC jar file no longer inlcudes MapR native library.
130887f	2016-05-10	MAPR-22222: The delay as a result of a single reduce task in the last stage is now fixed with a new parameter, `hive.groupby.limit.extrastep`, which can be used to enable or disable a new MR job for sorting the final output.
b48f8e6	2016-05-10	MAPR-23264: The issue with access to Hive 1.2 table data is now fixed.

Commit	Date (YYYY-MM-DD)	Comment
0b3fa65	2016-04-28	MAPR-23220: The issue that caused IndexOutOfBoundsException when running a query in Hive 1.2 is now fixed.
7976893	2016-04-28	MAPR-23029: Changed the Primary Key for table PARTITION_STATS_V2 from varchar(4000) to varchar(3072) as MySQL does not support varchar(4000) as Primary Key.
fa8e936	2016-04-26	HIVE-12469: The commons-collections 3.2.1 library that allowed invocation of arbitrary code is now replaced with commons-collections 3.2.2 to resolve this issue.
703c2f3	2016-04-25	MAPR-23153: Hive will no longer create scratch dir with incorrect permissions when a CTAS query is run. If CTAS query is run, now, the table root directory with correct permissions from `fs.permissions.umask-mode` will be created first and then the scratch dir with permissions from `hive.scrach.dir.permissions` will be created.
41a827a	2016-04-14	MAPR-23082: Beeline will no longer fail with NullPointerException when MAPR-SASL is used.
	2016-04-14	MAPR-23084: The `datanucleus.schema.autoCreateAll` property in `hive-site.xml` file is now set to `true` by default.
9f82b07	2016-04-01	MAPR-22994: The issue causing Hive table creation on existing HBase table to fail has been fixed.

Hive 1.2.1-1603 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform.

Hive Version	1.2.1
Release Date	April 4, 2016
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 .
Source on GitHub	https://github.com/mapr/hive/tree/1.2.1-mapr-1603
Maven Artifacts	https://repository.mapr.com/maven/

last-updated: Apr 11, 2022

Package Names	The following packages are associated with this release: • mapr-hive-1.2.201604040723-1.noarch.rpm • mapr_hive_1.2.201604040723_all.deb • mapr-hivemetastore-1.2.201604040723-1.noarch.rpm • mapr_hivemetastore_1.2.201604040723_all.deb • mapr-hiveserver2-1.2.201604040723-1.noarch.rpm • mapr_hiveserver2_1.2.201604040723_all.deb • mapr-hivewebhcat-1.2.201604040723-1.noarch.rpm • mapr_hivewebhcat_1.2.201604040723_all.deb

New in this Release

This release of Apache Hive includes the following behavior change that is specific to MapR:

New default value for `datanucleus.schema.autoCreateAll` property	The default value for the `datanucleus.schema.autoCreateAll` property was change from `true` to `false` in the hive-default.xml.template file.

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
9f82b075	2016-04-01	MAPR-22994: Hive 1.2.1 no longer fails to create and query a Hive table on an existing HBase 1.1.1 table.
fab6e30	2016-03-30	MAPR-22966: Hiveserver2 no longer fails to start when it uses LDAP authentication on a secure cluster.
8bff9ae	2016-03-28	MAPR-22945: HiveServer2 no longer fails to start when PAM authentication is enabled.
ea85d79	2016-03-25	MAPR-22928: The `datanucleus.schema.autoCreateAll` property is now set to `false` by default.
b74becc	2016-03-14	MAPR-22848: Backported HIVE-11940 to improve the performance of the INSERT OVERWRITE query.
7f8a9ef	2016-02-29	MAPR-21949: Constant propagation optimizer no longer returns string values in an incorrect format.
3899330	2016-03-09	MAPR-22810: Backported HIVE-11841 so that `KeyValuesInputMerger` no longer creates huge logs.
449a587	2016-03-02	MAPR-22701: Backported HIVE-12875 to reduce the chance of an authorization vulnerability.

GitHub Commit	Date (YYYY-MM-DD)	Comment
40e2baa	2016-03-02	MAPR-22759: Hive is now able to identify MapR filesystem scheme while adding resources through CLI.
8a9a3d2	2015-03-24	MAPR-22691: Default values are now provided for HiveServer2 Kerberos principal and keytab.
517e885	2016-02-12	MAPR-20523: Queries that use any type of join between avro tables no longer fail or give incorrect results.
15c00db	2016-02-12	MAPR-22560: Backported HIVE-10021 so that "Alter index rebuild" statements no longer fail when HiveServer2 uses Sentry authorization.
763e8614	2016-01-26	MAPR-22254: Hive only invokes the open method when the plan file exists. Therefore, it no longer prints the following message to the Hive shell when the query does not have a reduce phase: "`LookupFid error No such file or directory(2)`."

Hive 1.2.1-1601 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Hive 1.2.1 changelog or the Apache Hive homepage.

Version	1.2.1
Release Date	February 1, 2016
Source on GitHub	https://github.com/mapr/hive/tree/1.2.0-mapr-1601
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hive-1.2.201601281422-1.noarch.rpm • mapr-hive_1.2.201601281422_all.deb • mapr-hivemetastore-1.2.201601281422-1.noarch.rpm • mapr-hivemetastore_1.2.201601281422_all.deb • mapr-hiveserver2-1.2.201601281422-1.noarch.rpm • mapr-hiveserver2_1.2.201601281422_all.deb • mapr-hivewebhcat_1.2.201601281422_all.deb • mapr-hivewebhcat-1.2.201601281422-1.noarch.rpm

New in this Release

This release of Apache Hive includes the following behavior changes that are specific to MapR:

DataNucleus versions were updated.

The following DataNucleus versions were updated:

Component	Old Version	New Version
datanucleus-api-jdo	3.2.6	4.2.1

Component	Old Version	New Version
datanucleus-core	3.2.1.0	4.1.6
datanucleus-rdbms	3.2.9.0	4.1.7

DataNucleus properties were renamed.

The following DataNucleus properties were renamed:

Old Name	New Name
datanucleus.validateTables	datanucleus.schema.validateTable
datanucleus.validateConstraints	datanucleus.schema.validateCons
datanucleus.validateColumns	datanucleus.schema.validateColu
datanucleus.autoCreateSchema	datanucleus.schema.autoCreateAl

DataNucleus Property `datanucleus.fixedDatastore` was deleted.

Added integration with HBase 1.1.

For details on the features available in the open source version of this component, see the Apache Hive 1.2.1 changelog or the Apache Hive homepage.

Hive Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
fe18d11	2016-01-12	MAPR-22108 Hive insert to HBase table no longer fails with `java.lang.RuntimeException`: `java.lang.NoSuchMethodError: org.apache.hadoop.hbase.client.Put.setDurability`
b189fa2	2015-12-07	MAPR-21228: Hive queries no longer fail with exception: Table '`hive.DELETEME1xx`' doesn't exist.
13a8406	2015-12-11	MAPR-20207: Backported HIVE-9599 to remove extra, conflicting jars from `hive-jdbc-standalone.jar`.

GitHub Commit	Date (YYYY-MM-DD)	Comment
7646ece	2015-11-19	MAPR-20263: Hive Metastore no longer incorrectly determines the authentication method when Hive jobs are created by other components such as Sqoop, Oozie, or Spark.
c4345c3	2015-11-16	MAPR-21349: The `insert overwrite` directory command no longer fails.
b817902	2015-11-06	MAPR-21315 Backported HIVE-10802 so that table join queries with a constant field in the select statement no longer throw an `IndexOutOfBoundsException`.
03d2c74	2015-11-04	MAPR-21238: Backported Hive-11502 so that map side aggregation is no longer extremely slow.

Hive 1.2.1-1510 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Hive 1.2.1 changelog or the Apache Hive homepage.

Hive Version	1.2.1
Release Date	November 20, 2015
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/hive/tree/1.2.0-mapr-1510
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hive-1.2.201511180952-1.noarch.rpm • mapr-hive_1.2.201511180952_all.deb • mapr-hivemetastore-1.2.201511180952-1.noarch.rpm • mapr-hivemetastore_1.2.201511180952_all.deb • mapr-hiveserver2-1.2.201511180952-1.noarch.rpm • mapr-hiveserver2_1.2.201511180952_all.deb • mapr-hivewebhcat-1.2.201511180952-1.noarch.rpm • mapr-hivewebhcat_1.2.201511180952_all.deb

New in this Release

This release of Apache Hive includes the following behavior change that is specific to MapR:

Flexible HiveServer2 Authentication Support

When the cluster is secure, HiveServer2 accepts both MAPR-SASL and PAM for in-bound authentication. This allows a single instance of HiveServer2 to accept

	both MapR-SASL authentication from Hue and PAM authentication from JDBC/ODBC connections.

For details on the features available in the open source version of this component, see the Apache Hive 1.2.1 changelog or the Apache Hive homepage.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub:

GitHub Commit	Date (YYY-MM-DD)	Comment
ad20274	2015-10-23	MAPR-21015: Backported HIVE-11456 so that HCatStorer honors the `mapreduce.output.basename`.
b724320	2015-10-22	MAPR-21055: The insert overwrite command no longer fails to remove old data files before it overwrites the table.
3fb70d8	2015-10-21	MAPR-21035: Hive 1.2 queries on parquet files no longer fail.
3e36752	2015-10-16	MAPR 20096: SMB Join no longer fails on Hive 1.2.
923ab18	2015-10-16	MAPR-19370: On a secure cluster, HiveServer2 now accepts both MapR-SASL and PAM authentication.
12bca74	2015-09-21	MAPR-20191: On the MapR 5.0 sandbox, the Hive classpath no longer contains newlines.
d9f47eb	2015-09-09	HIVE-11688: Backported HIVE-11688 to resolve the issue where OrcRawRecordMerger does not close the primary reader.

Hive 1.2.1-1508 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Hive 1.2.1 changelog or the Apache Hive homepage.

Hive Version	1.2.1
Release Date	Sept 25, 2015
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	1.2.1-mapr-1508
MapR Version Compatibility	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

New in this Release

This is the initial release of Hive 1.2.1 for the MapR distribution for Hadoop. In addition to Hive 1.2.1 features, this release includes the following behavior changes and features:

- **HiveServer2 Authentication**

  On secure MapR clusters, HiveServer2 uses PAM by default.

- **WebHCat Changes**

  The warden.hcat.conf file is no longer installed with mapr-hive package. Instead, it is installed with the mapr-hivewebhcat package.

- **Cost-Based optimization**

  Cost-based optimization is enabled by default.

- **Hive ODBC Driver 2.1.8**

  For Hive 1.2.1, use Hive ODBC driver version 2.1.8 or above. This driver cannot be installed on CentOS 7.

Hive Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster and you can use SparkQL to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
bb40d76	2015-08-11	MAPR-19358: HiveServer2 and HiveMetastore no longer fail to start they are installed on MapR 5.0.
62592e8	2015-08-11	MAPR-19756: The Hive shell no longer fails to start.
d620ceb	2015-08-15	MAPR-19778: WebHCat service no longer displays a MalformedURLException in`/opt/mapr/logs/adminuiapp.log`.
e8f0545	2015-08-18	MAPR-20051: HiveMetastore no longer fails to create the tables correctly and no longer throws a lock exception when you try to run a query.
4769b27	2015-08-27	MAPR- 20197: templeton.storage.class property is now set to org.apache.hive.hcatalog.templeton.tool.HDFSStorage instead of org.apache.hcatalog.templeton.tool.HDFSStorage.
a546130	2015-08-31	MAPR-13215: On a secure cluster, hadoop.proxyuser.mapr.groups is now able to restrict the groups that can be impersonated.
2ad68d4	2015-09-01	Hadoop23Shims no longer includes HDFS encryption related code.

last-updated: Apr 11, 2022

Hive 1.0 Release Notes

The following release notes for the Hive 1.0 component are included in the MapR distribution for Apache Hadoop

Hive 1.0-1611 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform.

Version	1.0
Release Date	December 8, 2016
Source on GitHub	https://github.com/mapr/hive/tree/1.0.0-mapr-1611
MapR Version Compatibility	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	• mapr-hive-1.0.201612011449-1.noarch.rpm • mapr-hive_1.0.201612011449_all.deb • mapr-hivemetastore-1.0.201612011449-1.noarch.rpm • mapr-hivemetastore_1.0.201612011449_all.deb • mapr-hiveserver2-1.0.201612011449-1.noarch.rpm • mapr-hiveserver2_1.0.201612011449_all.deb • mapr-hivewebhcat-1.0.201612011449-1.noarch.rpm • mapr-hivewebhcat_1.0.201612011449_all.deb

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
e8300c35	2016-10-05	MAPR-24696: The issue that caused Hive partition directory URI to be double-encoded is now fixed.
98957a05	2016-10-04	MAPR-24425: The issue that caused `msck repair` command on partitioned table to return "Partitions not in metastore" error is now fixed.
e54cf84fc	2016-09-09	MAPR-24508: The `insert overwrite` command will now remove old data files before it overwrites the table.

Commit	Date (YYYY-MM-DD)	Comment
462ce1b5	2016-08-26	MAPR-24380: When the `load data local inpath <path_to_file> overwrite into <table>` command is run, Hive will now process each file separately instead of copying entire folder with files in it.

Hive 1.0-1608 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform.

Version	1.0
Release Date	September 1, 2016
Source on GitHub	https://github.com/mapr/hive/tree/1.0.0-mapr-1608
MapR Version Compatibility	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	• mapr-hive-1.0.201608311152-1.noarch.rpm • mapr-hive_1.0.201608311152_all.deb • mapr-hivemetastore-1.0.201608311152-1.noarch.rpm • mapr-hivemetastore_1.0.201608311152_all.deb • mapr-hiveserver2-1.0.201608311152-1.noarch.rpm • mapr-hiveserver2_1.0.201608311152_all.deb • mapr-hivewebhcat-1.0.201608311152-1.noarch.rpm • mapr-hivewebhcat_1.0.201608311152_all.deb

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
684b09a91	2016-08-18	MAPR-24275: The issue causing a query on hive to fail with an IllegalStateException is now fixed.
4a60eb1db	2016-08-12	MAPR-24248: The `isNonLocalScratchDirUsed` in `getLocalScratchDir()` method is now set to `false` to allow operations with scratch directories to use the local filesystem instead of the MapR filesystem.
e4f49cc398	2016-08-10	HIVE-11617: The issue causing explain job to (sometimes) not finish or be very slow when there are many lateral views is now fixed.

Commit	Date (YYYY-MM-DD)	Comment
3fcf5367	2016-07-16	MAPR-23970: When running a query, checks are now included to determine if current filesystem is non-local before running `getScratchDir()` method, which assumes the filesystem is not local.
d854ef03	2015-11-03	HIVE-12206: Hive no longer throws an exception when Kryo's classloader is set without the UDF jar.

Hive 1.0-1606 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform.

Version	1.0
Release Date	July 1, 2016
Source on GitHub	https://github.com/mapr/hive/tree/1.0.0-mapr-1606
MapR Version Compatibility	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	• mapr-hive-1.0.201606271053-1.noarch.rpm • mapr-hive_1.0.201606271053_all.deb • mapr-hivemetastore-1.0.201606271053-1.noarch.rpm • mapr-hivemetastore_1.0.201606271053_all.deb • mapr-hiveserver2-1.0.201606271053-1.noarch.rpm • mapr-hiveserver2_1.0.201606271053_all.deb • mapr-hivewebhcat-1.0.201606271053-1.noarch.rpm • mapr-hivewebhcat_1.0.201606271053_all.deb

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
2e82bd5	2016-06-09	HIVE-9839: If a HiveSQLException is thrown during a query, the opHandle is cleaned up if the query is synchronous and for asynchronous queries, the opHandle is retained and passed back for reporting errors.
9f51e20	2016-06-09	HIVE-12766: The objects created by Tez/Yarn will now be freed as TezTask will now close DagClient after execution.

Commit	Date (YYYY-MM-DD)	Comment
c1b67e6	2016-06-06	MAPR-23560: Symbolic links to the PID files to track memory utilization of HiveServer2 and HiveMetaStore processes are now available in `/opt/mapr/pid`.
9678f87	2016-06-02	MAPR-23029: Changed the Primary Key for table PARTITION_STATS_V2 from varchar(4000) to varchar(3072) as MySQL does not support varchar(4000) as Primary Key.
863c1a	2016-05-18	MAPR-21786: The issue with incorrect permissions (because of the fix for MAPR-19894) resulting in “Operation not permitted” error is now fixed.
71d345f	2016-05-16	MAPR-23302: ListBucketing tables with uppercase skew values can now be created.
9f67c78	2016-04-28	MAPR-23220: The issue that caused IndexOutOfBoundsException when running a query in Hive is now fixed.

Hive 1.0-1604 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform.

Version	1.0
Release Date	May 11, 2016
Source on GitHub	https://github.com/mapr/hive/tree/1.0.0-mapr-1604
MapR Version Compatibility	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	• mapr-hive-1.0.201605100956-1.noarch.rpm • mapr-hive_1.0.201605100956_all.deb • mapr-hivemetastore-1.0.201605100956-1.noarch.rpm • mapr-hivemetastore_1.0.201605100956_all.deb • mapr-hiveserver2-1.0.201605100956-1.noarch.rpm • mapr-hiveserver2_1.0.201605100956_all.deb • mapr-hivewebhcat-1.0.201605100956-1.noarch.rpm • mapr-hivewebhcat_1.0.201605100956_all.deb

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
55d0842	2016-04-27	MAPR-23141: The issue that caused UNION ALL and ORDER BY queries to fail with ArrayIndexOutOfBoundsException has been fixed.
0be26b8	2016-04-14	MAPR-23084: Added and set `datanucleus.schema.autoCreateAll` property to `true` by default in `hive-site.xml` file to create necessary schema on startup if schema does not already exist.
48a3e13	2016-03-15	MAPR-22840: SHOW TABLE EXTENDED will now show the correct lastUpdateTime of partition's filesystem.
2cd55b9	2016-03-22	MAPR-22913: Backported HIVE-9749 (MAPR-20650) to provide the option to configure whether or not the Hive Metastore version should be updated automatically.
7e8e6dc7	2016-04-08	MAPR-23004: The issue (HIVE-11351) with getting ColumnInfo using RowResolver is now fixed.
ab257c5	2016-04-12	MAPR-22930: Backported HIVE-11592 to set an upperbound on the protobuf message size.
a6bc387	2016-04-12	MAPR-23039: Changed properties to allow Primary Key for PARTITION_STATS_V2 from varchar(4000) to varchar(3072).
e06d690	2016-04-14	MAPR-23082: The issue causing Beeline to fail with java.lang.NullPointerException when MAPRSASL was used has been fixed.
a9b20b3	2016-04-21	MAPR-23153: The issue causing wrong permissions when using CTAS query (with `hive.optimize.insert.dest.volume` set to `true`) in Hive has been fixed.

Hive 1.0-1603 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform.

Hive Version	1.0
Release Date	April 4 2016
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 .
Source on GitHub	https://github.com/mapr/hive/tree/1.0.0-mapr-1603
Maven Artifacts	https://repository.mapr.com/maven/

Package Names	The following packages are associated with this release: • mapr-hive-1.0.201603301009-1.noarch.rpm • mapr-hive_1.0.201603301009_all.deb • mapr-hivemetastore-1.0.201603301009-1.noarch.rpm • mapr-hivemetastore_1.0.201603301009_all.deb • mapr-hiveserver2-1.0.201603301009-1.noarch.rpm • mapr-hiveserver2_1.0.201603301009_all.deb • mapr-hivewebhcat-1.0.201603301009-1.noarch.rpm • mapr-hivewebhcat_1.0.201603301009_all.deb

New in this Release

This release of Apache Hive includes the following behavior change that is specific to MapR:

New default value for `datanucleus.schema.autoCreateAll` property	The default value for the `datanucleus.schema.autoCreateAll` property was change from `true` to `false` in the hive-default.xml.template file.

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
da7ff8a	2016-03-30	MAPR-22966: Hiveserver2 no longer fails to start when it uses LDAP authentication on a secure cluster.
9937578	2016-03-28	MAPR-22945: HiveServer2 no longer fails to start when PAM authentication is enabled.
82b7a2d	2016-03-25	MAPR-22928: The `datanucleus.schema.autoCreateAll` property is now set to `false` by default.
56462ca	2016-03-09	MAPR-22810: Backported HIVE-11841 so that `KeyValuesInputMerger` no longer creates huge logs.
cbd034c	2016-03-03	MAPR-21949: Constant propagation optimizer no longer returns string values in an incorrect format.
fa75cd5	2016-03-02	MAPR-22701: Backported HIVE-12875 to reduce the chance of an authorization vulnerability.
502b02a	2016-02-12	MAPR-22560: Backported HIVE-10021 so that "Alter index rebuild" statements no longer fail when HiveServer2 uses Sentry authorization.
1f2e744	2016-02-12	MAPR-20523: Queries that use any type of join between avro tables no longer fail or give incorrect results.

GitHub Commit	Date (YYYY-MM-DD)	Comment
d3d73cc	2016-02-09	MAPR-22435: `beeline.sh` no longer includes an unnecessary "if" statement.

Hive 1.0-1601 Release Notes

Below are release notes for the Hive component included in the MapR Distribution for Apache Hadoop.

Hive Version	1.0
Release Date	February 1, 2016
MapR Version Interoperability	See the Hive and HCatalog Support Matrix and the Ecosystem Support Matrix
Source on GitHub	https://github.com/mapr/hive/tree/1.0.0-mapr-1601
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hive-1.0.201601281548-1.noarch.rpm • mapr-hive_1.0.201601281549_all.deb • mapr-hivemetastore-1.0.201601281548-1.noarch.rpm • mapr-hivemetastore_1.0.201601281549_all.deb • mapr-hiveserver2-1.0.201601281548-1.noarch.rpm • mapr-hiveserver2_1.0.201601281549_all.deb • mapr-hivewebhcat-1.0.201601281548-1.noarch.rpm • mapr-hivewebhcat_1.0.201601281549_all.deb

New in this Release

This release of Apache Hive includes the following behavior change that is specific to MapR:

DataNucleus versions were updated.

The following DataNucleus versions were updated:

Component	Old Version	New Version
datanucleus-api-jdo	3.2.6	4.2.1
datanucleus-core	3.2.1.0	4.1.6
datanucleus-rdbms	3.2.9	4.1.7

DataNucleus properties were renamed.

The following DataNucleus properties were renamed:

Old Name	New Name
datanucleus.validateTables	datanucleus.schema.validateTable
datanucleus.validateColumns	datanucleus.schema.validateColu
datanucleus.validateConstraints	datanucleus.schema.validateCons
datanucleus.autoCreateSchema	datanucleus.schema.autoCreateAl

DataNucleus Property datanucleus.fixedDatastore was deleted.

For details on the features available in the open source version of this component, see the Apache Hive 1.0 changelog or the Apache Hive homepage.

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
b3c72d3	2015-12-09	MAPR-21752: When you load data into a table partition, Operation not permitted messages no longer display for chgrp and chmod commands.
96c96e7	2015-12-07	MAPR-21228: Hive queries no longer fail with exception: Table 'hive.DELETEME1xx' doesn't exist.
b03fb23	2015-12-03	MAPR-20263: Hive Metastore no longer incorrectly determines the authentication method when Hive jobs are created by other components such as Sqoop, Oozie, or Spark.
18c12ac	2015-12-02	MAPR-21631: Backported HIVE-9867 to migrate usage of deprecated Calcite methods. This prevents the ClassCastException from occurring when you submit Hive queries.
49ecd8e	2015-11-18	MAPR-21352: Upgrading hive-0.7.1 to hive-1.0 no longer fails at 008-HIVE-2246.mysql.sql.
d75123c	2015-11-04	MAPR-21238: Backported Hive-11502 so that map side aggregation is no longer extremely slow.
f102c02	2015-11-03	MAPR-20207: Backported HIVE-9599 to remove extra, conflicting jars from hive-jdbc-standalone.jar.
5f8c059	2015-10-30	MAPR-19894: Fixed the java.lang.NullPointerException in the MoveTask class.

Hive 1.0-1510 Release Notes

Below are release notes for the Hive component included in the MapR Distribution for Apache Hadoop.

Hive Version	1.0
Release Date	November 20, 2015
MapR Version Interoperability	See the Hive and HCatalog Support Matrix and the Ecosystem Support Matrix
Source on GitHub	https://github.com/mapr/hive/tree/1.0.0-mapr-1510
Maven Artifacts	https://repository.mapr.com/maven/

last-updated: Apr 11, 2022

Package Names	The following packages are associated with this release: • mapr-hive-1.0.201511180941-1.noarch.rpm • mapr-hive_1.0.201511180941_all.deb • mapr-hivemetastore-1.0.201511180941-1.noarch.rpm • mapr-hivemetastore_1.0.201511180941_all.deb • mapr-hiveserver2-1.0.201511180941-1.noarch.rpm • mapr-hiveserver2_1.0.201511180941_all.deb • mapr-hivewebhcat-1.0.201511180941-1.noarch.rpm • mapr-hivewebhcat_1.0.201511180941_all.deb

New in this Release

This release of Apache Hive includes the following behavior change that is specific to MapR:

Flexible HiveServer2 Authentication Support	When the cluster is secure, HiveServer2 accepts both MAPR-SASL and PAM for in-bound authentication. This allows a single instance of HiveServer2 to accept both MapR-SASL authentication from Hue and PAM authentication from JDBC/ODBC connections.

For details on the features available in the open source version of this component, see thehttps://issues.apache.org/jira/secure/ReleaseNote.jspa?version=12324986&styleName=Text&projectId=12310843Apache Hive 1.0 changelog or the Apache Hive homepage.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
c582f58	2015-10-27	MAPR-19894: The insert overwrite table partition command no longer ignores fs.permission.umask-mode when hive.optimize.insert.dest.volume is enabled.
9ad855a	2015-10-23	MAPR-21035 Hive 1.2 queries on parquet files no longer fail.
20a0ba9	2015-10-13	MAPR-19370: On a secure cluster, HiveServer2 now accepts both MapR-SASL and PAM authentication.
462be0a	2015-09-21	MAPR-20191: On the MapR 5.x sandbox, the Hive classpath no longer contains newlines.
bdf9916	2015-09-08	HIVE-11688: Backported HIVE-11688 to resolve the issue where OrcRawRecordMerger does not close the primary reader.

Hive 1.0-1508 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Hive 1.0 changelog or the Apache Hive homepage.

Hive Version	1.0
Release Date	Sept 22, 2015
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	1.0.0-mapr-1508
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

New in this Release

This release of Hive 1.0 for the MapR distribution for Hadoop includes the following behavior changes:

- **HiveServer2 Authentication Changes** (Available for MapR cluster version 4.1 and above)

  On secure MapR clusters, MapR-SASL is not longer the default for HiveServer2. Instead, HiveServer2 uses PAM by default.

- **WebHCat Changes**

  The warden.hcat.conf file is no longer installed with mapr-hive package. Instead, it is installed with the mapr-hivewebhcat package.

- **Cost-based optimization**

  Cost-based optimization is disabled by default. For information on how to enable this option, see the Apache Hive documentation.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
3ce96ce	2015-05-18	MAPR-18725: Errors are no longer logged in Warden logs when Warden starts the WebHCat service.
75e1d7d	2015-06-16	MAPR-19150/Hive-9613: Left join query plan no longer outputs wrong column when using subquery.
5d3d28f	2015-06-16	MAPR-19155: On secure MapR clusters, HiveServer2 uses PAM by default.
80d5220	2015-06-23	MAPR-19208: The default log level for org.apache.hadoop.hive.serde2.avro.AvroDeserializer class is now debug instead of warn.
50fd1d3	2015-06-24	Includes HIVE-9976 fix.
547ddb2	2015-06-24	MAPR-19116: Includes HIVE-9976 and HIVE-10106 fixes.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
079d8a1	2015-06-24	MAPR-13215: The hadoop.proxyuser.mapr.groups property is now able to restrict the groups that can be impersonated.
bf6e670	2015-06-30	MAPR-19152: When you run a Hive UDF from within a Pig 0.15 console, Hive no longer logs a null pointer exception when SessionState.get() returns NULL.
11f1b69	2015-07-08	MAPR-19453: When you enable hive.warehouse.subdir.inherit.perms in hive-site.xml, Hive no longer creates tables with incorrect permissions. The tables now successfully inherit the permissions of the parent directory.
f580822	2015-07-27	MAPR-19698/HIVE-10929: On Tez, dynamic partitioning queries with union all statement no longer fail at org.apache.hadoop.hive.ql.exec.MoveTask with an "Invalid partition key & values" exception.
6db44f6	2015-07-30	MAPR-18222: Hive includes PARQUET-107 fix.
4f12a86 3a3c1d1	2015-08-03	MAPR-19029: HiveServer2 now honors the MapReduce Mode configuration in warden.hs2.conf.
9e4a5cd	2015-08-03	MAPR-19778: WebHCat service no longer displays a MalformedURLException in /opt/mapr/logs/adminuiapp.log.
3a516cd	2015-08-07	MAPR-19831/Hive-9855: Hive 1.0 skewjoin no longer fails with an "Invalid source or target" exception.
6f88d6e 1379459	2015-08-25	MAPR-19894: "Insert overwrite table partition" command no longer ignores fs.permission.umask-mode when hive.optimize.insert.dest.volume is enabled.
3d984ef	2015-09-01	MAPR-20281/HIVE-9199: DummyTxnManager can now determine the lockMode required for a DDL based on the output writetype.
60b1f39	2015-09-03	MAPR-20254/HIVE-10841: Join queries no longer produce incorrect data due to the failure to push down Hive predicate.

Hive 1.0-1504 Release Notes

The notes below relate specifically to the MapR distribution for Apache Hadoop. You may also be interested in the Apache Hive 1.0 changelog or the Apache Hive homepage.

Hive Version	1.0-1504
Release Date	May 20, 2015

Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	1.0.0-mapr-1504
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

New in this Release

This is the initial release of Hive 1.0 for the MapR distribution for Hadoop. In addition to Hive 1.0 features, this release includes the following features:

- **MapR-SASL Support**

  On secure MapR clusters, MapR-SASL is supported and configured by default for HiveServer2 and HiveMetastore. Previously, Simple SASL was the default and authentication had to be configured manually.

- **Hive Scratch Directory Changes**

  By default, the Hive scratch directory is created in the same volume as the target table (hive.optimize.insert.dest.volume=true). Previously, `hive.optimize.insert.dest.volume` was set to `false`.

- **WebHCat Changes**
  - The default WebHCat log directory is now `/opt/mapr/hive/<hive-version>/logs/<username>/webhcat`. Previously, it was `/tmp/<username>/webhcat`.
  - WebHCat is now managed by Warden. The warden.hcat.conf file is installed with the mapr-hive package.

- **JDBC SASL QOP Parameter Changes**

  When you need to connect via JDBC to hiveserver2 using the SASL QOP parameter, use salsQop instead of sals.qop.

Important: The Hive 1.0 cost-based optimization is a beta feature and is not fully supported.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
3cbeb8c	2015-05-01	**MAPR-14214:** WebHCat Server is now managed by Warden.
b6899d3	2015-05-01	**MAPR-17726 (HIVE-5672):** An insert overwrite to HDFS no longer fails.
263ab62	2015-04-28	**MAPR-17791:** MapR now sets hive.exec.submit.local.task.via.child=false to enable Beeline to execute mapjoin tasks when HiveServer2 uses MapR-SASL authentication.
4060cfd	2015-04-23	**MAPR-18202:** Transaction compaction error no longer appears in Hive logs.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
2578314	2015-04-22	**MAPR-18255:** Web browsers are now able to connect to WebHCat.
db10116	2015-04-16	**MAPR-18234:** Hive no longer tries to clean up MapReduce temp directories that do not exist.
672e375	2015-04-16	**MAPR-18058:** MapReduce queries no longer fail with a java.io.IOException.
dc00c63	2015-04-13	**MAPR-18166:** Errors no longer occur when doing select * on transaction enabled ORC table.
08a82e9	2015-04-01	**MAPR-17912 (HIVE-10083):** SMBJoin no longer fails in the case where one table is uninitialized.
ddaa3f0	2015-03-31	**MAPR-17982:** MapReduce queries on external tables no longer fail with a java.io.IOException.
76839e2	2015-03-30	**MAPR-17907:** Error no longer appears in Hive logs when transactions are enabled.
09c3cf9	2015-03-26	**MAPR-17854:** Hue can now access the Hive Metastore.
44edb1c	2015-03-19	**MAPR-17790:** HiveServer2 is now able to start when it is configured to use Kerberos authentication.
a048ceb	2015-03-18	**MAPR-17742:** MapReduce jobs started by WebHCat no longer fail when Zero-config Resource Manager HA is configured.
f7b9409	2015-03-10	**MAPR-17600:** Hive no longer fails with a NullPointerException when trying to access Amazon S3 storage.
70cab74	2015-03-04	**MAPR-17526:** Warden is now able to start Hiveserver2 and Metastore when they use MapR-SASL authentication.
9d8d495	2015-02-26	**MAPR-17458:** TABLESAMPLE query no longer fails.
b5037b0	2015-02-17	**MAPR-17276:** Hive Metastore no longer fails to start on Ubuntu
dc37fb2	2015-02-13	**MAPR-17272:** By default, hive.metastore.sasl.enabled is set to true for WebHCat when the cluster is secure.
43cd431	2015-02-12	**MAPR-17135:** HiveServer2 and Hive Metastore are automatically configured to use MapR-SASL when the cluster is secure.

Commit	Date (YYYY-MM-DD)	Comment
2edf083	2015-02-06	**MAPR-17127:** The following warning messages no longer appear in the hive shell:  **Warning:** org.apache.hadoop.metrics.jvm.EventCounter is deprecated. Please use org.apache.hadoop.log.metrics.EventCounter in all the log4j.properties files.
6949cc7	2015-02-06	**MAPR-16960:** The hive shell no longer displays warning messages related to duplicate slf*.jar files.
37536f3	2015-02-06	**MAPR-17120:** WebHCat log files are now written to the following folder: `/opt/mapr/hive/hive-<version>/logs/<user>/webhcat`
2a0367c	2015-02-03	**MAPR-16962:** The status of HiveServer2 and Hive Metastore now displays correctly in the Control System when the cluster runs on Ubuntu nodes.
68461ff	2015-01-27	**MAPR-16786:** By default, MapR sets hive.optimize.insert.dest.volume=true
be34b2b	2015-01-28	**MAPR-16952:** By default, MapR sets hive.server2.thrift.sasl.qop=auth-conf
3bf4741	2015-01-26	**MAPR-16899:** HiveServer2 now has support for MapR-SASL authentication when the cluster is secure.
d21a425	2015-01-26	**MAPR-15559:** Hive Metastore now has support for MapR-SASL authentication when the cluster is secure.
9dab4a5	2015-01-19	**MAPR-16762:** WebHCat no longer fails to start on a cluster that runs YARN services.
ca51e98	2015-01-12	**MAPR-15587 (HIVE-9119):** A single zookeeper instance can now be reused and shared by ZooKeeperHiveLockManagers to reduce the number of ZooKeeper clients created for every instance of HiveServer2.

Hive 0.13.0 Release Notes

The following release notes for the Hive 0.13 component are included in the MapR distribution for Apache Hadoop:

Hive 0.13.0-1611 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 0.13.0 changelog or the Apache Hive homepage.

last-updated: Apr 11, 2022

Hive Version	0.13.0
Release Date	December 8, 2016
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/hive/tree/0.13.0-mapr-1611
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hive-0.13.201612051242-1.noarch.rpm • mapr-hive_0.13.201612051243_all.deb • mapr-hivemetastore-0.13.201612051242-1.noarch.rpm • mapr-hivemetastore_0.13.201612051243_all.deb • mapr-hiveserver2-0.13.201612051242-1.noarch.rpm • mapr-hiveserver2_0.13.201612051243_all.deb • mapr-hivewebhcat-0.13.201612051242-1.noarch.rpm • mapr-hivewebhcat_0.13.201612051243_all.deb

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
d54769e7	2016-10-05	MAPR-24696: The issue that caused Hive partition directory URI to be double-encoded is now fixed.
0ec5dcc69	2016-09-08	HIVE-14696: The issue that caused Hive query to fail with MetaException when the metastore was overloaded is now fixed.
42cd70cb	2016-09-02	HIVE-8766: The issue that caused Hive query to fail with NucleusException when the metastore was overloaded is now fixed.

Hive 0.13.0-1608 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 0.13.0 changelog or the Apache Hive homepage.

Hive Version	0.13.0

Release Date	September 1, 2016
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/hive/tree/0.13.0-mapr-1608
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hive-0.13.201608311115-1.noarch.rpm • mapr-hive_0.13.201608311115_all.deb • mapr-hivemetastore-0.13.201608311115-1.noarch.rpm • mapr-hivemetastore_0.13.201608311115_all.deb • mapr-hiveserver2-0.13.201608311115-1.noarch.rpm • mapr-hiveserver2_0.13.201608311115_all.deb • mapr-hivewebhcat-0.13.201608311115-1.noarch.rpm • mapr-hivewebhcat_0.13.201608311115_all.deb

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
e6fb977	2016-08-11	MAPR-23970: When running a query, checks are now included to determine if current filesystem is non-local before running `getScratchDir()` method, which assumes the filesystem is not local.
8583651	2016-08-05	HIVE-11617: The issue causing explain job to (sometimes) not finish or be very slow when there are many lateral views is now fixed.
1c7123f	2016-06-06	MAPR-23560: Symbolic links to the PID files to track memory utilization of HiveServer2 and HiveMetaStore processes are now available in `/opt/mapr/pid`.
3494663	2015-02-05	HIVE-7175: For connecting to HiveServer2 with LDAP authentication enabled to batch run commands, a password file can now be used instead of typing the password openly in the command line.

Hive 0.13.0-1605 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform. You may also be interested in the Apache Hive 0.13.0 changelog or the Apache Hive homepage.

last-updated: Apr 11, 2022

Hive Version	0.13.0
Release Date	June 6, 2016
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/hive/tree/0.13.0-mapr-1605
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hive-0.13.201605311455-1.noarch.rpm • mapr-hive_0.13.201605311455_all.deb • mapr-hivemetastore-0.13.201605311455-1.noarch.rpm • mapr-hivemetastore_0.13.201605311455_all.deb • mapr-hiveserver2-0.13.201605311455-1.noarch.rpm • mapr-hiveserver2_0.13.201605311455_all.deb • mapr-hivewebhcat-0.13.201605311455-1.noarch.rpm • mapr-hivewebhcat_0.13.201605311455_all.deb

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
ae79594	2016-05-17	MAPR-23302: ListBucketing tables with uppercase skew values can now be created.
1878e90	2016-04-25	MAPR-23153: Hive will no longer create scratch dir with incorrect permissions when a CTAS query is run. If CTAS query is run, now, the table root directory with correct permissions from `fs.permissions.umask-mode` will be created first and then the scratch dir with permissions from `hive.scrach.dir.permissions` will be created.
5c67d97	2016-04-14	MAPR-23082: Beeline will no longer fail with NullPointerException when MAPR-SASL is used.
cb63c6d	2016-04-14	MAPR-23084: The `datanucleus.schema.autoCreateAll` property in `hive-site.xml` file is now set to `true` by default.

Commit	Date (YYYY-MM-DD)	Comment
63ce07f	2016-04-01	MAPR-22719: The issue causing Hive 0.13 to incorrectly determine the authentication mechanism when installed on MapR 4.0.1 has been fixed. Regression of MAPR-22677: Backported HIVE-8320 to fix the Hive 0.13 and Hue integration error.
1d2e15c	2016-01-29	MAPR-22222: The delay as a result of a single reduce task in the last stage is now fixed with a new parameter, `hive.groupby.limit.extrastep`, which can be used to enable or disable a new MR job for sorting the final output.
91b0053	2016-03-29	MAPR-22943: Pig will no longer try to receive Hive Metastore's functions and only receive standard functions, which are registered with FunctionRegistry class.
7d914fa	2015-08-19	HIVE-11592: Backported HIVE-11592 to set an upperbound on the protobuf message size.

Hive 0.13.0-1603 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform.

Version	0.13
Release Date	April 2016
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/hive/tree/0.13.0-mapr-1603
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release:

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
c97788c	2016-03-30	MAPR-22966: Hiveserver2 no longer fails to start when it uses LDAP authentication on a secure cluster.

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comment
69487f5	2016-03-28	MAPR-22945: HiveServer2 no longer fails to start when PAM authentication is enabled.
11efd3f	2016-03-25	MAPR-22928: The `datanucleus.schema.autoCreateAll` property is now set to `false` by default.
145d027	2016-02-19	MAPR-22677: Backported HIVE-8320 and HIVE-4625 so that Hcatalog jobs on secure clusters no longer fail due to the following reasons: • HiveServer2 tries to get delegation token from Hive Metastore even if the Hive Metastore is being used in embedded mode • Hiveserver2 session is called multiple times
9bfae6f	2016-02-25	MAPR-22719: The HiveServer2 authentication mechanisms is no longer determined incorrectly on MapR 4.0.1.
1df2fa6	2016-03-02	MAPR-22701: Backported HIVE-12875 to reduce the chance of an authorization vulnerability.

Hive 0.13.0-1602 Release Notes

Below are release notes for the Hive component included in the MapR Converged Data Platform.

Version	0.13
Release Date	March 4, 2016
MapR Version Interoperability	See Hive and HCatalog Support Matrix on page 5631 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/hive/tree/0.13.0-mapr-1602
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hive-0.13.201603021517-1.noarch.rpm • mapr-hive_0.13.201603021517_all.deb • mapr-hivemetastore-0.13.201603021517-1.noarch.rpm • mapr-hivemetastore_0.13.201603021517_all.deb • mapr-hiveserver2-0.13.201603021517-1.noarch.rpm • mapr-hiveserver2_0.13.201603021517_all.deb • mapr-hivewebhcat-0.13.201603021517-1.noarch.rpm

	• mapr-hivewebhcat_0.13.201603021517_all.deb

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
d6cf120115	2016-02-11	MAPR-22560/HIVE-10021: "Alter index rebuild" statements no longer fail when HiveServer2 uses Sentry authorization.
6f17d2836	2016-02-09	MAPR-22435: `beeline.sh` no longer includes an unnecessary "if" statement.

Hive 0.13.0-1601 Release Notes

Below are release notes for the Hive component included in the MapR Distribution for Apache Hadoop.

Version	0.13
Release Date	February 1, 2016
MapR Version Interoperability	See Hive and HCatalog Support Matrix and Ecosystem Support Matrix (Pre-5.2 Releases)
Source on GitHub	https://github.com/mapr/hive/tree/0.13.0-mapr-1601
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hive-0.13.201601281525-1.noarch.rpm • mapr-hive_0.13.201601281525_all.deb • mapr-hivemetastore-0.13.201601281525-1.noarch.rpm • mapr-hivemetastore_0.13.201601281525_all.deb • mapr-hiveserver2-0.13.201601281525-1.noarch.rpm • mapr-hiveserver2_0.13.201601281525_all.deb • mapr-hivewebhcat-0.13.201601281525-1.noarch.rpm • mapr-hivewebhcat_0.13.201601281525_all.deb

New in this Release

This release of Apache Hive includes the following behavior changes that are specific to MapR:

Drill can now read parquet text data.

last-updated: Apr 11, 2022

DataNucleus versions were updated.

The following DataNucleus versions were updated:

Component	Old Version	New Version
datanucleus-rdbms	3.2.9	4.1.7
datanucleus-core	3.2.1.0	4.1.6
datanucleus-api-jdo	3.2.6	4.2.1

DataNucleus properties were renamed.

The following DataNucleus properties were renamed:

Old Name	New Name
datanucleus.validateTables	datanucleus.schema.validateTable
datanucleus.validateColumns	datanucleus.schema.validateColu
datanucleus.validateConstraints	datanucleus.schema.validateCons
datanucleus.autoCreateSchema	datanucleus.schema.autoCreateAl

DataNucleus Property datanucleus.fixedDatastore was deleted.

For details on the features available in the open source version of this component, see the Apache Hive 0.13.0 changelog or the Apache Hive homepage.

Feature Support

MapR does not support Hive on Spark. Therefore, you cannot use Spark as an execution engine for Hive. However, you can run Hive and Spark on the same cluster. You can also use Spark SQL and Drill to query Hive tables.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
b9e8d66	2015-10-24	MAPR-21474: Backported HIVE-7735 which implements Char and Varchar in ParquetSerDe.
3b9b65f	2015-10-24	MAPR-21474: Backported HIVE-8205 so that using strings in the group type no longer causes failures in ParquetSerDe.
a0f183f	2015-11-13	MAPR-21228: Hive queries no longer fail with exception: Table 'hive.DELETEME1xx' doesn't exist
a793ea1	2015-11-28	MAPR-21192: Backported HIVE-6847 so that Hive creates user-specific scratch directories.
0b1b177	2015-12-03	MAPR-20263: Hive Metastore no longer incorrectly determines the authentication method when Hive jobs are created by other components such as Sqoop, Oozie, or Spark.
5a54172	2016-01-11	MAPR-22100: Resolved Spark build issued which was caused by the backporting of MAPR-19408.

Hive 0.13.0-1510 Release Notes

Below are release notes for the Hive component included in the MapR Distribution for Apache Hadoop.

Version	0.13
Release Date	November 20, 2015
MapR Version Interoperability	See the Hive and HCatalog Support Matrix and the Ecosystem Support Matrix
Source on GitHub	https://github.com/mapr/hive/tree/0.13.0-mapr-1510
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hive-0.13.201511180922-1.noarch.rpm • mapr-hive_0.13.201511180922_all.deb • mapr-hivemetastore-0.13.201511180922-1.noarch.rpm • mapr-hivemetastore_0.13.201511180922_all.deb • mapr-hiveserver2-0.13.201511180922-1.noarch.rpm • mapr-hiveserver2_0.13.201511180922_all.deb • mapr-hivewebhcat-0.13.201511180922-1.noarch.rpm • mapr-hivewebhcat_0.13.201511180922_all.deb

New in this Release

This release of Apache Hive includes the following behavior changes that are specific to MapR:

Additional Properties	The following Hive properties were backported into this release: • hive.metastore.schema.verification.record.version • hive.server2.thrift.http.max.idle.time • hive.server2.thrift.http.worker.keepalive.time • hive.server2.session.check.interval • hive.server2.idle.session.timeout • hive.exec.orc.default.block.size You can override the default values by configuring different values for these properties in the hive-site.xml file. For details on these properties, see the Apache Hive Documentation.
Additional orcfiledump Options	The orcfiledump utility includes the option to print the timezone in ORC metadata files and to dump the content of the ORC files.
Flexible HiveServer2 Authentication Support	When the cluster is secure, HiveServer2 accepts both MAPR-SASL and PAM for in-bound authentication. This allows a single instance of HiveServer2 to accept both MapR-SASL authentication from Hue and PAM authentication from JDBC/ODBC connections.

last-updated: Apr 11, 2022

Beeline can Submit Multiple Queries at a Time	Beeline can submit multiple queries at a time when you use a semi-colon(;) in-between queries. For details on the features available in the open source version of this component, see the Apache Hive 0.13.0 changelog or the Apache Hive homepage .

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
70e1c5f	2015-10-16	MAPR-19370: On a secure cluster, HiveServer2 now accepts both MapR-SASL and PAM authentication.
88b368f	2015-10-07	MAPR-20814: Backported HIVE-8746 to enable ORC timestamp columns to account for daylight savings time changes.
8ab8f06	2015-10-03	MAPR-20866: Queries on a subset of the columns in a parquet schema no longer return NULL column values.
4f7f3e3	2015-09-25	MAPR-19408: Backported HIVE-7353 so that HiveServer2 no longer has JDOPersistanceManager leaks when HiveServer2 uses the embedded MetaStore.
40c2a4e	2015-09-24	MAPR-20650: Backported HIVE-9749 to provide the option to configure whether or not the Hive Metastore version should be updated automatically.
0e3c9c4	2015-09-21	MAPR-20191: On the MapR 5.x sandbox, the Hive classpath no longer contains newlines.
c58f426	2015-09-17	MAPR-20297: Backported HIVE-9877 to allow Beeline to submit multiple queries at a time.

Hive 0.13.0-1508 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Hive 0.13.0 changelog or the Apache Hive homepage.

Version	0.13.0
Release Date	Sept 22, 2015
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	0.13.0-mapr-1508
Maven Artifacts	https://repository.mapr.com/maven/

last-updated: Apr 11, 2022

New in this Release

This MapR release of Hive 0.13 includes the following behavior changes:

- **HiveServer2 Authentication Changes** (Available for MapR cluster version 4.1 and above)

  On secure MapR clusters, MapR-SASL is not longer the default for HiveServer2. Instead, HiveServer2 uses PAM by default.

- **WebHCat Changes**

  The warden.hcat.conf file is no longer installed with mapr-hive package. Instead, it is installed with the mapr-hivewebhcat package.

Fixes

Commit	Date (YYYY-MM-DD)	Comment
4130e1e	2015-05-18	MAPR-18725: Errors are no longer logged in Warden logs when Warden starts webhcat.
28046ce	2015-06-17	MAPR-19155: On secure MapR clusters, HiveServer2 uses PAM by default.
fef6a67	2015-06-23	MAPR-19208: The default log level for org.apache.hadoop.hive.serde2.avro.AvroDeserializer class is now debug instead of warn.
1240cce bf16b43	2015-06-25	MAPR-19029: HiveServer2 now honors the MapReduce Mode configured in warden.hs2.conf.
1fe17a2	2015-07-30	MAPR-18222: Hive includes PARQUET-107 fix.
20ad928	2015-08-03	MAPR-19778: WebHCat service no longer displays a MalformedURLException in `/opt/mapr/logs/adminuiapp.log`.
65a7e10	2015-09-2-15	MAPR-20281/ HIVE-9199: DummyTxnManager can now determine the lockMode required for a DDL based on the output writetype.
979f420	2015-09-03	MAPR-20254/HIVE-10841: Join queries no longer produce incorrect data due to the failure to push down Hive predicate.
39ec9bd	2015-09-08	MAPR-20423: HiveQL queries with LPAD function no longer fail with the ClassCastException.
462acf4	2015-09-10	MAPR-20523 and MAPR-20329: Queries that use any type of join between avro tables no longer fail or give incorrect results.

Hive 0.13.0-1504 Release Notes

The notes below relate specifically to the MapR distribution for Apache Hadoop. You may also be interested in the Apache Hive 0.13.0 changelog or the Apache Hive homepage.

last-updated: Apr 11, 2022

Version	Hive 0.13.0-1504
Release Date	May 20, 2015
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	0.13.0-mapr-1504
Maven Artifacts	https://repository.mapr.com/maven/

Using Hive with Sentry and Impala

Hive 0.13 (1410 and later) works with Sentry 1.4 and Impala 1.4.1.

Using Hive with HBase

Hive 0.13 works with HBase 0.98.x.

New in this Release

- **MapR-SASL Support** (Available for MapR cluster version 4.1 and above)

  On secure MapR clusters, MapR-SASL is supported and configured by default for HiveServer2 and Hive Metastore.

- **WebHCat Changes**
  - The default WebHCat log directory is now /opt/mapr/hive/<hive-version>/logs/<username>/webhcat. Previously, it was /tmp/<username>/webhcat.
  - WebHCat is now managed by Warden. The warden.hcat.conf file is installed with the mapr-hive package.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
f562db3	2015-01-19	**MAPR-16762:** WebHCat no longer fails to start on a cluster that runs YARN services.
b246af9	2015-01-27	**MAPR-16786:** By default, MapR sets `hive.optimize.insert.dest.volume=true`.
c226a21	2015-01-22	**MAPR-16854:** WebHCat no longer returns a NoSuchMethodsError on startup.
1f583b0	2015-01-22	**MAPR-15559:** Hive Metastore now has support for MapR-SASL authentication when the cluster is secure.
6c37e83	2015-01-26	**MAPR-16899:** HiveServer2 now has support for MapR-SASL authentication when the cluster is secure.

Commit	Date (YYYY-MM-DD)	Comment
d617f6e	2015-01-28	**MAPR-16952:** By default, MapR sets hive.server2.thrift.sasl.qop=auth-conf.
0a8e267	2015-02-03	**MAPR-16962:** On Ubuntu systems, the service status for hiveserver2 and hivemetastore is now displayed correctly in the Control System.
c06f014	2015-02-06	**MAPR-16960:** The hive shell no longer displays warning messages related to duplicate slf*.jar files.
514f307	2015-02-06	**MAPR-17120:** WebHCat log files are now written to the following folder: `/opt/mapr/hive/hive-<version>/logs/<user>/webhcat`
499d865	2015-02-06	**MAPR-17127:** The following warning messages no longer appear in the hive shell: `WARNING: org.apache.hadoop.metrics.jvm.EventCounter is deprecated. Please use org.apache.hadoop.log.metrics.EventCounter in all the log4j.properties files.`
f8370ba	2015-02-11	**MAPR-17135:** HiveServer2 and Hive Metastore are automatically configured to use MapR-SASL when the cluster is secure.
608c855	2015-02-13	**MAPR-17272:** By default, hive.metastore.sasl.enabled is set to true for WebHCat when the cluster is secure.
d1d2300	2015-02-17	**MAPR-17276:** Hive Metastore no longer fails to start on Ubuntu.
5635c82	2015-02-26	**MAPR-17458:** A TABLESAMPLE query no longer fails.
b6941af	2015-03-04	**MAPR-17526:** Warden is now able to start HiveServer2 and Metastore when they use MapR-SASL authentication.
f7b9409	2015-03-10	**MAPR-17600:** Hive no longer fails with a NullPointerException when trying to access Amazon S3 storage.
dee7431	2015-03-13	**MAPR-17712:** Hive now returns a filtered database list when Sentry is enabled.
f5bf538	2015-03-19	**MAPR-17783:** Hive mapjoin queries no longer fail with a java.io.FileNotFoundException.

Commit	Date (YYYY-MM-DD)	Comment
9383589	2015-04-10	**MAPR-16186 (HIVE-8297):** Hive no longer returns wrong results for TIMESTAMP columns in RCFile format.
8f52355	2015-05-01	**MAPR-17726 (HIVE-5672):** An insert overwrite to HDFS no longer fails.
3cbeb8c	2015-05-01	**MAPR-14214:** WebHCat Server is now managed by Warden.

Hive 0.13.0-1501 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Hive 0.13.0 changelog or the Apache Hive homepage.

Hive Version	0.13.0-1501
Release Date	January 21, 2015
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	0.13-mapr-1501
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

Using Hive with Sentry and Impala

Hive 0.13 (1410 and 1501) works with Sentry 1.4 and Impala 1.4.1.

Using Hive with HBase

Hive 0.13 works with HBase 0.98.x.

New in this Release

The default log directory for Hivesever2 has changed from `/tmp/<user>` to `/opt/mapr/hive/hive-<version>/logs/<user>`. Users will not be impacted by this change if the log directory is set in `/opt/mapr/hive.hive-<version>/conf/hive-log4j.properties`.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
491c048	2015-01-19	MAPR-16784: Secure Hive metastore no longer fails to start on a MapR 4.x cluster that runs MapReduce v1.
83e5b3b	2015-01-12	MAPR-15587/Hive-9119: Reduced the number of ZooKeeper clients created per running instance of HiveServer2.
6cf6531	2015-01-12	MAPR-16706: Hive beeline no longer hangs on a secure MapR 3.1.0 cluster.

Commit	Date (YYYY-MM-DD)	Comment
35ffc63	2014-12-22	MAPR-16535: TypeMismatchException no longer appears when calling Hadoop23Shims.createJobContext(Configuration conf, JobId jobId) API.
c99933a	2014-12-09	MAPR-15982: Hive service status check requires less CPU.
37f75d4	2014-12-14	MAPR-16184: On MapR 4.0.1, Hive statements using relative path URI to the filesystem no longer fail due to incorrect URI creation.
313f016	2014-11-21	MAPR-14270/HIVE-7279: format_number now supports the decimal datatype.
608c461	2014-11-20	MAPR-16090: hcat CLI no longer fails on a cluster that runs YARN.
4dff42b	2014-11-20	MAPR-15833: The hive metatool updateLocation command no longer fails.

Hive 0.13.0-1410 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Hive 0.13.0 changelog or the Apache Hive homepage.

Hive Version	0.13.0-1410
Release Date	November 19, 2014
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	hive-0.13-mapr-1410
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

Using Hive with Sentry and Impala

Hive 0.13 (1410 and 1501) works with Sentry 1.4 and Impala 1.4.1.

Using Hive with HBase

Hive 0.13 works with HBase 0.98.x.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
bd18b73	2014-09-22	MAPR-15371: Partitions of type date behaved incorrectly with daylight savings time.

Commit	Date (YYYY-MM-DD)	Comment
5d96f2e	2014-10-02	MAPR-15481: When data was imported in Parquet format, Hive returned a null-pointer exception.
1485a0d	2014-10-24	MAPR-15696: Changes were made to support Sentry for Hive 13.
f1fe7ea	11-03-2014	MAPR-15816: HCatStorer failed to store data in a Hive table on a non-secure 4.0.1 unsecure running MRv1.
88a2938	11-03-2014	MAPR-15467: HiveServer2 failed to execute a join query when Kerberos authentication was enabled.

Hive 0.13.0-1409 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Hive 0.13.0 changelog or the Apache Hive homepage.

Hive Version	0.13
Release Date	Sept 30, 2014
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	0.13-mapr-1409
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
9c91b34	2014-09-13	MAPR-15246: Hive metastore starts in secure mode with Kerberos enabled.
7735233	2014-09-19	MAPR-15246: Hive metastore starts in secure mode with Kerberos enabled.
fac5bab	2014-09-19	MAPR-15193: Fixed Hive 0.13.0 runtime error.
bd18b73	2014-09-22	MAPR-15371: Partitions of type date handle daylight savings time correctly.
5e2334d	2014-09-22	MAPR-15393: Hive metastore picks up the correct principal name.

Hive 0.13.0-1408 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Hive 0.13.0 changelog or the Apache Hive homepage.

Hive Version	1.0

Release Date	Sept 22, 2015
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	1.0.0-mapr-1508
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
9daf945	4-Sept-2014	MAPR-15135: Created shims to handle hive queries in a mixed-mode cluster.
d798be2	2-Sept-2014	MAPR-14901: Change default port of HiveServer2 in warden.hs2.conf

Hive 0.13.0-1406 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Hive 0.13.0 changelog or the Apache Hive homepage.

Version	Hive 0.13.0-1406
Release Date	June 30, 2014
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	hive-0.13-mapr-1406
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
cf24ff5	23-Apr-2014	[HIVE-7009] Adds support for non-HDFS file systems for Hive+Tez jobs.

Hive 0.13.0-1405 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Hive 0.13.0 changelog or the Apache Hive homepage.

Version	Hive 0.13.0-1406
Release Date	June 6, 2014
Source on GitHub	https://github.com/mapr/hive
GitHub Release Tag	hive-0.13-mapr-1405

MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

New In This Release

This is the initial release of Hive 0.13 for the MapR distribution for Hadoop.

Known Issues

MapR Issue	Description
14397	Impala only runs with Hive 0.12. Earlier versions of Hive and Hive 0.13 are not supported with Impala.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

MapR 10677. Hive task temp files are now written to the task's spill directory in a MapRFS local volume instead of the local filesystem. To write temp files to the local filesystem, change the value of the hive.exec.tmp.maprfsvolume attribute in the hive-site.xml file to false.

Commit	Date (YYYY-MM-DD)	Comment
09529ed	14-May-2014	MapR 12909. Resolves scratchdir user permissions issues related to impersonation when starting HiveServer2 for the first time. When impersonation is enabled, HiveServer2 sets the permissions on the root scratch directory (by default, `/user/mapr-user/tmp/hive`) to 777. These permissions enable the creation of query-specific scratch subdirectories for impersonated users.
cbc8a7b	16-May-2014	[HIVE-6245] and MapR 13847. Resolves impersonation issues with remote MetaStore in HiveServer 2 when `hive.metastore.execute.setugi` property is set to true.
3e78db6	16-May-2014	MapR 13477. The LD_LIBRARY_PATH variable now includes the path to libjpam.so. Users no longer need to download and configure `libjpam.so` during HiveServer2 configuration.
7973de9	22-May-2014	MapR 13872. Improved support for Beeline with Hive 13 on MapR.
cf24ff5	23-Apr-2014	[HIVE-4133] Fixes a ClassNotFoundException error in the EventCounter.
a2f9ecd	23-Apr-2014	[HIVE-4969] and MapR 10948. The HCatLoader class now returns complete table information.

Commit	Date (YYYY-MM-DD)	Comment
5a02b20	23-Apr-2014	[HIVE-5631] and MapR 11291. Skewed tables can now have indexes created.
119d60c	23-Apr-2014	
fef36c1	24-Apr-2014	[HIVE-3844] and MapR 9663. Hive TIMESTAMP column type now supports required timestamp formats.
0efe07d	24-Apr-2014	MapR 8765. Hive now correctly computes the JobTracker location in case of failure.
be1996b	24-Apr-2014	MapR 11554. ASTNodeOrigin objects serialize and deserialize correclty.
b1d9ac7	24-Apr-2014	MapR 10677. Hive task temp files are now written to the task's spill directory in a MapRFS local volume instead of the local filesystem. To write temp files to the local filesystem, change the value of the hive.exec.tmp.maprfsvolume attribute in the `hive-site.xml` file to `false`.
27e9a32	25-Apr-2014	MapR 11012. Hive now uses scratch directories that are subdirectories of the target table's home directory, eliminating cross-volume copies and improving performance. This optimization is disabled by default. For details on how to enable the option, see Hive.

5bace77	25-Apr-2014	MapR 12673. When the cluster is secure, HiveServer2 has PAM authentication enabled with 'login' profile by default. The default configuration is equivalent to the following property values:
		`hive.server2.authentication=CUSTOM`
		`hive.server2.authentication.pam.profiles=login`
		`hive.server2.custom.authentication.class=org.apache.hive.service.auth.PamAuthenticationProvider`

a682974	01-May-2014	[HIVE-5677]. Fixes a condtion where Hive services generate errors when HIVE_OPTS is set.
5034b63	01-May-2014	[HIVE-4776] Adds an option to Hive to skip the first n rows of a file.
4b02a6a	01-May-2014	MapR 13010. Enhances support for Hive on YARN.

Commit	Date (YYYY-MM-DD)	Comment
23892ab	01-May-2014	MapR 12684. The /logs and /pids subdirectories are created when HiveServer 2 or Hive Metastore are launched if they do not already exist.
7d99556	05-May-2014	[HIVE-4629] Adds a GetLog() API to retrieve query logs.
54d8193	07-May-2014	[HIVE-6893] Resolves a concurrency error in the HiveMetastore server.
ff41cbf	07-May-2014	MapR 13024. Corrects a condition where upgrading the Hive metastore would incorrectly report the upgrade as failed due to a mismatch in the version string in the JAR file.

HttpFS Release Notes

The release notes for HttpFS included in the MapR Converged Data Platform contains notes specific to MapR only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

HttpFS 1.1.0.200 - 2201 (EEP 8.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hadoop.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.1.0.200
Release Date	January 2022
Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/httpfs/
GitHub Release Tag	1.1.0.200-eep-810
Maven Artifacts	http://repository.mapr.com/maven/
Package Names	Navigate tohttps://package.mapr.hpe.com/releases/MEP/ and select your MEP (EEP) and OS to view the list of package names

New in This Release

HttpFS 1.1.0.200-eep-810 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

- Adds FIPS support.
- Upgrades commons-io.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
e7946c91	2021-12-15	HTTPFS-96: log4j updated to 1.3.0-mapr due vulnerability CVE-2019-17571, CVE-2021-4104
34c838ff	2021-12-01	HTTPFS-94: Updated jdom-1.1.jar due vulnerability CVE-2021-33813
34312110	2021-11-25	Added execute permission to configuration script
4234dba2	2021-11-19	HTTPFS-93: commons-io-2.4.jar vulnerability CVE-2021-29425
e009fb22	2021-11-18	HTTPFS-92: Updated Hadoop and Jetty version to the latest
02824dda	2021-10-06	HTTPFS-87 - Fix unit tests
36235297	2021-09-13	HTTPFS-82: HttpFS can't load SSL config and start with NPE
13709650	2021-09-09	Httpfs 73: Add FIPS support

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

None

Resolved Issues

None

HttpFS 1.1.0.100 - 2110 (EEP 8.0.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hadoop.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.1.0.100
Release Date	October 2021
Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/httpfs/
GitHub Release Tag	1.1.0.100-eep-800
Maven Artifacts	http://repository.mapr.com/maven/
Package Names	Navigate tohttps://package.mapr.hpe.com/releases/MEP/ and select your MEP (EEP) and OS to view the list of package names

New in This Release

HttpFS 1.1.0.100-eep-800 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

- Adds XAttrs support (for details, see https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/WebHDFS.html#Extended_Attributes.28XAttrs.29_Operations)

last-updated: Apr 11, 2022

- Updates Jetty to 9.4.43.v20210629
- Updates Jackson v1 and v2 depenencies

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
49f1d8ab	2021-04-19	HTTPFS-68 - Fixed problem with starting httpfs on SLES
83f9eb5f	2021-04-29	Backport HDFS-6430 HTTPFS - Implement XAttr support (Yi Liu via tucu)
e371c077	2021-05-20	HTTPFS-69 fix bug with incremental install
eef38d0d	2021-07-12	Httpfs 70: Fixed unit tests
9e3db8fb	2021-07-28	HTTPFS-75: Update Jetty to 9.4.43.v20210629
e0dd2c6e	2021-08-03	HTTPFS-76: Remove sudo usage in HttpFS
e920552c	2021-08-17	HTTPFS-78: Update Jackson v1 and v2 depenencies
c1e7bcda	2021-09-02	HTTPFS-79: HttpFS can't find credential.provider.path and read encrypted password

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

None

Resolved Issues

None

HttpFS 1.1.0.0 - 2104 (EEP 7.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hadoop.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.1.0.0
Release Date	April 2021
HPE Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/httpfs/
GitHub Release Tag	1.1.0.0-mapr-710
Maven Artifacts	http://repository.mapr.com/maven/

Package Names	Navigate tohttps://package.mapr.hpe.com/releases/MEP/ and select your MEP and OS to view the list of package names

New in This Release

HttpFS 1.1.0.0-mapr-710 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- Added credential provider support.
- Moved from Tomcat to Jetty.
- Service verifier

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
46deff33	2021-04-13	HTTPFS-67: Jetty updated to 9.4.39.v20210325 due CVE-2021-28165
eb3a0da1	2021-03-22	HTTPFS-65/66: Updated Jetty to 9.4.38.v20210224 and Hadoop to 2.7.5.0-mapr-710
7f870717	2021-03-22	HTTPFS-62: Add service verifier to HTTPFS package
78da3bdd	2021-03-12	Backported HDFS-6849. Replace HttpFS custom proxyuser handling with common implementation (tucu)
d58d6210	2021-02-24	HTTPFS-60: Fixed problem with creating file in HDFS.
6b39a262	2021-02-18	HTTPFS-59: Enable ssl by default for secure cluster
992be133	2021-02-09	HTTPFS-57: Upgrade jersey to the version 1.19
bad9ae7d	2021-01-20	HTTPFS-47: Moving HTTPFS to 4-digit artifacts version
ff3e5cfa	2020-10-12	HTTPFS-46: Delete mapr-security-web as it not need any more
d1b7727e	2020-10-11	HTTPFS-46: Update Hadoop dependency to MEP-7.1.0
79836a55 9a66beb8	2020-10-01	HTTPFS-37: Update Jetty to 9.4.28.v20200408 version and change symbol links for ssl-client.xml and ssl-server.xml
8009cddf	2020-10-01	HTTPFS-38: Move MAPR configuration from Tomcat to Jetty
c664fdd3	2020-10-01	HTTPFS-38: Move ssl security configuration from Tomcat to Jetty

last-updated: Apr 11, 2022

341dfe38	2020-10-01	HTTPFS-37: Fix logging configuration
7cf88816	2020-10-01	HTTPFS-38: Add support Mapr PAM Authentication
bc7a07dd	2020-10-01	HTTPFS-37: Fix java.lang.NoClassDefFoundError: javax/servlet/http/ HttpSessionIdListener
ab34685b	2020-10-01	HTTPFS-37: Write new script for running httpfs, delete tomcat directory
7d232d39	2020-10-01	MAPRFS-37: Fix bugs from backport HDFS-10860. Create a copy of the some classes from hadoop-common version 3.0.0
59fb2bcb	2020-10-01	Backported HDFS-10860. Switch HttpFS from Tomcat to Jetty. Contributed
951d7d32	2020-10-01	Backported HADOOP-10075. Update jetty dependency to version 9 (rkanter)
72037d12	2020-10-01	Backported HADOOP-10771. Refactor HTTP delegation support out of httpfs to common, PART 2. (tucu)
f04a9495	2020-10-01	Backported HDFS-3113 amendment, removing incorrectly committed files
dba6fff8	2020-10-01	Backported HDFS-3113. httpfs does not support delegation tokens. (tucu)

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- HTTPFS-69: If HttpFS fails to start after incremental installation, set the `httpfs.ssl.enabled` property to `false` at `httpfs-site.xml` configuration file and restart the HttpFS service.

Resolved Issues

- None.

HttpFS 1.0 - 2101 (EEP 7.0.1) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hadoop.

Version	1.0
Release Date	January 2021
HPE Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/httpfs/
Maven Artifacts	http://repository.mapr.com/maven/
Package Names	Navigate tohttps://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

HttpFS 1.0 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- Tomcat v9.0.39

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
b383888	2020-11-12	HTTPFS-49: upgrade tomcat version to 9.0.39

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

HttpFS 1.0 - 2009 (EEP 7.0.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for HttpFS 1.0 - 2009 for EEP 7.0.0.

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.0
Release Date	September 2020
MapR Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/httpfs/
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
99ac817	2019-12-23	HTTPFS-27: Add security headers for HTTPFS
a56fdae	2020-03-13	HTTPFS-28: No TLS protocol logging for httpfs

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
33dfdcb	2020-03-31	HTTPFS-31: Backported HDFS-6404 [HttpFS should use a 000 umask for mkdir and create operations
efda36b	2020-06-16	HTTPFS-36: Support jdk 11
b8621d7	2020-07-01	MAPRHADOOP-83: Updated tomcat version to 9.0.36
5e3d1ad	2020-07-01	HTTPFS-39: "org.apache.log4j.ConsoleAppender" object is not assignable to a "org.apache.log4j.Appender" variable
a2742d0	2020-07-31	HTTPFS-40: Update SSL password from ssl-client.xnl

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

HttpFS 1.0 - 2201 (EEP 6.3.6) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hadoop.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.0
Release Date	January 2022
HPE Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/httpfs/
GitHub Release Tag	1.0-mapr-2201
Maven Artifacts	http://repository.mapr.com/maven/
Package Names	Navigate tohttps://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

HttpFS 1.0 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

- Bug fixes and updates.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment

f98d94ea	2022-01-25	HTTPFS-97: Updated log4j v1 to the 1.3.1-mapr
897a38bf	2021-12-15	HTTPFS-96: log4j updated to 1.3.0-mapr due vulnerability CVE-2019-17571, CVE-2021-4104
581be28e	2021-11-25	Added execute permission to configuration script
342b993d	2021-10-25	HTTPFS-89: upgrade Tomcat version to 9.0.54

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

HttpFS 1.0 - 2104 (EEP 6.3.4) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hadoop.

Version	1.0
Release Date	April 2021
HPE Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/httpfs/
Maven Artifacts	http://repository.mapr.com/maven/
Package Names	Navigate tohttps://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

HttpFS 1.0 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- Bug fixes and updates.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
890de403	2021-02-09	HTTPFS-57: upgrate jersey to the version 1.19
5feefda8	2021-02-08	HTTPFS-56: update tomcat to 9.0.43 version

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

HttpFS 1.0 - 2101 (EEP 6.3.2) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hadoop.

Version	1.0
Release Date	January 2021
HPE Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/httpfs/
Maven Artifacts	http://repository.mapr.com/maven/
Package Names	Navigate tohttps://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

HttpFS 1.0 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- Tomcat v9.0.39

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
963eca1	2020-11-12	HTTPFS-49: upgrade tomcat version to 9.0.39

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

HttpFS 1.0 - 2101 (EEP 5.0.6) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hadoop.

Version	1.0
Release Date	January 2021
MapR Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/httpfs/
Maven Artifacts	http://repository.mapr.com/maven/
Package Names	Navigate tohttps://package.mapr.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

HttpFS 1.0 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- Tomcat v9.0.39

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
963eca1	2020-11-12	HTTPFS-49: upgrade tomcat version to 9.0.39

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None

.

Resolved Issues

- None

.

HttpFS 1.0 - 2009 (EEP 6.3.1 and EEP 5.0.5) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for HttpFS 1.0 - 2009 for EEP 6.3.1 and EEP 5.0.5.

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.0
Release Date	September 2020
MapR Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/httpfs/
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Tomcat v9.0.36

Fixes

This MapR release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
99ac817	2019-12-23	HTTPFS-27: Add security headers for HTTPFS
a56fdae	2020-03-13	HTTPFS-28: No TLS protocol logging for httpfs
33dfdcb	2020-03-31	HTTPFS-31: Backported HDFS-6404 [HttpFS should use a 000 umask for mkdir and create operations
efda36b	2020-06-16	HTTPFS-36: Support jdk 11
b8621d7	2020-07-01	MAPRHADOOP-83: Updated tomcat version to 9.0.36
5e3d1ad	2020-07-01	HTTPFS-39: "org.apache.log4j.ConsoleAppender" object is not assignable to a "org.apache.log4j.Appender" variable

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

HttpFS-1.0-1904 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.0
Release Date	April 2019
MapR Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/httpfs/
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Tomcat is updated to v7.0.92.

Fixes

This MapR release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
d39f31a	2019-02-21	HTTPFS-22: Updated Tomcat version to 7.0.92

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

HttpFS-1.0-1901 (EEP 6.1.0) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.0
Release Date	February 2019
MapR Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/httpfs/
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

- None.

Known Issues and Limitations

- None.

Resolved Issues

- None.

HttpFS-1.0-1808 (EEP 6.0.0) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.0

Release Date	September 2018
MapR Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/httpfs/
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Upgraded Tomcat version from 6.0.32 to 7.0.82.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
4fe5706	2018-02-13	[MAPR-30661] HttpFS used vulnerable tomcat
c1645f6	2018-01-23	MAPR-25377-REOPEN warden.httpfs.conf should not have double quotes for service.env
8defb23	2018-01-22	MAPR-30557 Add support for arbitrary content type to WebHDFS
ee6641b	2018-01-20	MAPR-30449 configure.sh -R failed with HttpFS component
0697ffc	2018-01-20	MAPR-26377 warden.httpfs.conf should not have double quotes for service.env
2190de6	2017-10-28	MAPR-HTTPFS-11 HttpFS still configured on SSL after configure.sh --unsecure
029d8e2	2017-10-26	MAPR-HTTPFS-12 Update JSON lib at HttpFS
bf7c40a	2017-10-22	MAPR-HTTPFS-9 HttpFS should be restarted durring any configure.sh -R run
7fb9d19	2017-10-20	MAPR-HTTPFS-6 HttpFS Signature file is default and world readable
1ddaa89	2017-10-20	MAPR-29745-HttpFS configure.sh -R when HttpFS service added generates an error but does enable service
7a38196	2017-10-20	MAPR-HTTPFS-7 HttpFS is not using https by default
ca608a8	2017-10-18	MAPR-HTTPFS-4 Upload restart script into restart file
9695d4a	2017-10-12	MON-2236 HttpFS link in service page does not work

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
b4ceb9e	2017-10-11	MAPR-HTTPFS-4 Implement customSecure flag

Known Issues and Limitations

- None

Resolved Issues

- None

HttpFS-1.0-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	1.0
Release Date	March 2018
MapR Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	N/A
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Upgraded Tomcat version from 6.0.32 to 7.0.82.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
4fe5706	2018-02-13	[MAPR-30661] HttpFS used vulnerable tomcat
c1645f6	2018-01-23	MAPR-25377-REOPEN warden.httpfs.conf should not have double quotes for service.env
8defb23	2018-01-22	MAPR-30557 Add support for arbitrary content type to WebHDFS
ee6641b	2018-01-20	MAPR-30449 configure.sh -R failed with HttpFS component
0697ffc	2018-01-20	MAPR-26377 warden.httpfs.conf should not have double quotes for service.env
2190de6	2017-10-28	MAPR-HTTPFS-11 HttpFS still configured on SSL after configure.sh --unsecure
029d8e2	2017-10-26	MAPR-HTTPFS-12 Update JSON lib at HttpFS

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
bf7c40a	2017-10-22	MAPR-HTTPFS-9 HttpFS should be restarted durring any configure.sh -R run
7fb9d19	2017-10-20	MAPR-HTTPFS-6 HttpFS Signature file is default and world readable
1ddaa89	2017-10-20	MAPR-29745-HttpFS configure.sh -R when HttpFS service added generates an error but does enable service
7a38196	2017-10-20	MAPR-HTTPFS-7 HttpFS is not using https by default
ca608a8	2017-10-18	MAPR-HTTPFS-4 Upload restart script into restart file
9695d4a	2017-10-12	MON-2236 HttpFS link in service page does not work
b4ceb9e	2017-10-11	MAPR-HTTPFS-4 Implement customSecure flag

Known Issues and Limitations

- None

Resolved Issues

- None

HttpFS-1.0-1803 (EEP 3.0.3) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	1.0
Release Date	March 2018
MapR Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	N/A
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Upgraded Tomcat version from 6.0.32 to 7.0.82.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
4fe5706	2018-02-13	[MAPR-30661] HttpFS used vulnerable tomcat

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
c1645f6	2018-01-23	MAPR-25377-REOPEN warden.httpfs.conf should not have double quotes for service.env
8defb23	2018-01-22	MAPR-30557 Add support for arbitrary content type to WebHDFS
0697ffc	2018-01-20	MAPR-26377 warden.httpfs.conf should not have double quotes for service.env
029d8e2	2017-10-26	MAPR-HTTPFS-12 Update JSON lib at HttpFS
bf7c40a	2017-10-22	MAPR-HTTPFS-9 HttpFS should be restarted durring any configure.sh -R run
7fb9d19	2017-10-20	MAPR-HTTPFS-6 HttpFS Signature file is default and world readable
1ddaa89	2017-10-20	MAPR-29745-HttpFS configure.sh -R when HttpFS service added generates an error but does enable service
7a38196	2017-10-20	MAPR-HTTPFS-7 HttpFS is not using https by default
ca608a8	2017-10-18	MAPR-HTTPFS-4 Upload restart script into restart file
9695d4a	2017-10-12	MON-2236 HttpFS link in service page does not work
b4ceb9e	2017-10-11	MAPR-HTTPFS-4 Implement customSecure flag

Known Issues and Limitations

- None

Resolved Issues

- None

HttpFS-1.0-1710 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	1.0
Release Date	November 2017
MapR Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	N/A
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

The following features are new in this release:

- HttpFS now has its own `configure.sh` file.
- SSL is enabled by default on secure clusters (for mapr-core 6.0.0 only).

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
cb12cd7ec	2017-09-21	[MAPR-29298] Add support for content-type plain text in CheckUploadContentType Filter

Known Issues and Limitations

- None

Resolved Issues

- None

HttpFS 1.0-1703 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	1.0
Release Date	April 2017
MapR Version Compatibility	See EEP Components and OS Support on page 5536
GitHub Source	https://github.com/mapr/httpfs
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs).

New in This Release

No new features in this release.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
9a362ae	2017-03-28	[MAPR-26276] HttpFS can't stop correctly on secure clusters
008e322	2017-02-10	[MAPR-25758] CHECKACCESS operation missing
1a2835a	2017-02-15	[MAPR-22912] able to impersonate any user

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
5f84223	2016-10-24	[MAPR-25042] [HTTPFS] Set MapR-SASL as default authentication

Known Issues and Limitations

None

Resolved Issues

- None

HttpFS 1.0-1609 Release Notes

Below are release notes for the HttpFS component included in the MapR Converged Data Platform.

HttpFS Version	1.0
Release Date	September 30, 2016
MapR Version Interoperability	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/httpfs/tree/1.0-mapr-1609
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
0a81f04	2015-08-23	MAPR-24326: The issue causing the HttpFS service to shut down after an upgrade is fixed.

HttpFS 1.0-1606 Release Notes

Below are release notes for the HttpFS component included in the MapR Converged Data Platform.

HttpFS Version	1.0
Release Date	July 1, 2016
MapR Version Interoperability	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/httpfs/tree/1.0-mapr-1606
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-httpfs-1.0.201606271250-1.noarch.rpm • mapr-httpfs_1.0.201606271250_all.deb

New in this Release

This release of Apache HttpFS includes the following behavior change that is specific to MapR:

last-updated: Apr 11, 2022

- The `GET_BLOCK_LOCATIONS` operation is now supported.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
1a00513	2015-06-03	MAPR-23393: Backported HDFS-6874 so that HttpFS now supports the `GET_BLOCK_LOCATIONS` operation.
f33edb7	2016-05-19	MAPR-23400: The start and restart of the HttpFS service now behaves as expected on a Kerberos cluster.

HttpFS 1.0-1504 Release Notes

Version	1.0
Release Date	May 6, 2015
Source on GitHub	https://github.com/mapr/httpfs.git
GitHub Release Tag	1.0-mapr-1504
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

New in This Release

This release of HttpFS Version 1.0 for the MapR Distribution for Apache Hadoop includes the following feature:

- PAM Authentication

Fixes

Commit	Date (YYYY-MM-DD)	Comment
f691ac6	2015-05-05	MAPR-17413: Parameter 'len' saved for backward compatibility.
16305ed	2015-04-29	MAPR-17413 Change API to default Apache API: parameter len length.
6fd122e	2015-04-28	MAPR-17413: Add the ability to read chunk file from filesystem.
1da8cde	2015-04-15	MAPR-18036: [Hue 3.7] Exception at File Browser page.
c87c8f6	2015-04-06	MapR-SASL support
e3379c0	2015-02-04	MapR-SASL support
44849dc	2014-12-09	MAPR-13476: Certificate user mapping support for client cert authentication
0e488c6	2014-11-14	MAPR-13476: HttpFS authentication with client key

Commit	Date (YYYY-MM-DD)	Comment
8f0d394	2014-11-13	MAPR-13476: PAM authentication refactoring
bfacc5e	2014-11-04	MAPR-13476: PAM user&password authentication

HttpFS 1.0-1501 Release Notes

Version	1.0
Release Date	January 21, 2015
Source on GitHub	https://github.com/mapr/httpfs.git
GitHub Release Tag	1.0-mapr-1501
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
e2cc13d	2014-12-24	MAPR-16534: Copying a large file from Hue to a Hive table through HttpFS resulted in only 4096 bytes of the file being copied to the target.

HttpFS 1.0-1409 Release Notes

Version	1.0
Release Date	September 30, 2014
Source on GitHub	https://github.com/mapr/httpfs.git
GitHub Release Tag	1.0-mapr-1409
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

New in This Release

As of HttpFS 1.0-1409, you can configure httpFS with security. For more information, refer to MapR's HttpFS documentation for *SSL Security for HttpFS*.

HttpFS 1.0-1406 Release Notes

Version	HttpFS 1.0-1406
Release Date	June 30, 2014
Source on GitHub	https://github.com/mapr/httpfs.git
GitHub Release Tag	1.0-mapr-1406
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.

last-updated: Apr 11, 2022

Maven Artifacts	https://repository.mapr.com/maven/

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
54fe9d8	12-May-2014	[HDFS-4733] The username pattern for HttpFS is now configurable in the httpfs-site.xml file.
f142a44	27-Jun-2014	MapR 13570. Preferentially calls the IOUtils.copyBytes() method from the 2.3.0 Hadoop API when available, improving file browser response time with large files.

HttpFS 1.0-1401 Release Notes

Version	HttpFS 1.0
Release Date	January 13, 2014
Source on GitHub	https://github.com/mapr/httpfs
GitHub Release Tag	1.0-mapr-1401
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
e1bb581	10-Jan-2013	Fix for MapR. The Zookeeper JAR file is sourced from the Hadoop library instead of the HttpFS library.

Hue Release Notes

The release notes for Hue component included in the MapR Converged Data Platform contains notes specific to MapR only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Hue 4.6.0.300 - 2201 (EEP 8.1.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 4.6.0.300-2201.

The notes below relate specifically to the MapR Data Platform Distribution for Apache Hadoop. You can find additional information in the following change logs or the Hue homepage:

- Changelog for Hue 4.6

- Changelog for Hue 4.5
- Changelog for Hue 4.4

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.6.0.300
Release Date	January 2022
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hue/tree/4.6.0.300-eep-810
GitHub Release Tag	4.6.0.300-eep-810
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP (MEP) and OS to view the list of package names

New in This Release

This Hue release:

- Provides the ability to configure a custom port fo the S3-fs endpoint.
- Disables Impala, Pig, and Sqoop1 applications by default.
- Updates the list of dependencies in Hue to resolve CVE vulnerabilities.
- Allows connection to an S3 server that uses self-signed certificates.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description
6b36590	2021-10-28	MHUE-480 [drill] Backport Drill JDBC client refactoring from Hue 4.3
119d3ae	2021-11-03	MHUE-484 Fix ssl_password_script.sh to resolve issue on Core 7/ Ubuntu 18
2c3191d	2021-11-11	MHUE-238 Provide an ability to configure custom port for S3-fs endpoint
b3bee61	2021-11-18	PR1123 [aws] s3datetime_to_timestamp parse timestamp with Z(minio.io)
0fa24ae	2021-11-19	MHUE-477 [build-boxes] Fix Hue installation on Ubuntu Focal
422b8a4	2021-12-01	fix(boto): S3 region parser references unassigned variable when S3 is colocated
4472b8e	2021-12-01	HUE-9435 [aws] Fix issue with aws behind proxy and make S3_USE_SIGV4 default when region is set
7d3c9d6	2021-12-22	MHUE-491 Disable Sentry, Impala and Pig apps
862faa9	2021-12-24	MHUE-491 Disable Pig in interpreter list
007af8a	2021-12-28	MHUE-491 Disable Sqoop1 in interpreter list
2f5828c	2022-01-07	MHUE-487 Backport fix for CVE-2021-3177

last-updated: Apr 11, 2022

65f6d2a	2022-01-09	MHUE-487 Hue CVE fixes for Jan 2022 release
37dfea4	2022-01-09	HUE-5095 [backend] Python requests library should put port information in log message
c8ee955	2022-01-10	MHUE-487 Revert upgrade of cryptography because it breaks build
7a8c4b6	2022-02-03	MHUE-500 Allow to connect to S3 server that uses self-signed certificates

For complete details, refer to the commit log for this project in GitHub.

Resolved Issues

This release resolves the following issues:

- MHUE-480 - Configured to use Zookeeper connection type does not work for Drill
- MHUE-484 - Fix ssl_password_script.sh which breaks Hue on Ubuntu 18.04
- MHUE-477 - Fix Hue compatibility with Ubuntu 20.04

Known Issues and Limitations

- Hue 4.6 is not compatible with FIPS-enabled setup.
- MapR Data Platform does not support the integration between Hue 4.6.0 and the following components:
  - Solr Search
  - ZooKeeper
- MHUE-209 Hue cannot create a table from a `*.csv` file via importer from ADLS.
- Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.
- When the `[notebook]` section of the `hue.ini` contains a Drill entry that precedes the Hive entry, the Table Browser uses the Drill back end. This can be turned off by changing the `force_hs2_metadata=true` setting in the `[metastore]` section of the `hue.ini` file.

Note: In Hue 4.3.0-1912, support for the integration of Drill with the Table Browser in Hue was added as an experimental feature.

Hue 4.6.0.200 - 2110 (EEP 8.0.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 4.6.0.200-2110.

The notes below relate specifically to the MapR Data Platform Distribution for Apache Hadoop. You can find additional information in the following change logs or the Hue homepage:

- changelog for Hue 4.6
- changelog for Hue 4.5
- changelog for Hue 4.4

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.6.0.200
Release Date	October 2021
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hue/tree/4.6.0.200-eep-800
GitHub Release Tag	4.6.0.200-eep-800
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP (MEP) and OS to view the list of package names

New in This Release

No new features were introduced in this release.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description
64a5614	2021-05-21	MHUE-470 Hue not starts because of wrong permissions of metrics file
5dc5f7b	2021-09-21	MHUE-476 SSL key and certificate could not be found or have a problem
b4ebcb1	2021-09-21	MHUE-474 Remove usage of sudo in Hue

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- MapR Data Platform does not support the integration between Hue 4.6.0 and the following components:
  - Solr Search
  - ZooKeeper
- MHUE-209 Hue cannot create a table from a `*.csv` file via importer from ADLS.
- Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.
- When the `[notebook]` section of the `hue.ini` contains a Drill entry that precedes the Hive entry, the Table Browser uses the Drill back end. This can be turned off by changing the `force_hs2_metadata=true` setting in the `[metastore]` section of the `hue.ini` file.

Note: In Hue 4.3.0-1912, support for the integration of Drill with the Table Browser in Hue was added as an experimental feature.

Hue 4.6.0.0 - 2104 (EEP 7.1.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 4.6.0.0-2104.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information in the following change logs or the Hue homepage:

- changelog for Hue 4.4

last-updated: Apr 11, 2022

- changelog for Hue 4.5
- changelog for Hue 4.6

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.6.0.0
Release Date	April 2021
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hue/tree/4.6.0.100-mapr-710
GitHub Release Tag	4.6.0.0-mapr-710
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Hue 4.6.0.0 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- MHUE-436: Ability to disable the automatic creation of user home directories in the filesystem, by setting `ensure_home_directory` to `false` in the `[desktop] [[auth]]` section of the `hue.ini` file.
- Service verifier

Fixes

This MapR release includes the following fixes on the base release:

GitHubCommitNumber	Date (YYYY-MM-DD)	MapR Fix Number and Description
54cddb8	2021-01-12	MHUE-436 Make creation of users home directory configurable (fix for Hue 4.6)
bfa33a4	2021-02-10	MHUE-444 Upgrade virtualenv to 16.7.10 to fix PIP CVE vulnerability (fix for Hue 4.6)
afe0cc8	2021-02-12	MHUE-451 Various vulnerabilities in Hue (fix for Hue 4.6)
736efa2	2021-04-06	MHUE-455 Add service verifier to Hue package
3787ba0	2021-04-08	MHUE-459 Use CentOS 7 image to build Hue 4.6 to resolve SLES 15 issues

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- MapR does not support the integration between Hue 4.6.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 4.6.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.

- MHUE-209 Hue cannot create a table from *.csv file via importer from ADLS.
- Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.
- When the `[notebook]` section of the `hue.ini` contains a Drill entry that precedes the Hive entry, the Table Browser uses the Drill back end. This can be turned off by changing the `force_hs2_metadata=true` setting in the `[metastore]` section of the `hue.ini` file.

Note: In Hue 4.3.0-1912, support for the integration of Drill with the Table Browser in Hue was added as an experimental feature.

Hue 4.6.0.0 - 2009 (EEP 7.0.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 4.6.0.0-2009.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information in the following change logs or the Hue homepage:

- changelog for Hue 4.4
- changelog for Hue 4.5
- changelog for Hue 4.6

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.6.0.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hue/tree/4.6.0.0-mapr-700
GitHub Release Tag	4.6.0.0-mapr-700
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- The Hue package no longer includes the Simba Drill JDBC driver and instead includes the Drill JDBC driver from the mapr-drill-internal package (MHUE-383). To connect to Drill using the Drill JDBC Interpreter you need to have the mapr-drill-internal package installed on the node with the Hue server.
- The Drill ODBC interpreter is optionally available in Hue through the SQLAlchemy interface. It requires the Simba Drill ODBC drivers installed on the system and available only on RedHat/CentOS (MHUE-327).
- Environment configuration files for Hue server should now be configured in the `${HUE_HOME}/desktop/conf/env.d` directory instead of `${HUE_HOME}/bin/env.d`.
- Hue now has the following HTTP security headers configured by default (MAPR-CORE-307):
  - X-Content-Type-Options
  - X-XSS-Protection

last-updated: Apr 11, 2022

  - Strict-Transport-Security
  - Content-Security-Policy
- TLSv1 and TLSv1.1 were disabled (MHUE-328)
- In kerberized environments, the Hue kt_renewer process is started automatically with Hue server (MHUE-387).

Note: In Hue 4.3.0-1912, support for the integration of Drill with the Table Browser in Hue was added as an experimental feature.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
2d46a11	2020-02-24	MAPR-CORE-307: Add security headers to Hue
e032a6a	2020-03-05	Get rid of redundant logging of JDBC connections
b087bff	2020-03-11	MHUE-328 Disable TLSv1 and TLSv1.1 protocol for Hue 4.6
9ecba78	2020-03-11	MHUE-327 Remove trailing semicolon before execution queries through SQL Alchemy ODBC
68c801d	2020-03-16	MHUE-326 Update Protobuf version to 3.11.1
4fd06d8	2020-03-16	MHUE-332 Could not install table for Hive examples in Hue
74bb9b8	2020-03-18	MHUE-345 Dynamically set Hive version for hive_conf_dir in hue.ini
2c00d3a	2020-03-18	MHUE-328 Show list of enabled ciphers
7a7318b	2020-03-26	MHUE-327 Add SQLAlchemy Drill and its dependencies
d337bd8	2020-04-07	MHUE-348 Internal Server Error while open Hive page
b625f35	2020-04-08	MHUE-330 Error - Hue editor does not use database from databases block for requests
4b92ac5	2020-04-24	MHUE-366 Fix login issue for LDAP users accessing for the first time
c9de095	2020-04-28	MHUE-327 Provide an ability to do the user impersonation with Drill ODBC
2f44f81	2020-04-30	MHUE-356 Set right thrift_version in beeswax
94286d8	2020-05-06	HUE-9175 [core] Upgrade thrift-0.9.1 to thrift-0.13.0
04d9bbd	2020-05-06	HUE-9175 [editor] Regenerate impala and hive thrift from 0.9 to 0.13

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
52ab5ca	2020-05-06	Hue Aquascan CVEs (#1095)
30b92b9	2020-05-06	HUE-9223 [frontend] Fix high risk npm package vulnerabilities
94bc2f5	2020-05-08	MHUE-363 Update Hue 3d-party libraries to fix Security Vulnerabilities / fix of JS libraries
9e30862	2020-06-01	MHUE-384 Drill on HUE does not work when all ZK nodes are mentioned in the hue.ini
b991e7e	2020-06-26	MHUE-387 Add management tools for kt_renewer in Hue
050c5b4	2020-06-26	MHUE-391 Fix import of maprsecurity module on 11 Java
61325a9	2020-07-13	MHUE-393 Job Browser "TypeError: 'NoneType' object is not iterable" when non-admin user try to kill Tez job
29951d5	2020-07-17	MHUE-400 Hue-4.6 not starting on Centos-8.1 (mapr-sasl)
d2a27c0	2020-07-22	MHUE-397 An error occurred while watching the sqoop job running
e66fa5f	2020-07-23	MHUE-378 Hive do not work on non-secure setup
21204bc	2020-07-30	MHUE-411 Update Pig directory path to Pig 0.17.0
7c18d15	2020-07-30	HUE-9288 [editor] Fix selection type variable substitution in the editor
d5b1b67	2020-07-31	MHUE-412 Error while open Hive page in Hue
47397ee	2020-08-05	MHUE-412 Error while open Hive page in Hue
0ea47de	2020-08-06	MHUE-329 Error in logs after install Hue
55dfee0	2020-08-07	MHUE-419 Livy session cannot start on Kerberos cluster
8f83f95	2020-08-10	MHUE-413 Errors after run Impala query
5db3882	2020-08-12	HUE-9290 [frontend] Properly close certain html tags
76c2034	2020-08-12	MHUE-398 Error clicking on the job link in properties
fab2c58	2020-08-12	MHUE-417 Disable "Explain" button for interpreters that do not suppot this feature
b06c801	2020-08-14	MHUE-352 File browser in Hue does not open files that contain "#" symbol in the name

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
755de7c	2020-08-14	MHUE-429 Fix Livy examples for Python 3
7655382	2020-08-17	MHUE-431 TSocket read 0 bytes while opening HBase page in Hue on Kerberos
3672b30	2020-08-21	MHUE-388 Hue to leverage the usage of HA of Hiveserver2
c920c63	2020-08-21	MHUE-352 File browser in Hue does not open files that contain "#" symbol in the name
7bcd5a6	2020-08-26	MHUE-415 Execute USE statement for Drill ODBC and PostgreSQL queries
10652bd	2020-08-28	MHUE-373 Cannot confirm kill jobs in popup
de472b4	2020-08-29	MHUE-383 Switch to Drill drivers from mapr-drill-internal package
8bbaa2c	2020-09-01	MHUE-352 File browser in Hue does not open files that contain "#" symbol in the name
ec23886	2020-09-02	MHUE-435 Cannot connect to Drill on Ubuntu

Known Issues and Limitations

- MapR does not support the integration between Hue 4.6.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 4.6.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- MHUE-209 Hue cannot create a table from *.csv file via importer from ADLS.

Note:

- Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.
- When the `[notebook]` section of the `hue.ini` contains a Drill entry that precedes the Hive entry, the Table Browser uses the Drill back end. This can be turned off by changing the `force_hs2_metadata=true` setting in the `[metastore]` section of the `hue.ini` file.

Hue 4.3.0.500 - 2201 (EEP 6.3.6) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 4.3.0.500-2201.

The notes below relate specifically to the MapR Data Platform Distribution for Apache Hadoop. You can find additional information in the following change logs or the Hue homepage:

- Changelog for Hue 4.3

- Changelog for Hue 4.2
- Changelog for Hue 4.1
- Changelog for Hue 4.0

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.3.0.500
Release Date	January 2022
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hue/tree/4.3.0.500-mapr-636
GitHub Release Tag	4.3.0.500-mapr-636
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP (MEP) and OS to view the list of package names

New in This Release

This Hue release:

- Updates the list of Hue dependencies to resolve CVE vulnerabilities.
- Updates Log4j 1.2.17 to Log4j 1.3.1-mapr, which resolves a couple of minor vulnerabilities.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description
acd646f	2021-09-21	MHUE-470 Hue not starts because of wrong permissions of metrics file
b68660d	2021-10-26	MHUE-480 Configured to use Zookeeper connection type does not work for Drill
e9170cc	2021-10-26	MHUE-480 Fix dependencies in Drill connector
c2efd43	2021-10-27	MHUE-480 Fix Protobuf and Guava dependencies in Drill connector
3489d1e da6d2d1	2022-01-05 2022-01-31	MHUE-493 Switch to log4j 1.3.0-mapr Update log4j to 1.3.1-mapr
bcbf294	2022-01-09	MHUE-487 Hue CVE fixes for Jan 2022 release
d7ca610	2022-01-10	MHUE-487 Revert upgrade of cryptography as it breaks build

For complete details, refer to the commit log for this project in GitHub.

Resolved Issues

This release resolves the following issues:

- MHUE-470 - Hue not starts because of wrong permissions of metrics file
- MHUE-480 - Configured to use Zookeeper connection type does not work for Drill

last-updated: Apr 11, 2022

Known Issues and Limitations

- MapR Data Platform does not support the integration between Hue 4.3.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 4.3.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use data-fabric-SASL authentication.
- MHUE-209 Hue cannot create a table from *.csv file via importer from ADLS.
- Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.

Note: In Hue 4.3.0-1912, support for the integration of Drill with the Table Browser in Hue was added as an experimental feature.

Hue 4.3.0.400 - 2104 (EEP 6.3.4) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 4.3.0.400 - 2104.

The notes below relate specifically to the MapR Data Platform distribution for Apache Hadoop. You can find additional information in the following change logs or the Hue homepage:

- changelog for Hue 4.0
- changelog for Hue 4.1
- changelog for Hue 4.2

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.3.0.400
Release Date	April 2021
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hue/tree/4.3.0.400-mapr-634
GitHub Release Tag	4.3.0.400-mapr-634
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Hue 4.6.0.100 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- MHUE-451: Updated list of libraries in Hue to resolve CVE vulnerabilities.

Fixes

This release includes the following fixes on the base release:

GitHubCommitNumber	Date (YYYY-MM-DD)	MapRFixNumberandDescription

last-updated: Apr 11, 2022

ee6e06e	2021-02-09	MHUE-450 Fix Jackson Databind vulnerabilities
5da59b8	2021-02-10	MHUE-444 Upgrade virtualenv to 16.7.10 to fix PIP CVE vulnerability
941db66	2021-02-10	MHUE-441 Update Django to 1.11.29 to fix CVE vulnerabilities
66f2806	2021-02-11	MHUE-443 Update PyYAML to 5.3.1 to resolve CVE vulnerabilities
6d00d1b	2021-02-12	MHUE-451 Various vulnerabilities in Hue component
25146a2	2021-02-12	MHUE-451 Fix issues with updated cryptography and tablib
b2482d2	2021-02-12	DFDEVOPS-1549 Update CentOS 7 build images to 7.4 to have OpenSSL 1.0.2
d438ef9	2021-02-12	HUE-9225 [core] Upgrade certain third-party python libraries that has identified vulnerabilities (#1102)
65e02ac	2021-02-19	MHUE-456 update Zookeeper
7e18d7b	2021-02-20	MHUE-441 Fix Kerberos issues on Ubuntu

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- The MapR Data Platform does not support the integration between Hue 4.3.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 4.3.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use data-fabric-SASL authentication.
- MHUE-209 Hue cannot create a table from *.csv file via importer from ADLS.
- Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.

Resolved Issues

- MHUE-441: Kerberos on Ubuntu does not work for Hive/Impala and HBase/Data Fabric Database applications.

  Note: In Hue 4.3.0-1912, support for the integration of Drill with the Table Browser in Hue was added as an experimental feature.

Hue 4.3.0.300 - 2101 (EEP 6.3.2) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 4.3.0.300-2101.

The notes below relate specifically to the MapR Data Platform distribution for Apache Hadoop. You can find additional information in the following change logs or the Hue homepage:

- changelog for Hue 4.0
- changelog for Hue 4.1
- changelog for Hue 4.2

last-updated: Apr 11, 2022

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.3.0.300
Release Date	January 2021
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hue/tree/4.3.0.300-mapr-632
GitHub Release Tag	4.3.0.300-mapr-632
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- MHUE-445: The Thrift library was updated to version 0.13.0.
- MHUE-436: You can now disable creation of the home directory in DFS on the first user login by changing the value of the `ensure_home_directory` parameter in the `desktop.auth` section of `hue.ini`.
- MHUE-437: Hue 4.3 is now compatible with CentOS 8.

Fixes

This release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
900d872	2021-01-11	MHUE-437 Make Hue 4.3 compatible with CentOS 8
7c7d300	2021-01-11	MHUE-445 Update Thrift libraries to fix Security Vulnerabilities
0a17050	2021-01-12	MHUE-436 Make creation of users home directory configurable

Known Issues and Limitations

- The MapR Data Platform does not support the integration between Hue 4.3.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 4.3.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use data-fabric-SASL authentication.
- MAPR-28087: Livy cannot use the Hive Interpreter. To work around this issue, use the cluster mode. Set `livy.spark.deployMode=cluster` in the `${LIVY_CONF}/livy.conf` file. This issue is caused by Spark 11851:

```
Unable to start spark thrift server against secured hive metastore(GSS
initiate
                    failed)
```

- MHUE-209 Hue cannot create a table from *.csv file via importer from ADLS.

  Note: Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.

- LIVY-42: Livy UI is not accessible on Kerberos.

Resolved Issues

MHUE-386: Hue 4.3.0 in EEP 6.3.0 and 6.3.1 is not supported for Red Hat Enterprise Linux (RHEL) or CentOS 8.x. However, Hue 4.3.0.300 in EEP 6.3.2 can be used with RHEL or CentOS 8.x.

Note: In Hue 4.3.0-1912, support for the integration of Drill with the Table Browser in Hue was added as an experimental feature.

Hue 4.3.0.200 - 2009 (EEP 6.3.1) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 4.3.0.200-2009.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information in the following change logs or the Hue homepage:

- changelog for Hue 4.0
- changelog for Hue 4.1
- changelog for Hue 4.2

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.3.0.200
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hue/tree/4.3.0.200-mapr-631
GitHub Release Tag	4.3.0.200-mapr-631
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Hue now has the following HTTP security headers configured by default (MAPR-CORE-307):
  - X-Content-Type-Options
  - X-XSS-Protection
  - Strict-Transport-Security
  - Content-Security-Policy
- TLSv1 and TLSv1.1 were disabled (MHUE-328).
- In kerberized environments, the Hue kt_renewer process is started automatically with Hue server (MHUE-387).

last-updated: Apr 11, 2022

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
4107ec7	2020-01-10	MAPR-CORE-307: Add security headers to Hue
0597366	2020-03-11	MHUE-328 Disable TLSv1 and TLSv1.1 protocol
9bf9796	2020-03-18	MHUE-328 Show list of enabled ciphers
a9a42dc	2020-04-24	MHUE-366 Fix login issue for LDAP users accessing for the first time
2b5a0f9	2020-05-06	MHUE-357 Fix Jackson vulnerability
69c79b2	2020-06-01	MHUE-384 Drill on HUE does not work when all ZK nodes are mentioned in the hue.ini
cb7681a	2020-06-25	MHUE-387 Add management tools for kt_renewer
d203fe8	2020-07-13	MHUE-393 Job Browser "TypeError: 'NoneType' object is not iterable" when non-admin user try to kill Tez job
a010403	2020-07-24	HUE-8887 [editor] Hive LLAP and Hive Service Discovery connectors
239b56c	2020-07-30	HUE-8605 [metadata] Only show the Table Privilege tab when Sentry is enabled
08f2043	2020-08-07	MHUE-388 Hue to leverage the usage of HA of Hiveserver2

Known Issues and Limitations

- MapR does not support the integration between Hue 4.3.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 4.3.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- MAPR-28087: Livy cannot use the Hive Interpreter. To work around this issue, use the cluster mode. Set `livy.spark.deployMode=cluster` in the `${LIVY_CONF}/livy.conf` file. This issue is caused by Spark 11851:

```
Unable to start spark thrift server against secured hive metastore(GSS
initiate
                    failed)
```

- MHUE-209 Hue cannot create a table from *.csv file via importer from ADLS.

  Note: Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.

- LIVY-42: Livy UI is not accessible on Kerberos.

Hue 4.3.0.100-1912 (EEP 6.3.0) Release Notes
This section provides reference information, including new features, fixes, known issues, and limitations for Hue 4.3.0-1912.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on:

- changelog for Hue 4.0
- changelog for Hue 4.1
- changelog for Hue 4.2

or the Hue homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	Hue: 4.3.0.100 Livy: 0.5.0
Release Date	December 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Hue: https://github.com/mapr/hue/tree/4.3.0.100-mapr-630 Livy: https://github.com/mapr/livy/tree/0.5.0-mapr-1912
GitHub Release Tag	Hue: 4.3.0.100-mapr-630 Livy: 0.5.0-mapr-1912
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- In the Simba Drill Drivers included in the Hue package, `joda-time` library was updated to 2.10.3.
- `joda-time` was also updated to 2.10.3 in the Livy package.
- Jetty was updated to 9.4.22 in Livy to avoid CVE issues.

Fixes

This MapR release includes the following fixes on the base release:

Hue Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
a833c27	2019-05-08	HUE-8833: [editor] Error - hidden popup menu in the presentation section
8436327	2019-06-04	HUE-8826: [frontend] Can't close log block on services page

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
1bb10dc	2019-06-07	MHUE-289: Add sqoop2 to blacklist for hue
c2e89df	2019-06-11	HUE-8873 [jobbrowser] Auto refresh deselects your selection for rerun workflows and schedulers if job is running
4735c0f	2019-07-26	MHUE-291 Hue fails to start when Beeswax is blacklisted
e28cfc7	2019-07-29	MHUE-296: Database is not loading for Impala
a5f575c	2019-06-05	HUE-8713 [jb] Support TEZ jobs
694a63f	2019-09-24	MHUE-298: YarnOozieAttempt instance has no attribute 'finishedTime'
ef4e35d	2019-09-30	HUE-8713 [jb] Fetch log name list dynamically for Yarn jobs
30fa0c8	2019-09-30	HUE-8713 [assist] Add missing column in TEZ
4c84993	2019-09-30	HUE-8713 [notebook] Add progress updates for TEZ jobs
27e487	2019-11-04	MHUE-307: Change version from HBASE-1.1.8 to HBASE-1.1.13
412aca8	2019-11-06	MHUE-303 Table browser throws 500 for tables in Drill schema
f4b6d90	2019-11-06	HUE-8610 [tb] Convert GET format to POST for describe table
9fdb8d0	2019-11-08	MHUE-302 Broken link to Table Browser in table description in left assistant panel
d9136b5	2019-11-11	MHUE-302 Prettify "Page not found" error message
12ea63e	2019-11-15	MAPR-HBASE-132: '401 Client Error' when try to open HBase Browser in Hue on kerberos cluster
347e360	2019-11-18	MHUE-317: Job progress is not updated dynamically
de1fc42	2019-11-21	HUE-319 Update joda-time dependency for in Drill drivers

Livy Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
28d75b3	2019-11-08	[LIVY-526] Upgrade jetty version
345e789	2019-11-12	MLIVY-59 Update Livy Jetty version to 9.4
d81192a	2019-11-19	MLIVY-63 Replacing joda-time-*.jar with joda-time-2.10.3.jar

Known Issues and Limitations

- MapR does not support the integration between Hue 4.3.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 4.3.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- MAPR-2561: DB Query in Hue cannot execute more than one query.
- MAPR-28087: Livy cannot use the Hive Interpreter. To work around this issue, use the cluster mode. Set `livy.spark.deployMode=cluster` in the `{LIVY_CONF}/livy.conf` file. This issue is caused by Spark 11851:

```
Unable to start spark thrift server against secured hive metastore(GSS
initiate failed)
```

- MHUE-209 Hue cannot create a table from *.csv file via importer from ADLS.

Note: Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.

- LIVY-42: Livy UI is not accessible on Kerberos.

Resolved Issues

- MHUE-302 Broken link to Table Browser in table description in left assistant panel
- MHUE-303 Integration of Table Browser with Drill is not supported

Note: In Hue 4.3.0-1912, support for the integration of Drill with the Table Browser in Hue was added as an experimental feature.

Hue 4.3.0-1904 (EEP 6.2.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 4.3.0-1904.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on:

- changelog for Hue 4.0
- changelog for Hue 4.1
- changelog for Hue 4.2

or the Hue homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	Hue: 4.3.0 Livy: 0.5.0
Release Date	April 2019

last-updated: Apr 11, 2022

MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Hue: https://github.com/mapr/hue/tree/4.3.0.0-mapr-620 Livy: https://github.com/mapr/livy/tree/0.5.0-mapr-1904
GitHub Release Tag	Hue: 4.3.0.0-mapr-620 Livy: 0.5.0-mapr-1904
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

Hue Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
30d39a92	15.03.19	MHUE-242: HUE 4.2.0 has Drill JDBC 1.5.9.1018 which is not compatible with Drill 1.14.0 and 1.15.0
a9ee7e90	21.03.19	MHUE-248: Error with default value in variable for HUE-4.3
7471813e	21.03.19	MHUE-251: Boolean variable substitution fails with coercion error
8d74dedb	22.03.19	MHUE-252: Error with redirect to page for Hue-4.3
74e94383	01.03.19	MHUE-253: Oozie workflow submission failed
319ee1a8	02.01.19	MHUE-259: Error while run new Coordinator
522c28bc	01.04.19	MHUE-254: Can't reload job in mini job browser
f0ab1ba6	05.04.19	MHUE-254: Can't kill job in mini job browser
257cd92f	02.04.19	MHUE-257: Can't close context menu
0de424e5	01.04.19	MHUE-258: No redirect after changing url in command line in job browser
256cf1d7	07.01.19	MHUE-260: Can't create new database in tables browser for Hive
25c51d56	04.04.19	MHUE-261: Error while click 'Refresh' button in tables browser
061b0cdd	08.04.19	MHUE-266: Error while run oozie workflow job for HiveServer2 Script

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
b8437285	09.04.19	MHUE-272: Error while open sqoop2 page in hue-4.3
76b837eb	09.04.19	MHUE-271: Installer does not configure sqoop2
81785854	10.04.19	MHUE-268: Cannot create a table in a newly created database, also 'Refresh' button does not work.
de426eb9	11.04.19	MHUE-199: Error when changing owner for s3
f23deeed	12.04.19	MHUE-275: Can't save file after use 'Save as' button in Azure
be221bd0	08.04.19	MHUE-265: Error while share documents
91a74325	15.04.19	MHUE-187: R session for R Editor in Hue cannot start
9452fb7d	17.04.19	MHUE-141: Example maprdb tables are not downloaded on MAPR-SASL cluster
089d385f	22.01.19	MHUE-280: After adding new requests - the order of launching requests does not change
0445a1be	22.04.19	MHUE-282: Altus API is not configured while open Impala page in Hue
eef21af9	22.04.19	MHUE-182: Not active 'Run' button for Spark Submit Python

Livy Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
2733eac	2019-03-15	LIVY-60 Use Spark jars from cluster instead of pack it in Livy
f2ac34d	2019-04-16	MLIVY-61 Enable access-control in Livy by default on secure clusters

Known Issues and Limitations

- MapR does not support the integration between Hue 4.3.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 4.3.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- MAPR-2561: DB Query in Hue cannot execute more than one query.
- MAPR-28087: Livy cannot use the Hive Interpreter. To work around this issue, use the cluster mode. Set `livy.spark.deployMode=cluster` in the `{LIVY_CONF}/livy.conf` file. This issue is caused by Spark 11851: "Unable to start spark thrift server against secured hive metastore(GSS initiate failed)".
- MHUE-209 Hue cannot create a table from *.csv file via importer from ADLS.

Note: Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.

- LIVY-42: Livy UI is not accessible on Kerberos.
- MHUE-302: Broken link to Table Browser in table description in left assistant panel.
- MHUE-303: Integration of Table Browser with Drill is not supported.

  Note: When the `[notebook]` section of the `hue.ini` contains a Drill entry that precedes the Hive entry, the Table Browser uses the Drill back end. This can be turned off by setting `force_hs2_metadata=true` in the `[metastore]` section of the `hue.ini`.

Resolved Issues

- None.

Hue 4.2.0-1904 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 4.2.0-1904.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on:

- changelog for Hue 4.0
- changelog for Hue 4.1
- changelog for Hue 4.2

or the Hue homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	Hue: 4.2.0 Livy: 0.5.0
Release Date	April 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Hue: https://github.com/mapr/hue/tree/4.2.0-mapr-1904 Livy: https://github.com/mapr/livy/tree/0.5.0-mapr-1904
GitHub Release Tag	Hue: 4.2.0-mapr-1904 Livy: 0.5.0-mapr-1904
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

Hue Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
10db5dda	2019-03-15	MHUE-242: HUE 4.2.0 has Drill JDBC 1.5.9.1018 which is not compatible with Drill 1.14.0 and 1.15.0
2e790925	2018-05-11	MHUE-221: Hue not autocompleting column names in Drill query editor
e47150e1	2019-04-12	MHUE-275: Can't save file after use 'Save as' button in Azure
1d43ac36	2019-04-08	MHUE-265: Error while share documents
ad02c3a5	2019-04-15	MHUE-187: R session for R Editor in Hue cannot start

Livy Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
2733eac	2019-03-15	LIVY-60 Use Spark jars from cluster instead of pack it in Livy
f2ac34d	2019-04-16	MLIVY-61 Enable access-control in Livy by default on secure clusters

Known Issues and Limitations

- MapR does not support the integration between Hue 4.2.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 4.2.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- MAPR-2561: DB Query in Hue cannot execute more than one query.
- MAPR-28087: Livy cannot use the Hive Interpreter. To work around this issue, use the cluster mode. Set `livy.spark.deployMode=cluster` in the `{LIVY_CONF}/livy.conf` file. This issue is caused by Spark 11851: "Unable to start spark thrift server against secured hive metastore(GSS initiate failed)".
- MHUE-209 Hue cannot create a table from *.csv file via importer from ADLS.

Note: Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.

- LIVY-42: Livy UI is not accessible on Kerberos.
- MHUE-302: Broken link to Table Browser in table description in left assistant panel.
- MHUE-303: Integration of Table Browser with Drill is not supported.

Note: When the `[notebook]` section of the `hue.ini` contains a Drill entry that precedes the Hive entry, the Table Browser uses the Drill back end. This can be turned off by setting `force_hs2_metadata=true` in the `[metastore]` section of the `hue.ini`.

last-updated: Apr 11, 2022

Resolved Issues

- None.

Hue 4.2.0-1901 (EEP 6.1.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 4.2.0-1901.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on:

- changelog for Hue 4.0
- changelog for Hue 4.1
- changelog for Hue 4.2

or the Hue homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	Hue: 4.2.0 Livy: 0.5.0
Release Date	February 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Hue: https://github.com/mapr/hue/tree/4.2.0-mapr-1901 Livy: https://github.com/mapr/livy/tree/0.5.0-mapr-1901
GitHub Release Tag	Hue: 4.2.0-mapr-1901 Livy: 0.5.0-mapr-1901
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

Hue Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
7d19798	2018-12-14	MHUE-218 Configuration of HUE for creating json report with available metrics
ec9807c	2018-12-21	MHUE-216 An error occurred while watching the job running with Oozie 5.0

Livy Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
8643c36	2019-01-15	MLIVY-55 Update spark 2.3.2.0 jars for livy for MEP-6.0.1 and MEP-6.1.0 releases
3260882	2019-01-16	MLIVY-55 Add spark-2.3-mapr profile

Known Issues and Limitations

- MapR does not support the integration between Hue 4.2.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 4.2.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- MAPR-2561: DB Query in Hue cannot execute more than one query.
- MAPR-28087: Livy cannot use the Hive Interpreter. To work around this issue, use the cluster mode. Set `livy.spark.deployMode=cluster` in the `{LIVY_CONF}/livy.conf` file. This issue is caused by Spark 11851: "Unable to start spark thrift server against secured hive metastore(GSS initiate failed)".
- MHUE-141 Example MapR Database tables are not downloaded on MAPR-SASL cluster.
- MHUE-158 Error when change timezone in Oozie coordinator.
- HUE-7712 Livy-batch not available in HUE 4.1.
- MHUE-187 R session for R Editor in Hue cannot start.
- MHUE-192 Error, when create Impala table via Table Browser.
- MHUE-206 ERROR "Failed to extract json message" when try to export Hive query result to ADLS.
- MHUE-209 Hue cannot create a table from *.csv file via importer from ADLS.

Note: Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.

- LIVY-42: Livy UI is not accessible on Kerberos.
- MHUE-302: Broken link to Table Browser in table description in left assistant panel.
- MHUE-303: Integration of Table Browser with Drill is not supported.

Note: When the `[notebook]` section of the `hue.ini` contains a Drill entry that precedes the Hive entry, the Table Browser uses the Drill back end. This can be turned off by setting `force_hs2_metadata=true` in the `[metastore]` section of the `hue.ini`.

Resolved Issues

- None.

Hue 4.2.0-1808 (EEP 6.0.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 4.2.0-1808.

last-updated: Apr 11, 2022

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on:

- changelog for Hue 4.0
- changelog for Hue 4.1
- changelog for Hue 4.2

or the Hue homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	Hue: 4.2.0 Livy: 0.5.0
Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Hue: https://github.com/mapr/hue/tree/4.2.0-mapr-1808 Livy: https://github.com/mapr/livy/tree/0.5.0-mapr-1808
GitHub Release Tag	Hue: 4.2.0-mapr-1808 Livy: 0.5.0-mapr-1808
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Improved Drill integration, see Integrate Hue with Drill on page 3655.
- Improvements in scripts used to manage the Hue Database.
- Hue uses `/opt/mapr/conf/ssl_truststore.pem` file to verify SSL connections with other services on MapR cluster and `/opt/mapr/conf/ssl_keystore.pem` as its certificate.
- For Livy upgrades, from Livy 0.3 and above, user configuration files are saved during upgrade.
- Starting with EEP 6.0, MapR SASL authentication, encryption, and impersonation for Livy is enabled by default on secure clusters. For more information, see Configure Livy with Security on page 3839.
- Support for integrating Hue with ADLS. For more information, see Browsing ADLS data, querying it with SQL and exporting the results back in Hue 4.2.

Known Issues and Limitations

- MapR does not support the integration between Hue 4.2.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 4.2.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.

- MAPR-2561: DB Query in Hue cannot execute more than one query.
- MAPR-28087: Livy cannot use the Hive Interpreter. To work around this issue, use the cluster mode. Set `livy.spark.deployMode=cluster` in the `{LIVY_CONF}/livy.conf` file. This issue is caused by Spark 11851: "Unable to start spark thrift server against secured hive metastore(GSS initiate failed)".
- MHUE-141 Example maprdb tables are not downloaded on MAPR-SASL cluster.
- MHUE-158 Error when change timezone in oozie coordinator.
- HUE-7712 Livy-batch not available in HUE 4.1.
- MHUE-187 R session for R Editor in Hue cannot start.
- MHUE-192 Error, when create Impala table via Table Browser.
- MHUE-206 ERROR "Failed to extract json message" when try to export Hive query result to ADLS.
- MHUE-209 Hue cannot create a table from *.csv file via importer from ADLS.

Note: Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.

- LIVY-42: Livy UI is not accessible on Kerberos.
- MHUE-302: Broken link to Table Browser in table description in left assistant panel.
- MHUE-303: Integration of Table Browser with Drill is not supported.

Note: When the `[notebook]` section of the `hue.ini` contains a Drill entry that precedes the Hive entry, the Table Browser uses the Drill back end. This can be turned off by setting `force_hs2_metadata=true` in the `[metastore]` section of the `hue.ini`.

Resolved Issues

- HUE-6074: [notebook] Execute snippets as Oozie batch.

Fixes

This MapR release includes the following fixes on the base release:

Hue Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
212fe65	2018-05-01	MHUE-103 Hue querying Drill: Support for Outbound Impersonation
896aa92	2018-05-04	MHUE-119 Error when try to move/ copy large file via file browser
458a760	2018-05-08	MHUE-97 Hue-Livy secured by default
87f3d09	2018-05-16	MHUE-128 Error "Unable to authenticate" when open Job Browser page
0986577	2018-05-18	MHUE-139 Determine spark_historyserver_url from maprcli call

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
0c51b1d	2018-05-19	MHUE-142 Authenticate to Spark HistoryServer API with PAM credentials on secure cluster
7396e0b	2018-05-22	MHUE-140 Error while writing the path to the file in the field to load the request for hive
ba32f2a	2018-05-22	MHUE-131 Sample data is missing in database assistant
3c3e9dd	2018-05-22	MHUE-133 Error in sample pig script
47af5fb	2018-05-23	MHUE-135 Error - Could not resolve org.apache.pig.piggybank.evaluation.string.UPPER using imports
6824e44	2018-05-29	MHUE-152 Authenticate to Spark HistoryServer API with MapR-Security credentials on secure cluster
24fc2e6	2018-05-29	MHUE-153 Failed to determine active YARN HS after restart of Warden
f9155e0	2018-05-30	MD-2382 Drill Hue Integration
8e5f098	2018-05-31	MHUE-144 Errors in Hue on Kerberos cluster
8bade2e	2018-06-01	MHUE-161 Use core ssl_truststore.pem to certificate verification
7bd6b79	2018-06-01	MHUE-162 Resolving of Spark Job tracking url is broken after porting
42864e0	2018-06-01	HUE-132 Hue is trying to check certificates even if the verification is turned off
ec9db82	2018-06-04	MHUE-151 Error when starting query in oozie
1bc593a	2018-06-05	MHUE-117 Error while open maprdb table via File Browser
135bba2	2018-06-07	MHUE-157 Error when save bundle in Oozie editor
ea73663	2018-06-07	MHUE-170 Error, timer does not count execution time of the request
063b232	2018-06-08	MHUE-172 Prevent from too big size of hue.out log
bce0069	2018-06-08	MHUE-171 Add full path to ZK master ID in JDBC connection url
3b5a1c2	2018-06-08	MHUE-164 Error when click on the link 'Explain' in rdbms
0d210ee	2018-06-08	MHUE-164 [drill] Implement explain method for Drill API
35b1035	2018-06-15	MHUE-175 Remove ZK Cluster ID in JDBC URL when connecting directly to DrillBit

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
e8b5610	2018-07-03	MHUE-126 Enabling and Disabling tables does not work for MapR Database tables
e2d7c57	2018-07-03	MHUE-113 Remove links to the functional that is not used on the mapr
e173afd	2018-07-10	MHUE-132 Hue is trying to check certificates for HBase even if the verification is turned off
3da7775	2018-07-10	MHUE-188 "Session '-1' not found." for Spark Sample Notebook in Hue
dc4c6f0	2018-07-12	MHUE-188 Prevent from starting all sessions on opening of Sample Notebook
0e1bd1a	2018-07-13	MHUE-159 Port memory issue fix from Hue 3.12 to Hue 4.2
edd5d40	2018-07-16	MHUE-186 MaprFS jar should not be part of hue package - it should be taken from /opt/mapr/lib
39e7f1d	2018-07-17	MHUE-159 Include tools/ops/ into Hue distribution
e5e68b1	2018-07-18	MHUE-159 Add execution permission for scripts in tools/ops/
b860f7c	2018-07-25	MHUE-189 Use ssl_keystore.pem from core instead of certificates extracted from ssl_keystore JKS
1cd6581	2018-07-25	MHUE-181 Turn off debug logging at Hue
4962115	2018-07-25	MHUE-200 Could not update job in sqoop2
7feb4b4	2018-07-31	MHUE-205 Exception 'WebHdfs object has no attribute _mechanism' when try to create file in ADLS
ee07d1a	2018-07-31	MHUE-204 Error in Hue integration with Drill on Kerberos
4359222	2018-08-02	MHUE-207 Correct the comment for 'drill options' in hue.ini
308de58	2018-08-02	MLIVY-51 Hue service are not started after Incremental Install
264e8be	2018-08-07	MHUE-205 Exception 'WebHdfs object has no attribute _mutual_ssl_auth' when try to create file in ADLS
5028ed6	2018-08-07	MHUE-181 Turn off debug logging at Hue in log.conf

Livy Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
4fb484e	2018-08-07	MLIVY-51 Fix misconfiguration issue in livy.conf
4f09642	2018-08-02	MLIVY-51 Livy service are not started after Incremental Install
d4c1423	2018-07-20	MZEP-109 Move DSR startup scripts out of Livy
45e074f	2018-07-18	MLIVY-37 Set hadoop.scope to provided in pom.xml
f468f04	2018-07-17	MLIVY-37 The key store and trust store password for livy should be taken from ssl-server.xml
acb639f	2018-07-16	MLIVY-50 MaprFS jar should not be part of livy package - it should be taken from /opt/mapr/lib
e450053	2018-07-10	MLIVY-49 Add to livy conf livy.repl.enable-hive-context to true if Spark is configured on Hive
a51286e	2018-07-09	[LIVY-466][RSC] Fix RSCDriver exception during RPC shutdown
f949020	2018-07-09	MLIVY-47 Errors in SparkR with Spark 2.3
0325f78	2018-07-05	MLIVY-46 Livy session is dead due to old hadoop jar in livy
b4d413f	2018-06-08	MLIVY-36 Configure.sh fails during the first running
e5808a0	2018-06-06	MLIVY-35 Warning with message from Spark job that jpam was added multiple times
2a151d1	2018-05-30	MLIVY-30 Need to secure all the passwords in livy.conf using Hadoop CredentialProvider
97e17c0	2018-05-29	MLIVY-33 Enable SASL for Remote Spark Client
c54b292	2018-05-23	MLIVY-27 Livy service cannot start on 6.1.0 cluster
b3d1b41	2018-05-03	MLIVY-25 security-by-default changes for configure.sh
13cb262	2018-05-02	MLIVY-25 Add multiauth authentication

Hue 3.12.0 - 2009 (EEP 5.0.5) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 3.12.0-2009.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information in the following change logs or the Hue homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	3.12.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hue/tree/3.12.0-mapr-2009
GitHub Release Tag	3.12.0-mapr-2009
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
e8574e8	2020-03-05	MHUE-331 Hue to Hive communication over TLSv1.2 not working in Hue-3.12
ad72736	2020-03-11	MHUE-328 Disable TLSv1 and TLSv1.1 protocol for Hue
bced09a	2020-03-18	MHUE-328 Show list of enabled ciphers
e9b0034	2020-05-14	MHUE-366 Fix login issue for LDAP users accessing for the first time

Known Issues and Limitations

- MapR does not support the integration between Hue 4.3.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 3.12.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- MAPR-25661: DB Query in Hue cannot execute more than one query.
- HUE-6074: [notebook] Execute snippets as Oozie batch.
- MAPR-28087: Livy cannot use the Hive Interpreter. To work around this issue, use the cluster mode. Set `livy.spark.deployMode=cluster` in the `${LIVY_CONF}/livy.conf` file. This issue is caused by Spark 11851:

```
Unable to start spark thrift server against secured hive metastore(GSS
initiate failed)
```

Note: Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.

last-updated: Apr 11, 2022

Resolved Issues

None.

Hue 3.12.0-1912 (EEP 5.0.4) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 3.12.0-1912.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on this changelog or the Hue homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	Hue: 3.12.0
Release Date	December 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hue/tree/3.12.0-mapr-1912
GitHub Release Tag	3.12.0-mapr-1912
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
e0a7344	2019-04-30	HUE-7213 [editor] Mark the correct statement when executing selection after the first line.
0719b0	2019-05-03	HUE-8140 [editor] Improve multi-statement execution.

Known Issues and Limitations

- MapR does not support the integration between Hue 3.12.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 3.12.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- MAPR-2561: DB Query in Hue cannot execute more than one query.
- HUE-6074: [notebook] Execute snippets as Oozie batch.

- MAPR-28087: Livy cannot use the Hive Interpreter. To work around this issue, use the cluster mode. Set `livy.spark.deployMode=cluster` in the `{LIVY_CONF}/livy.conf` file. This issue is caused by Spark 11851:

```
Unable to start spark thrift server against secured hive metastore (GSS
initiate failed)
```

Note: Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.

Resolved Issues

- None.

Hue 3.12.0-1901 (EEP 4.1.3 and EEP 5.0.2) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 3.12.0-1901.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on this changelog or the Hue homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	Hue: 3.12.0 Livy: 0.3.0
Release Date	January 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Hue: https://github.com/mapr/hue/tree/3.12.0-mapr-1901 Livy: https://github.com/mapr/livy/tree/0.3.0-mapr-1901
GitHub Release Tag	Hue: 3.12.0-mapr-1901 Livy: 0.3.0-mapr-1901
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

Hue Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
84f92c7	2018-12-14	MHUE-220 Hue 3.12 with Oracle is broken

last-updated: Apr 11, 2022

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
384c389	2018-12-18	MAPR-32233 Backport fixes for XSS vulnerabilities HUE-6212 and HUE-6197

Livy Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
b08cad5	2018-11-12	MLIVY-53 Livy 0.3 not working with MapR 6.0.1 Fix installed
de8319a	2018-11-12	MLIVY-53 Remove unused dependencies
e972b9a	2018-11-12	[Security] Update to support pyspark and sparkr changes in Spark 2.3.1
4d66eb6	2018-11-16	MLIVY-54 Ensure that Livy use proper version of py4j

Known Issues and Limitations

- MapR does not support the integration between Hue 3.12.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 3.12.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- MAPR-2561: DB Query in Hue cannot execute more than one query.
- HUE-6074: [notebook] Execute snippets as Oozie batch.
- MAPR-28087: Livy cannot use the Hive Interpreter. To work around this issue, use the cluster mode. Set `livy.spark.deployMode=cluster` in the `{LIVY_CONF}/livy.conf` file. This issue is caused by Spark 11851: "Unable to start spark thrift server against secured hive metastore(GSS initiate failed)".

Note: Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.

Resolved Issues

- None.

Hue 3.12.0-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 3.12.0-1808.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on this changelog or the Hue homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	Hue: 3.12.0 Livy: 0.3.0 (EEP 4+)

Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Hue: https://github.com/mapr/hue/tree/3.12.0-mapr-1808 Livy: https://github.com/mapr/livy
GitHub Release Tag	Hue: 3.12.0-mapr-1808 Livy: 0.3.0-mapr-1803 (EEP 4+)
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

None.

Fixes

This MapR release includes the following fixes on the base release:

Hue Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
5228b39	2018-04-10	HUE-2656 is missing in Hue-3.12

Known Issues and Limitations

- MapR does not support the integration between Hue 3.12.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 3.12.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- MAPR-2561: DB Query in Hue cannot execute more than one query.
- HUE-6074: [notebook] Execute snippets as Oozie batch.
- MAPR-28087: Livy cannot use the Hive Interpreter. To work around this issue, use `cluster` mode. Set `livy.spark.deployMode=cluster` in `{LIVY_CONF}/livy.conf`. This issue is caused by Spark 11851:

```
Unable to start spark thrift server against secured hive metastore (GSS
initiate failed)
```

Note: Hue uses python parquet lib to read parquet files. This library does not support all possible parquet formats.

Resolved Issues

- None.

last-updated: Apr 11, 2022

Hue 3.12.0-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 3.12.0-1803.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on this changelog or the Hue homepage.

Version	Hue: 3.12.0 Livy: 0.3.0
Release Date	March 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Hue: https://github.com/mapr/hue.git Livy: https://github.com/mapr/livy.git
GitHub Release Tag	Hue: 3.12.0-mapr-1803 Livy: 0.3.0-mapr-1803
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

None.

Fixes

This MapR release includes the following fixes on the base release:

Hue Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
089e749	2018-03-05	MAPR-30824 - [Hue 3.12] Hue should be restarted via configure.sh script only after changes in secure type
8955cbd	2018-03-01	MHUE-93 Review file permissions for Hue
0fa9a98	2018-02-16	MAPR-30810 - [Hue 3.12] Could not connect to RM on Kerberos cluster
dad0475	2018-02-16	MAPR-30568 [Hue 3.12] Need import current user ticket for restart script
3e0f490	2017-11-22	MHUE-47 Fix of restart file in configure.sh
c7637b1	2017-11-15	MAPR-29965 [HUE-3.12] DB transaction failed because of atomic block on home page

Livy Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
4141b70	2018-03-02	IN-1317 Packaging improvements for restart and security

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
d778404	2018-02-27	MLIVY-19 Fixed file path to warden.livy.conf
73f2e22	2018-02-22	Clean up configure.sh
7f690bf	2018-02-22	LIVY-19: Fix Livy so it can start with spark 2.2.1

Known Issues and Limitations

- MapR does not support the integration between Hue 3.12.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 3.12.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- MAPR-2561: DB Query in Hue cannot execute more than one query.
- HUE-6074: [notebook] Execute snippets as Oozie batch.
- MAPR-28087: Livy cannot use the Hive Interpreter. To work around this issue, use `cluster` mode. Set livy.spark.deployMode=cluster in {LIVY_CONF}/livy.conf. This issue is caused by Spark 11851 ("Unable to start spark thrift server against secured hive metastore(GSS initiate failed").

Note: Hue uses python parquet lib to read the parquet files. This library does not support all possible parquet formats.

Resolved Issues

- None.

Hue 3.12.0-1803 (EEP 3.0.3) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 3.12.0-1803.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on this changelog or the Hue homepage.

Version	Hue: 3.12.0
Release Date	March 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Hue: https://github.com/mapr/hue.git
GitHub Release Tag	Hue: 3.12.0-mapr-1803
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

None.

last-updated: Apr 11, 2022

Fixes

This MapR release includes the following fixes on the base release:

Hue Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
089e749	2018-03-05	MAPR-30824 - [Hue 3.12] Hue should be restarted via configure.sh script only after changes in secure type
0fa9a98	2018-02-16	MAPR-30810 - [Hue 3.12] Could not connect to RM on Kerberos cluster
c7637b1	2017-11-15	MAPR-29965 [HUE-3.12] DB transaction failed because of atomic block on home page

Known Issues and Limitations

- MapR does not support the integration between Hue 3.12.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 3.12.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- MAPR-2561: DB Query in Hue cannot execute more than one query.
- HUE-6074: [notebook] Execute snippets as Oozie batch.
- MAPR-28087: Livy cannot use the Hive Interpreter. To work around this issue, use `cluster` mode. Set livy.spark.deployMode=cluster in {LIVY_CONF}/livy.conf. This issue is caused by Spark 11851 ("Unable to start spark thrift server against secured hive metastore(GSS initiate failed").

Note: Hue uses python parquet lib to read the parquet files. This library does not support all possible parquet formats.

Resolved Issues

- None.

Hue 3.12.0-1710 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 3.12.0-1710 and Livy 0.3.0-1710.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on this changelog or the Hue homepage.

Version	Hue: 3.12.0 Livy: 0.3.0
Release Date	November 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Hue: https://github.com/mapr/hue.git Livy: https://github.com/mapr/livy.git

GitHub Release Tag	Hue: 3.12.0-mapr-1710 Livy: 0.3.0-mapr-1710
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Livy was introduced as a separate package to replace Hue Livy. For MapR Livy documentation, see Livy on page 3839.

Fixes

This MapR release includes the following fixes on the base release:

Hue Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
49c77a3	2017-10-23	MHUE-90 Hue upgrade issue
041c398	2017-10-18	DEVOPS-1904 Build Hue for MEP-3.0.2 on CentOS 6
9541f9e	2017-10-16	MAPR-29572 Hue notebooks do not work when metastore blacklisted
56e0e00	2017-10-16	MAPR-29689 [Hue 3.12] validate property in hue.ini is ignored
0449e3d	2017-10-13	MAPR-29569 Ensure that Hue will work on CentOS 6 / SuSE 12
d08ccaf	2017-10-11	MHUE-81 Implement customSecure flag
bbac9f8	2017-10-09	MAPR-29569 Hue doesn't start after fresh install on suse via installer 1.7
6fc8630	2017-10-02	HUE-71 Adding Hue build images.
dfd31ff	2017-10-02	MHUE-47 Remove backuping of hue.ini in configure.sh
b674bbb	2017-10-02	MHUE-63 Fix issue with MySQL library on CentOS 7 / SUSE 12
4f89cde	2017-09-29	IN-1146 Installer set wrong hbase_conf_dir in hue.ini
05d5196	2017-09-27	MAPR-28638 (HUE-3.12] Hue Job Browser throws error when Beeswax is blacklisted
ac904ea	2017-09-26	MAPR-HUE-47 Refactoring Hue configuration steps and security by default
354768e	2017-09-21	MAPR-28561 [Hue 3.12] Failed to execute Hive Example in Workflow Editor on MapR-Secure cluster
de4b233	2017-09-19	Code cleanup after fixes for MHUE-57 and MHUE-66

last-updated: Apr 11, 2022

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
ff2f191	2017-09-12	MHUE-57 Remove obsolete maprsecurity.so library from Hue
120e2e9	2017-09-12	MHUE-66 HTTPS by default on secure cluster
4e2f0e8	2017-09-08	MHUE-62 Update internal Python version
047d492	2017-09-06	MAPR-28564 [HUE-3.12] Unable to submit HiveServer2 Oozie job from Hue in MAPRSASL secured cluster
e687ff5	2017-08-18	MAPR-HUE-60 Fix Hue sqoop tests to work on MapR

Livy Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
663d595	2017-10-27	MLIVY-14 Script that starts Livy in Zeppelin PACC does not work on Ubuntu
9338c58	2017-10-26	MZEP-51 Added ability to set livy.rsc.rpc.server.address on container startup
88bea98	2017-10-26	MZEP-51 Test Zeppelin with bridge Docker network
52179ff	2017-10-22	MSPARK-99 SparkVersion file is not present in /opt/mapr/spark
88c0220	2017-10-21	MZEP-47 No possibility to work with hive tables via spark (livy interpreter) in Zeppelin in PACC
b31031e	2017-10-20	MLIVY-13 Livy restart file can't restart service
e31bdc6	2017-10-20	LIVY-11 have configure.sh issue warning if security is turned on
08647ac	2017-10-17	MZEP-22 Possibility to work with different versions of Python Zeppelin
40407eb	2017-10-12	MLIVY-12 Permissions of livy-restart file
3a3d6a2	2017-10-05	MZEP-41 Zep PACC: Modify livy conf to configure spark (on yarn) configuration
2e339a7	2017-10-04	MZEP-30 Create script to start Livy in container environment
0e77b8b	2017-10-02	MLIVY-10 Add mapr classpath to livy CLASSPATH
7da2e2d	2017-10-02	MLIVY-10 Update Spark version in Livy dependencies
26bd8ff	2017-09-21	LIVY-8 Configure.sh doesn't work with new 6.0.0 builds
bb3ef3a	2017-09-07	LIVY-5 Adds configure.sh script

d6c4d15	2017-08-16	DEVOPS-1662 Fix of build on CentOS 6: update build dependencies of livy-python-api
fbcc2ff	2017-08-16	MAPR-LIVY-2 Livy should be a standalone project

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- MapR does not support the integration between Hue 3.12.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 3.12.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- MAPR-2561: DB Query in Hue cannot execute more than one query.
- HUE-6074: [notebook] Execute snippets as Oozie batch.
- MAPR-28087: Livy cannot use the Hive Interpreter. To work around this issue, use `cluster` mode. Set livy.spark.deployMode=cluster in {LIVY_CONF}/livy.conf. This issue is caused by Spark 11851 ("Unable to start spark thrift server against secured hive metastore(GSS initiate failed").

Note: Hue uses python parquet lib to read the parquet files. This library does not support all possible parquet formats.

Resolved Issues

- Issue MAPR-18668 "Hue+MySQL doesn't work on RH/CentOS7" fixed, as long as issue with broken `libffi.so` and `libsas12.so` libraries on RedHat/CentOS 7.

Hue 3.12.0-1707 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 3.12.0-1707.

The following notes relate specifically to the MapR distribution for Apache Hadoop. You can find additional information on this changelog or the Hue homepage.

Version	3.12.0
Release Date	August 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hue.git
GitHub Release Tag	3.12.0-mapr-1707
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

No new features.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
8b30312	2017-05-11	MAPR-27012 [Hue 3.12] Refactor how Interpreters are disabled in Notebook UI
3fa4304	2017-05-24	MAPR-27502 [Hue 3.12] Failed to get result size for Hive snippets
c733979	2017-05-26	MAPR-27503 [Hue 3.12] Hue cannot connect to RM API on first run on secure cluster
217651f	2017-06-07	MAPR-27488 [HUE-3.10] Unable to access saved notebook after upgrade from HUE-3.9 to HUE-3.10 (fix for 3.12.0)
939057f	2017-06-13	MAPR-27758 [Hue 3.12] Cannot install Impala examples when impersonation is enabled
b49f950	2017-06-13	MAPR-27616 User with read permissions for Notebook can remove another's permissions (fix for 3.12.0)
b7b32ea	2017-06-19	MAPR-27973 [Hue 3.12] Hue error when execute migrate to empty DB
d5cbe8	2017-06-19	APR-27972 [Hue 3.12] Test user cannot run jobs via Job Designer
eedf9ca	2017-06-22	MAPR-28019 [Hue 3.12] User cannot edit workflows in Job Designer
0476f17	2017-06-22	MAPR-28020 [Hue 3.12] Problem with layout in Oozie coordinator submit window
3361e28	2017-06-22	MAPR-28022 [Hue 3.12] User with privileges for use only notebooks has empty popup "Query Editor"

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- Livy is supported only for use with Hue. The Livy home directory is `/opt/mapr/hue-livy/hue-livy-3.12.0/`.
- MapR-28087: Hue Livy cannot use Hive Interpreter. To work around this issue, use cluster mode. Set `livy.spark.deployMode=cluster` in `{LIVY_CONF}/livy.conf`.
- MapR does not support the integration between Hue 3.12.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 3.12.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.

- MapR-18668: Hue does not work on RedHat 7 / CentOS 7 / SuSE 12 when it is configured to use a MySQL database. When this issue occurs, the Control System displays the "Hue Down Alarm." Use this workaround:

  1. Run the following commands to install MariaDB and the RedHat 6 compatibility library:

     ```
     yum install mariadb
     rpm -ivh --nodeps http://yum.mariadb.org/$(rpm -qa mariadb|cut -d-
     -f2)/rhel7-amd64/rpms/MariaDB-$(rpm -qa mariadb|cut -d-
     -f2)-centos7-x86_64-compat.rpm
     ```

  2. Run the following commands to reconfigure Hue:

     ```
     cd /opt/mapr/hue/hue-3.12.0/
     source ./build/env/bin/activate
     hue syncdb --noinput
     hue migrate
     deactivate
     ```

  3. Run the following command to restart Hue:

     ```
     maprcli node services -name hue -action restart -nodes <node_with_hue>
     ```

- MapR-2561: DB Query in Hue cannot execute more than one query.
- Hue-6074: [notebook] Execute snippets as Oozie batch.

Note: Hue uses python parquet lib to read the parquet files. This library does not support all possible parquet formats.

Resolved Issues

None.

Hue 3.12.0-1703 Release Notes

The following notes relate specifically to the MapR distribution for Apache Hadoop.

Version	3.12.0
Release Date	April 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hue.git
GitHub Release Tag	3.12.0-mapr-1703
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

The following is new in this release:

- Hue integration with Drill.

Important: Note these considerations for Hue 3.12.0:

- Livy is supported only for use with Hue. The Livy home directory is:

  ```
  /opt/mapr/hue-livy/hue-livy-3.12.0/
  ```

- MapR does not support the integration between Hue 3.12.0 and the following components:
  - Solr Search
  - Zookeeper
- Integration between Hue 3.12.0 and Sentry 1.7 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.

Fixes

This MapR release includes the following new fixes on the base release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
e3c9a57	2016-12-22	[MAPR-25612] Changed sh script permission, added output in stdout for start command.
e9f5ae9c	2016-12-30	[MAPR-25540] Hue adds the Sentry privilege of URIs for Hive with "maprfs:////", not "maprfs:///".
4479477	2017-01-17	MAPR-25588 Hue cannot rename Sqoop link.
c67c658fc	2017-01-20	MAPR-25668 Pig job cannot display succeed status after job finish via Hue.
be333ff8	2017-01-23	MAPR-25669 Spark Oozie example fails via Hue Workflows.
30e9d8	2017-01-23	MAPR-25684 Oozie job in Hue cannot be submitted by non-mapr user.
7505afa	2017-01-26	[MAPR-25653] Hue cannot execute correctly saved query in db query tab.
35acd20	2017-02-01	[MAPR-25673] List of Notebooks does not correspond to the actual.
acd9126	2017-02-01	[Mapr 25593] Update default paths to Hive, HBase, Pig and Sentry at hue.ini.
b2bd166	2017-02-02	[MAPR-25697] Select current time in Hue rdbms doesn't work.
697bec0	2017-02-02	MAPR-25935 No logs for some finished Oozie jobs.
64bd0baf	2017-02-02	MAPR-25652 Hue cannot save new query in DB Query tab.
f2a59d9c	2017-02-02	MAPR-25882 Job browser doesn't display logs of jobs via classic mode.
44dc41d	2017-02-06	[MAPR-25925] Can't move directory correctly on AWS.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
b634a15	2017-02-06	MAPR-25799 Hue Drill notebook does't work on Kerberos cluster.
df63f92c	2017-02-07	[MAPR-25921] Action "change owner/ group" in S3 Browser doesn't work.
55fd441	2017-02-07	MAPR-25782 Hue Drill notebook doesn't display correct result for dfs storage query.
bd8f011	2017-02-07	MAPR-25683 Error trying to export query result from Pig Notebook.
6f401a	2017-02-21	[MAPR-HUE-21] Link metadata doesn't have page.
94639b2	2017-02-21	[MAPR-HUE-12] Pig scripts incorrectly open in new tab.
5ef5d76	2017-02-22	[MAPR-HUE-10] Hue Hive editor cannot save query result to non-default DB.
7e6432f	2017-02-22	[MAPR-HUE-22] Excessive values of destination for copy/move should be removed.
f3b6805	2017-02-27	MAPR-HUE-32 Update default confs: set submit_to=True for yarn_cluster by default, and fix Oozie 4 compatibility issue.
5a2d6bc	2017-02-27	[MAPR-HUE-29] Hue cannot connect to JobTracker.
2a9f3ff	2017-02-27	MAPR-HUE-14 Hue cannot migrate some Oozie data to PostgreSQL database.
0e7bacb	2017-02-27	[MAPR-HUE-18] Hue doesn't display job details in JobBrowser after job starting.
35cc9fa	2017-02-27	MAPR-HUE-20 Hue opens the same page in new tab when you press scroll wheel on the title of drop-down lists.
b57616c	2017-03-01	MAPR-HUE-19 Search for Sqoop links doesn't work for all content.
3612987	2017-02-28	MAPR-HUE-31 Improve Hue with Drill configuration.
c15b81d	2017-03-01	[MAPR-HUE-16] Hue doesn't work with HBase 1.1.8 (maprdb) in mapr-secure cluster.
58ca6ee	2017-03-02	MAPR-HUE-34 Cannot work with Spark Notebooks from non-mapr user.
577a6a9	2017-03-02	[MAPR-HUE-16] [hue-3.12.0] Added "blob:" to default conf.
db01210	2017-03-06	[MAPR-HUE-21] Link metadata doesn't have page (reopen).

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
c52dd6d	2017-03-06	[MAPR-HUE-9] Hue doesn't correctly display progress of Oozie Job.
16b7912	2017-03-06	MAPR-HUE-13 New folder creation window has wrong position.
6e721fb	2017-03-06	[MAPR-HUE-28] Can't run any example for sample notebook.
cb5e3e4	2017-03-07	[MAPR-HUE-33] Hue cannot save Impala query result to file.
0561744	2017-03-10	[MAPR-HUE-39] Characters duplicated when typed in the notebooks.
ade1463	2017-03-10	[MAPR-HUE-9] Hue doesn't correctly display progress of Oozie Job (reopen).
0761b1b	2017-03-10	MAPR-HUE-20 Hue opens the same page in new tab when we press scroll wheel on the title of drop-down lists (reopen).
daddd3f	2017-03-13	MAPR-HUE-11 Hue cannot start on SUSE Kerberos cluster.
b5b1bf9	2017-03-14	[MAPR-HUE-17] Hue new create table wizard has an error when create table passed.
3409063	2017-03-16	MAPR-HUE-44 Hue doesn't work when Beeswax in blacklist.
f5172eb	2017-03-16	[DEVOPS-1323] Add java_home env to make file.
973691e	2017-03-16	MAPR-HUE-40 Can't submit Pig script via Notebooks.

Known Issues and Limitations

- **MAPR-18668:** Hue does not work on RedHat/CentOS 7 when it is configured to use a MySQL database. When this issue occurs, the Control System displays the "Hue Down Alarm." Use this workaround:

  1. Run the following commands to install MariaDB and the RedHat 6 compatibility library:

     ```
     yum install mariadb
     rpm -ivh --nodeps http://yum.mariadb.org/$(rpm -qa mariadb|cut -d-
     -f2)/rhel7-amd64/rpms/MariaDB-$(rpm -qa mariadb|cut -d-
     -f2)-centos7-x86_64-compat.rpm
     ```

  2. Run the following command to create a symlink for the Cyrus SASL library:

     ```
     ln -s /lib64/libsasl2.so.3.0.0 /lib64/libsasl2.so.2
     ```

3. Run the following commands to reconfigure Hue:

```
source /opt/mapr/hue/hue-3.10.0/build/env/bin/activate
/opt/mapr/hue/hue-3.10.0/build/env/bin/hue syncdb --noinput
/opt/mapr/hue/hue-3.10.0/build/env/bin/hue migrate
```

4. Run the following command to restart Hue:

```
maprcli node services -name hue -action restart -nodes <space
separated list of hostnames>
```

Note: Hue uses the python parque lib to read the parquet files. This library does not support all possible parquet formats.

- [MAPR-2561] DB Query in Hue can't execute more than one query.
- [HUE-6704] Hue is not able to execute Oozie snippets.

Resolved Issues

None.

Hue 3.10.0-1707 Release Notes
This section provides reference information, including new features, fixes, known issues, and limitations for Hue 3.10.0-1707.

The notes below relate specifically to the MapR distribution for Apache Hadoop. You can find additional information on this changelog or the Hue homepage.

Version	3.10.0
Release Date	August 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hue.git
GitHub Release Tag	3.10.0-mapr-1707
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

No new features.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
0bc600a	2017-05-26	MAPR-27488 [HUE-3.10] Unable to access saved notebook after upgrade from HUE-3.9 to HUE-3.10
af99c1d	2017-06-13	MAPR-27616 User with read permissions for Notebook can remove another permissions

last-updated: Apr 11, 2022

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
ab82658	2017-06-19	MAPR-27973 [Hue 3.12] Hue error when execute migrate to empty DB (fix for 3.10.0)

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- Hue integration with Spark is an experimental feature.
- The Livy version is a snapshot. Livy is supported only for use with Hue. The Livy home directory is `/opt/mapr/hue-livy/hue-livy-3.10.0/`.
- MapR does not support the integration between Hue 3.10.0 and the following components:
  - Solr Search
  - ZooKeeper
- Integration between Hue 3.10.0 and Sentry 1.6 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- MapR-18668: Hue does not work on RedHat 7 / CentOS 7 / SuSE 12 when it is configured to use a MySQL database. When this issue occurs, the Control System displays the "Hue Down Alarm." Use this workaround:

  1. Run the following commands to install Maria and the Red Hat 6 compatibility library:

     ```
     yum install mariadb
     rpm -ivh --nodeps http://yum.mariadb.org/$(rpm -qa mariadb|cut -d-
     -f2)/rhel7-amd64/rpms/MariaDB-$(rpm -qa mariadb|cut -d-
     -f2)-centos7-x86_64-compat.rpm
     ```

  2. Run the following commands to reconfigure Hue:

     ```
     cd /opt/mapr/hue/hue-3.12.0/
     source ./build/env/bin/activate
     hue syncdb --noinput
     hue migrate
     deactivate
     ```

  3. Run the following command to restart Hue:

     ```
     maprcli node services -name hue -action restart -nodes <node_with_hue>
     ```

Note: Hue uses python parquet lib to read the parquet files. This library does not support all possible parquet formats.

Resolved Issues

None.

Hue 3.10.0-1703 Release Notes

The notes below relate specifically to the MapR distribution for Apache Hadoop.

Version	3.10.0
Release Date	April 2017
MapR Version Interoperability	See the EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hue.git
GitHub Release Tag	3.10.0-mapr-1703
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

Important: Note these considerations for Hue 3.10.0:

- The Livy version is a snapshot.
- Hue integration with Spark is an experimental feature.
- Livy is supported only for use with Hue. The Livy home directory is:

```
/opt/mapr/hue-livy/hue-livy-3.10.0/
```

- MapR does not support the integration between Hue 3.10.0 and the following components:
  - Solr Search
  - Zookeeper
- Integration between Hue 3.10.0 and Sentry 1.6 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.

Fixes

This MapR release includes the following new fixes on the base release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
df551f0	2016-12-30	[MAPR-25540] Hue adds the Sentry privilege of URIs for Hive with "maprfs:////", not "maprfs:///"
d1dab93	2017-02-06	[MAPR-26005] Too many logs on migrate and syncdb operations

Known Issues and Limitations

- **MAPR-18668:** Hue does not work on RedHat/CentOS 7 when it is configured to use a MySQL database. When this issue occurs, the Control System displays the "Hue Down Alarm." Use this workaround:
  1. Run the following commands to install MariaDB and the RedHat 6 compatibility library:

```
yum install mariadb
rpm -ivh --nodeps http://yum.mariadb.org/$(rpm -qa mariadb|cut -d-
-f2)/rhel7-amd64/rpms/MariaDB-$(rpm -qa mariadb|cut -d-
-f2)-centos7-x86_64-compat.rpm
```

2. Run the following command to create a symlink for the Cyrus SASL library:

```
ln -s /lib64/libsasl2.so.3.0.0 /lib64/libsasl2.so.2
```

3. Run the following commands to reconfigure Hue:

```
source /opt/mapr/hue/hue-3.10.0/build/env/bin/activate
/opt/mapr/hue/hue-3.10.0/build/env/bin/hue syncdb --noinput
/opt/mapr/hue/hue-3.10.0/build/env/bin/hue migrate
```

4. Run the following command to restart Hue:

```
maprcli node services -name hue -action restart -nodes <space
separated list of hostnames>
```

Note: Hue uses the python parque lib to read the parquet files. This library does not support all possible parquet formats.

Resolved Issues

None.

Hue 3.9.0-1707 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Hue 3.9.0-1707.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on this changelog or the Hue homepage.

Version	3.9.0
Release Date	August 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/hue.git
GitHub Release Tag	3.9.0-mapr-1707
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

No new features.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
8532e7f	2017-06-01	MAPR-27630 [HUE-3.9] Hue unable to connect to thrift for some users

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- MapR does not support the integration between Hue 3.9.0 and the following components:

  - Sentry 1.4
  - Solr Search
  - ZooKeeper
- Integration between Hue 3.9.0 and Sentry 1.6 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- Spark is available as a beta feature. R with Spark 1.3.1 and 1.4.1 is not supported.
- Hive SQL only works with Hive 1.2 and Oozie 4.2.0.
- MAPR-20358: Hive job from Job Designer examples fail on YARN mode without the Oozie patch for 2015-10-23:
- HUE-2673: Hue displays a timeout error when a new session is not started within 60 seconds due to the lack of YARN resources.
- MapR-18668: Hue does not work on RedHat/CentOS 7 when it is configured to use a MySQL database. When this issue occurs, the MapR Control System (MCS) displays the "Hue Down Alarm." Use this workaround:
  1. Run the following commands to install MariaDB and the RedHat 6 compatibility library:

     ```
     yum install mariadb
     rpm -ivh --nodeps http://yum.mariadb.org/$(rpm -qa mariadb|cut -d-
     -f2)/rhel7-amd64/rpms/MariaDB-$(rpm -qa mariadb|cut -d-
     -f2)-centos7-x86_64-compat.rpm
     ```

  2. Run the following command to create a symlink for the Cyrus SASL library:

     ```
     ln -s /lib64/libsasl2.so.3.0.0 /lib64/libsasl2.so.2
     ```

  3. Run the following commands to reconfigure Hue:

     ```
     cd /opt/mapr/hue/hue-3.12.0/
     source ./build/env/bin/activate
     hue syncdb --noinput
     hue migrate
     deactivate
     ```

  4. Run the following command to restart Hue:

     ```
     maprcli node services -name hue -action restart -nodes <node_with_hue>
     ```

Resolved Issues

None.

Hue 3.9.0-1703 Release Notes

The notes below relate specifically to the MapR distribution for Apache Hadoop.

Version	3.9.0
Release Date	April 2017

MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hue.git
GitHub Release Tag	3.9.0-mapr-1703
Maven Artifacts	See Maven Artifacts for MapR on page 4155.
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

None.

Fixes

This MapR release includes the following new fixes on the base release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
382479f	2017-12-30	[MAPR-25540] Hue adds the Sentry privilege of URIs for Hive with "maprfs:////", not "maprfs:///"

Known Issues and Limitations

Note these important limitations:

- MapR does not support the integration between Hue 3.9.0 and the following components:
  - Sentry 1.4
  - Solr Search
  - Zookeeper
- The integration between Hue 3.9.0 and Sentry 1.6 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.
- Spark is available as a beta feature. R with Spark 1.3.1 and 1.4.1 is not supported.
- Hive SQL only works with Hive 1.2 and Oozie 4.2.0.
- **MAPR-20358:** Hive job from Job Designer examples fail on YARN mode without the Oozie patch for 2015-10-23:
  - HUE-2673: Hue displays a timeout error when a new session is not started within 60 seconds due to the lack of YARN resources.
  - MapR-18668: Hue does not work on RedHat/CentOS 7 when it is configured to use a MySQL database. When this issue occurs, the MapR Control System (MCS) displays the "Hue Down Alarm." Use this workaround:

    1. Run the following commands to install MariaDB and the RedHat 6 compatibility library:

```
yum install mariadb
rpm -ivh --nodeps http://yum.mariadb.org/$(rpm -qa mariadb|cut -d-
-f2)/rhel7-amd64/rpms/MariaDB-$(rpm -qa mariadb|cut -d-
-f2)-centos7-x86_64-compat.rpm
```

2. Run the following command to create a symlink for the Cyrus SASL library:

```
ln -s /lib64/libsasl2.so.3.0.0 /lib64/libsasl2.so.2
```

3. Run the following commands to reconfigure Hue:

```
source /opt/mapr/hue/hue-3.10.0/build/env/bin/activate
/opt/mapr/hue/hue-3.10.0/build/env/bin/hue syncdb --noinput
/opt/mapr/hue/hue-3.10.0/build/env/bin/hue migrate
```

4. Run the following command to restart Hue:

```
maprcli node services -name hue -action restart -nodes <space
separated list of hostnames>
```

Resolved Issues

None.

Hue 3.10.0-1611 Release Notes

The notes below relate specifically to the MapR distribution for Apache Hadoop. You may also be interested in the Cloudera Hue changelog or the Cloudera Hue home page.

Version	3.10.0
Release Date	December 9, 2016
MapR Version Interoperability	See the EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/hue.git
GitHub Release Tag	3.10.0-mapr-1611
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

Important: Note these considerations for Hue 3.10.0:

- The Livy version is a snapshot. Livy is supported only for use with Hue.
- MapR does not support the integration between Hue 3.10.0 and the following components:
  - Solr Search
  - Zookeeper
- Integration between Hue 3.10.0 and Sentry 1.6 is supported on secure clusters that use Kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.

The following are new in this release:

- Support for Spark 2.0.1. **Hue integration with Spark is an experimental feature**.
- Support for Sqoop2 1.99.7.

- Livy is moved to its own directory:

```
/opt/mapr/hue-livy/hue-livy-3.10.0/
```

Fixes

This MapR release includes the following new fixes on the base release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
2f44b6d	2016-08-05	[MAPR-2415][hue-3.10.0] - Added missing variable "hive_mechanism"
7dc6616	2016-09-05	[MAPR-24250] Hue 3.10 doesn't install example queries for hive 1.2 and Impala 2.5.0
7fbb091	2016-09-07	[MAPR-24153] Job browser doesn't display details of active job using classic mode
755bf6d	2016-09-08	MAPR-24493 Disabled adding/deleting users in Hue Web UI for PAM and editing user for LDAP (commit for Hue 3.10)
af8cb1c	2016-09-14	[MAPR-24010][hue-3.10.0] Updated hive, hbase, sentry and pig default version.
4480bf3	2016-09-14	MAPR-24153 Job browser doesn't display details of active job using classic mode
959320	2016-09-14	MAPR-24565 {Hue 3.10] Error in "Check configuration" tab for classic mode
8d553d7	2016-09-15	[MAPR-24010]{hue-3.10.0] Changed sentry version from 1.7.0 to 1.6.0
e50dc9a	2016-10-06	MAPR-24840 [Hue 3.10] Link "Location" does not work for tables from "Metastore Manager" tab.
4b91812	2016-10-07	MAPR-24788 Errors when trying to run Spark examples from Sample Notebook
db3b592	2016-10-11	MAPR-24872 [Hue 3.10] Error while opening temporary table details in "Metastore" stab
f1aa4db	2016-10-11	MAPR-24694 [Hue 3.10] Hue can't save result of hive query using Spark notebooks
773048e	2016-09-27	[MAPR-24950] Firefox doesn't support Hue feature dragging tables
09e49f3	2016-10-20	MAPR-25000 [Hue 3.10] Wrong display of Livy sessions in Spark Notebooks
13511d8	2016-10-24	MAPR-25040 [Hue 3.10] Hue build fails
92bfb08	2016-10-26	MAPR-24986 Stack of bugs in Hue Editor, Hue Sentry and Hue DBMS
c76ca23	2016-10-28	MAPR-24857 [Hue 3.10] Hue 3.10 doesn't work with new version of Sqoop2
443762c	2016-10-27	MAPR-24937 [Hue 3.9] Reflected XSS (Hue>Spark beta>>Notebook) (Commit for Hue 3.10)
083ec13	2016-11-02	MAPR-25017 [Hue 3.10] Cannot start session until close existing sessions on Sample Notebook page
923a602	2016-06-14	[MAPR-25109][Hue 3.10] Cannot add Impala to blacklist of hue.ini

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
e2a13c3	2016-08-19	MAPR-24936 - [Hue 3.9] Reflected XSS (Hue>>Hive>>Settings) (Commit for Hue 3.10)
42d715b	2016-11-08	MAPR-25019 [Hue 3.10] Hue + Hive + SSL doesn't work after switching mode on a Kerberos cluster
f7238e5	2016-11-09	[MAPR-25201] Add some commentary to the hue.ini file for regex
50fb043	2016-11-15	[MAPR-25248] Migrate Livy to external repository
9432387	2016-11-15	MAPR-25222 [Hue 3.10] Hue livy sessions don't work correctly when Spark mode is yarn (fix R sample)
0f5ba6f1	2016-11-22	MAPR-25284 Notebooks application is unavailable when beeswax was added to blacklist application
f8a5cdf7	2016-11-28	MAPR-25309 A user with notebook.access and spark.access privileges in HUE could not use notebooks
d0fde67	2016-11-30	MAPR-25363 [Hue 3.10] Failed to open parquet file using FileBrowser
73d3b14	2016-11-30	MAPR-25346 [Hue 3.10] Spark sample snippets do not work after upgrade from Hue 3.9 to Hue 3.10

Known Issues and Limitations

- **MAPR-18668:** Hue does not work on RedHat/CentOS 7 when it is configured to use a MySQL database. When this issue occurs, the Control System displays the "Hue Down Alarm." Use this workaround:

  1. Run the following commands to install MariaDB and the RedHat 6 compatibility library:

     ```
     yum install mariadb
     rpm -ivh --nodeps http://yum.mariadb.org/$(rpm -qa mariadb|cut -d-
     -f2)/rhel7-amd64/rpms/MariaDB-$(rpm -qa mariadb|cut -d-
     -f2)-centos7-x86_64-compat.rpm
     ```

  2. Run the following command to create a symlink for the Cyrus SASL library:

     ```
     ln -s /lib64/libsasl2.so.3.0.0 /lib64/libsasl2.so.2
     ```

  3. Run the following commands to reconfigure Hue:

     ```
     source /opt/mapr/hue/hue-3.10.0/build/env/bin/activate
     /opt/mapr/hue/hue-3.10.0/build/env/bin/hue syncdb --noinput
     /opt/mapr/hue/hue-3.10.0/build/env/bin/hue migrate
     ```

  4. Run the following command to restart Hue:

     ```
     maprcli node services -name hue -action restart -nodes <space
     separated list of hostnames>
     ```

  Note: Hue uses the python parque lib to read the parquet files, and this library doesn't support all possible parquet formats.

last-updated: Apr 11, 2022

Resolved Issues

Commit	Data (YYYY-MM-DD)	Comment
943238	2016-11-18	MAPR-25222 [Hue 3.10] Hue livy sessions don't work correctly when spark mode is yarn and fixed R example.

Hue 3.9.0-1609 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	3.9.0
Release Date	September 30, 2016
MapR Version Interoperability	See Hue Support Matrix on page 5634.
Source on GitHub	https://github.com/mapr/hue/tree/3.9.0-mapr-1609
GitHub Release Tag	3.9.0-mapr-1609
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hue-3.9.0.201609271316-1.noarch.rpm • mapr-hue_3.9.0.201609271316_all.deb • mapr-hue-base-3.9.0.201609271316-1.noarch.rpm • mapr-hue-base_3.9.0.201609271316_all.deb • mapr-hue-livy-3.9.0.201609271316-1.noarch.rpm • mapr-hue-livy_3.9.0.201609271316_all.deb

Important Notes

- MapR does not support the integration between Hue 3.9.0 and the following components: Sentry 1.4, Solr Search, and Zookeeper.

  Note: The integration between Hue 3.9.0 and Sentry 1.6 is supported on secure clusters that use kerberos authentication, but it is not supported on secure clusters that use MapR-SASL authentication.

- Spark is available as a beta feature. R with Spark 1.3.1 and 1.4.1 is not supported.
- Hive SQL example only works with Hive 1.2 and Oozie 4.2.0.

Fixes

This release by MapR includes the following fixes on the base release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comments
af8cb1cef	2016-09-01	MAPR-24010: This fix updates the default paths to Hive and HBase in the hue.ini.

GitHub Commit	Date (YYYY-MM-DD)	Comments
2db5883283	2016-09-09	MAPR-24493: This fix disables adding or deleting users in the Hue Web UI when the authentication type is PAM, and disables editing users for LDAP.

Known Issues

- MapR-20358: Hive job from Job Designer examples fails on YARN mode without Oozie patch for 2015-10-23.
- HUE-2673: Hue displays a timeout error when a new session is not started within 60 seconds due to the lack of YARN resources.
- MapR-18668: Hue does not work on RedHat/CentOS 7 when it is configured to use a MySQL database.

  When this issue occurs, the MapR Control System (MCS) displays the "Hue Down Alarm."

  Workaround:

  1. Run the following commands to install MariaDB and the Redhat 6 compatibility library:

     ```
     yum install mariadb
     ```

     ```
     rpm -ivh --nodeps http://yum.mariadb.org/$(rpm -qa
     mariadb|cut -d- -f2)/rhel7-amd64/rpms/MariaDB-$(rpm -qa mariadb|
     cut -d- -f2)-centos7-x86_64-compat.rpm
     ```

  2. Run the following command to create a symlink for the Cyrus SASL library:

     ```
     ln -s /usr/lib64/libsasl2.so.3.0.0 /usr/lib64/libsasl2.so.2
     ```

  3. Run the following commands to reconfigure Hue:

     ```
     source /opt/mapr/hue/hue-3.9.0/build/env/bin/activate
     ```

     ```
     /opt/mapr/hue/hue-3.9.0/build/env/bin/hue syncdb --noinput
     ```

     ```
     /opt/mapr/hue/hue-3.9.0/build/env/bin/hue migrate
     ```

  4. Run the following command to restart Hue:

     ```
     maprcli node services -name hue -action restart -nodes
     <space-separated list of hostnames>
     ```

Hue 3.9.0-1607 Release Notes

The notes below relate specifically to the MapR Converged Data Platform. You may also be interested in the Cloudera Hue changelog or the Cloudera Hue homepage.

Version	3.9.0
Release Date	July 29, 2016

MapR Version Interoperability	See Hue Support Matrix on page 5634.
Source on GitHub	https://github.com/mapr/hue/tree/3.9.0-mapr-1607
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-hue-3.9.0.201607121533-1.noarch.rpm • mapr-hue_3.9.0.201607121533_all.deb • mapr-hue-base-3.9.0.201607121533-1.noarch.rpm • mapr-hue-base_3.9.0.201607121533_all.deb • mapr-hue-livy-3.9.0.201607121533-1.noarch.rpm • mapr-hue-livy_3.9.0.201607121533_all.deb

Important Notes

- MapR does not support the integration between Hue 3.9.0 and the following components: Sentry 1.4, Solr Search, and Zookeeper.

  Note: The integration between Hue 3.9.0 and Sentry 1.6 is supported on secure clusters that use kerberos authentication but it is not supported on a secure clusters that use MapR-SASL authentication.

- Spark is available as a beta feature. R with Spark 1.3.1 and 1.4.1 is not supported.
- Hive SQL example only works with Hive 1.2 and Oozie 4.2.0.

Fixes

GitHub Commit	Date (YYYY-MM-DD)	Comments
b85a13d	2016-02-23	MAPR-22575: The Hue File Browser is now able to view the bz2 file format.
74f2099	2016-15-04	MAPR-23388: Hue links were changed from gethue.tumblr.com to gethue.com.
910fe6b	2016-05-19	MAPR-23375: The Spark python notebooks issue was fixed.
b1ad23e	2016-06-08	MAPR-23396: The installation of the Pig example no longer generates errors because the user id in the example is different from the default Hue user id.
5a7c872	2016-07-18	MAPR-24010: In the beeswax section of the hue.ini, the default Hive version was changed to 1.2 and the default HBase versions was changed to 1.1.1.

Known Issues

- MapR-20358: Hive job from Job Designer examples fails on YARN mode without oozie patch for 2015-10-23.
- HUE-2673: Hue displays a timeout error when a new session is not started within 60 seconds due to the lack of YARN resources.
- MapR-18668: Hue does not work on RedHat/CentOS 7 when it is configured to use a MySQL database.

  When this issue occurs, the Control System displays the "Hue Down Alarm."

  Workaround:

  1. Run the following commands to install MariaDB and the Redhat 6 compatibility library:

     ```
     yum install mariadb
     ```

     ```
     rpm -ivh --nodeps http://yum.mariadb.org/<mariadb_version>/rhel7-amd64/
     rpms/MariaDB-<mariadb_version>-centos7-x86_64-compat.rpm
     ```

  2. Run the following command to create a symlink for the Cyrus SASL library:

     ```
     ln -s /lib64/libsasl2.so.3.0.0 /lib64/libsasl2.so.2
     ```

  3. Run the following command to reconfigure Hue:

     ```
     bash -c "./opt/mapr/hue/hue-3.9.0/build/env/bin/activate; /opt/
     mapr/hue/hue-3.9.0/build/env/bin/hue
           syncdb --noinput; /opt/mapr/hue/hue-3.9.0/build/env/bin/hue
     migrate"
     ```

  4. Run the following command to restart Hue:

     ```
     maprcli node services -name hue -action restart -nodes <space
     separated list of
           hostnames>
     ```

Hue 3.9.0-1510 Release Notes

The following notes relate specifically to the MapR distribution for Apache Hadoop. You may also be interested in the Cloudera Hue changelog or the Cloudera Hue homepage.

Version	3.9.0
Release Date	November 20, 2015
MapR Version Interoperability	See Hue Support Matrix on page 5634.
Source on GitHub	https://github.com/mapr/hue/tree/3.9.0-mapr-1510
Maven Artifacts	https://repository.mapr.com/maven/

Package Names	The following packages are associated with this release: • mapr-hue-3.9.0.201511191616-1.noarch.rpm • mapr-hue_3.9.0.201511191616_all.deb • mapr-hue-base-3.9.0.201511191616-1.noarch.rpm • mapr-hue-base_3.9.0.201511191616_all.deb • mapr-hue-livy-3.9.0.201511191616-1.noarch.rpm • mapr-hue-livy_3.9.0.201511191616_all.deb

Important Notes

- MapR does not support integration between Hue 3.9.0 and the following components: Sentry, Solr Search, and Zookeeper.
- Spark is available as a beta feature. R with Spark 1.3.1 and 1.4.1 is not supported.
- Hive SQL example only works with Hive 1.2 and Oozie 4.2.0.

Fixes

GitHub Commit	Date (YYYY-MM-DD)	Comment
5a7c872	2015-10-20	MAPR-21019: When Hue submits Oozie jobs, Hue uses the default_jobtracker_host value (maprfs:///) instead of the jobtracker host:port address.
5f7812d	2015-09-15	MAPR-20441: Hue now automatically detects the Sqoop security settings and Hue.ini no longer contains the sqoop_conf_dir property.
569dbf5	2015-09-15	MAPR-20438: Hue is now able to connect to the ResourceManager on secure cluster because the path to ssl_cacerts in the hue.ini was corrected.
6fd0484	2015-09-07	MAPR-20357: Job browser is now able to display jobs when the cluster runs in classic mode.
ecdd711	2015-09-07	MAPR-20440: Hue now automatically detects the Hive security mechanism and therefore hive action statements in Oozie workflows no longer fail.

Known Issues

- HUE-2673: Hue displays a timeout error when a new session is not started within 60 seconds due to the lack of YARN resources.
- MapR-20358: Hive job from Job Designer examples fails on YARN mode without oozie patch for 2015-10-23.

- MapR-18668: Hue does not work on RedHat/CentOS 7 when it is configured to use a MySQL database.

  When this issue occurs, the Control System displays the "Hue Down Alarm."

  Workaround:

  1. Run the following commands to install MariaDB and the Redhat 6 compatibility library:

     ```
     yum install mariadb
     ```

     ```
     rpm -ivh --nodeps http://yum.mariadb.org/$(rpm -qa mariadb|cut -d-
             -f2)/rhel7-amd64/rpms/
     MariaDB-5.5.41-centos7_0-x86_64-compat.rpm
     ```

  2. Run the following command to create a symlink for the Cyrus SASL library:

     ```
     ln -s /lib64/libsasl2.so.3.0.0 /lib64/libsasl2.so.2
     ```

  3. Run the following command to reconfigure Hue:

     ```
     bash -c "./opt/mapr/hue/hue-3.9.0/build/env/bin/activate; /opt/
     mapr/hue/hue-3.9.0/build/env/bin/hue
           syncdb --noinput; /opt/mapr/hue/hue-3.9.0/build/env/bin/hue
     migrate"
     ```

  4. Run the following command to restart Hue:

     ```
     maprcli node services -name hue -action restart -nodes <space
     separated list of
           hostnames>
     ```

Hue 3.8.1 Release Notes

The following Hue 3.8.1 component release notes are included in the MapR distribution for Apache Hadoop.

Hue 3.8.1-1604 Release Notes

The following notes relate specifically to the MapR distribution for Apache Hadoop. You may also be interested in the Cloudera Hue changelog or the Cloudera Hue homepage.

Version	3.8.1
Release Date	May 4, 2016
Source on GitHub	https://github.com/mapr/hue.git
GitHub Release Tag	https://github.com/mapr/hue/tree/3.8.1-mapr-1604
MapR Version Compatibility	See Hue Support Matrix on page 5634.
Maven Artifacts	https://repository.mapr.com/maven/

Package Names	The following packages are associated with this release: • mapr-hue-3.8.1.201604271202-1.noarch.rpm • mapr-hue_3.8.1.201604271202_all.deb • mapr-hue-base-3.8.1.201604271202-1.noarch.rpm • mapr-hue-base_3.8.1.201604271202_all.deb • mapr-hue-livy-3.8.1.201604271202-1.noarch.rpm • mapr-hue-livy_3.8.1.201604271202_all.deb

Important:

- MapR does not support integration between Hue 3.8.1 and the following components: Sentry, Solr Search, and Zookeeper.

Fixes

GitHub Commit	Date (YYYY-MM-DD)	Comment
cb701de	2016-04-08	MAPR-23047: Hue no longer fails to delete Sqoop jobs by name when the name contain one or more white spaces.

Known Issues

- **MAPR-18668:** Hue does not work on RedHat/CentOS 7 when it is configured to use a MySQL database.

  When this issue occurs, the Control System displays the `"Hue Down Alarm."`

  Workaround:

  1. Run the following commands to install MariaDB and the Redhat 6 compatibility library:

     ```
     yum install mariadb
     ver=$(rpm -qa mariadb|cut -d- -f2)
     rpm -ivh --nodeps http://yum.mariadb.org/$ver/rhel7-amd64/rpms/
     MariaDB-$ver-centos7-x86_64-compat.rpm
     ```

  2. Run the following command to create a symlink for the Cyrus SASL library:

     ```
     ln -s /lib64/libsasl2.so.3.0.0 /lib64/libsasl2.so.2
     ```

  3. Run the following command to reconfigure Hue:

     ```
     bash -c "source /opt/mapr/hue/hue-<version>/build/env/bin/activate;
           /opt/mapr/hue/hue-<version>/build/env/bin/hue syncdb --noinput;
           /opt/mapr/hue/hue-<version>/build/env/bin/hue migrate"
     ```

  4. Run the following command to restart Hue:

     ```
     maprcli node services -name hue -action restart -nodes <space
     separated list of
           hostnames>
     ```

Hue 3.8.1-1507 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Cloudera Hue changelog or the Cloudera Hue homepage.

Version	3.8.1
Release Date	August 5, 2015
Source on GitHub	https://github.com/mapr/hue.git
GitHub Release Tag	3.8.1-mapr-1507
MapR Version Compatibility	See Hue Support Matrix on page 5634.
Maven Artifacts	https://repository.mapr.com/maven/

Important:

- MapR does not support integration between Hue 3.8.1 and the following components: Sentry, Solr Search, and Zookeeper.
- Hue 3.8.1 supports Sqoop2 on a MapR 5.0.

New in this Release

This release of Hue version 3.8.1 for the MapR Distribution for Apache Hadoop includes the following features:

- MapR-SASL security support between Hue 3.8.1 and Sqoop2 1.99.6
- Livy server can be managed by warden. The `warden.livy.conf` file is installed along with the `mapr-hue-livy` package.

Fixes

Commit	Date (YYYY-MM-DD)	Comment
a9b8e06	2015-07-17	MAPR-19601: The warden.livy.conf file is installed along with the mapr-hue-livy package so that the Livy Server can be managed by Warden.
781b2fc	2015-06-04	MAPR-18650: You no longer need to reload the HBase Browser in order to see newly created MapR Database tables. Instead, you need to re-open the directory that contains the table.
d37d3d6	2015-05-19	MapR-18605: MapR Database examples have been added.

Known Issues

- MapR-18668: Hue does not work on RedHat/CentOS 7 when it is configured to use a MySQL database.

  Workaround:

last-updated: Apr 11, 2022

1. Run the following commands to install MariaDB and the Redhat 6 compatibility library:

```
yum install mariadb
rpm -ivh --nodeps http://yum.mariadb.org/$
(rpm -qa mariadb|cut -d- -f2)/rhel7-amd64/rpms/
MariaDB-5.5.41-centos7_0-x86_64-compat.rpm
```

2. Run the following command to create a symlink for the Cyrus SASL library:

```
ln -s /lib64/libsasl2.so.3.0.0 /lib64/libsasl2.so.2
```

3. Run the following command to reconfigure Hue:

```
bash -c "./opt/mapr/hue/hue-3.8.1/build/env/bin/activate; /opt/
mapr/hue/hue-3.8.1/build/env/bin/hue syncdb --noinput; /opt/mapr/hue/
hue-3.8.1/build/env/bin/hue migrate"
```

4. Run the following command to restart Hue:

```
maprcli node services -name hue -action restart -nodes <space
separated list of hostnames>
```

Hue 3.7.0 Release Notes

The following Hue 3.7.0 component release notes are included in the MapR distribution for Apache Hadoop.

Hue 3.7.0-1506 Release Notes

The notes below relate specifically to the MapR distribution for Apache Hadoop. You may also be interested in the Hue github page and the release notes for Version 3.7.

Version	3.7.0
Release Date	July 10, 2015
Source on GitHub	https://github.com/mapr/hue.git
GitHub Release Tag	3.7.0-mapr-1506
MapR Version Compatibility	See Hue Support Matrix on page 5634.
Maven Artifacts	https://repository.mapr.com/maven/

Note: This version of Hue does not support Sentry.

Fixes

Commit	Date (YYYY-MM-DD)	Comment
65b2bb6	2015-05-25	MAPR-18823: Hue is now able to communicate with Hive when the hue.ini security settings are automatically configured to use MapR-SASL.
625bf6c	2015-06-12	MAPR-19099: On a secure MapR cluster, Impala is now able to work with Hue.

Known Issues

MapR-17229: The HBase examples provided in Hue 3.7 will not load in HBase 0.94.x because HBase 0.94.x uses a different thrift version than Hue 3.7.

MapR-17314: When you run Hue 3.7 with a Hadoop version that is less than 2.5.1, the Job Browser hangs if you attempt to kill running YARN applications from the Job Browser window. This issue occurs with MapR version 4.0.1 as it uses Hadoop 2.4.1.

MapR-18668: Hue does not work on RedHat/CentOS 7 when it is configured to use a MySQL database.

When this issue occurs, the Control System displays the "Hue Down Alarm."

Workaround:

1. Run the following commands to install MariaDB and the Redhat 6 compatibility library:

```
yum install mariadb
rpm -ivh --nodeps http://
yum.mariadb.org/$(rpm -qa mariadb|
cut -d- -f2)/rhel7-amd64/rpms/
MariaDB-5.5.41-centos7_0-x86_64-com
pat.rpm
```

2. Run the following command to create a symlink for the Cyrus SASL library:

```
ln -s /lib/64/libsasl2.so.3.0.0 /
lib64/libsasl2.so.2
```

3. Run the following command to reconfigure Hue:

```
bash -c "./opt/
mapr/hue/hue-3.7.0/build/env/bin/
activate; /opt/mapr/hue/hue-3.7.0/
build/env/bin/hue
syncdb --noinput; /opt/mapr/hue/
hue-3.7.0/build/env/bin/hue
migrate"
```

4. Run the following command to restart Hue:

```
maprcli node services -name
hue -action restart -nodes <space
separated list of hostnames>
```

Hue 3.7.0-1505 Release Notes

The notes below relate specifically to the MapR distribution for Apache Hadoop. You may also be interested in the Hue github page and the release notes for Version 3.7.

Version	3.7.0
Release Date	June 2, 2015
Source on GitHub	https://github.com/mapr/hue.git

last-updated: Apr 11, 2022

GitHub Release Tag	3.7.0-mapr-1505
MapR Version Compatibility	See Hue Support Matrix on page 5634.
Maven Artifacts	https://repository.mapr.com/maven/

Note: This version of Hue does not support Sentry.

New in this Release

This release of Hue version 3.7 for the MapR distribution for Apache Hadoop includes the following features:

- MapR-SASL security support between Hue and the following components:
  - YARN on MapR 4.0.1 or greater
  - HttpFS 1.0-1504 and greater on MapR 4.0.1 or greater
  - Hive 0.13-1504 and Hive 1.0-1504 or greater on MapR 4.1 or greater
  - Oozie 4.0.1 or greater on MapR 4.0.1 or greater

  The ability to configure a file size restriction for the File Browser. The file size is specified by the file_size property in the hue.ini file. The default is 1.0 GB.
- In a new installation, Hue automatically determines the following values that you would otherwise configure in the hue.ini file:
  - resourcemanager_api_url
  - proxy_api_url
  - history_server_api_url
  - security_enabled
  - mechanism
- Support for multiple PAM modules. By default, Hue is configured to use PAM authentication for new installations.
- Support for Hive 1.0
- The ability to configure Hue to display Hive 1.0 logs by setting the `use_get_log_api` property to true in the [beeswax] section of the `hue.ini` file.

Fixes

Commit	Date (YYYY-MM-DD)	Comment
3a912cc	2015-03-11	MAPR-13476: Support for SSL mutual certificate-based authentication to HttpFS.
f78b5ef	2015-03-26	MAPR-17765: The following error no longer displays while updating Hue: "IntegrityError: (1062, "Duplicate entry '1' for key 'PRIMARY'")".

Commit	Date (YYYY-MM-DD)	Comment
3cbc369	2015-04-03	MAPR-15976: secure.sh script auto-generates keys required for security configuration.
b5ed618	2015-04-08	MAPR-17413: Added a property to restrict the size of files that can be viewed from the File Browser.
e7d2328	2015-04-14	MapR-18179: Hue UI no longer throws an error after a successful installation performed with the MapR Installer.
fd7696	2015-04-28	MAPR-17413: Changed API parameter len to length.
853e44e	2015-05-05	MAPR-18334: Added getTablesWithoutMemorize function for loading list of tables without cache.

Known Issues

- **MapR-17229:** The HBase examples provided in Hue 3.7 will not load in HBase 0.94.x because HBase 0.94.x uses a different thrift version than Hue 3.7.
- **MapR-17314:** When you run Hue 3.7 with a Hadoop version that is less than 2.5.1, the Job Browser hangs if you attempt to kill running YARN applications from the Job Browser window. This issue occurs with MapR version 4.0.1 as it uses Hadoop 2.4.1.
- **MapR-18668:** Hue does not work on RedHat/CentOS 7 when it is configured to use a MySQL database.

  When this issue occurs, the Control System displays the "Hue Down Alarm."

  Workaround:

  1. Run the following commands to install MariaDB and the Redhat 6 compatibility library:

     ```
     yum install mariadb
     rpm -ivh --nodeps http://yum.mariadb.org/$
     (rpm -qa mariadb|cut -d- -f2)/rhel7-amd64/rpms/
     MariaDB-5.5.41-centos7_0-x86_64-compat.rpm
     ```

  2. Run the following command to create a symlink for the Cyrus SASL library:

     ```
     ln -s /lib/64/libsasl2.so.3.0.0 /lib64/libsasl2.so.2
     ```

  3. Run the following command to reconfigure Hue:

     ```
     bash -c "./opt/mapr/hue/hue-3.7.0/build/env/bin/activate; /opt/
     mapr/hue/hue-3.7.0/build/env/bin/hue syncdb --noinput; /opt/mapr/hue/
     hue-3.7.0/build/env/bin/hue migrate"
     ```

  4. Run the following command to restart Hue:

     ```
     maprcli node services -name hue -action restart -nodes <space
     separated list of hostnames>
     ```

- **MapR-18823:** Hue is unable to communicate with Hive when the hive-site.xml is empty and the hue.ini security settings are automatically configured to use MapR-SASL.

  Workaround: Add the following properties to hive-site.xml with values based on the configuration you require:

  - hive.metastore.sasl.enabled
  - hive.metastore.uris
  - hive.server2.authentication
  - hive.server2.enable.doAs (optional)
  - hive.server2.thrift.sasl.qop

  For example:

```
<property>
<name>hive.metastore.sasl.enabled</name>
<value>true</value>
<description>if true, the metastore thrift interface will be secured with
SASL.</description>
</property>

<property>
 <name>hive.server2.authentication</name>
 <value>MAPRSASL</value>
 <description>authentication type</description>
</property>

<property>
 <name>hive.server2.enable.doAs</name>
 <value>false</value>
</property>

<property>
  <name>hive.server2.thrift.sasl.qop</name>
  <value>auth</value>
<property>

<property>
  <name>hive.metastore.uris</name>
  <value>thrift://localhost:9083</value>
  <description> URI where clients contact Hive metastore server </
description>
</property>
```

Hue 3.7.0-1503 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Hue github page and the release notes for Version 3.7.

Version	3.7.0
Release Date	March 27, 2015
Source on GitHub	https://github.com/mapr/hue.git
GitHub Release Tag	3.7.0-mapr-1503

MapR Version Compatibility	See Hue Support Matrix on page 5634. **Note:** This version of Hue does not support Sentry.
Maven Artifacts	https://repository.mapr.com/maven/

Fixes

Commit	Date (YYYY-MM-DD)	Comment
6989a77	2015-03-13	MAPR-17586: ThriftJobTrackerPlugin is able to start on a MapR version 3.1.1 Kerberos secure cluster.
b8fd140	2015-03-13	MAPR-17533: You can view task logs in the Job Browser when you run Hue 3.7 with a secure MapR cluster.
ae62ff4	2015-03-13	MAPR-17413: Large files can be successfully uploaded and viewed in the Hue File Browser.

Known Issues

- MapR-17314:When you run Hue 3.7 with a Hadoop version that is less than 2.5.1, the Job Browser hangs if you attempt to kill running YARN applications from the Job Browser window. This issue occurs with MapR version 4.0.1 as it uses Hadoop 2.4.1.
- MapR-17229: The HBase examples provided in Hue 3.7 will not load in HBase 0.94.x because HBase 0.94.x uses a different thrift version than Hue 3.7.

Hue 3.7.0-1502 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Hue github page and the release notes for Version 3.7.

Version	3.7.0
Release Date	March 9, 2015
Source on GitHub	https://github.com/mapr/hue.git
GitHub Release Tag	3.7.0-mapr-1502
MapR Version Compatibility	See Hue Support Matrix on page 5634.
Maven Artifacts	https://repository.mapr.com/maven/

New in This Release

This is the initial release of Hue Version 3.7 for the MapR Distribution for Apache Hadoop.

It includes the following features in addition to those listed in the release notes for Version 3.7:

- With Hbase 0.98.7 or above, the Hbase Browser displays the MapR filesystem directories and MapR Database tables. For more information, see MapR's Hue documentation for *Managing MapR Database Tables in Hue 3.x*.
- With MapR 4.0.2 or above, Hue supports MapR-SASL security with JobTracker. For more information, see MapR's Hue documentation to *Configure Hue to use MapR-SASL*.

Note: This version of Hue does not support Sentry.

last-updated: Apr 11, 2022

Fixes

- MapR-17314:When you run Hue 3.7 with a Hadoop version that is less than 2.5.1, the Job Browser hangs if you attempt to kill running YARN applications from the Job Browser window. This issue occurs with MapR version 4.0.1 as it uses Hadoop 2.4.1.
- MapR-17229: The HBase examples provided in Hue 3.7 will not load in HBase 0.94.x because HBase 0.94.x uses a different thrift version than Hue 3.7.
- MapR-17533: You cannot view task logs in the Job Browser when you run Hue 3.7 with a secure MapR cluster. Instead, you can view the task logs in the hadoop user logs directory (/opt/mapr/hadoop/hadoop-<version>/logs/userlogs).

Impala Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The release notes for Impala, included in the MapR Data Platform, contain notes specific to HPE only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior HPE releases, see Previous Versions on page 6578.

Impala 2.12.0.600 - 2110 (EEP 6.3.5) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Impala. For more information, see Apache Impala 2.12 Release Notes or the Apache Impala Guide.

Version	2.12.0.600
Release Date	October 2021
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/private-apache-impala
GitHub Release Tag	2.12.0.600-mapr-635
Maven Artifacts	None
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

Impala 2.12.0.600 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base Apache release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
d178dbc08	2021-07-21	IMPALA-116 Exclude logging jars from sentry dependencies

dda99d456	2021-09-17	IMPALA-117 Fix impala-64 - exit if deadlocked threads were found

Known Issues and Limitations

- None.

Resolved Issues

- None.

Impala 2.12.0.400 - 2101 (EEP 7.0.1) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Impala. See Apache Impala 2.12 Release Notes or the Apache Impala Guide for more information.

Version	2.12.0.400
Release Date	January 2021
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/private-apache-impala
GitHub Release Tag	2.12.0.400-mapr-701
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

Impala 2.12.0.400 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
d18bb87b8	2020-11-20	MapR [IMPALA-105] Override vulnerable ant dependency version with the newer one

Known Issues and Limitations

- See Impala Limitations.

Resolved Issues

- None.

Impala 2.12.0.300 - 2101 (EEP 6.3.2) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

last-updated: Apr 11, 2022

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Impala. See Apache Impala 2.12 Release Notes or the Apache Impala Guide for more information.

Version	2.12.0.300
Release Date	January 2021
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/private-apache-impala
GitHub Release Tag	2.12.0.300-mapr-632
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

Impala 2.12.0.300 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
`bb223e212`	2020-11-20	MapR [IMPALA-105] Override vulnerable ant dependency version with the newer one
`76a5ff977`	2020-10-29	MapR [IMPALA-104] Copy openssl, krb .so files to package to make it compatible both with CentOS 7 and 8 versions

Known Issues and Limitations

- See Impala Limitations.

Resolved Issues

- None.

Impala 2.12.0.200 - 2009 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Impala 2.12.0.200-2009.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information in the changelog for 2.12 or the Impala homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Impala Version	2.12.0.200

Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/impala
Release Tag	impala-2.12.0.200
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit	Date (YYYY-MM-DD)	Comment
7133eb2	2020-05-18	IMPALA-74 Disable build of libhdfspp for Apache ORC
214915a7	2020-07-02	IMPALA-94 Add support to build with CentOS 8
296f641	2020-07-13	IMPALA-95 Fix name of openssl shared library
2158f37	2020-08-06	IMPALA-96 Bump jQuery from 1.12.4 to 3.5.1
8722ef1	2020-06-16	IMPALA-91 Override vulnerable Jackson transitive dependencies to newer ones

Known Issues

- None.

Limitations

- See Impala Limitations.

Fixed Issues

- None.

Impala 2.12.0.100-1912 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Impala 2.12.0.100-1912.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on:

- changelog for 2.11

last-updated: Apr 11, 2022

- changelog for 2.12

or the Impala homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Impala Version	2.12.0.100
Release Date	December 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/impala
Release Tag	impala-2.12.0.100
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit	Date (YYYY-MM-DD)	Comment
a04964f	2019-11-05	IMPALA-79 Fix HBase / MaprDB binary integration

Known Issues

- None.

Limitations

- See Impala Limitations.

Fixed Issues

- None.

Impala 2.12-1904 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Impala 2.12.0-1904.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on:

- changelog for 2.11
- changelog for 2.12

or the Impala homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Impala Version	2.12
Release Date	April 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/impala
Release Tag	impala-2.12.0-1904
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- None.

Fixes

This MapR release includes the following new fixes since the latest MapR Impala 2.10.0 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
911aac8	2019-04-04	IMPALA-46 Adapt Impala into MapR ecosystem
21f9461	2019-03-15	IMPALA-62 Add workaround for Protobuf runtime conflict
6633f14	2018-02-18	IMPALA-5690: Part 1: Rename ostream operators for thrift types
16a5762	2018-02-13	IMPALA-5690: Part 2: Upgrade thrift to 0.9.3-p4
9c26182	2018-08-29	IMPALA-7502: ALTER TABLE RENAME should require ALL on the old table
66700de	2019-04-04	IMPALA-3307: Add support for IANA time-zone db
69367fc	2019-04-04	IMPALA-6857: Add Jvm pause/GC Monitor utility and expose JMX metrics
5133c99	2019-08-24	IMPALA-7483: Abort stuck impalad/ catalogd on JVM deadlock

Known Issues

- None.

Limitations

- See Impala Limitations.

last-updated: Apr 11, 2022

Fixed Issues

- None.

Impala 2.10.0-1808 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Impala 2.10.0-1808.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Impala component included in the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Impala Version	2.10.0
Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/impala/tree/2.10.0-mapr-1808
Release Tag	2.10.0-mapr-1808
Hive support	2.3
HBase support	1.1.8
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

Implemented preserving configuration during package update.

Fixes

This MapR release includes the following new fixes since the latest MapR Impala 2.10.0 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
e3641783	2018-08-29	MAPR-IMPALA-44: Impala cannot work with secure hbase

Known Issues

None.

Limitations

See Impala Limitations.

Fixed Issues

None.

Impala 2.10.0-1803 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Impala 2.10.0-1803.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Impala component included in the MapR Converged Data Platform.

Impala Version	2.10.0
Release Date	March 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/impala
Release Tag	2.10.0-mapr-1803
Hive support	2.1
HBase support	1.1.8
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

This release of Impala includes new features and behavior changes that are specific to MapR. See New Features in Impala 2.10 for MapR on page 3691 for a full list and descriptions of new features and behaviors.

Fixes

This release of Impala on MapR includes the following fixes. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
aa72637c	2018-01-19	[MAPR-30570] Fixed LOAD DATA errors.
3de65a1e	2018-01-23	[MAPR-30548] Impala could not start on CentOS with jdk 1.7.
716875ac	2018-01-29	[MAPR-29157] Impala could not get credentials to work with AWS from core-site.xml in Impala conf directory.

Known Issues

None

Limitations

See Impala Limitations.

Fixed Issues

None

last-updated: Apr 11, 2022

Impala 2.7.0-1710 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Impala 2.7.0-1710.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Impala component included in the MapR Converged Data Platform.

Impala Version	2.7.0
Release Date	November 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/impala
Release Tag	2.7.0-mapr-1710
Hive support	2.1
HBase support	1.1.8
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

No new features in this release.

Fixes

This release of Impala on MapR includes the following fixes. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
2e33fe3	2017-08-02	[MAPR-28575] Impala cannot access MapR Database tables
66948f1	2017-08-18	[MAPR-28621/IMPALA-4363] Add Parquet timestamp validation
aacdbd5e	2017-08-18	Fix a potential pitfall with DiskIoMgr::CancelContext().

Known Issues

[MAPR-29157] Impala can't get credentials to work with AWS from core-site.xml in Impala conf directory.

[MAPR-29618] Impala catalog cannot start on SUSE 12.2 mapr-core-6.0 secure cluster.

Limitations

No new limitations in this release.

Fixed Issues

[MAPR-27403] Impala 2.7.0 does not work with the AWS S3 filesystem on systems running Java 8. This issue relates to a library conflict with joda-time.

last-updated: Apr 11, 2022

Impala 2.7.0-1707 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Impala 2.7.0-1707.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Impala component included in the MapR Converged Data Platform.

Impala Version	2.7.0
Release Date	August 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/impala
Release Tag	2.7.0-mapr-1707
Hive support	2.1
HBase support	1.1.8
Package Names	See Package Names for MapR Ecosystem Packs (EEPs).

New in This Release

No new features in this release.

Fixes

This release of Impala on MapR includes the following fixes. For complete details, refer to the commit log for this project in GitHub.

Commit	Date(YYYY-MM-DD)	Comment
4b567ccc	2017-07-21	[IMPALA-5253/MAPR-28508] The appropriate Thrift transport in the StatestoreSubscriber is used.
bd61e00d	2017-06-30	[IMPALA-3641/MAPR27796] DROP IF EXISTS does not return the object name in the RPC response a table or database is not actually dropped.
6c8d67a6	2017-04-19	[IMPALA-5005/MAPR-26889] The server can send an out of order SASL COMPLETE message.

Known Issue

[MAPR-27403] Impala 2.7.0 does not work with the AWS S3 filesystem on systems running Java 8. This issue relates to a library conflict with joda-time.

Limitations

No new limitations in this release.

Fixed Issues

No fixed issues in this release.

last-updated: Apr 11, 2022

Impala 2.7.0 - 1703 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Impala component included in the MapR Converged Data Platform.

Impala Version	2.7.0
Release Date	April 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/impala
Release Tag	2.7.0-mapr-1703
Hive support	2.1
HBase support	1.1.8
Package Names	See Package Names for MapR Ecosystem Packs (EEPs).

New in This Release

This release of Impala includes new features and behavior changes that are specific to MapR. See New Features in Impala 2.7.0 for MapR for a complete list of new features and feature descriptions.

In addition to support on RedHat and CentOS, Impala 2.7.0 is also supported on SUSE 12 SP1. See EEP 3.0 Components and OS Support.

Fixes

This release of Impala on MapR includes the following fixes. For complete details, refer to the commit log for this project in GitHub.

Commit	Date(YYYY-MM-DD)	Comment
15ab181a	2017-02-02	[MAPR-25934] Altering a partition no longer fails in Impala.
3cf7720e	2017-02-03	[MAPR-25985] Impala starts on SUSE clusters.
3cf7720e	2017-02-03	[MAPR-25988] Running COMPUTE STATS on partitioned tables no longer fails.
220a6d7f	2017-02-06	[MAPR 16579] The MapR Impala configuration script is no longer hard coded to use hadoop-2.4.1.
08cb0217	2017-02-17	[MAPR-26034] CTAS queries no longer remove all files from the /user volume.
2339fca5	2017-03-01	[MAPR-26182] Impala 2.7 works with Hue 3.12.
294d4c59	2017-03-07	[MAPR-26022] The Impala daemon starts on SUSE clusters.

Known Issue

Performance regression for TPCH query 10. See IMPALA-1.

last-updated: Apr 11, 2022

Limitations

See Impala Limitations.

Fixed Issues

None

Impala 2.5.0-1703 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Impala 2.5.0-1703.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Impala component included in the MapR Converged Data Platform.

Impala Version	2.5.0
Release Date	April 2017
MapR Version Interoperability	See Impala Support Matrix on page 5635 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/impala
Release Tag	2.5.0-mapr-1703
Hive support	1.2
HBase support	1.1.x
Package Names	See Package Names for MapR Ecosystem Packs (EEPs).

Fixes

This release of Impala on MapR includes the following fixes. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
741d346	2017-02-16	[MAPR-26034] CTAS queries no longer remove all files from the /user volume.

Known Issues

[MAPR-27463] Impala 2.5.0 does not support SSL when configured on Kerberos.

Impala 2.5.0 - 1606 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Impala component included in the MapR Distribution for Apache Hadoop.

Version	2.5.0
Release Date	June 30, 2016
MapR Version Interoperability	Works with MapR version 5.1.0 and later. See Impala Support Matrix on page 5635 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.

last-updated: Apr 11, 2022

Source on GitHub	https://github.com/mapr/impala
Release Tag	2.5.0-mapr-1606
Hive Support	1.2
HBase Support	1.1.1
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) if you are running 5.2 and install EEP 1.1.1, EEP 2.0, or any subsequent EEP release: The following packages are associated with Impala 2.5.0 - 1606 if you are running 5.1 and you install Impala from the https://package.mapr.com/releases/ecosystem-5.x/ repository, or if you are running 5.2 and install EEP 1.0.0 or 1.1.0: • mapr-impala-2.5.0.201606222231-1.noarch.rpm • mapr-impala-statestore-2.5.0.201606222231-1.noarch.rpm • mapr-impala-server-2.5.0.201606222231-1.noarch.rpm • mapr-impala-catalog-2.5.0.201606222231-1.noarch.rpm

New in this Release

This release of Impala includes new features and behavior changes that are specific to MapR. See New Features in Impala 2.5.0 for MapR for a full list and descriptions of new features and behaviors.

Fixed Issue

The performance issue tracked by IMPALA-1755 is now resolved.

Fixes

This release by MapR includes the following fixes. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
fc0b0aa2	2016-05-20	[IMPALA-3375] Improve TopN performance with a trivial Compare object.
0d3bd54f	2016-05-20	[IMPALA-2548] Codegen Tuple::MaterializeExprs() and use in TopN node.
75a048e	2016-05-20	[IMPALA-3385] Fix crashes on accessing error_log.
8ce0ce6	2016-05-20	[IMPALA-2076] Correct execution time tracking for DataStreamSender.
4605ee4	2016-05-20	[IMPALA-2399] Check for mem limit in allocations in parquet scanner and decompressor.
9ce2136	2016-05-20	[IMPALA-3141] Send dummy filters when filter production is disabled.

last-updated: Apr 11, 2022

8a6dfa0d	2016-05-20	[IMPALA-3395] Old HT filter code uses wrong expr type.
5d134fb7	2016-05-20	[IMPALA-3270] & [IMPALA-3237] Improve handling of unsupported data types.
0a09b2a	2016-05-20	[IMPALA-3194] Allow queries materializing scalar type columns in RC/sequence files.
99896f6a	2016-05-20	[IMPALA-1928] Fix Thrift client transport wrapping order.

Impala 2.2.0 - 1608 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Impala component included in the MapR Distribution for Apache Hadoop.

Version	2.2.0
Release Date	September 1, 2016
MapR Version Interoperability	See Impala Support Matrix on page 5635 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/impala
Release Tag	2.2.0-mapr-1608
Hive Support	1.2
HBase Support	98.12, 1.1
Package Names	The following packages are associated with this release: • mapr-impala-2.2.0.201608311322-1.noarch.rpm • mapr-impala-catalog-2.2.0.201608311322-1.noarch.rpm • mapr-impala-server-2.2.0.201608311322-1.noarch.rpm • mapr-impala-statestore-2.2.0.201608311322-1.noarch.rpm

Known Issue

Impala 2.2.0 has a known performance bug tracked by IMPALA-1755.

Fixes

This release by MapR includes the following fixes. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
2de4d7f	2016-06-22	MAPR-23749/IMPALA-2348: The Catalog service closes the connection to the Hive metastore after table invalidation to allow new connections for subsequent operations.

Commit	Date (YYYY-MM-DD)	Comment
777a46d	2016-08-05	MAPR-23828: The `DESCRIBE [FORMATTED] table` statement correctly displays column metadata.

Impala 2.2.0 - 1602 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Impala component included in the MapR Distribution for Apache Hadoop.

Version	2.2.0
Release Date	February 24, 2016
MapR Version Interoperability	See Impala Support Matrix on page 5635 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/impala.
Release Tag	2.2.0-mapr-1602
Hive Support	1.2
Package Names	The following packages are associated with this release: • mapr-impala-2.2.0.201602231403-1.noarch.rpm • mapr-impala-statestore-2.2.0.201602231403-1.noarch.rpm • mapr-impala-server-2.2.0.201602231403-1.noarch.rpm • mapr-impala-catalog-2.2.0.201602231403-1.noarch.rpm

New in this Release

This release of Impala includes new features and behavior changes that are specific to MapR. See New Features in Impala 2.2.0 for MapR for a full list and descriptions of new features and behaviors.

Known Issue

Impala 2.2.0 has a known performance bug tracked by IMPALA-1755.

Fixes

This release by MapR includes the following fixes. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
3c89c74	2016-01-22	MAPR-22240: You can create and use UDFs when the filesystem changes from C++ to JAVA.
e33bdf3	2016-01-14	MAPR-22160: You can successfully run a SELECT query on partitioned Hive tables after issuing the COMPUTE STATS statement in Impala.

Commit	Date (YYYY-MM-DD)	Comment
713305a	2015-12-07	MAPR-21559: Impala can connect to the Hive metastore when each component is running on a different node and they both have Kerberos enabled.
97eed5f	2015-12-07	MAPR-21532: JAVA_HOME points to the correct library path in the Impala 2.2.0 package.
4711700	2015-12-03	MAPR-21261: You can open Parquet tables created in Impala from the Hive shell.
2b45524	2015-12-03	MAPR-21187: You can issue the LOAD DATA statement through the Impala shell.
4e52e73	2015-12-03	MAPR-21340: You can use Sentry 1.6 to configure Sentry authorization for Impala,
9739ca1	2015-10-21	MAPR-21088: Impala can query MapR Database files.
8714b15	2015-10-21	MAPR-21083: You can created user-defined functions (UDFs) in Impala.

Impala 1.4.1-1501 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Version	1.4.1
Release Date	January 14, 2015
Source on GitHub	https://github.com/mapr/impala
GitHub Release Tag	1.4.1-mapr-1501
MapR Version Interoperability	See Impala Support Matrix on page 5635 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.

This release of Impala includes some updated package files that do not affect Impala performance or functionality.

Important:

- Impala on MapR version 4.0.1 requires mapr-patch-4.0.1.27334.GA-28668.x86_64.rpm. Contact support to obtain the patch. If Oozie is running on the cluster, after you apply the patch you must regenerate the Oozie WAR file to pick up the new binary.
- Impala on MapR version 4.0.2 does not require a patch.
- This release of Impala only works with Sentry in file-based storage mode.

Impala 1.4.1-1410 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Version	1.4.1
Release Date	November 19, 2014
Source on GitHub	https://github.com/mapr/impala
GitHub Release Tag	1.4.1-mapr-1410
MapR Version Interoperability	See Impala Support Matrix on page 5635 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

Important:

- Impala 1.4.1 requires MapR version 4.0.1 with mapr-patch-4.0.1.27334.GA-28668.x86_64.rpm. You must contact support to obtain the patch.
- If Oozie is running on the cluster, after you apply the patch you must regenerate the Oozie WAR file to pick up the new binary.

New Features

Impala 1.4.1 offers some new features for improved performance and functionality, including admission control. For a full list of new features and their descriptions, refer to New Features in Impala 1.4.1 for MapR.

Important: This release of Impala only works with Sentry in file-based storage mode.

Resolved Issues

Impala 1.4.1 on MapR version 4.0.1 with patch `mapr-patch-4.0.1.27334.GA-28668.x86_64.rpm` resolves the following known issues:

MapR Issue	Description
15155	Impala returns results when you run queries that include the UNION ALL clause.
14863	Installing Impala packages no longer updates the mapr-hive package with Hive 0.12.
14397	Impala supports Hive 0.13.
14391	The following message that was inapplicable to MapR filesystem no longer appears in the Impala message log: `hdfs-scan-node.cc:401] Unknown disk id. This will negatively affect performance. Check your hdfs settings to enable block location metadata.`
13802	Warden starts all services when you install Impala on Ubuntu systems.
12793	The Impala daemon uses memory without exceeding the limits set for Impala.

Fixes

Commit	Date (YYYY-MM-DD)	Comment
2e768c9	28-Oct-2014	MapR 14870. The `-authorized_proxy_user_config=mapr=*` option is set by default in env.sh to enable impersonation for Impala.
5fe101f	07-Sept-2014	MapR 13802. Warden starts all Impala services on Ubuntu.

Livy Release Notes

The release notes for Livy component included in the MapR Data Platform contain notes specific to MapR Data Platform only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR Data Platform releases, see Previous Versions on page 6578.

Livy 0.7.0.200 - 2201 (EEP 8.1.0) Release Notes

The notes below relate specifically to the MapR Data Platform Distribution for Apache Hadoop. You can find additional information on the Livy release notes page or Livy homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.7.0.200
Release Date	January 2022
HPE Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/livy/tree/0.7.0.200-eep-810
GitHub Release Tag	0.7.0.200-eep-810
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP(MEP) and OS to view the list of package names.

New in This Release

- MLIVY-96: Updated dependencies in Livy to be compatible with Spark 3.2 .
- MLIVY-98: Fixed incompatibility of Livy Python modules with Python 3.8.
- MLIVY-97, MLIVY-99: Ensured Livy worked on FIPS-enabled cluster. Added support of SCRAM-SHA-256 SASL mechanism for communication between Livy server and Livy session Spark Applications.
- MLIVY-92: Updated dependencies in Livy to resolve CVE vulnerabilities.
- MLIVY-100: Updated `log4j 1.2.17` to `log4j 1.3.1-mapr` to resolve vulnerabilities.

Fixes

This MapR release includes the following fixes on the base release:

last-updated: Apr 11, 2022

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
1daea08	2021-12-09	MLIVY-96 Update dependencies to be compatible with Spark 3.2
96d6457	2021-12-24	Mock ast.Module to work with Python 3.8
78fc627	2022-01-04	MLIVY-70 Switch to log4j 1.3.0-mapr
7b018f7	2022-01-21	MLIVY-97 Fix Livy on FIPS-enabled cluster
f28d269	2022-01-24	MLIVY-99 Fix Livy for older cores
80417bf	2022-01-24	MLIVY-92 CVE fixes
4922f62	2022-01-25	MLIVY-92 Fix Livy after last round of CVE fixing
5c2abf2	2022-01-25	MLIVY-92 Resolve dependencies issues
ad6697d	2022-01-25	MLIVY-100 Update log4j v1 to the 1.3.1-mapr
700c1bb	2022-01-27	MLIVY-97 Change the way of enabling SCRAM-SHA-256 on FIPS setup
70263f8	2022-02-02	MLIVY-101 Build Livy ECO EEP 8.1.0 components with DF v 6.2.0

For complete details, refer to the commit log for this project in GitHub.

Resolved issues

- MLIVY-98: Fixed incompatibility of Livy Python modules with Python 3.8.

Known Issues and Limitations

- When you enable the SSL in a mixed (FIPS and non-FIPS) configuration, Spark application run fails. To run Spark applications, set `spark.ssl.ui.enabled` option to `false` in `spark-defaults.conf` configuration file.
- Hive-compatible JDBC / ODBC server introduced in Livy 0.7 is not available in MapR distribution.

Livy 0.7.0.100 - 2110 (EEP 8.0.0) Release Notes

The notes below relate specifically to the MapR Data Platform Distribution for Apache Hadoop. You can find additional information on the Livy release notes page or Livy homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.7.0.100
Release Date	October 2021
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/livy/tree/0.7.0.100-eep-800
GitHub Release Tag	0.7.0.100-eep-800

Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP(MEP) and OS to view the list of package names.

New in This Release

- Support for Spark 3.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
4068bf1	2021-09-14	SPARK-884 Update Kryo version
9e57289	2021-09-14	SPARK-884 Backport changes for org.apache.livy.rsc.driver.SparkEntries from upstream
6677a0e	2021-09-15	MLIVY-90 No result after executing the script for SparkR
713c93a	2021-09-21	MLIVY-88 Remove usage of sudo in Livy

Known Issues and Limitations

- Hive-compatible JDBC / ODBC server introduced in Livy 0.7 is not available in MapR distribution.

Livy 0.7.0.0 - 2104 (EEP 7.1.0) Release Notes

The notes below relate specifically to the MapR Data Platform Distribution for Apache Hadoop. You can find additional information on the Livy release notes page or Livy homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.7.0.0
Release Date	April 2021
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/livy/tree/0.7.0.0-mapr-710
GitHub Release Tag	0.7.0-mapr-710
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Livy Server was updated to 0.7 version.
- Service verifier
- Configuration of MapR authentication was refactored.

last-updated: Apr 11, 2022

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
34aaf00	2021-04-06	MLIVY-84 Add verify_service script
5d9ad7f	2021-04-15	MLIVY-87 Add missing Kerberos properties to livy.conf

Known Issues and Limitations

- Hive-compatible JDBC / ODBC server introduced in Livy 0.7 is not available in MapR distribution.
- MAPR-28087: Livy cannot use the Hive Interpreter. Workaround: Use the cluster mode. Set `livy.spark.deployMode=cluster` in the `${LIVY_CONF}/livy.conf` file. This issue is caused by Spark 11851 - Unable to start Spark thrift server against secured hive metastore (GSS initiate failed).
- LIVY-42: Livy UI is not accessible on Kerberos.

Livy 0.5.0 - 2009 (EEP 7.0.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Livy 0.5.0-2009.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You can find additional information on the Livy release notes page or Livy homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.5.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/livy/tree/0.5.0-mapr-2009
GitHub Release Tag	0.5.0-mapr-2009
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
05c464b	2020-05-14	MLIVY-64 Add security headers to Livy
df56acd	2020-05-14	MLIVY-71 Fix Guava to be compatible with Guava in Core/Hadoop
a2a2afe	2020-05-15	MLIVY-72 Update jQuery to fix vulnerability

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
4613caa	2020-05-15	MLIVY-66 Update bootstrap to fix vulnerability
b6358a2	2020-07-02	MLIVY-73 Errors in Livy on CentOS 8 with java11
a128805	2020-07-13	MLIVY-73 Errors in Livy on CentOS 8 with java11
89fe852	2020-07-13	MLIVY-74 Session can not start
73837c2	2020-07-15	MLIVY-74 Session can not start
960a78c	2020-07-29	MLIVY-76 Error in Livy PySpark example
63ceb98	2020-08-14	MLIVY-75 Error in Livy during service start

Known Issues and Limitations

- MAPR-28087: Livy cannot use the Hive Interpreter. Workaround: Use the cluster mode. Set `livy.spark.deployMode=cluster` in the `${LIVY_CONF}/livy.conf` file. This issue is caused by Spark 11851 - Unable to start Spark thrift server against secured hive metastore (GSS initiate failed).
- LIVY-42: Livy UI is not accessible on Kerberos.

Livy 0.5.0 - 2201 (EEP 6.3.6) Release Notes

The notes below relate specifically to the MapR Technologies Distribution for Apache Hadoop. You can find additional information on the Livy release notes page or Livy homepage.

These release notes contain only MapR Technologies specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.5.0
Release Date	January 2022
HPE Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/livy/tree/0.5.0-mapr-2201
GitHub Release Tag	0.5.0-mapr-2201
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- MLIVY-100: Updated `log4j 1.2.17` to `log4j 1.3.1-mapr` to resolve vulnerabilities.

Fixes

This MapR Technologies release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description
14b7aaf	2022-01-06	MLIVY-70 Switch to log4j 1.3.0-mapr

last-updated: Apr 11, 2022

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description
1ac7153	2022-01-25	MLIVY-100 Update log4j to the 1.3.1-mapr

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- MAPR-28087: Livy cannot use the Hive Interpreter. Workaround: Use the cluster mode. Set `livy.spark.deployMode=cluster` in the `${LIVY_CONF}/livy.conf` file. This issue is caused by Spark 11851 - Unable to start Spark Thrift Server against secured Hive Metastore (GSS initiate failed).

Livy 0.5.0 - 2104 (EEP 6.3.4) Release Notes

The notes below relate specifically to the MapR Data Platform Distribution for Apache Hadoop. You can find additional information on the Livy release notes page or Livy homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.5.0
Release Date	April 2021
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/livy/tree/0.5.0-mapr-2009
GitHub Release Tag	0.5.0-mapr-2104
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Backported fix for LIVY-547: [LIVY-547][SERVER] Livy kills session after `livy.server.session.timeout` even if the session is active.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
a68df98	2021-03-24	MLIVY-85 Backport [LIVY-547] [SERVER] Livy kills session after livy.server.session.timeout even if the session is active

Known Issues and Limitations

- MAPR-28087: Livy cannot use the Hive Interpreter. Workaround: Use the cluster mode. Set `livy.spark.deployMode=cluster` in the `${LIVY_CONF}/livy.conf` file. This issue is caused by Spark 11851 - Unable to start Spark thrift server against secured hive metastore (GSS initiate failed).
- LIVY-42: Livy UI is not accessible on Kerberos.

Mahout Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The release notes for Mahout component included in the MapR Converged Data Platform contains notes specific to MapR only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Mahout 0.12.0 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following Mahout 0.12.0 component release notes are included in the MapR distribution for Apache Hadoop.

Mahout 0.12.0-1611 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Mahout component included in the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Mahout home page.

Version	0.12.0
Release Date	December 9, 2016
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/mahout/tree/0.12.0-mapr-1611
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

Note: Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2.

New in This Release

This release of Mahout 0.12.0 is only available with EEP 2.0 and includes support for Spark v2.0.1.

Note: Spark versions prior to v2.0.1 are not compatible.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
7a6d7c4	2016-11-21	MAPR-24808: Spark version was moved to v2.0.1 and Scala to v2.11.

Mahout 0.12.0-1609 Release Notes

last-updated: Apr 11, 2022

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Mahout component included in the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Mahout home page.

Version	0.12.0
Release Date	September 30, 2016
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/mahout/tree/0.12.0-mapr-1609
Package Names	• `mapr-mahout-0.12.0.201609271030-1.noarch.rpm` • `mapr-mahout_0.12.0.201609271030_all.deb`

New in this Release

This release of Mahout 0.12.0 includes backports of all the patches included in Apache Mahout 0.12.1 and 0.12.2. For details on the fixes available in the open source version of this component, see changelog for Apache Mahout 0.12.1 and Apache Mahout 0.12.2.

Mahout 0.12.0-1605 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Mahout component included in the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Mahout home page.

Version	0.12.0
Release Date	June 6, 2016
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/mahout/tree/0.12.0-mapr-1605
Package Names	• mapr-mahout-0.12.0.201605311533-1.noarch.rpm • mapr-mahout_0.12.0.201605311533_all.deb

New in This Release

This is the initial release of version 0.12.0 of Mahout for the MapR Distribution for Hadoop. For more information, see the Release Notes.

Mahout 0.11.0 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following Mahout 0.11.0 component release notes are included in the MapR distribution for Apache Hadoop.

Mahout 0.11.0-1604 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Mahout component included in the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Mahout home page.

Mahout Version	0.11.0
Release Date	May 4, 2016
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/mahout/tree/0.11.0-mapr-1604
Package Names	• mapr-mahout-0.11.0.201604271224-1.noarch.rpm • mapr-mahout_0.11.0.201604271224_all.deb

New in this Release

All patches from Apache Mahout 0.11.2 have been backported into this release.

Fixes

This release from MapR includes the following fixes on the base Apache release. These fixes were back-ported from Mahout version 0.11.2. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
a075065	2016-03-23	Changed Spark version from Apache to MapR (Spark 1.5.2).
e43e3b3	2016-03-09	MAHOUT-1802: The optimizer, instead of generating checkpoints, now uses the cached checkpoints if available.
14a84d4	2016-03-09	MAHOUT-1801: Improved the speed of Sparse Matrix operations using FastUtil.
fdcb104	2016-03-09	MAHOUT-1640: Improved speed in vector benchmarks through better collections.
f2e763c	2016-03-08	Upgraded dependency to Spark 1.5.2.
91a2456	2016-03-08	MAHOUT-1800: Pared down overuse of classtag.
06f5993	2016-01-16	MAHOUT-1797: Fixed typos in `SPARK_ASSEMBLY_BIN`.
47e46b7	2016-01-06	MAHOUT-1785: Replaced (deprecated) `spark.kryoserializer.buffer.mb` parameter in the Spark configuration with `spark.kryoserializer.buffer`.

Mahout 0.11.0-1601 Release Notes

last-updated: Apr 11, 2022

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Mahout component included in the MapR Distribution for Apache Hadoop.

Mahout Version	0.11.0
Release Date	February 1, 2016
Source on GitHub	https://github.com/mapr/mahout/tree/0.11.0-mapr-1601
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-mahout-0.11.0.201601281739-1.noarch.rpm • mapr-mahout_0.11.0.201601281739_all.deb

New in this Release

This release of Apache Mahout includes the following behavior changes that are specific to MapR:

- All patches from Apache Mahout 0.11.1 have been backported into this release.
- Includes support for Spark 1.4.1 and Spark 1.5.2.

For details on the features available in the open source version of this component, see Apache Mahout Release Notes and the Apache Mahout homepage.

Note: When using Mahout-Samsara, the cluster requires Spark 1.3.1.

Mahout 0.11.0-1509 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Version	0.11.0
Release Date	Oct 5, 2015
Source on GitHub	https://github.com/mapr/mahout/tree/0.11.0-mapr-1509
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-mahout-0.11.0.201509301721-1.noarch.rpm • mapr-mahout_0.11.0.201509301721_all.deb

New in this Release

For details on the features available in the open source version of this component, see Apache Mahout Release Notes and the Apache Mahout homepage.

Important: When using Mahout-Samsara, the cluster requires Spark 1.3.1.

Fixes

This release from MapR includes the following fixes on the base Apache release. These fixes were back-ported from Mahout version 0.10.1 For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
078edc1	2015-09-10	MAPR-20481: Mahout Spark job no longer fails after switching the cluster MapReduce mode to classic mode.

Mahout 0.10.0 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following Mahout 0.10.0 component release notes are included in the MapR distribution for Apache Hadoop.

Mahout 0.10.0-1507 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the Mahout component included in the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Mahout homepage.

Version	0.10.0
Release Date	August 5, 2015
Source on GitHub	https://github.com/mapr/mahout.git
GitHub Release Tag	0.10-mapr-1507
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

Fixes

This release from MapR includes the following fixes on the base Apache release. These fixes were back-ported from Mahout version 0.10.1 For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
c9ee4f3	2015-06-10	MAHOUT-1704: Minimized the dependency jar for h2o so that it only includes necessary run-time classes.
d880b7a	2015-06-10	MAHOUT-1697:Math-scala and spark module docs are no longer packaged under the wrong path in the bin distribution archive.
358620d	2015-06-10	MAHOUT-1696: QRDecomposition.solve(...) no longer returns incorrect Matrix types.
e6d7525	2015-06-10	MAHOUT-1690: Resolved the issue where vector dumper flags are expecting arguments.

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
7907998	2015-06-10	MAHOUT-1693: FunctionalMatrixView no longer materializes row vectors when it is run in scala shell.

Mahout 0.10.0-1505 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the Mahout component included in the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Mahout homepage.

Version	0.10.0
Release Date	June 2, 2015
Source on GitHub	https://github.com/mapr/mahout.git
GitHub Release Tag	0.10-mapr-1505
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

New in This Release

This is the initial release of version 0.10.0 of Mahout for the MapR Distribution for Hadoop.

Fixes

This release from MapR includes the following fixes on the base Apache release. These fixes were back-ported from Mahout version 0.10.1 For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
ef5e912	2015-Apr-20	$MAHOUT_HOME and $SPARK_HOME are now exported automatically.

MapR Event Store For Apache Kafka Client Release Notes

The release notes for MapR Event Store For Apache Kafka clients included in the MapR Converged Data Platform.

MapR Event Store For Apache Kafka C Client 0.11.3 - 1803 Release Notes

Release notes for the MapR Event Store For Apache Kafka C client included in the MapR Converged Data Platform. The notes below relate specifically to the MapR Converged Data Platform.

Version	0.11.3
Release Date	March 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

This is a release of the MapR Event Store For Apache Kafka C Client for EEP 5.0 (and above) that suppported by MapR cluster version 6.0.1 (and above). This C Client is is a binding for librdkafka 0.11.3.

Fixes

- N/A

Known Issues and Limitations

none

Resolved Issues

None.

MapR Event Store For Apache Kafka Python Client 0.11.3 - 1803 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for MapR Event Store For Apache Kafka Python Client as of EEP 5.0 or above.

Release notes for the MapR Event Store For Apache Kafka Python client included in the MapR Converged Data Platform. The notes below relate specifically to the MapR Converged Data Platform. You can use MapR Event Store For Apache Kafka Python Client EEP 5.0 (and above) on MapR cluster version 6.0.1 (and above).

Version	0.11.3
Release Date	March 2018
Source on GitHub	
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

This is a release of the MapR Event Store For Apache Kafka Python Client for EEP 5.0 (and above) that suppported by MapR cluster version 6.0.1 (and above). This Python Client is is a binding for librdkafka 0.11.3.

Fixes

- N/A

Known Issues and Limitations

- You cannot use the MapR Installer to install the MapR Event Store For Apache Kafka Python Client. To install the MapR Event Store For Apache Kafka Python Client, use pip to manually install the package. See Installing MapR Event Store For Apache Kafka Python Client on page 198 for more information.

MapR Event Store For Apache Kafka C#/.NET 0.11.3 - 1803 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for MapR Event Store For Apache Kafka C#/.NET Client as of EEP 5.0 or above.

Release notes for the MapR Event Store For Apache Kafka C#/.NET client included in the MapR Converged Data Platform. The notes below relate specifically to the MapR Converged Data Platform. You can use MapR Event Store For Apache Kafka Python Client EEP 5.0 (and above) on MapR cluster version 6.0.1 (and above).

last-updated: Apr 11, 2022

Version	0.11.3
Release Date	March 2018
Source on GitHub	
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

This is a new release of the MapR Event Store For Apache Kafka C#/.NET Client for EEP 5.0 (and above) that suppported by MapR cluster version 6.0.1 (and above). This C#/.NET Client is a binding for librdkafka 0.11.3.

Fixes

- N/A

Known Issues and Limitations

- You cannot use the MapR Installer to install the MapR Event Store For Apache Kafka C#/.NET Client. See Installing MapR Event Store For Apache Kafka C#/.NET Client on page 200 for more information.

MapR Event Store For Apache Kafka C Client 0.9.1 - 1703 Release Notes

Release notes for the MapR Event Store For Apache Kafka C client included in the MapR Converged Data Platform. The notes below relate specifically to the MapR Converged Data Platform.

Version	0.9.1
Release Date	April 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

This is the initial release of the MapR Event Store For Apache Kafka C Client. You can use MapR Event Store For Apache Kafka C Client on MapR cluster version 5.2.1 and above.

Fixes

- Fix for bug 778 (Double free when using "topic.blacklist" setting) has been backported into this release.

Known Issues and Limitations

- **MapR-26629:**Application memory usage may be high due to fragmentation when Glibc Malloc is in use. In this case, consider tuning the MAX_MALLOC_ARENA environment variable or the M_ARENA_MAX tunable to a lower value, such as 1. When the value is 1, the amount of fragmentation decreases and the memory usage generally decreases by 3 or 4 times as well.
- **MapR-26602:**Kafka script `bin/kafka-consumer-groups.sh` cannot be used to obtain the consumer group list.
- **MapR-26331:**MapR Event Store For Apache Kafka C Client commits when "enable.auto.commit"= true and "auto.commit.interval.ms"=0. Apache librdkafka, does not commit in this case.

- **MapR-26279:**When a producer application creates multiple threads using pthread_create() and also calls rd_kafka_destroy(), the Streams C client logs the following error:

  ```
  rdkafka.c:527:rd_kafka_destroy_app: assert: !*"failed to join main
  thread"
  ```

  This issue also occurs with Apache librdkafka 0.9.1.

Resolved Issues

None.

MapR Event Store For Apache Kafka Python Client 0.9.2-1707 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for MapR Event Store For Apache Kafka Python Client 0.9.2-1707.

Release notes for the MapR Event Store For Apache Kafka Python client included in the MapR Converged Data Platform. The notes below relate specifically to the MapR Converged Data Platform.

Version	0.9.2
Release Date	August 2017
Source on GitHub	https://github.com/mapr/confluent-kafka-python/tree/0.9.2-mapr-1707
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

There are no new features in this release.

You can use MapR Event Store For Apache Kafka Python Client on MapR cluster version 5.2.1 and above.

Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
ee1add2	Jun 2, 2017	MAPR-KAFKA-12 Change package markings

Known Issues and Limitations

- You cannot use the MapR Installer to install the MapR Event Store For Apache Kafka Python Client. To install the MapR Event Store For Apache Kafka Python Client, use pip to manually install the package. See the MapR Installation documentation.

MapR Event Store For Apache Kafka Python Client 0.9.2 - 1703 Release Notes

Release notes for the MapR Event Store For Apache Kafka Python client included in the MapR Converged Data Platform. The notes below relate specifically to the MapR Converged Data Platform.

Version	0.9.2
Release Date	April 2017
Source on GitHub	https://github.com/mapr/confluent-kafka-python/tree/0.9.2-mapr-1703

MapR Version Interoperability	See the EEP Components and OS Support on page 5536.
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

This is the initial release of the MapR Event Store For Apache Kafka Python Client. You can use MapR Event Store For Apache Kafka Python Client on MapR cluster version 5.2.1 and above.

Fixes

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
96dc364	2017-3-23	MAPR-26494: Renamed package and refactored code.

Known Issues and Limitations

- You cannot use the MapR Installer to install the MapR Event Store For Apache Kafka Python Client. To install the MapR Event Store For Apache Kafka Python Client, use pip to manually install the package. See the MapR Installation documentation.

MapR Event Store For Apache Kafka Tools Release Notes

The release notes for MapR Event Store For Apache Kafka tools included in the MapR Converged Data Platform.

Kafka Streams Release Notes

The release notes for the Kafka Streams component included in the MapR Converged Data Platform contains notes specific to MapR only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Kafka Streams 2.6.1.100 - 2201 (EEP 8.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka. See Apache Kafka 2.6.1 release notes or the Apache Kafka Streams homepage for more information.

Version	2.6.1.100
Release Date	January 2022
HPE Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/kafka
GitHub Release Tag	2.6.1.100-eep-810
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Kafka Streams 2.6.1.100 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

- CVE fixes
- Bug fixes

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
0d14450	2022-01-25	MAPR-KAFKA-839 Updated log4j version to 1.3.1-mapr
20b0965	2022-01-12	MAPR-KAFKA-771 Make ListConsumerGroupsResult constructor accessible from outside the package.
cabe9f4	2021-12-16	MAPR-KAFKA-826 Added force override of log4j version
6b76e00	2021-12-15	MAPR-KAFKA-826 Updated log4j version
f59ab85	2021-11-19	MAPR-KAFKA-800 Skip already existing stream creation
17e8dbe	2021-11-19	MAPR-KAFKA-804 Netty CVE for kafka components
022472e	2021-11-16	MAPR-KAFKA-786 Added publishing of kafka-streams-test-utils
7a256fd	2021-11-15	MAPR-KAFKA-799 Backport KAFKA-12211 NoSuchFileException will be thrown if hasPersistentStores is false when creating stateDir
8e73665	2021-11-11	MAPR-KAFKA-796 Update dependencies versions for MEP-8.1
6f93f7e	2021-11-09	MAPR-KAFKA-793 KafkaProducer throws NPE to spring-kafka (additional commit)
44b509a	2021-11-08	MAPR-KAFKA-793 KafkaProducer throws NPE to spring-kafka

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- Transactions are not supported.
- Pattern subscription is not supported.
- The application reset tool hangs if it runs when the Kafka Streams application is running.
- The application reset tool may throw a Null Pointer Exception when the date or duration parameter is used.
- The application reset tool does not reset to intermediate offset if the topic has multiple partitions.

- MAPR-KAFKA-581: Stream hangs in rebalancing state. The workaround is to set a larger timeout. This issue is caused by MS-915: “MapR Stream application hangs inside cycle”

Resolved Issues

- None.

Kafka Streams 2.6.1.0 - 2110 (EEP 8.0.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka. See Apache Kafka 2.6.1 release notes or the Apache Kafka Streams homepage for more information.

Version	2.6.1.0
Release Date	October 2021
HPE Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/kafka
GitHub Release Tag	2.6.1.0-eep-800
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Kafka Streams 2.6.1.0 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

- Kafka-664: When a Consumer application (in HPE Ezmeral Data Fabric Event Data Streams) calls `consumer.poll()`, the Consumer does not read any data from the topic if the timeout (`request.timeout.ms`) is set to 0. In previous releases, Consumers read one message.

If you plan to upgrade from Kafka 2.1.1 to 2.6.1, you may want to review Changes in Kafka 2.6.1 on page 3863.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
`360bf7ab, b0507115`	2021-10-11	KAFKA-783 Add log4j properties for MM2.
`c9a89245`	2021-10-06	KAFKA-781 Avoid overwriting task user name.
`e3f52c9394`	2021-09-25	KAFKA-770 Configuration topic which stores offsets should store the topics' names without Mapr Stream name.
`69565a337e`	2021-09-23	KAFKA-761 Seek to beginning of Mapr partitions before reading logs from configuration's topics.
`343c082fbc`	2021-09-16	KAFKA-762 NPE while starting kafka-connect service on core 7.0.0
`ba61df9ded`	2021-07-17	KAFKA-654 Make MirrorMaker 2 fully functional with Mapr Streams

bd0c93efd2	2021-09-08	KAFKA-757 mapr-security-web jar should be taken from the cluster
6ad6204cdf	2021-09-05	KAFKA-758 Update the maven artifact version strings to eep
20f177c8fb	2021-09-01	KAFKA-752 KafkaConsumer.pause() throws NoSuchElementException
cc0c8eaf13	2021-08-25	KAFKA-751 Protobuf and JsonSchema formats were added to Schema Registry 6.0.0
290bf71f61	2021-08-19	KAFKA-746 Remove workaround implemented due to the difference in work of poll(0) between Mapr and Apache Kafka.
32cfddfd2a	2021-08-18	KAFKA-725 Update hadoop dependency
9910062bb7	2021-08-17	KAFKA-745 Update Jackson dependencies
61e6625b43	2021-08-13	KAFKA-686 Service verifier was added to Kafka Connect
ecc67384f7	2021-08-10	KAFKA-680 Jetty CVE for kafka components
8fb65fa77b	2021-07-08	KAFKA-724 Avoid appearing NPE when using MirrorMaker
ca2adcf2d9	2021-07-02	KAFKA-654 The value of property AUTO_COMMIT_INTERVAL_MS_CONFIG has Long type
e326b73eb2	2021-05-25	KAFKA-714 mapr-streams dependency was replaced with kafka-eventstreams
29e9fe9f14	2021-05-24	KAFKA-715 Hadoop dependency was updated to 2.7.5.100-mapr-720-SNAPSHOT
e6a7e80417	2021-03-03	KAFKA-681 Avoid appearing NPE when using MirrorMaker
00b0b4b438	2021-02-24	KAFKA-679 Hadoop version was changed to 2.7.4.0-mapr-710
b02d49328f	2021-02-23	KAFKA-676 Workaround for not implemented method MarlinAdminClientImpl.describeConfigs
939b524487	2021-02-19	KAFKA-650 Kafka connect JDBC code base was upgraded to version 10.0.1
6dff118ef9	2021-02-15	KAFKA-649 Kafka connect code base was upgraded to version 10.0.0
18c41383c0	2021-02-11	KAFKA-670 Avoid potential use of Apache methods in Mapr Kafka 2.6
210c161329	2021-01-14	KAFKA-660 TaskManager should be inited before adding records to tasks.

last-updated: Apr 11, 2022

c2898ea40c	2021-01-21	KAFKA-666 Synchronized blocks with monitor object 'taskmanager' were removed.
0882ab8b3a	2021-01-12	KAFKA-662 InternalStreamCompacted attribute was removed from StoreChangelogReader
5bd960288c	2021-01-05	KAFKA-659 Avoid appearing NullPointerException

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- Transactions are not supported.
- Pattern subscription is not supported.
- The application reset tool hangs if it runs when the Kafka Streams application is running.
- The application reset tool may throw a Null Pointer Exception when the date or duration parameter is used.
- The application reset tool does not reset to intermediate offset if the topic has multiple partitions.
- MAPR-KAFKA-581: Stream hangs in rebalancing state. The workaround is to set a larger timeout. This issue is caused by MS-915: “MapR Stream application hangs inside cycle”

Resolved Issues

- None.

Kafka Streams 2.1.1.200 - 2104 (MEP 7.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka. See Apache Kafka 2.1.1 Release Notes or the Apache Kafka Streams homepage for more information.

Version	2.1.1.200
Release Date	April 2021
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka
GitHub Release Tag	2.1.1.200-mapr-710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka Streams 2.1.1.200 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
4d9def8021	2021-04-12	MAPR-KAFKA-696 Partition is returned in consumer.assignment(), but seekToEnd fails with "No current assignment for partition"
651c04c420	2021-03-31	MAPR-KAFKA-686 Service verifier was added to Kafka Connect
23e005ac30	2021-04-01	MAPR-KAFKA-692 Update Hadoop dependency to 2.7.5.0-mapr-710-SNAPSHOT
0af5f9d9d4	2021-03-30	MAPR-KAFKA-685 CVE-2018-10237,CVE-2020-8908 vulnerabilities in Kafka
83081395	2021-03-03	MAPR-KAFKA-681 Avoid appearing NPE when using MirrorMaker

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- Pattern subscription is not supported.
- The application reset tool hangs if it runs when the Kafka Streams application is running.
- The application reset tool may throw a Null Pointer Exception when the date or duration parameter is used.
- The application reset tool does not reset to intermediate offset if the topic has multiple partitions.
- MAPR-KAFKA-581: Stream hangs in rebalancing state. The workaround is to set a larger timeout. This issue is caused by MS-915: “MapR Stream application hangs inside cycle”

Resolved Issues

- None.

Kafka Streams 2.1.1.100 - 2101 (MEP 7.0.1) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka. See Apache Kafka 2.1.1 Release Notes or the Apache Kafka Streams homepage for more information.

Version	2.1.1.100
Release Date	January 2021
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka
GitHub Release Tag	2.1.1.100-mapr-701
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka Streams 2.1.1.100 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
87b900a79	2021-01-09	MAPR-KAFKA-661 Bump jetty to v9.4.35
a25b751d2	2020-11-09	MAPR-KAFKA-639 ClassCastException: Non-string value for key task.user: null

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- Pattern subscription is not supported.
- The application reset tool hangs if it runs when the Kafka Streams application is running.
- The application reset tool may throw a Null Pointer Exception when the date or duration parameter is used.
- The application reset tool does not reset to intermediate offset if the topic has multiple partitions.
- MAPR-KAFKA-581: Stream hangs in rebalancing state. The workaround is to set a larger timeout. This issue is caused by MS-915: “MapR Stream application hangs inside cycle”

Resolved Issues

- None.

Kafka Streams 2.1.1.0 - 2009 (MEP 7.0.0) Release Notes

The following notes relate specifically to the MapR distribution for Apache Kafka. See Apache Kafka 2.1.1 Release Notes or the Apache Kafka Streams homepage for more information.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	2.1.1.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka
GitHub Release Tag	2.1.1.0-mapr-700
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

N/A

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
fa80ee2	2019-03-30	MAPR-KAFKA-280 mapr version is changed to 6.2.0-mapr-SNAPSHOT
9a9e3ce	2019-05-15	MAPR-KAFKA-331 Kafka uses 2.7.4 version of hadoop.
4ffa437	2019-07-05	MAPR-KAFKA-399 Jetty version was updated at Kafka to latest 9.4 (as of 06/28 - it is 9.4.19.v20190610).
941f2b8	2020-04-06	MAPR-KAFKA-564 Update gradle version and fix config files in order to build with java 11
7a51ece	2020-07-09	MAPR-KAFKA-582 Update ZK to v3.5.6
33152eb	2020-07-09	MAPR-KAFKA-588 Create only absent internal streams for KStreams

Known Issues and Limitations

- Pattern subscription is not supported.
- The Application Reset tool hangs if it runs when the Kafka Streams application is running.
- The Application Reset tool may throw a Null Pointer Exception if the date or duration parameter is used.
- The Application Reset tool does not reset to intermediate offset if the topic has multiple partitions.
- MAPR-KAFKA-581: Stream hangs in rebalancing state; caused by MS-915: MapR Stream application hangs inside cycle. Workaround: Increase the value of the `streams.rpc.timeout.ms` parameter, as described in Enabling Soft Mount and Setting the Timeout on page 420 and Configuration Parameters on page 2772.

Resolved Issues

- None.

Kafka Streams 1.1.1 - 2201 (EEP 6.3.6) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka. See Apache Kafka 1.1 ReleaseNotes or the Apache Kafka Streams homepage for more information.

Version	1.1.1
Release Date	January 2022
MapR Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka
GitHub Release Tag	1.1.1-mapr-2201
Maven Artifacts	https://repository.mapr.com/maven/

last-updated: Apr 11, 2022

Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka Streams 1.1.1 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

- CVE fixes
- Bug fixes

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
07937bf	2022-01-10	MAPR-KAFKA-835 Updated log4jv2 version
7dbed35	2021-04-13	MAPR-KAFKA-699 Hadoop mapreduce client core jar was added to the classpath for executing connectors
cdef034	2021-04-12	MAPR-KAFKA-696 Partition is returned in consumer.assignment(), but seekToEnd fails with "No current assignment for partition"

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- Pattern subscription is not supported.
- The application reset tool hangs if it runs when the Kafka Streams application is running.
- The application reset tool may throw a Null Pointer Exception if the date or duration parameter is used.
- The application reset tool does not reset to intermediate offset if the topic has multiple partitions.
- MAPR-KAFKA-581: Stream hangs in rebalancing state. The workaround is to set a larger timeout. This issue is caused by MS-915: MapR Stream application hangs inside cycle

Resolved Issues

- None.

Kafka Streams 1.1.1 - 2104 (MEP 6.3.4) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka. See Apache Kafka 1.1 ReleaseNotes or the Apache Kafka Streams homepage for more information.

Version	1.1.1
Release Date	April 2021
MapR Version Interoperability	See MEP Components and OS Support

Source on GitHub	https://github.com/mapr/kafka
GitHub Release Tag	1.1.1-mapr-2104
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka Streams 1.1.1 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
7a7afdda1b	2021-03-30	MAPR-KAFKA-685 CVE-2018-10237,CVE-2020-8908 vulnerabilities
d63fe60ca4	2021-03-03	MAPR-KAFKA-681 Avoid appearing NPE when using MirrorMaker

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- Pattern subscription is not supported.
- The application reset tool hangs if it runs when the Kafka Streams application is running.
- The application reset tool may throw a Null Pointer Exception if the date or duration parameter is used.
- The application reset tool does not reset to intermediate offset if the topic has multiple partitions.
- MAPR-KAFKA-581: Stream hangs in rebalancing state. The workaround is to set a larger timeout. This issue is caused by MS-915: MapR Stream application hangs inside cycle

Resolved Issues

- None.

Kafka Streams 1.1.1 - 2101 (MEP 6.3.2) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka. See Apache Kafka 1.1 ReleaseNotes or the Apache Kafka Streams homepage for more information.

Version	1.1.1
Release Date	January 2021
MapR Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka
GitHub Release Tag	1.1.1-mapr-2101

last-updated: Apr 11, 2022

Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka Streams 1.1.1 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
264d8d8dd	2020-12-24	MAPR-KAFKA-657 Update rest-utils version
811794f99	2020-12-15	MAPR-KAFKA-657 Fix SSLUtils after updating jetty
0c1497b10	2020-12-11	MAPR-KAFKA-383 Fix ACE validation
a27fef715	2020-12-10	MAPR-KAFKA-655 Cherry-pick commits from MAPR-KAFKA-392
9e9d80398	2020-12-10	MAPR-KAFKA-655 Cherry-pick commits from MAPR-KAFKA-387
df812b11c	2020-12-01	EMEP-85 Jetty version was changed to 9.4.35.v20201120
ae4667858	2020-11-09	MAPR-KAFKA-639 ClassCastException: Non-string value for key task.user: null

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- Pattern subscription is not supported.
- The application reset tool hangs if it runs when the Kafka Streams application is running.
- The application reset tool may throw a Null Pointer Exception if the date or duration parameter is used.
- The application reset tool does not reset to intermediate offset if the topic has multiple partitions.
- MAPR-KAFKA-581: Stream hangs in rebalancing state. The workaround is to set a larger timeout. This issue is caused by MS-915: “MapR Stream application hangs inside cycle”

Resolved Issues

- None.

Kafka Streams 1.1 - 2009 (MEP 6.3.1) Release Notes

The following notes relate specifically to the MapR distribution for Apache Kafka. See Apache Kafka 1.1 ReleaseNotes or the Apache Kafka Streams homepage for more information.

last-updated: Apr 11, 2022

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.1
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka
GitHub Release Tag	1.1-mapr-2009
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
08f5ca8	2020-03-05	MAPR-KAFKA-555 Ability to disable log compaction was added
c18c4a2	2020-05-04	MAPR-KAFKA-577 bcprov-jdk15on version was upgraded to 1.60
24b542f	2020-05-05	MAPR-KAFKA-569 Jackson dependencies was updated to 2.11.0
75dbe04	2020-07-10	MAPR-KAFKA-616 Server Banner is disabled in in HTTP Responses

Known Issues and Limitations

- Pattern subscription is not supported.
- The Application Reset tool hangs if it runs when the Kafka Streams application is running.
- The Application Reset tool may throw a Null Pointer Exception if the date or duration parameter is used.
- The Application Reset tool does not reset to intermediate offset if the topic has multiple partitions.
- MAPR-KAFKA-581: Stream hangs in rebalancing state; caused by MS-915: MapR Stream application hangs inside cycle. Workaround: Increase the value of the `streams.rpc.timeout.ms` parameter, as described in Enabling Soft Mount and Setting the Timeout on page 420 and Configuration Parameters on page 2772.

Kafka Streams 1.1-1912 Release Notes

The following notes relate specifically to the MapR distribution for Apache Kafka. See Apache Kafka 1.1 Release Notes or the Apache Kafka Streams homepage for more information.

last-updated: Apr 11, 2022

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.1
Release Date	December 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka/tree/1.1.1-mapr-1912
GitHub Release Tag	1.1-mapr-1912
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

N/A

Fixes

N/A

Known Issues and Limitations

- Pattern subscription is not supported.
- Application Reset Tool hangs if it runs when Kafka Streams application is running.
- Application Reset Tool may throw Null Pointer Exception if date or duration parameter is used.
- Application Reset Tool doesn't reset to intermediate offset if topic has multiple partitions.

Resolved Issues

- KAFKA-505: kafka-connect configure.sh processes --EC option incorrectly.
- MS-911: MEP6 with Kafka 1.1.1 fails interact through SSL.

Kafka Streams 1.1-1808 Release Notes

The following notes relate specifically to the MapR distribution for Apache Kafka. See Apache Kafka 1.1 Release Notes or the Apache Kafka Streams homepage for more information.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.1
Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka/tree/1.1.1-mapr-1808
GitHub Release Tag	1.1-mapr-1808

Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- This is the first release of MapR Kafka Streams.

  The MapR Kafka Streams is included in EEP repositories beginning with MEP-6.0.0. Kafka Streams is a Java library for building streaming applications and microservices, specifically ones that transform input MapR Event Store For Apache Kafka topics into output MapR Event Store For Apache Kafka topics.

  Feature Support

  - Application Reset Tool
  - Secure internal topic creation

Fixes

N/A

Known Issues and Limitations

- Pattern subscription is not supported.
- Application Reset Tool hangs if it runs when Kafka Streams application is running.
- Application Reset Tool may throw Null Pointer Exception if date or duration parameter is used.
- Application Reset Tool doesn't reset to intermediate offset if topic has multiple partitions.

Resolved Issues

- None

KSQL Release Notes

The release notes for the KSQL component included in the MapR Converged Data Platform contains notes specific to MapR only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

KSQL 6.0.0.100 - 2201 (EEP 8.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	6.0.0.100
Release Date	January 2022
HPE Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/ksql
GitHub Release Tag	6.0.0.100-eep-810
Maven Artifacts	https://repository.mapr.com/maven/

last-updated: Apr 11, 2022

Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

KSQL 6.0.0.100 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

- Federal Information Processing Standards (FIPS) support (valid for core 7.0.0 and later). .
- CVE fixes.
- Bug fixes.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
59d3a86	2022-01-26	MAPR-KAFKA-839 Updated log4j version to 1.3.1-mapr
8ed2a69	2022-01-25	MAPR-KAFKA-837 Avoid IndexOutOfBounds error
e475458	2022-01-25	MAPR-KAFKA-837 Update vert.x version to the one which is works with netty 4.1.69.Final
3036b52	2022-01-19	MAPR-KAFKA-836 Add providers before vert.x starts to read security properties
a69ebf6	2021-12-21	MAPR-KAFKA-792 FIPS support was added with changes in vertx-core
7f7bce7	2021-12-17	MAPR-KAFKA-803 Changed kafka-streams-test-utils version to eep snapshot
a1edebe	2021-12-01	MAPR-KAFKA-808 Kafka-ksql CVE fixes (additional fix)
2f50a62	2021-11-25	MAPR-KAFKA-808 Kafka-ksql CVE fixes
0446c5e	2021-11-24	MAPR-KAFKA-800 Skip already existing stream creation by KSQL
19ffc24	2021-11-15	MAPR-KAFKA-798 KSQL cli logs contain errors about metrics submission
ea4cd7a	2021-11-11	MAPR-KAFKA-796 Update dependencies versions
82d7c5f	2021-10-27	MAPR-KAFKA-784 Add security java options if FIPS mode is enabled
cf49faf	2021-10-15	MAPR-KAFKA-782 All eep jars that can be taken from /opt/mapr/kafka was excluded

For complete details, refer to the commit log for this project in Github.

Known Issues and Limitations

- You cannot upgrade KSQL from 4.x to 5.x/6.x versions; you must uninstall KSQL 4.x and then install the newer version.
- Concurrent queries on a table can result in a null pointer exception.
- The SHOW TOPICS command does not print information about active consumers and consumer groups.
- MAPR-KAFKA-437: Dropping streams/tables may take up to five minutes. This issue is caused by MS-915: “MapR Stream application hangs inside cycle”
- MAPR-KAFKA-427: KSQL server periodically responds with a 403 code only. This issue is caused by MS-915: “MapR Stream application hangs inside cycle”

Resolved Issues

- None.

KSQL 6.0.0.0 - 2110 (EEP 8.0.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	6.0.0.0
Release Date	October 2021
HPE Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/ksql
GitHub Release Tag	6.0.0.0-eep-800
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

KSQL 6.0.0.0 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

- None

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
ae254aca1c	2021-09-08	MAPR-KAFKA-757 mapr-security-web and maprdb jars should be taken from the cluster
5da53e1e6b	2021-09-05	MAPR-KAFKA-758 Update the maven artifact version strings to eep
1f57181607	2021-08-31	MAPR-KAFKA-749 Incorrect access mode for scripts in bin directory
51fd694f20	2021-08-31	MAPR-KAFKA-737 Implement authentication.cookie.expiration property support

3d16826412	2021-08-26	MAPR-KAFKA-750 Cannot create table because producer is clossed
6228621cf4	2021-08-19	MAPR-KAFKA-741 Make verify_service executable
608c4ec5e5	2021-08-18	MAPR-KAFKA-745 Update Jackson dependencies
95381a5e8b	2021-08-17	MAPR-KAFKA-725 Update hadoop dependency version
c99c6f314e	2021-08-16	MAPR-KAFKA-744 Vulnerabilities in http-client
4cdd4d68c0	2021-08-13	MAPR-KAFKA-741 Add service verifier to Kafka KSQL
428cba4c9f	2021-08-11	MAPR-KAFKA-680 Jetty CVE for kafka components
553dba7156	2021-08-03	MAPR-KAFKA-736 KSQL start fails in non-interactive mode
a244d69947	2021-08-02	MAPR-KAFKA-735 KSQL server crashes if truststore is in not default location

For complete details, refer to the commit log for this project in Github.

Known Issues and Limitations

- You cannot upgrade KSQL from 4.x to 5.x/6.x versions; you must uninstall KSQL 4.x and then install the newer version.
- Concurrent queries on a table can result in a null pointer exception.
- The SHOW TOPICS command does not print information about active consumers and consumer groups.
- MAPR-KAFKA-437: Dropping streams/tables may take up to five minutes. This issue is caused by MS-915: “MapR Stream application hangs inside cycle”
- MAPR-KAFKA-427: KSQL server periodically responds with a 403 code only. This issue is caused by MS-915: “MapR Stream application hangs inside cycle”

Resolved Issues

- None.

KSQL 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	5.1.2.200
Release Date	April 2021
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/ksql
GitHub Release Tag	5.1.2.200-mapr-710
Maven Artifacts	https://repository.mapr.com/maven/

Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

KSQL 5.1.2.200 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- Service verifier

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
58c7cf9912	2021-04-05	MAPR-KAFKA-690 Add service verifier to Kafka KSQL
b6ca1ab18d	2021-04-01	MAPR-KAFKA-697 Jersey 1.19 was excluded from the project
da5dd67088	2021-03-19	MAPR-KAFKA-692 Update Hadoop dependency to 2.7.5.0-mapr-710-SNAPSHOT

For complete details, refer to the commit log for this project in Github.

Known Issues and Limitations

- The SHOW TOPICS command does not print information about active consumers and consumer groups.
- MAPR-KAFKA-437: Dropping streams/tables may take up to five minutes. This issue is caused by MS-915: “MapR Stream application hangs inside cycle”
- MAPR-KAFKA-427: KSQL server periodically responds with a 403 code only. This issue is caused by MS-915: “MapR Stream application hangs inside cycle”

Resolved Issues

- None.

KSQL 5.1.2.100 - 2101 (MEP 7.0.1) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	5.1.2.100
Release Date	January 2021
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/ksql
GitHub Release Tag	5.1.2.100-mapr-701
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

last-updated: Apr 11, 2022

New in This Release

KSQL 5.1.2.100 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
e5dd2e9f	2021-01-08	KAFKA-661 Bump jetty to v9.4.35

For complete details, refer to the commit log for this project in Github.

Known Issues and Limitations

- The SHOW TOPICS command does not print information about active consumers and consumer groups.
- MAPR-KAFKA-437: Dropping streams/tables may take up to five minutes. This issue is caused by MS-915: “MapR Stream application hangs inside cycle”
- MAPR-KAFKA-427: KSQL server periodically responds with a 403 code only. This issue is caused by MS-915: “MapR Stream application hangs inside cycle”

Resolved Issues

- None.

KSQL 5.1.2.0 - 2009 (MEP 7.0.0) Release Notes

The following notes relate specifically to the KSQL component included in the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	5.1.2.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/ksql
GitHub Release Tag	5.1.2.0-mapr-700
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Additional security features.
- KSQL works with Avro format using Mapr Schema Registry.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
b855332	2019-03-19	MAPR-KAFKA-281 Symlink to /opt/mapr/ksql-$version/bin/ksql to run as a default application is created.
e05a761	2019-03-28	MAPR-KAFKA-255 Security (Basic, Kerberos and MapRSasl) is enabled on server side
835473f	2019-04-05	MAPR-KAFKA-256 Impersonation for KSQL was added.
e5bc4df	2019-04-08	MAPR-KAFKA-251 SSL/TLS for KSQL CLI
7d79754	2019-04-09	MAPR-KAFKA-253 Authenticate with entering username and hidden password
89565f2	2019-04-11	MAPR-KAFKA-252 Maprsasl for KSQL cli
5b7da76	2019-04-22	MAPR-KAFKA-305 Cookies authentication for KSQL
42536b1	2019-04-22	MAPR-KAFKA-194 Authentication for KSQL
1dc8734	2019-05-11	MAPR-KAFKA-331 KSQL uses 6.2 version of mapr.
1d69d2a	2019-05-17	MAPR-KAFKA-318 Authorization filter for KSQL.
d1189f0	2019-06-05	MAPR-KAFKA-360 Ksql client does not start when auth type is "basic" and user without ticket
e2636b9	2019-06-11	MAPR-KAFKA-377 Add separate configuration property to enable or disable authorization property
49af0ec	2019-06-19	MAPR-MAPR-KAFKA-382: Replace "authentication.method" property in ksql config with "authentication.enable"
82986b6	2019-07-05	MAPR-KAFKA-396 KSQL reads info about SR urls from zookeeper
f622cd8	2019-10-06	MAPR-CORE-327 Security headers and custom headers for KSQL
f063e37	2020-04-29	MAPR-KAFKA-564 Edit pom files to build ksql with java 11
f8ba9d9	2020-05-05	MAPR-KAFKA-573 Update version of commons-beanutils jar
ee957f9	2020-05-05	MAPR-KAFKA-582 Update ZK to v3.5.6

Known Issues and Limitations

- The `show topics` command doesn't print information about active consumers and consumer groups.

- MS-915 (MapR Stream application hangs inside cycle) causes the following issues:
  - MAPR-KAFKA-437: DROP statements run on streams or tables can delay for approximately five minutes.
  - MAPR-KAFKA-427: The KSQL server periodically responds with a 403 code.
    - Workaround: Increase the value of the `streams.rpc.timeout.ms` parameter, as described in Enabling Soft Mount and Setting the Timeout on page 420 and Configuration Parameters on page 2772.

Resolved Issues

- None

KSQL 4.1.1-2201 (EEP 6.3.6) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	4.1.1
Release Date	January 2022
HPE Version Interoperability	See EEP Components and OS Support.
Source on GitHub	https://github.com/mapr/ksql
GitHub Release Tag	4.1.1-mapr-2201
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for Ezmeral Ecosystem Packs (EEPs).

New in This Release

KSQL 4.1.1 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

- CVE fixes
- Bug fixes

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description
92024b7	2022-01-26	MAPR-KAFKA-839 Updated log4j version to 1.3.1-mapr
4a7f2c5	2021-12-21	MAPR-KAFKA-826 Updated log4j version
9b5a9c3	2020-12-11	MAPR-KAFKA-383 Set public read permissions for KSQL directories
c75b19b	2020-12-11	MAPR-KAFKA-656 Update jetty version
4c71935	2020-05-05	MAPR-KAFKA-573 Update versions of commons-*.jar
9e64cb0	2020-03-17	MAPR-KAFKA-562 Create health check for KSQL
279deab	2020-03-05	MAPR-KAFKA-555 Add option to control log compaction for KStreams
95cc6ca	2020-02-12	MAPR-KAFKA-542 Fix for class not found exception (versions of connect jars are calculated dinamically)

6f8cf8e	2018-11-29	MAPR-KAFKA-158 Copyright 2017 Confluent Inc message doesn't appear when starting up KSQL.
f220e4c	2018-10-17	MAPR-MS-513 Fix for configs overriding
7297843	2018-10-05	MAPR-MS-515 Mapr specific jars are taken from cluster.
d1d7b3e	2018-09-19	MAPR-MS-511 Fix for Error: File exists

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- The SHOW TOPICS command does not print information about active consumers and consumer groups.
- MAPR-KAFKA-437: Dropping streams/tables may take up to five minutes. This issue is caused by MS-915: “MapR Stream application hangs inside cycle.”
- MAPR-KAFKA-427: KSQL server periodically responds with a 403 code only. This issue is caused by MS-915: “MapR Stream application hangs inside cycle”

Resolved Issues

- None

KSQL 4.1.1-1808 Release Notes

The following notes relate specifically to the KSQL component include in the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.1.1
Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	n/a
GitHub Release Tag	4.1.1-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- This release is the first MapR release of KSQL for MapR Event Store For Apache Kafka.

  The MapR KSQL is included in EEP repositories beginning with MEP-6.0.0. KSQL is the streaming SQL engine that provides interactive/non-interactive SQL interface for stream processing on Kafka. This allows launching Kafka Streams applications with no need to write code in Java/Python.

  Feature Support

  - Secure creation of command store topic
  - Default stream

Fixes

N/A

Known Issues and Limitations

- The `show topics` command doesn't print information about active consumers and consumer groups.

Resolved Issues

- None

Kafka Connect Release Notes

The release notes for the Kafka Connect component included in the MapR Converged Data Platform contains notes specific to MapR only.

Note: To identify the EEP to which a specific release note belongs, seeEEP Release Notes on page 5658 . To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Kafka Connect HDFS 10.0.0.100 - 2201 (EEP 8.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	10.0.0.100
Release Date	January 2022
HPE Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/kafka-connect-hdfs
GitHub Release Tag	10.0.0.100-eep-810
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Kafka Connect HDFS 10.0.0.100 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

- CVE fixes

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
4e4f3cd	2021-12-31	MAPR-KAFKA-834 Updated log4jv2 version
8e5cd0d	2021-12-14	MAPR-KAFKA-824 CVE-2021-44228 - Log4j vulnerability in Kafka HDFS Con
d654f94	2021-11-25	MAPR-KAFKA-805 Compress-commons CVE for kafka components

2a4633e	2021-11-25	MAPR-KAFKA-805 Gson CVE for kafka components
ab56cff	2021-11-25	MAPR-KAFKA-805 Netty CVE for kafka components

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Connect HDFS 10.0.0.0 - 2110 (EEP 8.0.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	10.0.0.0
Release Date	October 2021
HPE Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/kafka-connect-hdfs
GitHub Release Tag	10.0.0.0-eep-800
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Kafka Connect HDFS 6.0.0.0 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

- None

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
df8d31b	2021-09-09	MAPR-KAFKA-704 Kafka Connect lib dir contains Kafka client jar
d0eb929	2021-09-07	MAPR-KAFKA-757 mapr-security-web and maprdb jars should be taken from the cluster
041943d	2021-09-05	MAPR-KAFKA-758 Update the maven artifact version strings to eep
cb558bd	2021-08-26	MAPR-KAFKA-751 Protobuf and Json Schema connect converters were added.
5ac9da4	2021-08-18	MAPR-KAFKA-745 Update Jackson v1 and v2 dependencies

28ab440	2021-08-16	MAPR-KAFKA-744 Vulnerabilities in http-client
f36b2e4	2021-08-11	MAPR-KAFKA-725 Update hadoop, hive, hbase dependencies
1fbf6c5	2021-05-25	MAPR-KAFKA-715 Hadoop dependency was updated to 2.7.5.100-mapr-720-SNAPSHOT
5472301	2021-02-25	MAPR-KAFKA-675 Skip checking the creation of mapr hadoop directory.
4afca95	2021-02-24	MAPR-KAFKA-678 hadoop-client and hadoop-yarn-client version should be 2.7.4.0-mapr

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Connect HDFS 5.1.2.200 - 2104 (EEP 7.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	5.1.2.200
Release Date	April 2021
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka-connect-hdfs
GitHub Release Tag	5.1.2.200-mapr-710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka Connect HDFS 5.1.2.200 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- Service verifier

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
3c82f9f	2021-04-02	MAPR-KAFKA-698 Update Hive dependency to 2.3.8-mapr-SNAPSHOT

173cd5c	2021-03-22	MAPR-KAFKA-692 Update Hadoop dependency to 2.7.5.0-mapr-710-SNAPSHOT

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Connect HDFS 5.1.2.100 - 2101 (EEP 7.0.1) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	5.1.2.100
Release Date	January 2021
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka-connect-hdfs
GitHub Release Tag	5.1.2.100-mapr-701
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka Connect HDFS 5.1.2.100 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
4e01a19	2021-01-18	MAPR-KAFKA-661 Change version of Hadoop dependencies and remove jetty notices
c0efe3b	2021-01-08	MAPR-KAFKA-661 Exclude jetty libs

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Connect HDFS 5.1.2.0 - 2009 (EEP 7.0.0) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	5.1.2.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka-connect-hdfs
GitHub Release Tag	5.1.2.0-mapr-700
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
7b90819	2020-05-05	MAPR-KAFKA-582 Update ZK to v3.5.6
b8e46ef	2020-06-04	MAPR-KAFKA-603 Update jacoco and spotbugs plugin
17a3f04	2020-06-04	MAPR-KAFKA-596 Netty-all version was upgraded to 4.1.42

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka Connect HDFS 4.1.0 - 2201 (EEP 6.3.6) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	4.1.0
Release Date	January 2022
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka-connect-jdbc
GitHub Release Tag	4.1.0-mapr-2201
Maven Artifacts	https://repository.mapr.com/maven/

Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka Connect HDFS 4.1.0 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

- CVE fixes
- Bug fixes

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
04a5cc4	2022-01-25	MAPR-KAFKA-839 Updatel log4j version to 1.3.1-mapr
ddae28f	2021-12-31	MAPR-KAFKA-834 Updated log4jv2 version
a45ddc7	2021-12-21	MAPR-KAFKA-826 Updated log4j version

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Connect HDFS 4.1.0 - 2104 (EEP 6.3.4) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	4.1.0
Release Date	April 2021
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka-connect-jdbc
GitHub Release Tag	4.1.0-mapr-2104
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka Connect HDFS 4.1.0 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- None

last-updated: Apr 11, 2022

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
3e6adec	2021-04-13	KAFKA-702 Hadoop version updated to 2.7.0-mapr-1808

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Connect HDFS 4.1.0 - 2101 (EEP 6.3.2) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hadoop.

Version	4.1.0
Release Date	January 2021
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka-connect-hdfs
GitHub Release Tag	4.1.0-mapr-2101
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka Connect HDFS 4.1.0 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
8a5bfa9	2020-12-24	MAPR-KAFKA-656 Update Hive version
f264feb	2019-04-19	MAPR-KAFKA-301: exclude non-mapr hadoop libs

For complete details, refer to the commit log for this project in Github.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Connect HDFS 4.1.0-1808 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.1.0
Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	n/a
GitHub Release Tag	4.1.0-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Added support of SSL encryption, impersonation, and PAM authentication for Kafka Connect REST API.
- Added support security by default for Kafka Connect.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
b76f3038	2018-05-25	KAFKA-102 HDFS Connector rewrites files in MapR filesystem after re-balancing
ce7c368d	2018-06-08	KAFKA-120 Build connect-hdfs with Hive v2.3

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka Connect HDFS 4.0.0-1808 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

last-updated: Apr 11, 2022

Version	4.0.0
Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	n/a
GitHub Release Tag	4.0.0-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release
Multiple issues were fixed.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
caaa7386	2018-03-26	KAFKA-88 Remove assignment before rebalancing
eb03bd0	2018-04-19	KAFKA-102 HDFS Connector rewrites files in MapR filesystem after re-balancing

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka Connect HDFS 4.0.0-1803 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	4.0.0
Release Date	March 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	n/a
GitHub Release Tag	4.0.0-mapr-1803
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release
Multiple issues were fixed.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
f46a3a2	Feb 9, 2018	KAFKA-72 NullPointerException after creating the HDFS connector
0afc78f	Feb 15, 2018	KAFKA-80 Error when create HDFS connector with ParquetFormat
6c04272	Feb 16, 2018	KAFKA-82 Remove jar that contains libMapRClient

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka Connect HDFS 2.0.1-1710 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	2.0.1
Release Date	November 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	n/a
GitHub Release Tag	2.0.1-mapr-1710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Kafka Connect HDFS 2.0.1-1710 introduces the following enhancements or MapR platform-specific behavior changes:

- You can use configure.sh to configure this component.

Fixes

Changes have been made to the mapr-kafka package and to the packaging process

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka Connect JDBC 10.0.1.100 - 2201 (EEP 8.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hadoop.

Version	10.0.1.100
Release Date	January 2022
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/kafka-connect-jdbc
GitHub Release Tag	10.0.1.100-eep-810
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

- CVE fixes

Fixes

This release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
aac690b	2021-11-22	MAPR-KAFKA-806 Gson CVE for kafka components
c573a05	2021-11-22	MAPR-KAFKA-806 PostgreSQL CVE for kafka components
68b2963	2021-11-22	MAPR-KAFKA-806 Commons compress CVE for kafka components

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka Connect JDBC 10.0.1.0 - 2110 (EEP 8.0.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hadoop.

Version	10.0.1.0
Release Date	October 2021
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/kafka-connect-jdbc
GitHub Release Tag	10.0.1.0-eep-800
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

- None

Fixes

This release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
1b52890	2021-09-05	MAPR-KAFKA-758 Update the maven artifact version strings to eep
d0b64f5	2021-08-26	MAPR-KAFKA-751 Protobuf and Json Schema connect converters were added.
e2e9b83	2021-08-18	MAPR-KAFKA-745 Update Jackson dependency

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka Connect JDBC 5.1.2.0 - 2009 (MEP 7.0.0) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	5.1.2.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka-connect-jdbc
GitHub Release Tag	5.1.2.0-mapr-700
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Hive connector supports Avro format.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
ff47526	2019-02-28	MAPR-KAFKA-285 Kafka-connect-jdbc uses kafka: 2.1.1.0-mapr
4794ac5	2020-05-05	MAPR-KAFKA-582 Update ZK to v3.5.6

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
ae5f25e	2020-06-04	MAPR-KAFKA-603 Update jacoco and spotbugs plugin

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka Connect JDBC 4.1.0 - 2201 (EEP 6.3.6) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	4.1.0
Release Date	January 2022
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka-connect-jdbc
GitHub Release Tag	4.1.0-mapr-2201
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka Connect JDBC 4.1.0 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

- CVE fixes

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
9077d4a	2022-01-25	MAPR-KAFKA-839 Updated log4j version to 1.3.1-mapr
8a3fc0d	2021-12-21	MAPR-KAFKA-826 Updated log4j version

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Connect JDBC 4.1.0 - 2101 (EEP 6.3.2) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	4.1.0
Release Date	January 2021
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka-connect-jdbc
GitHub Release Tag	4.1.0-mapr-2101
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka Connect JDBC 4.1.0 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- The Hive connector supports Avro format.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
a0289cde	2021-01-20	MAPR-KAFKA-665 Kafka Connect JDBC artifact version is incorrect
978012d2	2019-12-02	MAPR-KAFKA-507 support of java.sql.BIT

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Connect JDBC 4.1.0-1808 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.1.0
Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	n/a
GitHub Release Tag	4.1.0-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/

Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Added support of SSL encryption, impersonation, and PAM authentication for Kafka Connect REST API.
- Added support security by default for Kafka Connect.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
7274afd3	2018-06-08	KAFKA-67 Add workaround for isSigned for Hive

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka Connect JDBC 4.0.0-1803 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	4.0.0
Release Date	March 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	n/a
GitHub Release Tag	4.0.0-mapr-1803
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

The JDBC Sink Connector was added.

Fixes

Changes have been made to the mapr-kafka package and to the packaging process.

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
7523a17	Feb 9, 2018	KAFKA-67 JDBC-source connector could not produce data from Hive table into the topic.

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka Connect JDBC 2.0.1-1710 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Confluent home Kafka Connect JDBC page

Version	2.0.1-1710
Release Date	November 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	n/a
GitHub Release Tag	2.0.1-mapr-1710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Kafka Connect JDBC 2.0.1-1710 introduces the following enhancements or MapR platform-specific behavior changes:

- You can use configure.sh to configure this component.

Fixes

Changes have been made to the mapr-kafka package and to the packaging process

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka Connect 10.0.0.100 - 2201 (EEP 8.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka. See Apache Kafka 2.6.1 release notesor the Apache Kafka Streams homepagefor more information.

Version	10.0.0.100
Release Date	January 2022

HPE Version Interoperability	See EEP Components and OS Support on page 5536.
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Kafka Connect 10.0.0.100-2201 introduces the following enhancements or HPE platform-specific behavior changes:

- Federal Information Processing Standards (FIPS) support (valid for core 7.0.0 and later). .

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
5ac2089	2022-02-01	MAPR-KAFKA-842 Change default value for "listeners" property for kafka-connect
7c5c9cc	2021-12-22	MAPR-KAFKA-774 Kafka-connect service does not restart after running configure.sh script
078d514	2021-11-16	MAPR-KAFKA-794 Kafka Connect doesn't use field which is not present in core 6.2
53e77e1	2021-11-12	MAPR-KAFKA-785 Fix Kafka Connect work with enabled FIPS
e76de0a	2021-10-25	MAPR-KAFKA-785 Kafka Connect works with enabled FIPS

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Connect 5.1.2.0 - 2009 (EEP 7.0.0) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	5.1.2.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Maven Artifacts	https://repository.mapr.com/maven/

Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Additional security features.

Fixes

Changes have been made to the mapr-kafka package and to the packaging process.

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
d2b01a7	2019-05-22	MAPR-KAFKA-338 Enable authentication and impersonation by default only for secure clusters
72d879a	2019-06-14	MAPR-KAFKA-338 Impersonate Kafka Connect with authenticated user otherwise with current
ba6b3ac	2019-06-14	MAPR-KAFKA-339 Add multi-authentication mechanism for Connect REST API
dbdf61b	2019-10-19	MAPR-CORE-322 Security headers and custom headers for kafka rest
f2433b7	2020-07-09	MAPR-KAFKA-405 Automatic url retrieval for SR
c2d8cb0	2020-08-10	MAPR-KAFKA-617 Exclude kerberos auth authentication type from MultiMechsAuthenticationHandler

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka Connect 4.1.0-1808 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.1.0
Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	n/a
GitHub Release Tag	4.1.0-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/

last-updated: Apr 11, 2022

Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Support for SSL encryption, impersonation and PAM authentication.
- Support for security by default.
- Multiple issues were fixed.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
87cb472	May 14, 2018	KAFKA-100 Adds PAM support for Kafka Connect REST
4afd64a	May 23, 2018	KAFKA-99 Add impersonation support to Kafka Connect
1c817f0	June 8, 2018	KAFKA-114 Implement Secure by default for Kafka Connect

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka Connect 2.0.1-1801 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Kafka Connect 2.0.1-1801.

These notes are for the Kafka Connect for MapR Streams component included in the MapR Converged Data Platform.

Important: Use the latest patch for core MapR v6.0.0.

Version	2.0.1
Release Date	February 2018
MapR Version Compatibility	See EEP Components and OS Support on page 5536
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

This release of Kafka Connect includes fixes to Kafka distributed mode for MapR Streams.

Fixes

This MapR release includes the following new fixes since the latest Kafka Connect 2.0.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
0a88073	2017/11/28	MAPR-30006 Replace seektoend and set poll timeout
cdedaee	2017/08/07	MAPR-30290 Set TTL of metadata stream to infinity

Known Issues and Limitations

Distributed mode is only supported on MapR 5.2.1 and above.

Kafka Connect 2.0.1-1707 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Kafka Connect 2.0.1-1707.

These notes are for the Kafka Connect for MapR Streams component included in the MapR Converged Data Platform.

Version	2.0.1
Release Date	August 2017
MapR Version Compatibility	See EEP Components and OS Support on page 5536
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

The Kafka Connect for MapR Streams service can be started, restarted, and stopped via the maprcli nodes services command.

Fixes

There are no fix updates in this release.

Known Issues and Limitations

Distributed mode is only supported on MapR 5.2.1 and above.

Kafka Connect 2.0.1-1703 (EEP 3.x) Release Notes

Release notes for the Kafka Connect for MapR Event Store For Apache Kafka component included in the MapR Converged Data Platform.

Version	2.0.1
Release Date	April 2017
MapR Version Compatibility	See EEP Components and OS Support on page 5536
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

This release of Kafka Connect includes updates to the HDFS Connector. The JDBC Connector was not updated.

Fixes

MapR Fix	Description
MAPR-26184	Removed close of topic partition writers in DataWriter close.

last-updated: Apr 11, 2022

MapR Fix	Description
MAPR-26606	Kafka Connect HDFS is now built with Hive 2.1.1.

Known Issues and Limitations

Distributed mode is only supported on MapR 5.2.1 and above.

Kafka Connect 2.0.1-1703 (EEP 2.x) Release Notes

Release notes for the Kafka Connect for MapR Event Store For Apache Kafka component included in the MapR Converged Data Platform.

Version	2.0.1
Release Date	April 2017
MapR Version Compatibility	See EEP Components and OS Support on page 5536
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

This release of Kafka Connect includes updates to the HDFS Connector. The JDBC Connector was not updated.

Fixes

MapR Fix	Description
MAPR-26184	Removed close of topic partition writers in DataWriter close.

Known Issues and Limitations

Distributed mode is only supported on MapR 5.2.1 and above.

Kafka Connect 2.0.1-1611 Release Notes

Release notes for the Kafka Connect for MapR Event Store For Apache Kafka component included in the MapR Converged Data Platform.

Version	2.0.1
Release Date	December 9, 2016
MapR Version Compatibility	See Ecosystem Support Matrix
Source on GitHub	n/a
GitHub Release Tag	2.0.1-mapr-1611
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in the Release

This release is the first MapR release of the Kafka Connect for MapR Event Store For Apache Kafka. The `kafka-connect-jdbc` package is a Kafka Connector used to import data from any relational database with a JDBC driver into MapR Event Store For Apache Kafka or Apache Kafka topics. The `kafka-connect-hdfs` package is a Kafka Connector for copying data between MapR Event Store For Apache Kafka or Apache Kafka and MapRFS. See the Apache Kafka Connect for more information.

Known Issues and Limitations

The following are either known issues or limitations.

- Ensure that the latest MapR patch version is installed before using the JDBC and HDFS connectors. The latest MapR patch contains a fix associated with Kafka Connect for MapR streams functionality.
- Distributed mode is not available in MapR version 5.2.0.

Kafka REST Release Notes

The release notes for the Kafka REST component included in the MapR Converged Data Platform contains notes specific to MapR only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Kafka REST Proxy 6.0.0.100 - 2201 (EEP 8.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka. You may also be interested in the Apache Kafka REST Proxy 6.0.0.0 changelog or the Apache Kafka REST Proxy project homepage.

Version	6.0.0.100
Release Date	January 2022
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	6.0.0.100-eep-810
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Kafka REST Proxy 6.0.0.100 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

- Federal Information Processing Standards (FIPS) support (valid for core 7.0.0 and later). .
- CVE fixes.

Fixes

This release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
c8b56ab	2021-11-26	MAPR-KAFKA-809 Gson CVE for kafka components
8f413c0	2021-11-11	MAPR-KAFKA-796 Update dependencies versions
b4d85bc	2021-11-04	MAPR-KAFKA-787 Change error message for not supporting API

8d06f39	2021-10-27	MAPR-KAFKA-784 Add security java options if FIPS mode is enabled

For complete details, refer to the commit log for this project in Github.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka REST Proxy 6.0.0.0 - 2110 (EEP 8.0.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka. You may also be interested in the Apache Kafka REST Proxy 6.0.0.0 changelog or the Apache Kafka REST Proxy project homepage.

Version	6.0.0.0
Release Date	October 2021
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	6.0.0.0-eep-800
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Kafka REST Proxy 6.0.0.0 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
f4361740	2021-09-27	MAPR-KAFKA-729 Replace sudo command with maprexecute in Kafka REST Server
98f4487f	2021-09-21	MAPR-KAFKA-753 Incorrect pid in the /opt/mapr/pid/kafka-rest.pid after fresh install kafka-rest
245062f3	2021-09-14	MAPR-KAFKA-756 Consumer doesn't read data from topic
99280f05	2021-09-08	MAPR-KAFKA-757 mapr-security-web jars should be taken from the cluster

851ea5d3	2021-09-05	MAPR-KAFKA-758 Update the maven artifact version strings to eep
e9a8c7ed	2021-08-18	MAPR-KAFKA-745 Update Jackson dependency version
4e56e605	2021-08-13	MAPR-KAFKA-689 Service verifier was added to Kafka Rest
9d91ef73	2021-08-09	MAPR-KAFKA-725 Update hadoop, hive, hbase dependencies for kafka
f62d66e3	2021-05-27	MAPR-KAFKA-714 mapr-streams dependency was replaced with kafka-eventstreams
89b52ebb	2021-05-25	MAPR-KAFKA-715 Hadoop dependency was updated to 2.7.5.100-mapr-720-SNAPSHOT
f9b4747a	2021-03-02	MAPR-KAFKA-677 Responses for API v3 negative cases are produced correctly
de7d0181	2021-02-26	MAPR-KAFKA-680 Jetty version differ from the default ones was excluded from project.
2dcc3cc8	2021-02-24	MAPR-KAFKA-679 Hadoop version was changed to 2.7.4.0-mapr-710
d0fb611e	2021-02-10	MAPR-KAFKA-673 AdminClient instance should be created for certain user

For complete details, refer to the commit log for this project in Github.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka REST Proxy 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka. You may also be interested in the Apache Kafka REST Proxy 5.1.2 changelog or the Apache Kafka REST Proxy project homepage.

Version	5.1.2.200
Release Date	April 2021
MapR Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	5.1.2.200-mapr-710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka REST Proxy 5.1.2.200 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- Kafka Rest works with the Avro format through the Schema Registry.
- Service verifier

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
16ee375f	2021-03-31	MAPR-KAFKA-689 Service verifier was added to Kafka Rest
4dcb0bd7	2021-03-19	MAPR-KAFKA-692 Update Hadoop dependency to 2.7.5.0-mapr-710-SNAPSHOT
fef0187d	2021-03-05	MAPR-KAFKA-683 Topic's existence is checked by the mdObserver specific for the current user.

For complete details, refer to the commit log for this project in Github.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka REST Proxy 5.1.2.100 - 2101 (MEP 7.0.1) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka. You may also be interested in the Apache Kafka REST Proxy 5.1.2 changelog or the Apache Kafka REST Proxy project homepage.

Version	5.1.2.100
Release Date	January 2021
MapR Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	5.1.2.100-mapr-701
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka REST Proxy 5.1.2.100 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- Kafka Rest works with the Avro format through Schema Registry.

Fixes

This HPE release includes the following fixes on the base release.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
42fe966b	2021-01-18	KAFKA-661 Override version of connect-runtime dependency
7732d93b	2020-11-11	MAPR-KAFKA-640 Multiple thread-leaks if impersonation enabled

For complete details, refer to the commit log for this project in Github.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka REST Proxy 5.1.2.0 - 2009 (MEP 7.0.0) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Kafka REST Proxy 5.1.2 changelog or the Apache Kafka REST Proxy project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	5.1.2.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	5.1.2.0-mapr-700
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Kafka REST works with the Avro format using Mapr Schema Registry.

Fixes

This release includes bug fixes for configuration scripts. For complete details, refer to the commit log for this project in Github.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
8dd460c	2019-03-12	MAPR-KAFKA-283 MULTIAUTH will be used as a default authentication method

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
7b533f9	2019-03-12	MAPR-KAFKA-285 Kafka Rest uses rest utils 5.1.2.0-mapr and kafka-client 2.1.1.0-mapr
088de14	2019-03-12	MAPR-KAFKA-293 Kafka Rest uses 2.1.1.0-mapr version of Kafka_scala.
6f00a3e	2019-04-17	MAPR-KAFKA-302 Refactoring of impersonation for Kafka Rest 5.1.2
18549f1	2019-04-23	MAPR-KAFKA-313 Avro in Kafka Rest works with default stream
4911ede	2019-04-26	MAPR-KAFKA-316 Kafka Rest uses default Mapr Schema Registry port's settings.
9ad58ae	2019-05-11	MAPR-KAFKA-331 Kafka Rest uses 6.2.0 version of mapr and 2.7.4 version of hadoop.
c41d9d5	2019-06-18	MAPR-KAFKA-354 Kafka REST was integrated with secure Schema Registry
581957	2019-06-19	MAPR-KAFKA-382: "authentication.method" property was replaced in Kafka Rest config with "authentication.enable"
39f7d50	2019-07-08	MAPR-KAFKA-404 Automatic Setup of Schema Registry url for Kafka Rest
03a2df8	2019-08-09	MAPR-KAFKA-188 Enable JMX authentication by default
29bea42	2019-10-18	MAPR-CORE-329 Security headers and custom headers for Kafka Rest
d83d02e	2020-04-13	MAPR-KAFKA-568 update log4j artifact for kafka-rest 5.1.2
3d2ef4f	2020-05-05	MAPR-KAFKA-582 Update ZK to v3.5.6
f852ffd	2020-06-10	MAPR-KAFKA-604 Update spotbugs plugin to build kafka rest with java 11
df090d1 3f1bff8 cef0836	2020-06-26 2020-07-02 2020-08-13	MAPR-KAFKA-607 Schema Registry Discovery options was added
0cc7c69 4d64bab 3a8ab95 0f311ab	2020-08-25 2020-08-25 2020-08-28 2020-08-28	MAPR-KAFKA-556 JMX long term support was added
4cf21e7 1a5f1eb 4bc803d	2020-08-25 2020-08-25 2020-08-26	MAPR-KAFKA-387 Partition exists checks for /offsets API was added

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka REST 4.1.0 - 2201 (EEP 6.3.6) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hadoop. You may also be interested in the Apache Kafka REST Proxy 4.1.0 changelog or the Apache Kafka REST Proxy project homepage.

Version	4.1.0
Release Date	January 2022
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	4.1.0-mapr-2201
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka REST 4.1.0 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

- CVE fixes

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
4bc6188	2021-12-30	MAPR-KAFKA-833 Updated log4jv2 version
7293406	2021-12-15	MAPR-KAFKA-825 CVE-2021-44228 - Log4j vulnerability in Kafka REST

For complete details, refer to the commit log for this project in Github.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka REST 4.1.0 - 2101 (MEP 6.3.2) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Hadoop. You may also be interested in the Apache Kafka REST Proxy 4.1.0 changelog or the Apache Kafka REST Proxy project homepage.

last-updated: Apr 11, 2022

Version	4.1.0
Release Date	January 2021
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	4.1.0-mapr-2101
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka REST 4.1.0 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
da80f42a	2020-12-30	MAPR-KAFKA-655 Hadle ConfigException in createonsumer and check for null in committedOffsets method
8eb0f872	2020-12-30	MAPR-KAFKA-655 Remove topic existence check for subscription and assignments methods
726413f3	2020-12-24	MAPR-KAFKA-390 Add topic and partition existence verification
04d2b82d	2020-12-24	MAPR-KAFKA-389 Return 404 on nonexistent stream for /streams API
1e800440	2020-12-24	MAPR-KAFKA-655 Cherry-pick commits from MAPR-KAFKA-387
da0264d5	2020-12-24	MAPR-KAFKA-656 Set rest-utils version to SNAPSHOT and set jetty version in props
56fa472e	2020-10-30	MAPR-KAFKA-640 Multiple thread-leaks if impersonation enabled (addition)
bbc480a6	2020-10-29	MAPR-KAFKA-640 Multiple thread-leaks if impersonation enabled

For complete details, refer to the commit log for this project in Github.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka REST Proxy 4.1.0 - 2009 (MEP 6.3.1) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Kafka REST Proxy 4.1.0 changelog or the Apache Kafka REST Proxy project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.1.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	4.1.0-mapr-2009
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This release includes bug fixes for configuration scripts. For complete details, refer to the commit log for this project in Github.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
7b1f855	2020-04-13	MAPR-KAFKA-568 log4j version was updated to 2.13.1
0e689d4	2020-04-16	MAPR-KAFKA-580 Only client and consumer properties are passed to SimpleConsumerConfig
dbb8c8f	2020-05-05	MAPR-KAFKA-569 Version of jackson artifacts was updated to 2.10.3
a81924a	2020-06-22	MAPR-KAFKA-609 OPTIONS method is disabled

Known Issues and Limitations

- None.

Resolved Issues

- MAPR-KAFKA-608: Server Banner is disabled in HTTP Responses

Kafka REST Proxy 4.1.0-1912 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Kafka REST Proxy 4.1.0 changelog or the Apache Kafka REST Proxy project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.1.0
Release Date	December 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	4.1.0-mapr-1912
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None

Fixes

This release includes bug fixes for configuration scripts. For complete details, refer to the commit log for this project in Github.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
7630302	2019-11-22	KAFKA-504 – Parsing of configure.sh options is fixed. --EC | -EC option is processed correctly

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka REST 4.1.0-1808 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.1.0
Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	4.1.0-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/

Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Remove certificate generation, kafka should use certs from the {MAPR_HOME}/conf/ directory.
- Remove key/keystore passwords and keystore values from config file.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
7b8f23883651b9a6c341a1a4c8f120cc322013d8	July 11, 2018	KAFKA-138 Kafka should not generate certificates and keys for ssl to "${KAFKA_HOME}/conf" folder
7b8f23883651b9a6c341a1a4c8f120cc322013d8	July 11, 2018	KAFKA-143 SSL properties should be taken directly from ssl-client.xml

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka REST 4.0.0-1803 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	4.0.0
Release Date	March 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	4.0.0-mapr-1803
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Added support for Kafka API v2

- Impersonation
- PAM authentication
- You can use configure.sh to configure this component

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
1eca085	Feb 5, 2018	KAFKA-60 Does not consume messages from one partition of the topic
08519b3	Feb 5, 2018	KAFKA-68 Added impersonation for requests to APIv2
be88ae4	Feb 5, 2018	KAFKA-85 Fixed delay 5 seconds for "/topics"; request

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka REST 2.0.1-1803 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	2.0.1
Release Date	March 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	2.0.1-mapr-1710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

N/A

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
53ce7ea	March 19, 2018	KAFKA-96 Could not produce message to topic in hybrid mode.
bb917b2	March 17, 2018	KAFKA-75 Kafka-REST gateway always eventually runs out of memory.

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka REST 2.0.1-1710 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	2.0.1
Release Date	November 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	2.0.1-mapr-1710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Kafka REST 2.0.1-1710 introduces the following enhancements or MapR platform-specific behavior changes:

- Impersonation
- PAM authentication
- You can use configure.sh to configure this component

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
<commit_tag>	<yyyy-mm-dd>	<description_of_fix>
5d6d2cd	Sep 27, 2017	KAFKA-26 Enable impersonation for Kafka REST by default
735ca38	Sep 21, 2017	KAFKA-23 Configure.sh should remove not_configured_yet
c45d729	Sep 15, 2017	KAFKA-5 Adds impersonation to Kafka REST
d7a960b	Sep 6, 2017	KAFKA-1 Removes restart kafka-rest from configure.sh
aad565d	Aug 30, 2017	KAFKA-1 Add configure.sh script
8959902	Aug 21, 2017	MAPR-28814 Kafka-rest doesn't work in hybrid mode on MapR sasl

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
4fd710d	Aug 14, 2017	MAPR-28771 Fixed Kafka REST java.lang.NoClassDefFoundError: com/mapr/fs/MapRFileSystem on 6.0 core

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None

Resolved Issues

- None

Kafka REST 2.0.1-1707 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Kafka REST Proxy 2.0.1-1707.

Release notes for the Kafka REST Proxy for MapR Streams component included in the MapR Converged Data Platform.

These notes relate specifically to the MapR distribution for Apache Hadoop. You may also be interested in the Kafka REST Proxy info.

Version	2.0.1
Release Date	August 2017
MapR Version Compatibility	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	2.0.1-mapr-1707
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in this Release

You can configure excluded protocols and ciphers with the following added properties:

- ssl.disabled.protocols.
- ssl.cipher.suites.exclude

Fixes

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
69eaa0c	Jun 20, 2017	MAPR-27995 - Service restart via 'maprcli' does not work on Suse 12.1
c05287b	Jun 20, 2017	KAFKA-2 Change Kafka REST default ssl.protocol to TLSv1.2

Known Issues and Limitations

- Impersonation is not supported for Kafka REST Proxy for MapR Streams. Kafka REST is run as **mapr user** which means that you must explicitly give **mapr user** permissions to consume and produce from/to streams. For example, the `consumeperm` parameter must be set to **mapr user** to be able to consume and the `produceperm` parameter must be set to **mapr user** to be able to produce.

Kafka REST 2.0.1-1703 Release Notes

Release notes for the Kafka REST Proxy for MapR Event Store For Apache Kafka component included in the MapR Converged Data Platform.

Version	2.0.1
Release Date	April 2017
MapR Version Compatibility	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/kafka-rest
GitHub Release Tag	2.0.1-mapr-1703
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in this Release

This release of includes the following performance-related behavior changes:

- By default, `streams.buffer.max.time.ms` is now 0. Previously, it was 3000.
- By default, `consumer.request.timeout.ms` is now 1. Previously, it was 1000.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
ec4daf1	2017-01-16	MAPR-25552: Updated default configuration to improve performance.
12d47e5	2016-12-01	MAPR-25319: Removed the check for topic existence with KafkaConsumer.listTopics() when producing to topic or partition
96a74a4	2016-12-01	MAPR-24904 Adjusted Producer/ Consumer performance tools for streams topics

Known Issues and Limitations

- Impersonation is not supported for Kafka REST Proxy for MapR Event Store For Apache Kafka. Kafka REST is run as **mapr user** which means that you must explicitly give **mapr user** permissions to consume and produce from/to streams. For example, the `consumeperm` parameter must be set to **mapr user** to be able to consume and the `produceperm` parameter must be set to **mapr user** to be able to produce.

Kafka REST 2.0.1-1611 Release Notes

Release notes for the Kafka REST Proxy for MapR Event Store For Apache Kafka component included in the MapR Converged Data Platform.

Version	2.0.1
Release Date	December 9, 2016
MapR Version Compatibility	See Ecosystem Support Matrix
Source on GitHub	n/a
GitHub Release Tag	2.0.1-mapr-1611

Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in the Release

This release is the first MapR release of the Kafka REST Proxy for MapR Event Store For Apache Kafka.

Known Issues and Limitations

The following are either known issues or limitations.

- Impersonation is not supported for Kafka REST Proxy for MapR Event Store For Apache Kafka. Kafka REST is run as **mapr user** which means that you must explicitly give **mapr user** permissions to consume and produce from/to streams. For example, the `consumeperm` parameter must be set to **mapr user** to be able to consume and the `produceperm` parameter must be set to **mapr user** to be able to produce.

Kafka Schema Registry Release Notes

This feature is presented as a developer preview. Developer previews are not tested for production environments, and should be used with caution.

The release notes for the Kafka Schema Registry component included in the MapR Converged Data Platform contains notes specific to MapR only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Kafka Schema Registry 6.0.0.100 - 2201 (EEP 8.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	6.0.0.100
Release Date	January 2201
HPE Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/schema-registry
GitHub Release Tag	6.0.0.100-eep-810
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Kafka Schema Registry 6.0.0.100 - 2201 introduces the following enhancements or HPE platform-specific behavior changes:

- Federal Information Processing Standards (FIPS) support (valid for core 7.0.0 and later). .
- CVE fixes.

Fixes

This HPE release includes the following fixes on the base release:

last-updated: Apr 11, 2022

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
5e9ecf0	2021-11-30	MAPR-KAFKA-810 Gson CVE for kafka components
1a10948	2021-11-30	MAPR-KAFKA-810 Commons compress CVE for kafka components
51edd36	2021-11-17	MAPR-KAFKA-802 Schema Registry starts on the cluster with FIPS disabled
19eca68	2021-11-11	MAPR-KAFKA-796 Update dependencies versions
0639c9b	2021-11-04	MAPR-KAFKA-788 Removed old Kafka licenses and notices
70314bd	2021-10-27	MAPR-KAFKA-784 Add security java options if FIPS mode is enabled

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Schema Registry 6.0.0.0 - 2110 (EEP 8.0.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	6.0.0.0
Release Date	October 2021
HPE Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/schema-registry
GitHub Release Tag	6.0.0.0-eep-800
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

Kafka Schema Registry 6.0.0.0 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
90a62fe2c	2021-09-09	MAPR-KAFKA-757 mapr-security-web jar should be taken from the cluster
895247849	2021-09-05	MAPR-KAFKA-758 Update the maven artifact version strings to eep

3ae046f00	2021-08-25	MAPR-KAFKA-751 Protobuf and Json Schema jars is present in Schema Registry package as Avro jars.
6d241d46d	2021-08-19	MAPR-KAFKA-741 Make verify_service executable
20e5affeb	2021-08-18	MAPR-KAFKA-745 Update Jackson dependencies
28e444832	2021-08-03	MAPR-KAFKA-717 kafka-eventstreams.jar was added to classpath
bbed535c7	2021-05-29	MAPR-KAFKA-715 Hadoop jars should be taken from cluster
189220edd	2021-05-27	MAPR-KAFKA-714 mapr-streams dependency was replaced with kafka-eventstreams
e995cab97	2021-05-25	MAPR-KAFKA-715 Hadoop dependency was updated to 2.7.5.100-mapr-720-SNAPSHOT
0cdde9a77	2021-02-24	MAPR-KAFKA-679 Hadoop version was changed to 2.7.4.0-mapr-710
869273aa6	2021-02-03	MAPR-KAFKA-671 Kafka version was changed from 2.6.0 to 2.6.1
94c2442c6	2021-01-25	MAPR-KAFKA-667 The usage of option BOOTSTRAP_SERVERS was removed.

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Schema Registry 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	5.1.2.200
Release Date	April 2021
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/schema-registry
GitHub Release Tag	5.1.2.200-mapr-710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka Schema Registry 5.1.2.200 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- Service verifier

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
e6351ffcd	2021-04-02	MAPR-KAFKA-691 Service verifier was added to Schema Registry (data is pasted in correct log file)
5cd141cef	2021-04-01	MAPR-KAFKA-697 Jersey 1.19 was excluded from the project
e8c9940f6	2021-03-23	MAPR-KAFKA-695 Kafka unit tests fail with UnknownHostException
9842a8471	2021-03-22	MAPR-KAFKA-694 Build fails because of bug MPLUGIN-336
bd9dd9aaf	2021-03-22	MAPR-KAFKA-692 Update Hadoop dependency to 2.7.5.0-mapr-710-SNAPSHOT

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Schema Registry 5.1.2.100 - 2101 (MEP 7.0.1) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Kafka.

Version	5.1.2.100
Release Date	January 2021
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/schema-registry
GitHub Release Tag	5.1.2.100-mapr-701
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (MEPs).

New in This Release

Kafka Schema Registry 5.1.2.100 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
e5dd2e9f	2021-01-08	KAFKA-661 Bump jetty to v9.4.35

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Schema Registry 5.1.2.0 - 2009 (MEP 7.0.0) Release Notes

The following notes relate specifically to the Kafka Schema Registry component included in the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	5.1.2.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/schema-registry
GitHub Release Tag	5.1.2.0-mapr-700
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/labs/mapr-schema-registry and select your EEP and OS to view the list of package names

New in This Release

- The MapR Schema Registry is included in MEP repositories beginning with MEP-7.0.0.
- Additional security features.

For more information, see Kafka Schema Registry.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
0a91336	2019-02-28	MAPR-KAFKA-285 Schema Registry uses kafka: 2.1.1.0-mapr and rest utils 5.1.2.0-mapr
5384e6c	2019-03-16	MAPR-KAFKA-272 Enable security in SR for secure clusters
f6bfe4c	2019-03-17	MAPR-KAFKA-296 MULTIAUTH authentication is enabled by default
4cd9935	2019-03-18	MAPR-KAFKA-206: Create Authorization filter for Schema Registry
8b4f207	2019-04-16	MAPR-KAFKA-247 Mapr Sasl For schema registry
aff241b	2019-04-25	MAPR-KAFKA-314 Authorization filter for cookies auth
5683f85	2019-04-25	MAPR-KAFKA-309 Cookie authentication for Schema Registry

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
0e66c3f	2019-05-11	MAPR-KAFKA-331 Mapr streams version is 6.2.0-mapr.
e9f1b08	2019-05-23	MAPR-KAFKA-343 Log compaction is enabled for Schema Registry kafka store internal stream
f02e245	2019-06-11	MAPR-KAFKA-377: Add separate configuration property to enable or disable authorization property
7074c09	2019-06-17	MAPR-KAFKA-363 schema.registry.service.id property is added.
a0291bd	2019-06-19	MAPR-KAFKA-382: Replace "authentication.method" property in ksql config with "authentication.enable"
6d0a12d	2019-06-24	MAPR-KAFKA-371 Utils to read SR Urls on client side
7762fc2	2019-10-15	MAPR-CORE-328 Security and custom headers for Schema Registry
7.56E+08	2020-04-06	MAPR-KAFKA-564 Build schema registry with java 11
95e12ac	2020-05-05	MAPR-KAFKA-582 Update ZK to v3.5.6

Known Issues and Limitations

- None.

Resolved Issues

- None.

Kafka Schema Registry 4.1.1-1901 Release Notes

The following notes relate specifically to the Kafka Schema Registry component included in the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.1.1
Release Date	February 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	n/a
GitHub Release Tag	4.1.1-mapr-1901
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/labs/mapr-schema-registry and select your EEP and OS to view the list of package names

New in This Release

- This release is the first MapR release of Kafka Schema Registry for MapR Event Store For Apache Kafka.
- This feature is presented as a developer preview. Developer previews are not tested for production environments, and should be used with caution.
- Because it is a developer preview feature, the Kafka Schema Registry is not provided in the MEP 6.1 repository. You must download the packages from https://package.mapr.hpe.com/labs/mapr-schema-registry. Using the MapR Installer to install the Kafka Schema Registry is not currently supported.
- For more information, see Kafka Schema Registry.

Fixes

N/A

Known Issues and Limitations

- None.

Resolved Issues

- None.

MapR Monitoring Release Notes

The release notes for MapR Monitoring components included in the MapR Converged Data Platform contains notes specific to MapR only.

Monitoring Components - EEP 8.1.0 Release Notes

The notes below relate specifically to the MapR Data Platform.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 8.1.0 release contains the following monitoring-component versions: • Collectd 5.12.0.310 • Fluentd 1.10.3.300 • Opentsdb 2.4.1.300 • Elasticsearch 6.8.8.410 • Grafana 7.5.10.310 • Kibana 6.8.8.400
Release Date	January 2022
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- EEP 8.1.0 includes updates to Collectd, Elasticsearch, and Grafana.

  To compare Monitoring component versions, see Component Versions for Released EEPs on page 5586.

Fixes

Principal fixes in this release include:

- COLD-218 - nfs3 errors in collectd_daemon.log
- COLD-219 - Need to add configurable filter for jmx to specify process names we want to attach to for jmx collection
- ES-88 - Fix vulnerabilities including CVE-2021-44228

Known Issues and Limitations

Log monitoring is not supported in installations with FIPS-enabled nodes in EEP 8.1.0.

Resolved Issues

None.

Monitoring Components - EEP 8.0.0 Release Notes

The notes below relate specifically to the MapR Data Platform.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 8.0.0 release contains the following monitoring-component versions: • Collectd 5.12.0.300 • Fluentd 1.10.3.300 • Opentsdb 2.4.1.300 • Elasticsearch 6.8.8.400 • Grafana 7.5.10.300 • Kibana 6.8.8.400
Release Date	October 2021
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- EEP 8.0.0 includes updates to all of the monitoring components. In particular, the Collectd, Grafana, and Open TSDB versions changed significantly. Open TSDB now uses a four-digit version. See About the MapR Patch Version on page 5520.

  To compare Monitoring component versions, see Component Versions for Released EEPs on page 5586.

last-updated: Apr 11, 2022

- In Grafana 7.5.x, the steps to display sample dashboards are different from the steps used in previous versions of Grafana. For more information, see Sample Dashboards in Grafana on page 1384.

Fixes

Principal fixes in this release include:

- COLD-206: Activate Java logging in collectd
- COLD-213: need to update types.db with new metrics
- KIB-55: configure.sh fails to extract certs when clustername is not all lowercase
- OTSDB-121: CVE-2018-10237,CVE-2020-8908 vulnerabilities in Guava
- OTSDB-130: ot_purgeData.log log rotation fails when selinux is enabled and enforcing

Known Issues and Limitations

None.

Resolved Issues

None.

Monitoring Components - EEP 7.1.1 Release Notes

The notes below relate specifically to the MapR Data Platform.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 7.1.1 release contains the following monitoring-component versions: • Collectd 5.10.0.0 • Fluentd 1.10.3.200 • Opentsdb 2.4.0 • Elasticsearch 6.8.8.300 • Grafana 7.5.2.200 • Kibana 6.8.8.300
Release Date	October 2021
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- EEP 7.1.1 includes updates to all monitoring components.
- In Grafana 7.5.x, the steps to display sample dashboards are different from the steps used in previous versions of Grafana. For more information, see Sample Dashboards in Grafana on page 1384.

Fixes

Principal fixes in this release include:

- COLD-206: Activate Java logging in collectd
- COLD-208: Collectd Log rotation fails when SELinux is enabled
- ES-85: Configure.sh fails to get ip address of interface on SLES
- FLUD-54: Fluentd log rotation fails when SELinux is enabled
- FLUD-59: Configure.sh failed to extract certs when clustername is not all lowercase
- KIB-54: Fix es_mgmt.sh not getting correct return codes - 688 changed format
- KIB-55: Configure.sh fails to extract certs when clustername is not all lowercase
- OTSDB-121: CVE-2018-10237,CVE-2020-8908 vulnerabilities in Guava
- OTSDB-128: Opentsdb.err file is overwritten / not preserved on restart
- OTSDB-130: Ot_purgeData.log log rotation fails when selinux is enabled and enforcing
- SPYG-1146: Centos8.4 Searchguard - SearchGuardSSLNettyHttpServerTransport - Exception during establishing a SSL connection: javax.net.ssl.SSLHandshakeException: Received fatal alert: bad_certificate

Known Issues and Limitations

This release contains the following known issues and limitations:

- **SPYG-1146:** In Elasticsearch 6.8.8, Elasticsearch generates a fatal exception when used with Java 11.0.12 on RHEL/CentOS 8.3 or 8.4. This issue can occur in EEP 7.0.0, 7.0.1, or 7.1.0. The following message can be seen in the monitoring log in the Elasticsearch log directory:

  ```
  Unable to check whether cluster is sane: None of the configured nodes are
  available: [{#transport#-1}{mMuDVg1_TFytqJ-sgcTNjQ}
  {m2-mapreng-vm167242.mip.storage.hpecorp.net}{10.163.167.242:9300}]
  06:17:07.986 [elasticsearch[_client_][transport_worker][T#1]] ERROR
  com.floragunn.searchguard.ssl.http.netty.SearchGuardSSLNettyHttpServerTran
  sport - Exception during establishing a SSL connection:
  javax.net.ssl.SSLHandshakeException: Received fatal alert: bad_certificate
  javax.net.ssl.SSLHandshakeException: Received fatal alert: bad_certificate
      at sun.security.ssl.Alert.createSSLException(Alert.java:131) ~[?:?]
      at sun.security.ssl.Alert.createSSLException(Alert.java:117) ~[?:?]
      at sun.security.ssl.TransportContext.fatal(TransportContext.java:336)
  ```

  Workaround: Use one of the following workarounds:

  - Do not run Elasticsearch with Java 11.0.12. If you see this problem, reinstall 11.0.11 or an earlier Java release.

- If you want to continue running Elasticsearch with Java 11.0.12, use the following `keytool` command to import the certificates. You must provide the truststore password and cluster name:

```
cat /opt/mapr/conf/ca/signing-ca.pem | keytool \
    -import \
    -v \
    -keystore /opt/mapr/conf/ssl_usertruststore \
    -storepass <truststore_password> \
    -noprompt -alias <cluster_name>-root-signing-ca
```

- **ES-85:** In EEP 7.x.x with SLES 15 SP2, the Elasticsearch `configure.sh` module generates an error saying it cannot find a DNS or IP address. The `configure.sh` command succeeds, but Elasticsearch fails to start because it is not configured correctly.

  Workaround: Upgrade to EEP 7.1.1 or later, where this issue is fixed.

Resolved Issues

- None.

Monitoring Components - EEP 7.1.0 Release Notes

The notes below relate specifically to the MapR Data Platform.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 7.1.0 release contains the following monitoring-component versions: • Collectd 5.10.0.0 • Fluentd 1.10.3.0 • Opentsdb 2.4.0 • Elasticsearch 6.8.8.300 • Grafana 7.5.2.200 • Kibana 6.8.8.300
Release Date	May 2021
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- EEP 7.1.0 includes updates to Collectd, Fluentd, Opentsdb, Elasticsearch, Grafana, and Kibana. In particular, Grafana was updated from version 6.7.4.0 to version 7.5.2.200. To compare Monitoring component versions, see Component Versions for Released EEPs on page 5586.
- In Grafana 7.5.x, the steps to display sample dashboards are different from the steps used in previous versions of Grafana. For more information, see Sample Dashboards in Grafana on page 1384.

Fixes

Principal fixes in this release include:

- COLD-205: RM queue metrics is not seen through Collectd RESTApi Plugin due to commons-codec upgrade
- ES-75: /opt/mapr/elasticsearch/elasticsearch-6.8.8/bin/curator missing
- ES-76: bump curator to 5.8.3 version - official 5.8.2 has a couple of merge issues
- ES-77: incorrect mapping of 4th digit cause upgrade problems from mep6 to mep7
- GRAF-63: CVE-2020-13430,CVE-2020-24303 vulnerabilities in Grafana up to 7.0.0
- OTSDB-117: CVE-2020-25649,CVE-2020-35728 etc. vulnerabilities in Jackson Databind
- OTSDB-119: CVE-2017-5929 logback vulnerability

Known Issues and Limitations

This release contains the following known issues and limitations:

- None.

Resolved Issues

- None.

MapR Monitoring Components - EEP 7.0.1 Release Notes

The notes below relate specifically to the MapR Data Platform.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 7.0.1 release contains the following monitoring-component versions: • Collectd 5.10.0.0 • Fluentd 1.10.3.0 • Opentsdb 2.4.0 • Elasticsearch 6.8.8.0 • Grafana 6.7.4.0 • Kibana 6.8.8.0
Release Date	January 2021
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- EEP 7.0.1 includes updates to Collectd, Fluentd, Opentsdb, Elasticsearch, Grafana, and Kibana. To compare Monitoring component versions, see Component Versions for Released EEPs on page 5586.

Fixes

Principal fixes in this release include:

- COLD-202: fast-jmx using outdated and insecure version of jackson databind
- GRAF-61: unable to login in grafana on a fresh cluster install with core-6.2 + MEP-7.0
- OTSDB-115: fix zk jar version for build and remove from pkg
- OTSDB-116: bump jackson version to 2.9.10 or newer due to CVEs

Known Issues and Limitations

This release contains the following known issues and limitations:

- **ES-77**: During an upgrade from EEP 6.x to EEP 7.0.0 or EEP 7.0.1, some monitoring components do not get updated because of an error in the fourth digit of the package version. This issue can occur during manual upgrades or upgrades performed using the Installer. The affected components can include any or all of the following:
  - Elasticsearch
  - Fluentd
  - Grafana
  - Kibana

  Workaround: See Reinstalling Monitoring Components After an Upgrade on page 364.

Resolved Issues

- None.

MapR Monitoring Components - EEP 7.0.0 Release Notes

The notes below relate specifically to the MapR Data Platform.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 7.0.0 release contains the following monitoring-component versions: • Collectd 5.10.0.0 • Fluentd 1.10.3.0 • Opentsdb 2.4.0 • Elasticsearch 6.8.8.0 • Grafana 6.7.4.0 • Kibana 6.8.8.0
Release Date	September 2020
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- EEP 7.0.0 includes updates to Collectd, Fluentd, Opentsdb, Elasticsearch, Grafana, and Kibana. To compare Monitoring component versions, see MEP Components and OS Support.

Fixes

Principal fixes in this release include:

- **ES-20**: Elastic Search Key management seems unnecessarily complex. For more information, see "Simplified Installation for Log Monitoring" in Installation and Upgrade Notes (MapR 6.1.0) on page 39.
- **COLD-168**: Collectd process leak memory.
- **GRAF-57**: Grafana: SSRF Incorrect access control vulnerability (CVE-2020-13379).

Known Issues and Limitations

This release contains the following known issues and limitations:

- **COLD-162**: Graphs in the MapR Control System do not display metrics for MapR File System operations when hostname aliases are used. **Workaround:** Use fully qualified domain names when you install the Data Fabric through the MapR Installer. Run `hostname -f` to verify that the fully qualified domain name is returned. If the command returns a short hostname, the issue will persist.
- **ES-77**: During an upgrade from EEP 6.x to EEP 7.0.0 or EEP 7.0.1, some monitoring components do not get updated because of an error in the fourth digit of the package version. This issue can occur during manual upgrades or upgrades performed using the Installer. The affected components can include any or all of the following:
  - Elasticsearch
  - Fluentd
  - Grafana
  - Kibana

  Workaround: See Reinstalling Monitoring Components After an Upgrade on page 364.
- **FLUD-51**: In the `fluentd.conf` file, the log file name for Kibana is `kibana.log`. The file name is incorrect; the file name should be `kibana_daemon.log`. As long as the `fluentd.conf` file has the incorrect log-file name, you will not see Kibana log entries in Elasticsearch. **Workaround:** In the `fluentd.conf` file, change `kibana.log` to `kibana_daemon.log`, and restart Fluentd.
- **KIB-45**: The Kibana default index and example dashboard are not loaded at system startup. **Workaround:** Run the following command, and restart Fluentd:

  ```
  chmod +x /opt/mapr/kibana/kibana-6.8.8/bin/es_mgmt.sh
  ```

See Known Issues at Release (MapR 6.1.0) on page 47 and Troubleshoot MapR Monitoring Installation Errors on page 172.

Resolved Issues

- None.

Monitoring Components - EEP 6.3.6 Release Notes

The notes below relate specifically to the MapR Data Platform.

last-updated: Apr 11, 2022

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 6.3.6 release contains the following monitoring-component versions: • Collectd 5.8.1.210 • Fluentd 1.10.3.110 • Opentsdb 2.4.0 • Elasticsearch 6.8.8.110 • Grafana 7.5.2.110 • Kibana 6.8.8.110
Release Date	January 2022
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- In EEP 6.3.6, minor changes were made to the monitoring component versions. To compare monitoring component versions, see Component Versions for Released EEPs on page 5586.

Fixes

Principal fixes in this release include:

- COLD-218 - nfs3 errors in collectd_daemon.log
- COLD-219 - Need to add configurable filter for jmx to specify process names we want to attach to for jmx collection
- ES-88 - Fix vulnerabilities including CVE-2021-44228

Known Issues and Limitations

This release contains the following known issues and limitations:

MFS-10783

After a manual installation of release 6.1.0 or 6.1.1 on RHEL or CentOS 8.x, Collectd fails to start. Error messages indicate that `libcrypto.so.10` cannot open a shared object file.

Workaround: Install the `compat-openssl10` package using the following command, and restart Collectd:

```
yum install compat-openssl10
```

Alternatively, you can install the `compat-openssl10` package during the course of manual installation before installing the `mapr-*` packages. See Step 3: Install Cluster Service Packages on page 150.

last-updated: Apr 11, 2022

Resolved Issues

- None.

Monitoring Components - EEP 6.3.5 Release Notes

The notes below relate specifically to the MapR Data Platform.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 6.3.5 release contains the following monitoring-component versions: • Collectd 5.8.1.201 • Fluentd 1.10.3.100 • Opentsdb 2.4.0 • Elasticsearch 6.8.8.100 • Grafana 7.5.2.100 • Kibana 6.8.8.100
Release Date	October 2021
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- In EEP 6.3.5, none of the monitoring component versions changed. To compare monitoring component versions, see Component Versions for Released EEPs on page 5586.
- In Grafana 7.5.x, the steps to display sample dashboards are different from the steps used in previous versions of Grafana. For more information, see Sample Dashboards in Grafana on page 1384.

Fixes

Principal fixes in this release include:

- COLD-206: Activate Java logging in collectd
- OTSDB-121: CVE-2018-10237,CVE-2020-8908 vulnerabilities in Guava
- OTSDB-130: ot_purgeData.log log rotation fails when selinux is enabled and enforcing

Known Issues and Limitations

This release contains the following known issues and limitations:

- None.

Resolved Issues

- None.

MapR Monitoring Components - EEP 6.3.4 Release Notes

The notes below relate specifically to the MapR Data Platform.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 6.3.4 release contains the following monitoring-component versions: • Collectd 5.8.1.201 • Fluentd 1.10.3.100 • Opentsdb 2.4.0 • Elasticsearch 6.8.8.100 • Grafana 7.5.2.100 • Kibana 6.8.8.100
Release Date	May 2021
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- EEP 6.3.4 includes updates to Collectd, Fluentd, Opentsdb, Elasticsearch, Grafana, and Kibana. In particular, Grafana was updated from version 6.7.4.100 to version 7.5.2.100. To compare Monitoring component versions, see Component Versions for Released EEPs on page 5586.
- In Grafana 7.5.x, the steps to display sample dashboards are different from the steps used in previous versions of Grafana. For more information, see Sample Dashboards in Grafana on page 1384.

Fixes

Principal fixes in this release include:

- COLD-205: RM queue metrics is not seen through Collectd RESTApi Plugin due to commons-codec upgrade
- ES-75: /opt/mapr/elasticsearch/elasticsearch-6.8.8/bin/curator missing
- ES-76: bump curator to 5.8.3 version - official 5.8.2 has a couple of merge issues
- ES-77: incorrect mapping of 4th digit cause upgrade problems from mep6 to mep7
- GRAF-63: CVE-2020-13430,CVE-2020-24303 vulnerabilities in Grafana up to 7.0.0
- OTSDB-117: CVE-2020-25649,CVE-2020-35728 etc. vulnerabilities in Jackson Databind
- OTSDB-119: CVE-2017-5929 logback vulnerability

Known Issues and Limitations

This release contains the following known issues and limitations:

- None.

Resolved Issues

- None.

MapR Monitoring Components - EEP 6.3.3 Release Notes

The notes below relate specifically to the MapR Data Platform.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 6.3.3 release contains the following monitoring-component versions: • Collectd 5.8.1.201 • Fluentd 1.10.3.100 • Opentsdb 2.4.0 • Elasticsearch 6.8.8.100 • Grafana 6.7.4.100 • Kibana 6.8.8.100
Release Date	March 2021
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- EEP6.3.3 includes updates to Collectd, Fluentd, Opentsdb, Elasticsearch, Grafana, and Kibana. An update can include a change to the package timestamp and does not necessarily include a change to the package version. To compare Monitoring component versions, see Component Versions for Released EEPs on page 5586.

Fixes

Principal fixes in this release include:

- COLD-184: collectd-5.10.0.0.202006121008 crash in inflateReset2 -> free @ fs/common/zlib/inflate.c:169
- COLD-205: RM queue metrics is not seen through Collectd RESTApi Plugin due to commons-codec upgrade
- ES-75: /opt/mapr/elasticsearch/elasticsearch-6.8.8/bin/curator missing
- ES-76: bump curator to 5.8.3 version - official 5.8.2 has a couple of merge issues
- GRAF-28: The node dashboard fqdn dropbox do not show all the nodes in cluster

Known Issues and Limitations

This release contains the following known issues and limitations:

- None.

Resolved Issues

- None.

MapR Monitoring Components - EEP 6.3.2 Release Notes

The notes below relate specifically to the MapR Data Platform.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 6.3.2 release contains the following monitoring-component versions: • Collectd 5.8.1.201 • Fluentd 1.10.3.100 • Opentsdb 2.4.0 • Elasticsearch 6.8.8.100 • Grafana 6.7.4.100 • Kibana 6.8.8.100
Release Date	January 2021
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- EEP6.3.2 includes updates to Collectd, Fluentd, Opentsdb, Elasticsearch, Grafana, and Kibana. To compare Monitoring component versions, see Component Versions for Released EEPs on page 5586.

Fixes

Principal fixes in this release include:

- COLD-202: fast-jmx using outdated and insecure version of jackson databind
- GRAF-61: unable to login in grafana on a fresh cluster install with core-6.2 + MEP-7.0
- OTSDB-115: fix zk jar version for build and remove from pkg
- OTSDB-116: bump jackson version to 2.9.10 or newer due to CVEs

Known Issues and Limitations

This release contains the following known issues and limitations:

- None.

Resolved Issues

- None.

MapR Monitoring Components - EEP 6.3.1 Release Notes

The notes below relate specifically to the MapR Converged Data Platform.

These release notes contain only Data Fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 6.3.1 release contains the following monitoring-component versions: • Collectd 5.8.1.200 • Fluentd 1.10.3.0 • Opentsdb 2.4.0 • Elasticsearch 6.8.8.0 • Grafana 6.7.4.0 • Kibana 6.8.8.0
Release Date	September 2020
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

None

Fixes

None.

Known Issues and Limitations

None

See Known Issues at Release (MapR 6.1.0) on page 47 and Troubleshoot MapR Monitoring Installation Errors on page 172.

Resolved Issues

None.

MapR Monitoring Components - EEP 6.3.0 Release Notes

The notes below relate specifically to the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

last-updated: Apr 11, 2022

Version	The EEP 6.3.0 release contains the following monitoring-component versions: • Collectd 5.8.1.200 • Fluentd 1.4.0.100 • Opentsdb 2.4.0 • Elasticsearch 6.5.3.200 • Grafana 6.0.2.100 • Kibana 6.5.3.200
Release Date	December 2019
MapR Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- EEP 6.3.0 includes minor updates to Collectd, Fluentd, Opentsdb, Elasticsearch, Grafana, and Kibana. To compare MapR Monitoring component versions, see MEP Components and OS Support.

Fixes

None.

Known Issues and Limitations

This release contains the following known issues and limitations:

- COLD-162: Graphs in the MapR Control System do not display metrics for MapR Filesystem operations when hostname aliases are used. **Workaround:** Use fully qualified domain names when you install MapR through the MapR Installer. Run `hostname -f` to verify that the fully qualified domain name is returned. If the command returns a short hostname, the issue will persist.

See Known Issues at Release (MapR 6.1.0) on page 47 and Troubleshoot MapR Monitoring Installation Errors on page 172.

Resolved Issues

- None.

MapR Monitoring Components - EEP 6.2.0 Release Notes

The notes below relate specifically to the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 6.2.0 release contains the following monitoring component versions: • Collectd 5.8.1.100 • Fluentd 1.4.0.0 • Opentsdb 2.4.0 • Elasticsearch 6.5.3.100 • Grafana 6.0.2.0 • Kibana 6.5.3.100
Release Date	April 2019
MapR Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- EEP 6.2.0 includes minor updates to Collectd, Elasticsearch, and Kibana, and new versions of Fluentd and Grafana. To compare MapR Monitoring component versions, see MEP Components and OS Support.
- MapR Monitoring now displays updated Hive JMX Metrics.
- MapR Monitoring now displays new Spark JMX Metrics.

Fixes

None.

Known Issues and Limitations

See Known Issues at Release (MapR 6.1.0) on page 47 and Troubleshoot MapR Monitoring Installation Errors on page 172.

Resolved Issues

- None.

MapR Monitoring Components - EEP 6.1.0 Release Notes

The notes below relate specifically to the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 6.1.0 release contains the following monitoring component versions: • Collectd 5.8.1.0 • Fluentd 1.3.2.0 • Opentsdb 2.4.0 • Elasticsearch 6.5.3.0 • Grafana 5.4.2.0 • Kibana 6.5.3.0
Release Date	February 2019
MapR Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- EEP 6.1.0 includes new versions of the following MapR Monitoring components: Collectd, Fluentd, Elasticsearch, and Kibana. To compare MapR Monitoring component versions, see Component Versions for Released EEPs.
- MapR Monitoring now displays the Hive JMX Metrics on page 1341.

Fixes

None.

Known Issues and Limitations

See Known Issues at Release (MapR 6.1.0) on page 47 and Troubleshoot MapR Monitoring Installation Errors on page 172.

Resolved Issues

- None.

MapR Monitoring Components - EEP 6.0.0 Release Notes

The notes below relate specifically to the MapR Converged Data Platform.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 6.0.0 release contains the following monitoring component versions: • Collectd 5.8.0 • Fluentd 1.1.2 • Opentsdb 2.4.0 • Elasticsearch 6.2.3 • Grafana 4.6.1 • Kibana 6.2.3
Release Date	September 2018
MapR Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

EEP 6.0.0 includes new versions of the following MapR Monitoring components: Collectd, Fluentd, Elasticsearch, and Kibana. To compare MapR Monitoring component versions, see Component Versions for Released EEPs.

If you are upgrading from another release to MapR 6.1, you might need to perform extra steps to convert or preserve the Elasticsearch and Kibana indexes before upgrading. See Pre-Upgrade Steps for MapR Monitoring on page 342.

Fixes

None.

Known Issues and Limitations

See Known Issues at Release (MapR 6.1.0) on page 47 and Troubleshoot MapR Monitoring Installation Errors on page 172.

Resolved Issues

- None.

MapR Monitoring Components - EEP 5.0.7 Release Notes

The notes below relate specifically to the MapR Data Platform.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 5.0.7 release contains the following monitoring-component versions: • Collectd 5.7.2 • Fluentd 0.14.20 • Opentsdb 2.4.0 • Elasticsearch 5.4.1 • Grafana 7.5.2.0 • Kibana 5.4.1
Release Date	May 2021
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- EEP 5.0.7 includes updates to Collectd, Fluentd, Opentsdb, Elasticsearch, Grafana, and Kibana. In particular, Grafana was updated from version 6.7.4.200 to version 7.5.2.0. To compare Monitoring component versions, see Component Versions for Released EEPs on page 5586.
- In Grafana 7.5.x, the steps to display sample dashboards are different from the steps used in previous versions of Grafana. For more information, see Sample Dashboards in Grafana on page 1384.

Fixes

Principal fixes in this release include:

- COLD-205: RM queue metrics is not seen through Collectd RESTApi Plugin due to commons-codec upgrade
- ES-75: /opt/mapr/elasticsearch/elasticsearch-6.8.8/bin/curator missing
- ES-76: bump curator to 5.8.3 version - official 5.8.2 has a couple of merge issues
- ES-77: incorrect mapping of 4th digit cause upgrade problems from mep6 to mep7
- GRAF-63: CVE-2020-13430,CVE-2020-24303 vulnerabilities in Grafana up to 7.0.0
- OTSDB-117: CVE-2020-25649,CVE-2020-35728 etc. vulnerabilities in Jackson Databind
- OTSDB-119: CVE-2017-5929 logback vulnerability

Known Issues and Limitations

This release contains the following known issues and limitations:

- None.

Resolved Issues

- None.

MapR Monitoring Components - EEP 5.0.6 Release Notes

The notes below relate specifically to the MapR Data Platform.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	The EEP 5.0.6 release contains the following monitoring-component versions: • Collectd 5.7.2 • Fluentd 0.14.20 • Opentsdb 2.4.0 • Elasticsearch 5.4.1 • Grafana 6.7.4.200 • Kibana 5.4.1
Release Date	January 2021
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

- EEP5.0.6 includes updates to Collectd, Fluentd, Opentsdb, Elasticsearch, Grafana, and Kibana. To compare Monitoring component versions, see Component Versions for Released EEPs on page 5586.

Fixes

Principal fixes in this release include:

- COLD-202: fast-jmx using outdated and insecure version of jackson databind
- GRAF-61: unable to login in grafana on a fresh cluster install with core-6.2 + MEP-7.0
- OTSDB-115: fix zk jar version for build and remove from pkg
- OTSDB-116: bump jackson version to 2.9.10 or newer due to CVEs

Known Issues and Limitations

This release contains the following known issues and limitations:

- None.

Resolved Issues

- None.

MapR Monitoring Components - EEP 5.0.5 Release Notes

The notes below relate specifically to the MapR Converged Data Platform.

These release notes contain only Data Fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

last-updated: Apr 11, 2022

Version	The EEP 5.0.5 release contains the following monitoring-component versions: • Collectd 5.7.2 • Fluentd 0.14.20 • Opentsdb 2.4.0 • Elasticsearch 5.4.1 • Grafana 6.7.4.0 • Kibana 5.4.1
Release Date	September 2020
Version Interoperability	Component Versions for Released EEPs
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

None

Fixes

None.

Known Issues and Limitations

None

See Known Issues at Release (MapR 6.1.0) on page 47 and Troubleshoot MapR Monitoring Installation Errors on page 172.

Resolved Issues

None.

MapR Monitoring Components - EEP 5.0.0 Release Notes

The notes below relate specifically to the MapR Converged Data Platform.

Version	The EEP 5.0.0 release contains the following monitoring component versions: • Collectd 5.7.2 • Fluentd 0.14.20 • Opentsdb 2.4.0 • Elasticsearch 5.4.1 • Grafana 4.6.1 • Kibana 5.4.1
Release Date	March 2018
MapR Version Interoperability	EEP Components and OS Support on page 5536

Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

MapR introduces enhanced security settings for the following Monitoring components:

- Grafana - For more information, see Logging on to Grafana on page 1382.
- Kibana - For more information, see Logging on to Kibana on page 1396.

Fixes

None.

Known Issues and Limitations

See Known Issues at Release (MapR 6.1.0) on page 47.

Resolved Issues

- None.

MapR Monitoring Components - EEP 4.1.0 Release Notes

The notes below relate specifically to the MapR Converged Data Platform.

Version	The EEP 4.1.0 release contains the following monitoring component versions: • Collectd 5.7.2 • Fluentd 0.14.20 • Opentsdb 2.4.0 • Elasticsearch 5.4.1 • Grafana 4.4.2 • Kibana 5.4.1
Release Date	February 2018
MapR Version Interoperability	EEP Components and OS Support on page 5536
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

MapR Monitoring in the EEP 4.1.0 release introduces the following enhancements:

- File permissions changed to ensure compliance with the MapR Security Policy.
- The following rules are added for specifying metric names:
  - Strings are case sensitive: for example, "Sys.Cpu.User" is stored separately from "sys.cpu.user".
  - Spaces are not allowed.
  - Only the following characters are allowed: a to z, A to Z, 0 to 9, -, _, ., / or Unicode letters (as per the specification).

- Metric and tags are not limited in length, though you must try to keep the values fairly short.

Fixes

None

Known Issues and Limitations

See Known Issues at Release (MapR 6.1.0) on page 47

Resolved Issues

- **FLUD-17:** Fluent SSL verification enabled for HTTPS requests.
- **FLUD-21:** Administrator passwords removed from the FluentD log files.
- **FLUD-20:** Log rotate is fixed.

MapR Monitoring Components - EEP 4.0 Release Notes

The notes below relate specifically to the MapR Converged Data Platform.

Version	The EEP 4.0.0 release contains the following monitoring component versions: • Collectd 5.7.2 • Fluentd 0.14.20 • Opentsdb 2.4.0 • Elasticsearch 5.4.1 • Grafana 4.4.2 • Kibana 5.4.1
Release Date	November 2017
MapR Version Interoperability	EEP Components and OS Support on page 5536
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

MapR Monitoring in the EEP 4.0.0 release introduces the following enhancements:

- An upgraded component stack.
- A default security feature that secures the MapR cluster and supported ecosystem components, including MapR Monitoring, using MapR-SASL and wire-level encryption, as well as the ability to configure custom security.
- A Spyglass on Streams feature that uses Streams as the default mechanism through which metrics flow from collectd to OpenTSDB.

  Note: Writing to an external OpenTSDB is not supported from release 6.0 onwards.

- MapR Monitoring now provides metrics for the YARN Resource Manager and Node Manager.

Fixes

None

Known Issues and Limitations

See Known Issues at Release (MapR 6.1.0) on page 47

Resolved Issues

- **COLD-4:** By default, collectd reads files from the custom configuration file directory (/opt/mapr/collectd/conf/) on nodes without support for the wordexp function, eliminating the need to include Filter "*.conf" in the collectd.conf configuration file.
- **SPYG-838:** On secure clusters, JMX ports can no longer be accessed without authentication, eliminating the need to setup a password on each Node Manager and Resource Manager node.
- **KIB-1:** A syntax error in export_cert.sh no longer causes an error message when you run configure.sh on a node with Kibana installed in a secure cluster.

MapR Monitoring Components - EEP 3.x.x Release Notes

The notes below relate specifically to the MapR Converged Data Platform.

Version	The EEP 3.0.0 release contains the following monitoring component versions: • Collectd 5.7.1 • Fluentd 0.14.00 • Opentsdb 2.3.0 • Elasticsearch 2.3.3 • Grafana 4.1.2 • Kibana 4.5.4
Release Date	April 2017
MapR Version Interoperability	EEP Components and OS Support on page 5536
Package Names	Package Names for MapR Ecosystem Packs (EEPs) on page 5737

New in This Release

MapR Monitoring - EEP 3.0.0 release introduces the following enhancements:

- New metrics are now available for MapR Event Store For Apache Kafka, disks, and topology.
- You can now configure fluentd to forward logs to syslog servers.
- Metrics are now tagged by default with ClusterName.
- You can store custom collectd configuration files under the /opt/mapr/collectd/conf/ directory. Collectd will include configuration files from this directory at runtime and the files will not be altered during an upgrade.

Fixes

None

last-updated: Apr 11, 2022

Known Issues and Limitations

- **COLD-4: On nodes without support for the wordexp function, collectd does not read files from the custom configuration file directory (/opt/mapr/collectd/conf/).**

  Workaround:

  1. Open the /opt/mapr/collectd/collectd-5.7.1/etc/collectd.conf file.
  2. Add the following lines to the end of the file:

     ```
     <Include "/opt/mapr/collectd/conf/">
      Filter "*.conf"
     </Include>
     ```

  3. Restart the collectd service.

     ```
     maprcli node services -name collectd -nodes <space separated list of
     collectd nodes> -action restart
     ```

- **KIB-1: On secure clusters, a syntax error in export_cert.sh causes in the following error message when you run configure.sh on a node where Kibana is installed:**

  ```
  <TIMESTAMP>: ERROR: Failed to configure ssl for kibana
  ```

  This error appears in both the configure.sh console output and in the log file (/opt/mapr/logs/configure.log).

  Workaround:

  1. Open the /opt/mapr/kibana/kibana-4.5.4/bin/export_cert.sh file.
  2. Locate the following line:

     ```
     CLUSTERNAME=$(cat /opt/mapr/conf/mapr-clusters.conf | awk '{print $1}
     | head -n 1')
     ```

  3. Replace that line with the following line:

     ```
     CLUSTERNAME=$(cat /opt/mapr/conf/mapr-clusters.conf | awk '{print $1}'
     | head -n 1)
     ```

     Notice that the location of the second backtick is different in this line.

  4. Save changes to the file and re-run configure.sh.

- **SPYG-838: On secure clusters, JMX ports can be accessed without authentication.**

  Workaround: Complete the following steps to setup a password on each Node Manager and Resource Manager node:

  1. Under `/opt/mapr/conf`, create the following files:
     - jmxremote.access
     - jmxremote.password

2. Set the owner and group to the `mapr` user:

```
chown mapr:mapr jmxremote.access
chown mapr:mapr jmxremote.password
```

3. Set file permissions to read-only:

```
chmod 400 jmxremote.access
chmod 400 jmxremote.password
```

4. Add the following entries to jmxremote.password:

```
mapr mapr
root mapr
```

5. Add the following entries to jmxremote.access:

```
mapr    readonly
```

6. In the `#Enable JMX for MaprMonitoring` section of the yarn file (/opt/mapr/hadoop/hadoop-2.x.x/bin/yarn), update the `JMX_OPTS` parameter to the following:

```
JMX_OPTS="-Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote
.authenticate=true
-Dcom.sun.management.jmxremote.password.file=/opt/mapr/conf/
jmxremote.password
-Dcom.sun.management.jmxremote.access.file=/opt/mapr/conf/
jmxremote.access
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.port"
```

7. In collectd.conf file (/opt/mapr/collectd/collectd-<version>/etc/collectd.conf), add the following connection parameters:

```
<Connection>
user "mapr"
password "mapr"
ServiceURL "service:jmx:rmi:///jndi/rmi://mfs82.qa.lab:8025/jmxrmi"
#IncludePortInHostname true
Collect "QueueMetrics"
ttl 120
</Connection>
<Connection>
user "mapr"
password "mapr"
ServiceURL "service:jmx:rmi:///jndi/rmi://mfs82.qa.lab:8027/jmxrmi"
#IncludePortInHostname true
Collect "NodeManagerMetrics"
ttl 120
</Connection>
```

8. Restart the Collectd, NodeManager, and Resource Manager services running on this node.

Resolved Issues

- SPYG-757: CollectD no longer fails to retrieve Resource Manager metrics. CollectD determines which is the active Resource Manager before it collects ResourceManager metrics.
- SPYG-806: After you install OpenTSDB, you no longer need to configure the clusterID, fqdnID, or VolumeID in the default dashboards.
- SPYG-811: Grafana no longer expects OpenTSDB versions less than or equal to version 2.1. Dashboards now load as expected.
- SPYG-609: Users no longer need to configure an index pattern before viewing logs in Kibana.

S3 Gateway Release Notes

The S3 Gateway was formerly known as the *Object Store with S3-Compatible API*. The release notes for the S3 Gateway component included in the MapR Data Platform contain notes specific to MapR only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior releases, see Previous Versions on page 6578.

S3 Gateway 2.2.0.0 - 2110 (EEP 8.0.0) Release Notes

This section provides reference information, including new features, patches, and known issues for the 2.2.0.0 release of the S3 Gateway.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

S3 Gateway Version	2.2.0.0
Release Date	October 2021
Version Interoperability	See EEP Components and OS Support on page 5536.
Maven Artifacts	https://repository.mapr.com/maven/
Source on GitHub	https://github.com/mapr/minio/
GitHub Release Tag	2.2.0.0-eep-800
Package Names	• mapr-objectstore-client-2.2.0.0 • mapr-objectstore-gateway-2.2.0.0 To view the list of package names, navigate to Package Names for MapR Ecosystem Packs (EEPs) on page 5737, and select your EEP and OS.
Documentation	• S3 Gateway on page 3959 • Installing S3 Gateway on page 204 • Configuring S3 Gateway on page 3961

New in This Release

S3 Gateway 2.2.0.0 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

- Objectstore with S3-Compatible API is now called S3 Gateway in MapR Data Platform product documentation.
- FS mode added for LDAP integration.
- MinIO updated to RELEASE.2021-04-22T15-44-28Z.
- MC (MinIO Client) updated to RELEASE.2021-04-22T17-40-00Z.

Fixes

This data-fabric release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
5b18bbda	2021-04-29	S3-247: Added LDAP integration support for FS mode
6e50767b	2021-08-25	S3-268: Changed log rotation

For complete details, refer to the commit log for this project in GitHub.

Known Issues

- S3-261: The MinIO password is stored in cleartext.

Resolved Issues

- S3-268: Fixed an issue in logrotate that caused the S3 gateway to stop running in the cluster.

S3 Gateway 2.1.0.0 - 2104 (EEP 7.1.0) Release Notes

This section provides reference information, including new features, patches, and known issues for the 2.1.0.0 release of the S3 Gateway.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

S3 Gateway Version	2.1.0.0
Release Date	April 2021
Version Interoperability	See EEP Components and OS Support on page 5536.
Maven Artifacts	https://repository.mapr.com/maven/
Source on GitHub	https://github.com/mapr/minio/
GitHub Release Tag	2.1.0.0-mapr-710
Package Names	• mapr-objectstore-client-2.1.0.0 • mapr-objectstore-gateway-2.1.0.0 To view the list of package names, navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS.
Documentation	• S3 Gateway on page 3959 • Installing S3 Gateway on page 204 • Configuring S3 Gateway on page 3961

New in This Release

S3 Gateway 2.1.0.0 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- Multi-volume support
- Distributed mode support
- Service verifier
- Minio updated to RELEASE.2021-03-17T02-33-02Z
- MC updated to RELEASE.2021-03-23T05-46-11Z

Fixes

This data-fabric release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
69ef372e	2021-03-24	[S3-244] Added new ldap configuration and config migrations
a693008f	2021-03-15	[S3-240] Added service verifier
96c85fbf	2021-03-03	[S3-236] Added support of multi volumes
07af4968	2021-02-26	[S3-167] Added implementation of distributed mode for MapR FS
d6d2bf0d	2021-01-25	[S3-233] Added skip of Streams config during migration
d71a60f9	2021-01-25	[S3-234] Fixed clean start in FS mode

For complete details, refer to the commit log for this project in GitHub.

Known Issues

- None.

Resolved Issues

- S3-234: The objectstore no longer fails to start the first time in FS mode.
- S3-241: The old folder is no longer present after upgrade to new minor version.
- S3-244: CVE-2021-21362 MinIO vulnerability resolved.

S3 Gateway 2.0.0.0 - 2009 (EEP 7.0.0) Release Notes

This section provides reference information, including new features, patches, and known issues for the 2.0.0.0 release of the S3 Gateway.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

S3 Gateway Version	2.0.0.0
Release Date	September 2020
Version Interoperability	See EEP Components and OS Support on page 5536.
Maven Artifacts	https://repository.mapr.com/maven/
Source on GitHub	https://github.com/mapr/minio/

GitHub Release Tag	2.0.0.0-mapr-7.0.0
Package Names	• `mapr-objectstore-client-2.0.0.0` • `mapr-objectstore-gateway-2.0.0.0` To view the list of package names, navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS.
Documentation	• S3 Gateway on page 3959 • Installing S3 Gateway on page 204 • Configuring S3 Gateway on page 3961

New in This Release

- S3 gateway - support NFS in S3 OUT
- Added support for MinIO LDAP integration
- Added support of virtual-hosted–style URI
- Updated the MinIO code base to RELEASE.2020-04-10T03-34-42Z
- Added the MinIO client (mc) to `mapr-objectstore-client` package

Fixes

This data-fabric release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
5f2661441	2020-07-30	[S3-215] Added possibility to enable virtual-hosted–style URI
6a9b7c4d5	2020-07-29	[S3-208] Added passthrough for LDAP properties for
5eb644086	2020-07-29	[S3-195] Removed hardcode of password for ssl
e08e2eabd	2020-05-15	[S3-164] Added FS mode
f5fd04758	2020-04-17	[S3-163] Added support of UID and GID
6a07e9cb2	2020-04-14	[S3-162] Added integration with MapR Kafka
bdb193cff	2020-04-14	[S3-152] Add metadata for files uploaded directly to file system

For complete details, refer to the commit log for this project in GitHub.

Known Issues

- None.

Resolved Issues

- YARN-113: The MapR File System does not have access to s3a when fs.s3a.endpoint property contains the domain as a string.

S3 Gateway 1.0.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes

This section provides reference information, including new features, patches, and known issues for the 1.0.1 release of the S3 Gateway.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

MapR Object Store with S3-compatible API Version	1.0.1
Release Date	February 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	• mapr-objectstore-client-1.0.1 • mapr-objectstore-gateway-1.0.1 Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

None.

Fixes

This MapR release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
7d6a339	2018-12-17	S3-136: Minio GUI defaults to Open first bucket in multi tenant environment

For complete details, refer to the commit log for this project in GitHub.

Known Issues

- YARN-113: The MapR FileSystem does not have access to `s3a` when `fs.s3a.endpoint` property contains the domain as a string.
- S3-122: Events sent by Kafka notification does not contain bucket owner information, in case S3 Gateway is running in `S3-Only` mode.

Resolved Issues

None.

S3 Gateway 1.0.0-1808 (EEP 6.0.0) Release Notes

This section provides reference information, including new features, patches, and known issues for the 1.0.0 release of the S3 Gateway.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

MapR Object Store with S3-compatible API Version	1.0.0

Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	• mapr-objectstore-client-1.0.0 • mapr-objectstore-gateway-1.0.0 Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

This is the first release of the MapR Object Store with S3-compatible API (MapR Object Store). The MapR Object Store is included in EEP repositories beginning with EEP 6.0.0. The MapR Object Store is a S3 REST API compatible object store that enables you to use the S3 API to store and retrieve S3 objects from your cluster. Support for S3 REST API fundamentally provides an S3 REST API proxy server for MapR Data Platform.

Known Issues

- YARN-113: The MapR FileSystem does not have access to `s3a` when `fs.s3a.endpoint` property contains the domain as a string.
- S3-122: Events sent by Kafka notification does not contain bucket owner information, in case S3 Gateway is running in `S3-Only` mode.

Resolved Issues

None.

Myriad Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The release notes for the Myriad component included in the MapR Converged Data Platform contain notes specific to MapR only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Myriad 0.2-1710 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Release notes for the Myriad 0.2 component included in the MapR Converged Data Platform.

Myriad Version	0.2
Release Date	November 2017
MapR Version Interoperability	Pre-MapR 5.2: Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 MapR 5.2 and later: EEP Components and OS Support

last-updated: Apr 11, 2022

Source on GitHub	https://github.com/apache/incubator-myriad
Source Documentation	https://cwiki.apache.org/confluence/display/MYRIAD/Myriad+Home
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in this Release

The following features are new in this release:

- Accept GPU reservations
- Secure Myriad REST API
- Install from a tarball.

GPU Reservation

The Myriad 0.2 GPU reservation feature provides the capability to allocate resources from GPU-capable machines to Myriad. It allows Myriad tasks (Resource manager, Node Manager) to run in Mesos containers with GPU resources. This feature is enabled by default; configuration is not needed.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Description
0c2f74f	2017-08-23	Accept GPU reservations.
970e756	2016-10-12	Add https/ssl support for Myriad REST APIs and UI.
b9c1fda	2016-11-04	Address review comments.
c027b95	2016-11-04	Support Authentication in Myriad UI and REST APIs.

Myriad 0.1-1602 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Release notes for the Myriad 0.1 component included in the MapR Converged Data Platform.

Myriad Version	0.1
Release Date	February 29, 2016
MapR Version Interoperability	Pre-MapR 5.2: Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 MapR 5.2 and later: EEP Components and OS Support
Source on GitHub	https://github.com/apache/incubator-myriad
Source Documentation	https://cwiki.apache.org/confluence/display/MYRIAD/Myriad+Home

Package Names	The following repository is associated with this release: • Mesosphere repository: http://repos.mesosphere.io See Package Names for MapR Ecosystem Packs (EEPs).

Note: MapR does not support Myriad on secure clusters.

New in this Release

This release is the first MapR release of the Myriad component.

Oozie Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The release notes for Oozie component included in the MapR Converged Data Platform contains notes specific to MapR only. More details are available on the Apache Oozie project website.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Oozie 5.2.1.0 Release Notes

The following Oozie 5.2.1.0 component release notes are included in the HPE Ezmeral Data Fabric.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Oozie 5.2.1.200 - 2201 (EEP 8.1.0) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Data Platform Distribution for Apache Oozie. You may also be interested in the Apache Oozie 5.2.1 changelog or the Apache Oozie project homepage.

Version	5.2.1.200
Release Date	January 2022
HPE Version Interoperability	See the Interoperability Matrix, Ecosystem Support Matrix, and Oozie Support Matrix
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	5.2.1.200-eep-810
Maven Artifacts	http://repository.mapr.com/maven/.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Oozie 5.2.1.200 - 2201 introduces the following new feature:

- Starting from EEP-8.1.0, Oozie supports FIPS.

last-updated: Apr 11, 2022

- Updated the following:
  - Netty
  - Derby
  - Jython Standalone
  - JUnit
  - Log4j
  - Jetty
  - Gson
  - Graphviz
  - Commons Collections
  - XML Graphics Commons
  - Commons Compress
  - Netty4
  - Spark to version 3.2.0.0
- Starting from EEP-8.1.0, Oozie does not support Pig and Sqoop.

Fixes

This release by MapR includes the following patches on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
0035192a	2022-02-04	OOZ-331: JMX SSL options always false for MEP7+ releases
a6be3d7d	2022-01-26	OOZ-328: Added mapr-security-web jar to Jetty for JMX
5ca08852	2022-01-25	OOZ-327: Updated log4j v1 to the 1.3.1-mapr
15eb8fc6	2022-01-14	OOZ-326: Fixed error during initial configuration
204aa00b	2022-01-04	OOZ-325: Use mapr-shaded-avatica instead of apache avatica
f6f88010	2021-12-30	OOZ-324: Oozie cannot start on Core-6.2 with MEP-8.1.0
6d9a00bd	2021-12-24	OOZ-322 - fix findAndCopyJar function
4d9572da	2021-12-24	OOZ-285 - drop pig and sqoop from examples
7eef76a6	2021-12-22	OOZ-319: Oozie web UI issue manipulation of pop-up window
1a1a366c	2021-12-21	OOZ-321 Upgrade netty to 4.1.72.Final version

86011971	2021-12-20	OOZ-320: Excluded Log4j 1 from Hive, Hive2 and HCat actions
0f8b97d8	2021-12-20	OOZ-316: Added disruptor for HCatalog action for Log4j 2
95324f7a	2021-12-20	OOZ-316: Added disruptor to Hive2 action for compatibility with Log4j 2.16.0
189dbbfd	2021-12-15	OOZ-314 Upgrade org.apache.derby
7f4e9f15	2021-12-15	OOZ-313 Upgrade jython-standalone
8c128735	2021-12-15	OOZ-310: Updated log4j to 1.3.0-mapr version
abd9e759	2021-12-14	OOZ-311 Upgrade junit to version 4.13.1
6a2a4620	2021-12-13	OOZ-310: log4j updated to 1.2.17-mapr due vulnerability CVE-2019-17571
cc1b5370	2021-12-10	OOZ-307: Fixed mapreduce job on fips node
7d19a060	2021-12-01	OOZ-308: Updated jdom to org.apache.servicemix.bundles.jdom v2.0.5_1
ee9bfd9c	2021-11-19	OOZ-299: Removed netty-3 due vulnerability CVE-2019-20444
1de5c5d2	2021-11-19	OOZ-303: httpclient-4.5.7.jar vulnerability CVE-2020-13956
6bbaa988	2021-11-18	OOZ-301: Updated gson-2.8.5.jar due vulnerability WS-2021-0419
de222d4e	2021-11-18	Updated Jetty version to the latest
185c602e	2021-11-18	OOZ-297: Fixed build after graphviz update
8c78e419	2021-11-18	Updated Hive version for MEP 8.1 release
6263f34f	2021-11-18	OOZ-295: commons-collections4-4.0.jar vulnerability: CVE-2015-4852, CVE-2015-6420, CVE-2015-7501
ed65f36d	2021-11-18	OOZ-297: xmlgraphics-commons-2.3.jar vulnerability: CVE-2020-11988
7490e709	2021-11-18	Updated Hadoop version to correct for this release
1590ff1d	2021-11-18	OOZ-298: Updated commons-compress-1.20.jar due vulnerabilities CVE-2021-35515, CVE-2021-35516, CVE-2021-35517, CVE-2021-36090
6a3c2042	2021-11-18	OOZ-293: Updated Netty4 to the latest version
1e88365a	2021-11-18	OOZ-288: Upgrade Spark version to 3.2.0.0

last-updated: Apr 11, 2022

698577c8	2021-11-18	OOZ-291: Updated HBase version
f6bfb54a	2021-11-01	OOZ-285: Drop Pig and sqoop support
260fc242	2021-10-12	OOZ-283: Excluded hive-exec from spark sharelib
5e5bf269	2021-09-22	OOZ-280: Oozie service failed to start, because jmx agent can't get ssl credentials
1b10e878	2021-09-22	OOZ-254 add BouncyCastle support

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 5.2.1.100 - 2110 (EEP 8.0.0) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Oozie. You may also be interested in the Apache Oozie 5.2.1 changelog or the Apache Oozie project homepage.

Version	5.2.1.100
Release Date	October 2021
HPE Version Interoperability	See the Interoperability Matrix, Ecosystem Support Matrix, and Oozie Support Matrix
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	5.2.1.100-eep-800
Maven Artifacts	http://repository.mapr.com/maven/.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Oozie 5.2.1.100 - 2110 introduces the following new feature:

- Updated Hive libraries to version 2.3.9.
- Updated Jetty to version 9.4.43.v20210629 .
- Updated Spark to version 3.1.2.0.

Fixes

This release by MapR includes the following patches on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment

bd629f49	2021-05-27	OOZ-252: Oozie status server failed with Error message: Insufficient configured threads
51eefe7d	2021-07-05	OOZ-235 - upgrade spark up to 3.1.1
dfe42c6c	2021-07-05	OOZ-255: Update Hive libs to v2.3.9
1b2d0e77	2021-07-12	OOZ-258: update hadoop up to 2.7.6.0-mapr-720 version
c532ee4c	2021-07-19	OOZ-259: Updated jetty to the 9.4.41.v20210516
e7c22985	2021-07-28	OOZ-261: Update Jetty to 9.4.43.v20210629
3bfb1df0	2021-08-10	OOZ-262: Update Jackson v1 and v2 depenencies
99f50b41	2021-08-12	OOZ-263 Update Spark version to 3.1.2.0
9673c8d2	2021-08-16	OOZ-264: Removed avatica from common sharelib to avoid jackson versions conflict
0c9967ed	2021-08-17	OOZ-266: Drop Hive v1.2 from Spark dependencies
c845882c	2021-08-17	OOZ-265: force update dependencies
5f8b87ad	2021-08-17	Updated Hive SNAPSHOTS to correct version
e5aec661	2021-08-18	OOZ-267/OOZ-266: Removed log4jv2 from spark action and added missing libraries

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 5.2.1.0 - 2104 (EEP 7.1.0) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Oozie. You may also be interested in the Apache Oozie 5.2.1 changelog or the Apache Oozie project homepage.

Version	5.2.1.0
Release Date	April 2021
HPE Version Interoperability	See theInteroperability Matrix,Ecosystem Support Matrix, andOozie Support Matrix
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	5.2.1.0-mapr-710
Maven Artifacts	http://repository.mapr.com/maven/.
Package Names	Navigate tohttps://package.mapr.hpe.com/releases/MEP/ and select your MEP and OS to view the list of package names

New in This Release

Oozie 5.2.1.0 - 2104 introduces the following new feature:

- Service verifier

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
0425addf	2021-04-13	OOZ-244: Jetty updated to 9.4.39.v20210325 due CVE-2021-28165
15973d0b	2021-03-29	OOZ-242: Cleaned Sqoop sharelib from extra dependencies after update Avro dependencies
84854cb9	2021-03-24	OOZ-243: Update HBase dependency to 1.4.13.0-mapr-SNAPSHOT
019b3cab	2021-03-22	OOZ-240/241: Updated Hadoop version to 2.7.5.0 and Avro to 1.9.2
70dc23b7	2021-03-22	OOZ-231: add service verifier to Oozie package
b4b6b472	2021-03-18	OOZ-239: warden.oozie.conf fix
bda832b7	2021-03-17	OOZ-236: Remove oozie, oozie-error, oozie-instrumentation and oozie-audit logs older than 2 weeks
f9cb8c03	2021-03-15	OOZ-237: Updated Jetty to 9.4.38.v20210224
bd4bc437	2021-03-01	Backported OOZIE-3601 Upgrade quartz to 2.3.2 (dengliming via matijhs)
c563a682	2021-03-01	Backported OOZIE-3549 Add back support for truststore passwords (matijhs via asalamon74)

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 5.2.0.0 Release Notes

The following Oozie 5.2.0.0 component release notes are included in the MapR Converged Data Platform.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Oozie 5.2.0.100 - 2101 (EEP 7.0.1) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Oozie. You may also be interested in the Apache Oozie 5.2.0 changelog or the Apache Oozie project homepage.

Version	5.2.0.100
Release Date	January 2020
HPE Version Interoperability	See theInteroperability Matrix,Ecosystem Support Matrix, andOozie Support Matrix

Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	5.2.0.100-mapr-701
Maven Artifacts	http://repository.mapr.com/maven/.
Package Names	Navigate tohttps://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Oozie 5.2.0.100 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
3dcde87	2020-09-10	OOZ-212 Fix SSLServerConnectorFactory setting key password instead of keystore password
e2c99a8	2020-10-09	OOZ-196 Migrate UI to JQuery 3.5.1
dde8750	2020-10-09	OOZ-214 Detect OOZIE_URL for secured clusters
a9a8c5f	2020-11-10	OOZ-216 Use Jackson BOM
De4482b b139099	2020-12-01	OOZ-218 Update Spark to 2.4.7.0 and Scala to 2.12
cae4631	2020-12-04	OOZ-221: Update XercesImpl to 2.12.0
8ac9eb9	2020-12-09	OOZ-222: Update maprbuildversion*.jar at Oozie_home after configure.sh
cbcd590	2020-12-14	OOZ-219: Remove redundant warning messages in oozie status command
6327a0e	2020-12-17	OOZ-224: Updated jetty to 9.4.35.v20201120
d40fce6	2020-12-22	OOZ-225: Fixed spark-standalone mode for spark action
623164e	2021-01-05	OOZ-227: Fixed spark-hive actions

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 5.2.0.0 - 2009 (EEP 7.0.0) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Oozie 5.2.0.0 for EEP 7.0.0.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You may also be interested in the Apache Oozie 5.2.0 changelog or the Apache Oozie project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	5.2.0.0
Release Date	September 2020
MapR Version Compatibility	See EEP Components and OS Support on page 5536 and Oozie Support Matrix on page 5636.
GitHub Source	https://github.com/mapr/oozie
GitHub Release Tag	5.2.0.0-mapr-700
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
1264425	2020-05-26	OOZ-171 Update Fasterxml Jackson to 2.9.10
577ef4e5	2020-04-29	OOZ-173 Update Netty v3 to to 3.10.6, Netty v4 to 4.1.49
1bdd3da	2020-04-29	OOZ-177 Update Quartz to v2.3.2
0aa596c	2020-05-07	OOZ-183 Update Guava to 28.2-jre
d89e5c8	2020-05-19	OOZ-184 Update htrace to 4.2.0
d89e5c8	2020-08-04	OOZ-200 Update JLine
006bde8	2020-04-29	OOZ-159 Remove classifier for Pig dependency as deprecated
afb9706	2020-05-04	OOZ-185 Exclude org.apache.tomcat artifacts from hive-service
b6ee783	2020-08-18	OOZ-188 Secure JMX Support
c25b4081	2020-06-05, 2020-06-16	OOZ-187 Disable TLSv1, SSLv2Hello, TLSv1.1 by default

Commit	Date (YYYY-MM-DD)	Comment
189c4f0	2019-05-30	OOZ-141 Avoid copying commons-configuration since it causes uninstallation warning
1bb20e0c	2020-07-07	OOZ-191 Use oozie_fqdn in base url only. Configure 11000 to 11443 redirection
9a2c512	2020-07-17	OOZ-192: OOZIE-3592 Do not print misleading SecurityException for successful jobs
fb203475	2020-07-24	OOZ-197 Copy mapr-ojai driver to common sharelibs
e215051a	2020-08-12	OOZ-189: Uncomment OOZIE_CLIENT_OPTS at oozie-client-env.sh for secure client node
df79451b	2020-08-07	OOZ-198 Update Avatica to 1.17.0 (to conform Hive version), add Avatica to common libs.
6239016b	2020-08-13	OOZ-205: Oozie spark action fails at yarn-cluster/client mode
245cf50	2020-08-13	OOZ-206: Sqoop action on unsecure cluster fails
e4c23e5	2020-08-26	OOZ-211 Update Disruptor lib to 3.4.2 for Oozie Hive Sharelibs
7f21deb0	2020-08-27	OOZ-207 Update slf4j to 1.7.25

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 5.1.0 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following Oozie 5.1.0 component release notes are included in the MapR Converged Data Platform.

Oozie 5.1.0.800 - 2201 (EEP 6.3.6) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following notes relate specifically to the MapR Technologies Distribution for Apache Oozie. You may also be interested in the Apache Oozie 5.1.0 changelog or the Apache Oozie project homepage.

Version	5.1.0.800
Release Date	January 2022

HPE Version Interoperability	See the Interoperability Matrix,Ecosystem Support Matrix, and Oozie Support Matrix
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	5.1.0.800-mapr-636
Maven Artifacts	http://repository.mapr.com/maven/.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP(MEP) and OS to view the list of package names.

New in This Release

Oozie 5.1.0.800 - 2201 introduces dependency updates, but no significant new features.

Fixes

This release by MapR Technologies includes the following patches on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
86c67575	2022-01-25	OOZ-327: Updated log4j v1 to the 1.3.1-mapr
0b757355	2022-01-18	OOZ-315: Excluded jackson-core-asl and jackson-mapper-asl 1.9.13 version
a1400fe0	2022-01-14	OOZ-326: Fixed error during initial configuration
9e26bd3c	2022-01-04	OOZ-325: Use mapr-shaded-avatica instead of apache avatica
f10dbe6d	2021-12-24	OOZ-322: fix findAndCopyJar function
2f0edd31	2021-12-22	OOZ-319: Oozie web UI issue manipulation of pop-up window
942d8f97	2021-12-21	OOZ-321 Upgrade netty to 4.1.72.Final version
11a72a1b	2021-12-20	OOZ-320: Excluded Log4j 1 from Hive, Hive2 and HCat actions
740c9141	2021-12-17	OOZ-316: Updated disruptor to 3.4.2 version for compatibility with Log4j 2.16.0
ad8b7fcc	2021-12-17	OOZ-318: Exclude avatita from Oozie core
5d916535	2021-12-15	OOZ-314: Upgrade org.apache.derby
64ff4a92	2021-12-15	OOZ-313: Upgrade jython-standalone
6c5c7a93	2021-12-14	OOZ-311: Upgrade junit to version 4.13.1
6859fb84	2021-12-13	OOZ-310: log4j updated to 1.2.17-mapr due vulnerability CVE-2019-17571

ec4ea5ac	2021-12-01	OOZ-308: Updated jdom to org.apache.servicemix.bundles.jdom v2.0.5_1
a724355e	2021-11-23	OOZ-299: Removed netty-3 due vulnerability CVE-2019-20444
20917fec	2021-11-23	OOZ-303: httpclient-4.5.7.jar vulnerability CVE-2020-13956
997f84c5	2021-11-23	OOZ-301: Updated gson-2.8.5.jar due vulnerability WS-2021-0419
50747319	2021-11-23	OOZ-305: Updated jackson-databind to 2.11.1
996fd1e0	2021-11-23	Backport OOZIE-3507 Upgrade to Dozer 6 (asalamon74 via kmarton)
d5d75394	2021-11-23	OOZ-304: jackson-mapper-asl-1.9.13.jar vulnerability: CVE-2019-10172
596ff42f	2021-11-23	OOZ-295: commons-collections4-4.0.jar vulnerability: CVE-2015-4852, CVE-2015-6420, CVE-2015-7501
b11f43c5	2021-11-23	OOZ-300: Updated Jetty to 9.4.44.v20210927
cc9bac3d	2021-11-23	OOZ-297: xmlgraphics-commons-2.3.jar vulnerability: CVE-2020-11988
7af70b42	2021-11-23	OOZ-293: Updated Netty4 to the latest version

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 5.1.0.700 - 2110 (EEP 6.3.5) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following notes relate specifically to the MapR Data Platform Distribution for Apache Oozie. You may also be interested in the Apache Oozie 5.1.0 changelog or the Apache Oozie project homepage.

Version	5.1.0.700
Release Date	October 2021
HPE Version Interoperability	See the Interoperability Matrix,Ecosystem Support Matrix, and Oozie Support Matrix
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	5.1.0.700-mapr-635
Maven Artifacts	http://repository.mapr.com/maven/.

last-updated: Apr 11, 2022

Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP(MEP) and OS to view the list of package names.

New in This Release

Oozie 5.1.0.700 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

- Updated Jetty to version 9.4.41.v20210516 .

Fixes

This release by MapR includes the following patches on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
51f298db	2021-07-19	OOZ-259 - Update jetty to the 9.4.41.v20210516

Known Issues and Limitations

- You must use `hive-exec-core`(not `hive-exec`) library to apply Sqoop action for Hive import in Parquet and Avro formats.

Resolved Issues

- None.

Oozie 5.1.0.600 - 2104 (EEP 6.3.4) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Oozie. You may also be interested in the Apache Oozie 5.1.0 changelog or the Apache Oozie project homepage.

Version	5.1.0.600
Release Date	April 2021
HPE Version Interoperability	See theInteroperability Matrix,Ecosystem Support Matrix, andOozie Support Matrix
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	5.1.0.600-mapr-634
Maven Artifacts	http://repository.mapr.com/maven/.
Package Names	Navigate tohttps://package.mapr.hpe.com/releases/MEP/ and select your MEP and OS to view the list of package names

New in This Release

Oozie 5.1.0.600 - 2104 introduces bug fixes and updates, but no significant new features.

Fixes

This HPE release includes the following fixes on the base release:For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
751e7d6f	2021-04-13	OOZ-244: Jetty updated to 9.4.39.v20210325 due CVE-2021-28165
ef7d0f6c	2021-03-15	OOZ-237: Updated Jetty to 9.4.38.v20210224

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 5.1.0.500 - 2101 (EEP 6.3.2) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Oozie. You may also be interested in the Apache Oozie 5.1.0 changelog or the Apache Oozie project homepage.

Version	5.1.0.500
Release Date	January 2020
HPE Version Interoperability	See theInteroperability Matrix,Ecosystem Support Matrix, andOozie Support Matrix
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	5.1.0.500-mapr-632
Maven Artifacts	http://repository.mapr.com/maven/.
Package Names	Navigate tohttps://package.mapr.hpe.com/releases/MEP/ and select your MEP and OS to view the list of package names

New in This Release

Oozie 5.1.0.500 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
e3b785c	2020-09-10	OOZ-212 Fix SSLServerConnectorFactory setting key password instead of keystore password
e5d40c7	2020-10-09	OOZ-214 Detect OOZIE_URL for secured clusters
3248997	2020-11-10	OOZ-216 Use Jackson BOM
0f62d15	2020-12-04	OOZ-221: Update XercesImpl to 2.12.0

8a2bdb8	2020-12-09	OOZ-222: Update maprbuildversion*.jar at Oozie_home after configure.sh
5c47f96	2020-12-17	OOZ-224: Updated jetty to 9.4.35.v20201120
88b49b1	2020-12-30	OOZ-226: Oozie hive action protobuf error
05da0bf	2021-01-05	OOZ-227: Fixed spark-hive actions

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 5.1.0.400 - 2009 (EEP 6.3.1) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Oozie 5.1.0.400 for EEP 6.3.1.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

You may also be interested in the Apache Oozie 5.1.0 changelog or the Apache Oozie project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	5.1.0.400
Release Date	September 2020
MapR Version Compatibility	See EEP Components and OS Support on page 5536 and Oozie Support Matrix on page 5636.
GitHub Source	https://github.com/mapr/oozie
GitHub Release Tag	5.1.0.400-mapr-631
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
e024242	2020-04-21	OOZ-178 Update Apache Commons Libs

Commit	Date (YYYY-MM-DD)	Comment
6385250	2020-04-29	OOZ-171 Update Fasterxml Jackson to 2.7.9 (Jackson Databind to 2.7.9.7)
91e080d	2020-04-29	OOZ-173 Update Netty v3 to to 3.10.6, Netty v4 to 4.1.49
2aa707f	2020-04-29	OOZ-177 Update Quartz to v2.3.2
8eaf813	2020-04-29	OOZ-159 Remove classifier for Pig dependency as deprecated
026da11	2020-05-19	OOZ-185 Exclude org.apache.tomcat artifacts from hive-service
2f9c303 651050d	2020-05-20 2020-05-23	OOZ-167 add JMX option handling
8a5899f	2020-06-05	OOZ-187 Disable TLSv1, SSLv2Hello, TLSv1.1 by default
cfd1d17	2020-06-22	OOZ-141 Avoid copying commons-configuration since it causes uninstallation warning
737f517	2020-07-08	OOZ-191 Use oozie_fqdn in base url only. Configure 11000 to 11443 redirection
ae7903b	2020-07-17	OOZ-192: OOZIE-3592 Do not print misleading SecurityException for successful jobs
9dca36e	2020-07-24	OOZ-197 Copy mapr-ojai driver to common sharelibs
2fcfd03	2020-08-12	OOZ-189: Uncomment OOZIE_CLIENT_OPTS at oozie-client-env.sh for secure client node

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 5.1.0.300-1912 (EEP 6.3.0) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 5.1.0 changelog or the Apache Oozie project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	5.1.0.300

Release Date	December 2019
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/
GitHub Release Tag	5.1.0.300-mapr-630
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- Added configurable HTTP security headers. For more information, see Configuring Security Headers for Web Servers for Oozie on page 3993.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
50117f1	2019-09-24	OOZ-147: Update Commons BeanUtils to 1.9.4
f4e3f0a	2019-06-06	OOZ-138: OozieDB only updates when DB version is different
8bcbd31	2019-07-23	OOZ-145: Update Hive to 2.3.5, add 'hive-maprdb-json-handler' dependency
5f30c55	2019-08-07	OOZ-145: Add 'hive-maprdb-json-common' dependency
fe7108bb	2019-06-05	OOZ-140: Backport fix Apache's OOZIE-3415 from 5.2.0
fe717f57 48d4995a	2019-05-19 2019-06-04	OOZ-136: Prevent scripts execution with incomplete libs
646ad22a	2019-05-30	OOZ-141: Exclude unused artifacts from libs
16d9ae2a	2019-10-08	CORE-306: HTTP Security headers
0e5e119a	2019-09-25	OOZ-146: Avoid SLF4J implementations in outcome libs

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 5.1.0.200-1904 (EEP 6.2.0) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 5.1.0.0 changelog or the Apache Oozie project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	5.1.0.200
Release Date	April 2019
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/
GitHub Release Tag	5.1.0.200-mapr-620
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- Spark artifacts version updated to version 2.4.0.
- Jetty version upgrade version 9.3.25.v20180904.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
72cd9f5	2019-03-06	OOZ-114:Update Spark version at Oozie to v2.4.0
bde4bd1	2019-03-06	OOZ-123: version used by latest version is lower than previous version
7b6421a	2019-03-05	OOZ-120: Jetty version upgrade v9.3.25.v20180904
9e1113	2019-02-28	OOZ-121: Duplicate jersey-core jars make the AM link unavailable
1367fd0	2019-05-16	OOZ-137: YARN jobs are submitted as the submitter instead of administrator, if they are different users. Impersonation did not work only for several cases

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 5.1.0-1904 (EEP 6.1.1) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

last-updated: Apr 11, 2022

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 5.1.0.0 changelog or the Apache Oozie project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	5.1.0
Release Date	April 2019
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/
GitHub Release Tag	oozie-5.1.0-mapr-1904
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- Spark artifacts version updated to version 2.3.3.
- Jetty version upgrade version 9.3.25.v20180904.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
0fca106c	2019-03-25	OOZ-125:Update Spark version at Oozie to v2.3.3
31e71be	2019-03-06	OOZ-123: version used by latest version is lower than previous version
017e3ef	2019-03-05	OOZ-120: Jetty version upgrade v9.3.25.v20180904
bb24c68	2019-02-28	OOZ-121: Duplicate jersey-core jars make the AM link unavailable
1367fd0	2019-05-16	OOZ-137: YARN jobs are submitted as the submitter instead of administrator, if they are different users. Impersonation did not work only for several cases

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 5.1.0.0-1901 (EEP 6.1.0) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 5.1.0.0 changelog or the Apache Oozie project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	5.1.0.0
Release Date	February 2019
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/
GitHub Release Tag	oozie-5.1.0-mapr-1901
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • `mapr-oozie-5.1.0<>.rpm` • `mapr-oozie_5.1.0<>.deb`

New in This Release

None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
c8769ae	2018-12-14	MAPR-OOZIE-96: Oozie sends callback to wrong URL
a01531d	2018-12-18	MAPR-OOZIE-93: Oozie Spark action executor accesses local volumes on remote nodes via NFS
a016307	2018-12-21	MAPR-OOZIE-97: Support 4 digits version scheme for Oozie
75c9fa2	2018-12-27	MAPR-OOZIE-82: spark-sql-kafka jar is missing in oozie shared lib for Spark
24f6724	2018-12-28	MAPR-OOZIE-95: 'JavaPackage' object is not callable when launching Oozie Workflow with SparkSession

Known Issues and Limitations

None.

Resolved Issues

None.

last-updated: Apr 11, 2022

Oozie 4.3.0 Release Notes

The following Oozie 4.3.0 component release notes are included in the MapR Converged Data Platform.

Oozie 4.3.0 - 2009 (EEP 5.0.5) Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Oozie 4.3.0 for EEP 5.0.5.

You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.3.0
Release Date	September 2020
MapR Version Compatibility	See EEP Components and OS Support on page 5536 and Oozie Support Matrix on page 5636.
GitHub Source	https://github.com/mapr/oozie
GitHub Release Tag	oozie-4.3.0-mapr-2009
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
80ca7c6 d502bd6	2020-03-27 2020-04-03	OOZ-163: Disable TLSv1 and TLSv1.1 protocol for Oozie
306ef78	2020-06-04	OOZ-186: Oozie Java JMX Server Insecure Configuration Remote Code Execution Vulnerability
d35258a	2020-06-24	OOZ-178 Update Apache Commons Libs

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 4.3.0-1912 (EEP 5.0.4) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.3.0
Release Date	December 2019
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/
GitHub Release Tag	oozie-4.3.0-mapr-1912
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
bd171ed3 55ce50ff	2019-09-24 2019-10-28	OOZ-147: Update Commons Configuration to 1.10
5043e183	2019-06-06	OOZ-138: OozieDB only updates when DB version is different
eddaa80c	2019-09-25	OOZ-146: Avoid SLF4J implementations in outcome libs

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 4.3.0-1904 (EEP 6.0.2) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.3.0
Release Date	April 2019

MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/
GitHub Release Tag	oozie-4.3.0-mapr-1904
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- Added new option: oozie.action.null.args.allowed.
- Spark artifacts version updated to version 2.3.3.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
19825b4f	2019-04-08	OOZ-126 oozie jobs are failing with Null pointer
d9ee745	2019-02-28	OOZ-121: Duplicate jersey-core jars make AM link unavailable
95b685c	2019-03-25	OOZ-125: Update Spark version at Oozie to v2.3.3
882b9f4	2019-03-05	OOZ-122: multiple versions of metrics-core in oozie.war

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 4.3.0-1904 (EEP 5.0.3) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.3.0
Release Date	April 2019
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636

Source on GitHub	https://github.com/mapr/oozie/
GitHub Release Tag	oozie-4.3.0-mapr-1904
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- Added new option: oozie.action.null.args.allowed.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
be25c85	2019-04-08	OOZ-126 oozie jobs are failing with Null pointer
e6b28e7	2019-04-11	OOZ-121 Duplicate jersey-core jars make AM link unavailable

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 4.3.0-1904 (EEP 4.1.4) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.3.0
Release Date	April 2019
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/
GitHub Release Tag	oozie-4.3.0-mapr-1904
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

last-updated: Apr 11, 2022

New in This Release

- Added new option: oozie.action.null.args.allowed.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
e2f66dd	2019-04-08	OOZ-126 oozie jobs are failing with Null pointer
1c18337	2019-04-11	OOZ-121 Duplicate jersey-core jars make AM link unavailable

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 4.3.0-1901 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.3.0
Release Date	February 2019
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/
GitHub Release Tag	oozie-4.3.0-mapr-1901
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • `mapr-oozie-4.3.0<>.rpm` • `mapr-oozie_4.3.0<>.deb`

New in This Release

None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
07ef7d5	2018-12-18	MAPR-OOZIE-93: Oozie Spark action executor accesses local volumes on remote nodes via NFS

Known Issues and Limitations

None.

Resolved Issues

None.

Oozie 4.3.0-1808 (EEP 6.0.0) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.3.0
Release Date	September 2018
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/tree/4.3.0-mapr-mep-6.x-1808
GitHub Release Tag	oozie-4.3.0-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • `mapr-oozie-4.3.0<>.rpm` • `mapr-oozie_4.3.0<>.deb`

New in This Release

Updated sharelib versions:

- Spark 2.3.0
- Hive 2.3.0
- Sqoop 1.4.7

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
5f8e992	2018-03-14	MAPR-30894: High frequency LAST_ONLY coord job with many past time actions killed Oozie server
0a697e5	2018-04-27	Updated TDCH version to 1.5.4
cb45b30	2018-05-16	MAPR-OOZIE-42: Updated Hive version to 2.3
9385ca6	2018-05-18	MAPR-31411: Updated Tomcat version to 6.0.53
26d94f7	2018-06-08	MAPR-OOZIE-43: Updated Spark version to 2.3
b1f2b87	2018-06-14	MAPR-OOZIE-60: Used mapr-security-web for getting SSL keystore password
d0ac12c	2018-08-05	Backport OOZIE-2807 Oozie gets RM delegation token even for checking job status

Known Issues and Limitations

None.

Resolved Issues

None.

Oozie 4.3.0-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	4.3.0
Release Date	September 2018
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/tree/4.3.0-mapr-mep-3.x-1808 https://github.com/mapr/oozie/tree/4.3.0-mapr-mep-4.x-1808 https://github.com/mapr/oozie/tree/4.3.0-mapr-mep-5.x-1808
GitHub Release Tag	oozie-4.3.0-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • `mapr-oozie-4.3.0<>.rpm` • `mapr-oozie_4.3.0<>.deb`

New in This Release

None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
5f8e992	2018-03-14	MAPR-30894: High frequency LAST_ONLY coord job with many past time actions killed Oozie server
9385ca6	2018-05-18	MAPR-31411: Updated Tomcat version to 6.0.53
d0ac12c	2018-08-05	Backport OOZIE-2807 Oozie gets RM delegation token even for checking job status

Known Issues and Limitations

None.

Resolved Issues

None.

Oozie 4.3.0-1803 (EEP 5.0.0) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage.

Version	4.3.0
Release Date	March 2018
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	oozie-4.3.0-mapr-1803
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • `mapr-oozie-4.3.0<>.rpm` • `mapr-oozie_4.3.0<>.deb`

New in This Release

This release includes fix for [CVE-2017-15712] Apache Oozie Server vulnerability.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
7f91032	2018-02-06	MAPR-OOZIE-37: Enable Oozie client impersonation by default

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
855a250	2018-02-09	Backport OOZIE-3147 Misleading documentation of oozie.service.PurgeService.purge.limit configuration property
6aed912	2018-02-12	MAPR-30596: Include maprbuildversion jar into oozie shared lib for spark
92c4b24	2018-02-27	MAPR-OOZIE-38: Oozie should be able to work with Spark 2.2

Known Issues and Limitations

None.

Resolved Issues

None.

Oozie 4.3.0-1803 (EEP 4.1.1) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage.

Version	4.3.0
Release Date	March 2018
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	oozie-4.3.0-mapr-1803
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • `mapr-oozie-4.3.0<>.rpm` • `mapr-oozie_4.3.0<>.deb`

New in This Release

This release includes fix for [CVE-2017-15712] Apache Oozie Server vulnerability.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
7f91032	2018-02-06	MAPR-OOZIE-37: Enable Oozie client impersonation by default
855a250	2018-02-09	Backport OOZIE-3147 Misleading documentation of oozie.service.PurgeService.purge.limit configuration property

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
6aed912	2018-02-12	MAPR-30596: Include maprbuildversion jar into oozie shared lib for spark

Known Issues and Limitations

None.

Resolved Issues

None.

Oozie 4.3.0-1803 (EEP 3.0.3) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage.

Version	4.3.0
Release Date	March 2018
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	oozie-4.3.0-mapr-1803
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • `mapr-oozie-4.3.0<>.rpm` • `mapr-oozie_4.3.0<>.deb`

New in This Release

This release includes fix for [CVE-2017-15712] Apache Oozie Server vulnerability.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
7f91032	2018-02-06	MAPR-OOZIE-37: Enable Oozie client impersonation by default
855a250	2018-02-09	Backport OOZIE-3147 Misleading documentation of oozie.service.PurgeService.purge.limit configuration property
6aed912	2018-02-12	MAPR-30596: Include maprbuildversion jar into oozie shared lib for spark
92c4b24	2018-02-27	MAPR-OOZIE-38: Oozie should be able to work with Spark 2.2

Known Issues and Limitations

None.

Resolved Issues

None.

Oozie 4.3.0-1801 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage.

Version	4.3.0
Release Date	February 2018
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	oozie-4.3.0-mapr-1801
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • `mapr-oozie-4.3.0<>.rpm` • `mapr-oozie_4.3.0<>.deb`

New in This Release

No new features.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
774411c	2017-12-29	Backport OOZIE-1401: Enable PurgeCommand to purge the workflow jobs without end_time
d55de42	2018-01-16	MAPR-OOZIE-28: Add restart file for Oozie
75d0793	2018-01-16	MAPR-30440: Add mysql jdbc library by default

Known Issues and Limitations

MAPR-30627: Oozie Sqoop-Hive job fails on unsecured clusters starting from the Oozie 4.3.0-1710 release.

Note: The `sqoop jar` is released earlier than Oozie 4.3.0-1710.

Workaround: Manually update `sqoop share lib` in Oozie by using these commands:

```
cp /opt/mapr/sqoop/sqoop-1.4.6/sqoop-1.4.6-mapr-&lt;sqoop_version&gt;.jar
          /opt/mapr/oozie/oozie-4.3.0/share/lib/sqoop/
```

```
{OOZIE_HOME}/bin/oozie-setup.sh sharelib create -fs
maprfs:/// -locallib /opt/mapr/oozie/oozie-4.3.0/share
```

```
{OOZIE_HOME}/bin/oozie admin -sharelibupdate
```

Resolved Issues

None.

Oozie 4.3.0-1710 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage

Version	4.3.0
Release Date	November 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	oozie-4.3.0-mapr-1710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- Simplified Security - Starting in the MapR 6.0 and EEP 4.0 releases, you can use the "Enable Security" check box in the installer to enable security for the core platform and the installed ecosystem components. Alternatively, you can use the `configure.sh -secure` command to enable security for the core and the ecosystem components. See Configuring Oozie on a Secure Cluster on page 3990 for details on how this impacts Oozie.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
815c5c9	2017-08-08	MAPR-28720: Fix loading of Spark Driver UI
d22a566	2017-08-22	MAPR-28880: Fix Oozie so it works with TimeLine Server
b559b59	2017-08-23	MAPR-27486: Fix Oozie SSL configuration so it is not reset after calling ./configure.sh -R
2e92787	2017-09-07	MAPR-29044: Allow setting custom keystore for SSL

Known Issues and Limitations

- Oozie works only with Spark 2.1.0 and Hive 2.1.

Resolved Issues

- None.

Oozie 4.3.0-1707 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage

Version	4.3.0
Release Date	August 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	oozie-4.3.0-mapr-1707
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

No new features.

Fixes

No new fixes.

Known Issues and Limitations

- Oozie works only with Spark 2.1.0 and Hive 2.1.

Resolved Issues

- None.

Oozie 4.3.0-1703 Release Notes

The notes below relate specifically to thhive MapR Distribution for Apache Hadoop. The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.3.0 changelog or the Apache Oozie project homepage

Version	4.3.0
Release Date	April 2017
MapR Version Interoperability	See EEP 3.0 Components and OS Support on page 5575
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	oozie-4.3.0-mapr-1703
Maven Artifacts	https://repository.mapr.com/maven/

Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release
Added support for Hive 2.1 and Spark 2.1.0.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
583940c	2017-01-26	MAPR-25776: Oozie Hive2 action failed with " Delegation token only supported over remote client with kerberos authentication" on kerberized cluster
6855624	2017-01-27	MAPR-23180: Oozie temporary files moved to System.getProperty("java.io.tmpdir")/oozieTmp
33c6fb0	2017-02-07	MAPR-25989: Added Tez dependencies to Hive action
3c478bb	2017-02-20	MAPR-26195: Add default Tez configuration to Oozie
c537c84	2017-02-21	MAPR-26191: Oozie servers cannot authenticate on MapR-SASL cluster when HA enabled
7ba0945	2017-02-27	MAPR-26256: Changed Hive dependencies for Spark-Hive action
ca6b2ca	2017-03-13	APR-26454: Sqoop import to Hive from Oozie failed
b8097aa	2017-03-13	MAPR-26429: Oozie web UI failed HTTP Status 500

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- Oozie works only with Spark 2.1.0 and Hive 2.1.

Resolved Issues

- None.

Oozie 4.2.0 Release Notes

The following Oozie 4.2.0 component release notes are included in the MapR Converged Data Platform.

Oozie 4.2.0-1710 (EEP 2.x) Release Notes

The notes below relate specifically to MapR's Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.2.0 changelog or the Apache Oozie project http://oozie.apache.org/

Version	4.2.0
Release Date	November 2017

MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	oozie-4.2.0-mapr-mep-2.x-1710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

No new features.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
7ba93ff	2017-10-16	MAPR-29674: Fixes Oozie sqoop action failure that results in java.lang.NoClassDefFoundError: org/apache/commons/lang3/StringUtils

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 4.2.0-1710 (EEP 1.x) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	4.2.0
Release Date	November 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	oozie-4.2.0-mapr-1710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

No new features.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
102af73c	2017-10-16	MAPR-29674: Fixes Oozie sqoop action failure that results in java.lang.NoClassDefFoundError: org/apache/commons/lang3/StringUtils

Oozie 4.2.0-1707 (EEP 2.x) Release Notes

The notes below relate specifically to MapR's Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.2.0 changelog or the Apache Oozie project http://oozie.apache.org/

Version	4.2.0
Release Date	August 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	oozie-4.2.0-mapr-mep-2.x-1707
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

No new features.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
7964b2e	2017/07/20	IN-751: Removes ExtJS wget command from Oozie installation

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 4.2.0-1707 (EEP 1.x) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	4.2.0
Release Date	August 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536

last-updated: Apr 11, 2022

Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	oozie-4.2.0-mapr-1707
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

No new features.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
7c24a1f	2017/07/20	IN-751: Removes ExtJS wget command from Oozie installation

Oozie 4.2.0-1703 (EEP 2.x) Release Notes

The notes below relate specifically to MapR's Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.2.0 changelog or the Apache Oozie project http://oozie.apache.org/

Version	4.2.0
Release Date	April 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	4.2.0-mapr-mep-2.x-1703
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

No new features.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
c3f5aed	2017-03-13	OOZIE-2581: Oozie should reset SecurityManager in finally block
ec1cc26	2017-03-14	MAPR-26454: Sqoop import to hive from Oozie failed

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Oozie 4.2.0-1703 (EEP 1.x) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	4.2.0
Release Date	April 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	oozie-4.2.0-mapr-1703
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) on page 5737.

New in This Release

No new features.

Fixes

This MapR release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
06a891e	2017-02-24	OOZIE-2581: Oozie should reset SecurityManager in finally block
c5e2d2b	2017-03-13	MAPR-26454: Sqoop import to hive from Oozie failed

For complete details, refer to the commit log for this project in GitHub.

Oozie 4.2.0-1611 (EEP 2.x) Release Notes

The notes below relate specifically to the MapR Converged Data Platform. You may also be interested in the Apache Oozie 4.2.0 changelog or the Apache Oozie project homepage.

Oozie Version	4.2.0
Release Date	December 9, 2016
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/oozie/tree/4.2.0-mapr-mep-2.x-1611
GitHub Release Tag	4.2.0-mapr-mep-2.x-1611
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

last-updated: Apr 11, 2022

New in This Release

Oozie 4.2.0-1611 introduces the following enhancements or MapR platform-specific behavior changes:

- Oozie 4.2.0 bundled with Pig 0.16
- Oozie 4.2.0 bundled with Spark 2.0.1

Fixes

This MapR release includes the following fixes on the base Apache release.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
abbd4b9	2016-10-19	MAPR-24455: Changed the Pig version to 0.16.0.
900615e	2016-10-26	MAPR-24455: Changed the Spark version to 2.0.1.

For details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

This version of Oozie 4.2.0-1611, which is included in EEP 2.0, is not compatible with Spark 1.6.1 or earlier versions of Spark.

Resolved Issues

None.

Oozie 4.2.0-1611 (EEP 1.x) Release Notes

The notes below relate specifically to the MapR distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.2.0 changelog or the Apache Oozie project homepage.

Version	4.2.0
Release Date	December 9, 2016
MapR Version Interoperability	See EEP Components and OS Support
Source on GitHub	https://github.com/mapr/oozie
GitHub Release Tag	oozie-4.2.0-mapr-1611
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • `mapr-oozie-4.2.0.201612061048-1.noarch.rpm` • `mapr-oozie-internal-4.2.0.201612061048-1.noarch.rpm` • `mapr-oozie_4.2.0.201612061048_all.deb` • `mapr-oozie-internal_4.2.0.201612061048_all.deb`

New in This Release

No new features.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
f53f142	2016-11-14	OOZIE-2277: Honor oozie.action.sharelib.for.spark in Spark jobs.
5549352	2016-11-16	MAPR-24344: Fix Spark yarn-cluster and yarn-client jobs.
ade8bdb	2016-11-17	MAPR-25283: Oozie failed to install if configure.sh was not executed.

Known Issues and Limitations

None.

Resolved Issues

None.

Oozie 4.2.0-1609 Release Notes

The notes below relate specifically to the MapR Converged Data Platform. You may also be interested in the Apache Oozie 4.2.0 changelog or the Apache Oozie project homepage.

Oozie Version	4.2.0
Release Date	September 30, 2016
MapR Version Interoperability	See theInteroperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/tree/4.2.0-mapr-1609
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • `mapr-oozie-4.2.0.201609272055-1.noarch.rpm` • `mapr-oozie_4.2.0.201609272055_all.deb` • `mapr-oozie-internal-4.2.0.201609272055-1.noarch.rpm` • `mapr-oozie-internal_4.2.0.201609272055_all.deb`

Fixes

GitHub Commit	Data (YYYY-MM-DD)	Comment
0af6095	2016-08-31	MAPR-24393: The issue causing Oozie jobs to fail when Centralized Logging is enabled has been fixed.
bd9aa7c	2016-09-05	MAPR-24269: Oozie 4.2.0 is now bundled with Pig 0.15-1608.

Oozie 4.2.0-1608 Release Notes

The notes below relate specifically to the MapR Converged Data Platform. You may also be interested in the Apache Oozie 4.2.0 changelog or the Apache Oozie project homepage.

Oozie Version	4.2.0
Release Date	September 1, 2016
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Oozie Support Matrix on page 5636

last-updated: Apr 11, 2022

Source on GitHub	https://github.com/mapr/oozie/tree/4.2.0-mapr-1608
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • `mapr-oozie-4.2.0.201608311221-1.noarch.rpm` • `mapr-oozie_4.2.0.201608311221_all.deb`

Fixes

GitHub Commit	Data (YYYY-MM-DD)	Comment
d3052f7	2016-08-22	MAPR-24316: Oozie's shared library now includes the additional JARs required to complete Sqoop actions with Parquet and Avro data.
5612917	2016-08-30	MAPR-24416: oozie.sh now appends the `hadoop classpath` to the OOZIECPPATH environment variable.

Oozie 4.2.0-1607 Release Notes

Below are release notes for the Oozie component included in the MapR Converged Data Platform.

Oozie Version	4.2.0
Release Date	July 29, 2016
MapR Version Interoperability	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/tree/4.2.0-mapr-1607
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-oozie-4.2.0.201607281407-1.noarch.rpm • mapr-oozie_4.2.0.201607281407_all.deb • mapr-oozie-internal-4.2.0.201607281407-1.noarch.rpm • mapr-oozie-internal_4.2.0.201607281407_all.deb

New in this Release

This release of Apache Oozie includes the following behavior change that is specific to MapR:

Logging Enhancement	When you set the `user.name` parameter in the `job.properties` file and user impersonation is enabled, the `oozie.log` file indicates the following information for each job: • The name of the user that submits the job. • The name of the user that actually runs the job.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Data (YYYY-MM-DD)	Comment
0a24c8e	2016-07-11	MAPR-23948: Oozie jobs no longer fail with the following error: `java.lang.NoSuchFieldError: HADOOP_CLASSPATH`.
bf46e56	2016-07-01	MAPR-23558: Oozie now stores PID files in the following directory: `/opt/mapr/pid`.
96efbda	2016-04-14	MAPR-23037: When you set the `user.name` parameter in the `job.properties` file and user impersonation is enabled, the oozie.log file includes the name of the user that submits the job and the user that actually runs the job.
696f2e2	2016-02-24	MAPR-22703: Log messages about installing Oozie extensions no longer display when you install and remove MapR's Oozie packages.

Oozie 4.2.0-1602 Release Notes

Below are release notes for the Oozie component included in the MapR Distribution for Apache Hadoop.

Oozie Version	4.2.0
Release Date	March 2, 2016
MapR Version Interoperability	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/tree/4.2.0-mapr-1602
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-oozie-4.2.0.201603011312-1.noarch.rpm • mapr-oozie_4.2.0.201603011312_all.deb • mapr-oozie-internal-4.2.0.201603011312-1.noarch.rpm • mapr-oozie-internal_4.2.0.201603011312_all.deb

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Data (YYYY-MM-DD)	Comment
2e3367a	2016-02-24	MAPR-22703: Log messages about installing Oozie extensions no longer display when you install and remove MapR's Oozie packages.
cd2770e	2016-02-13	MAPR-22592: oozie-setup.sh no longer hangs when it is unable to download the extjs library.

Oozie 4.2.0-1601 Release Notes

Below are release notes for the Oozie component included in the MapR Distribution for Apache Hadoop.

Oozie Version	4.2.0
Release Date	February 1, 2016
MapR Version Interoperability	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Oozie Support Matrix on page 5636

Source on GitHub	https://github.com/mapr/oozie/tree/4.2.0-mapr-1601
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-oozie-4.2.0.201601281619-1.noarch.rpm • mapr-oozie_4.2.0.201601281619_all.deb • mapr-oozie-internal-4.2.0.201601281619-1.noarch.rpm • mapr-oozie-internal_4.2.0.201601281619_all.deb

New in this Release

This release of Apache Oozie includes the following behavior change that is specific to MapR:

- Whenever you upgrade or get the latest Oozie package, you must update the shared libraries on the MapR filesystem. When you restart Oozie, the shared libraries are no longer automatically updated unless you change the cluster MapReduce mode or the default sharedlib directory (/oozie/share/) does not exist. See MapR's Oozie upgrade documentation for details.

For details on the features available in the open source version of this component, see the Apache Oozie 4.2.0 changelog or the Apache Oozie project homepage.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Data (YYYY-MM-DD)	Comment
5e596f4	2016-01-22	MAPR-22258: When Oozie restarts, the sharedlib is not updated unless the default sharedlib directory does not exist or the cluster MapReduce mode was changed.
a416cf9	2016-01-08	MAPR-21606: oozie-setup.sh automatically installs the ExtJS jar if it does not exist on the node.
9cb6c59	2015-12-28	MAPR-21466: Oozie jobs no longer fail with error: E0721: `Invalid signal/transition, decision node [decision_node] signal [OK]`
106b426	2015-12-23	MAPR-21814: Backported Oozie-2380 so that Oozie jobs do not fail with the FileNotFoundException.
030d3f9	2015-12-18	MAPR-21848: Duplicate instances of Oozie jobs no longer get submitted after an HTTP connection reset.
e49cf00	2015-12-11	MAPR-21755: On a MapR cluster that uses Kerberos authentication, Oozie jobs no longer fail the following error: `java.lang.NoClassDefFoundError: org/apache/hive/hcatalog/api/HCatClient`
c73fe4e	2015-12-03	MAPR-21654: Oozie is now able to submit jobs to HiveServer2.

Oozie 4.2.0-1510 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.2.0 changelog or the Apache Oozie project homepage.

Oozie Version	4.2.0
Release Date	November 20, 2015
MapR Version Interoperability	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/tree/4.2.0-mapr-1510
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-oozie-4.2.0.201511171640-1.noarch.rpm • mapr-oozie_4.2.0.201511171640_all.deb • mapr-oozie-internal-4.2.0.201511171640-1.noarch.rpm • mapr-oozie-internal_4.2.0.201511171640_all.deb

New in This Release

This release of Apache Oozie includes the following behavior change that is specific to MapR:

- Oozie uses the oozie-setup.sh script to copy shared libraries to MapR File System.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Data (YYYY-MM-DD)	Comment
795fb7c	2015-10-29	MAPR- 21184: Oozie now uses the oozie-setup.sh script to copy shared libraries to MapR File System.
813be78	2015-10-16	MAPR- 20358: The path to the tmp directory was changed from a relative path to an absolute path.
580b058	2015-10-08	MAPR- 20945: The Oozie admin command no longer displays the "Unknown hadoop version" message when HADOOP_HOME is set.

Oozie 4.2.0-1508 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.2.0 changelog or the Apache Oozie project homepage.

Version	4.2.0
Release Date	Sept 25, 2015
Source on GitHub	https://github.com/mapr/oozie.git
GitHub Release Tag	4.2.0-mapr-1508
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Oozie Support Matrix on page 5636.
Maven Artifacts	https://repository.mapr.com/maven/

last-updated: Apr 11, 2022

New in This Release

This is the initial release of Oozie 4.2.0 for MapR. It includes the following behavior changes:

- The oozie-site.xml is empty by default. All default configuration properties located at oozie-default.xml. You can add your customized properties to oozie-site.xml. See Oozie-1890 for more information.
- The Oozie 4.2.0-1508 package includes shared libraries for the following components: Hive 1.2, Spark 1.4.1, Pig 0.15, and Sqoop 1.4.6.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
63f02b4	2015-08-21	MAPR-19942 Hiveserver2 jobs submitted by Oozie no longer fail when Hive is configured to use MapR-SASL.
75e8204	2015-08-21	MAPR-19819: Oozie includes shared libraries for the following components: Hive 1.2, Spark 1.4.1, Pig 0.15, Sqoop 1.4.6

Oozie 4.1.0 Release Notes

The following Oozie 4.1.0 component release notes are included in the MapR Converged Data Platform.

Oozie 4.1.0-1606 Release Notes

Below are release notes for the Oozie component included in the MapR Converged Data Platform.

Oozie Version	4.1.0
Release Date	July 1, 2016
MapR Version Interoperability	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/tree/4.1.0-mapr-1606
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-oozie-4.1.0.201606271017-1.noarch.rpm • mapr-oozie_4.1.0.201606271017_all.deb • mapr-oozie-internal-4.1.0.201606271017-1.noarch.rpm • mapr-oozie-internal_4.1.0.201606271017_all.deb

New in this Release

This release of Apache Oozie includes the following behavior change that is specific to MapR:

Logging Enhancement	When you set the `user.name` parameter in the `job.properties` file and user impersonation is enabled, the `oozie.log` file indicates the following information for each job:

- The name of the user that submits the job.
- The name of the user that actually runs the job.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Data (YYYY-MM-DD)	Comment
4e3ea4b	2016-04-14	MAPR-23037: When you set the `user.name` parameter in the `job.properties` file and user impersonation is enabled, the oozie.log file includes the name of the user that submits the job and the user that actually runs the job.
a011a11	2016-04-07	MAPR-23036: Since Oozie is managed by Warden, Oozie init files are no longer included in MapR's Oozie packages.

Oozie 4.1.0-1601 Release Notes

Below are release notes for the Oozie component included in the MapR Distribution for Apache Hadoop.

Oozie Version	4.1.0
Release Date	February 1, 2016
MapR Version Interoperability	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Oozie Support Matrix on page 5636
Source on GitHub	https://github.com/mapr/oozie/tree/4.1.0-mapr-1601
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-oozie-4.1.0.201601281728-1.noarch.rpm • mapr-oozie_4.1.0.201601281728_all.deb • mapr-oozie-internal-4.1.0.201601281728-1.noarch.rpm • mapr-oozie-internal_4.1.0.201601281728_all.deb

New in this Release

This release of Apache Oozie includes the following behavior change that is specific to MapR:

- Whenever you upgrade or get the latest Oozie package, you must update the shared libraries on the MapR File System. When you restart Oozie, the shared libraries are no longer automatically updated unless you change the cluster MapReduce mode or the default sharedlib directory (/oozie/share/) does not exist. See MapR's Oozie upgrade documentation for details.

For details on the features available in the open source version of this component, see the Apache Oozie 4.1.0 changelog or the Apache Oozie project homepage.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

last-updated: Apr 11, 2022

GitHub Commit	Data (YYYY-MM-DD)	Comment
3cc5d64	2016-01-22	MAPR-22258: When Oozie restarts, the sharedlib is not updated unless the default sharedlib directory on the MapR filesystem does not exist or the cluster MapReduce mode was changed.
7a93809	2016-01-21	MAPR- 21184: Oozie now uses the oozie-setup.sh script to copy shared libraries to MapR filesystem.
f5e4fdf	2016-01-14	MAPR-20515: A stale PID file no longer prevents configure.sh -R from rebuilding the WAR file.
6423a90	2016-01-14	MAPR-21606: oozie-setup.sh automatically installs the ExtJS jar if it does not exist on the node.
b4315f3	2015-12-18	MAPR-21848: Duplicate instances of Oozie jobs no longer get submitted after an HTTP connection reset.
70fb5bf	2015-11-21	MAPR-21466: Oozie jobs no longer fail with error: E0721: Invalid signal/transition, decision node [decision_node] signal [OK]
a8ab0ba	2015-10-13	MAPR- 20945: The Oozie admin command no longer displays the "Unknown hadoop version" message when HADOOP_HOME is set.

Oozie 4.1.0-1506 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.1.0 changelog or the Apache Oozie project homepage.

Version	4.1.0
Release Date	July 10, 2015
Source on GitHub	https://github.com/mapr/oozie.git
GitHub Release Tag	4.1.0-mapr-1506
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Oozie Support Matrix on page 5636..
Maven Artifacts	https://repository.mapr.com/maven/

New in This Release

This is a re-release of Oozie 4.1.0 for MapR which includes the following:

- Support for Spark jobs.

Important: When Oozie 4.1 runs on a MapR 5.0 cluster, it can automatically detect the active ResourceManager. Oozie 4.1 jobs no longer fail with the following exception when Zero Configuration Failover is configured for the ResourceManager:

```
java.lang.RuntimeException: Unable to determine ResourceManager service
address from Zookeeper at localhost:5181
```

Upgrade Notes

Before you upgrade to Oozie 4.1.0-1506 and above, you must remove `org.apache.oozie.action.hadoop.EmailActionExecutor` from `oozie.service.ActionService.executor.ext.classes` in the oozie-site.xml.

For complete upgrade instructions, see MapR's documentation for Oozie.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
cfd1dd3	2015-05-08	OOZIE-1983: Added a spark action executor.
a95f110	2015-05-08	OOZIE-2071: Added a Spark example.
f10551a	2015-05-15	Added MapR Zookeeper to the Spark share lib.
c566e95	2015-05-27	MAPR-18989: Oozie 4.1.0 is now able to work with mapr-core 5.0.0.
98b676f	2015-06-23	Changed the Spark version to 1.3.1 in the Oozie share lib.

Oozie 4.1.0-1502 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Oozie 4.1.0 changelog or the Apache Oozie project homepage.

Version	4.1.0
Release Date	March 6, 2015
Source on GitHub	https://github.com/mapr/oozie.git
GitHub Release Tag	4.1.0-mapr-1502
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Oozie Support Matrix on page 5636.
Maven Artifacts	https://repository.mapr.com/maven/

Upgrade Notes

Before upgrading from Oozie 4.0.1 to Oozie 4.1.0, remove the old share libraries and examples from:

```
maprfs://oozie/share
        maprfs://user/${user.name}/examples
```

For more details, see the complete MapR's Oozie documentation for upgrade instructions.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
21f4ac7	2014-12-23	MAPR-16541: A distcp job failed when MapR was running in classic mode (MRv1).

Pig Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

last-updated: Apr 11, 2022

The release notes for Pig component, included in the MapR Converged Data Platform, contains notes specific to MapR only. More details are available on the Apache Pig website.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Pig 0.17.0.0 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following Pig 0.17.0.0 component release notes are included in the MapR distribution for Apache Hadoop.

Pig 0.17.0.100 - (EEP 8.0.0) 2110 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Pig. You may also be interested in the Apache Pig 0.17.0 changelog or the Apache Pig homepage.

Pig Version	0.17.0.100
Release Date	October 2021
Source on GitHub	https://github.com/mapr/pig/
GitHub Release Tag	0.17.0.100-eep-800
Version Compatibility	See EEP Components and OS Support on page 5536
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

- None.

Fixes

This release by HPE includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
226ae101a	2021-09-14	MAPR-PIG-61 : Error messages during starting Pig with using Hcatalog option
7e29a0fdf	2021-09-10	MAPR-PIG-60 : Unable to enter pig shell after updating jars to eep suffix
1dfec233f	2021-09-03	MAPR-PIG-59 : Update the maven artifact version strings to eep

Known Issues and Limitations

- None.

Resolved Issues

- None.

Pig 0.17.0.0 - (EEP 7.0.0) 2009 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Distribution for Apache Pig. You may also be interested in the Apache Pig 0.17.0 changelog or the Apache Pig homepage.

Pig Version	0.17.0.0
Release Date	September 2020
Source on GitHub	https://github.com/mapr/pig/
GitHub Release Tag	0.17.0.0-mapr-700
MapR Version Compatibility	See EEP Components and OS Support on page 5536
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

- None.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
7f320d5	2020-04-29	MAPR-PIG-25 : Pig failed to use ORC Storage. Pig - Hive dependency issue
7ac734b	2020-04-29	MAPR-PIG-35 : Fix org.codehaus.jackson vulnerability
a20b81a	2020-04-29	MAPR-PIG-40 : Move Pig to protobuf.version 3.11.1
3958b68	2020-05-14	MAPR-PIG-38 : update commons-collections* to v4.1 / 3.2.2
1e87a5a	2020-05-14	MAPR-PIG-41 : ZK updates to v3.5.6 at MEP7.0.0
b098d7f	2020-05-16	MAPR-PIG-36 : update io.netty to v3.9.8
42b771d	2020-06-04	MAPR-PIG-47 :https://github.com/advisories/GHSA-vmqm-g3vh-847mupdate xercesImpl to v2.12.0
6a85d20	2020-06-04	MAPR-PIG-48 : update jython-standalone to v2.7-rc1
8ea1bfd	2020-06-04	MAPR-PIG-51 : CVE-2014-0107 update xalan to 2.7.2

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
e04ef30	2020-06-04	MAPR-PIG-50 : CVE-2017-1000487 update plexus-utils to 3.0.16
66fcb9a	2020-06-04	MAPR-PIG-49 :CVE-2018-1320: update libthrift to 0.12.0
df66eea	2020-07-09	MAPR-PIG-52 : CVE-2014-3643: jersey-* to v1.13
c24ece3	2020-07-20	MAPR-PIG-46: Update Guava version to 28.2-jre
9b0d02d	2020-07-22	MAPR-PIG-54: [Pig-Hbase integration] ERROR 1066: Unable to open iterator for alias data.
17acc61	2020-08-09	MAPR-PIG-55: Backport PIG-5269 MapReduceLauncher and MRJobStats imports org.python.google.common.collect.Lists

Known Issues and Limitations

- None.

Resolved Issues

- None.

Pig 0.16.0 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following Pig 0.16.0 component release notes are included in the MapR distribution for Apache Hadoop.

Pig 0.16.0 - 2110 (EEP 6.3.5) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Pig. You may also be interested in the Apache Pig 0.16.0 changelog or the Apache Pig homepage.

Pig Version	0.16.0
Release Date	October 2021
Source on GitHub	https://github.com/mapr/pig/
GitHub Release Tag	0.16.0-mapr-2110
Version Compatibility	See EEP Components and OS Support on page 5536
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

- None.

Fixes

This release by HPE includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
dd2dbef06	2021-08-04	PIG-4967: NPE in PigJobControl.run() when job status is null

Known Issues and Limitations

- None.

Resolved Issues

- None.

Pig 0.16.0 - 2009 (EEP 6.3.1) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Pig. You may also be interested in the Apache Pig 0.16.0 changelog or the Apache Pig homepage.

Pig Version	0.16.0
Release Date	September 2020
Source on GitHub	https://github.com/mapr/pig/tree/0.16-mapr-2009
MapR Version Compatibility	See EEP Components and OS Support on page 5536
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

- None.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
477969f	2020-04-29	MAPR-PIG-25 : Pig failed to use ORC Storage. Pig - Hive dependency issue
ab5e106	2020-05-14	MAPR-PIG-38 : update commons-collections* to v4.1 / 3.2.2
7420fdd	2020-05-16	MAPR-PIG-36 : update io.netty to v3.9.8
6595dad	2020-06-04	MAPR-PIG-47 :https://github.com/advisories/GHSA-vmqm-g3vh-847mupdate xercesImpl to v2.12.0

last-updated: Apr 11, 2022

Commit	Date (YYYY-MM-DD)	Comment
0787ae3	2020-06-04	MAPR-PIG-48 : update jython-standalone to v2.7-rc1
2decff4	2020-06-04	MAPR-PIG-51 : CVE-2014-0107 update xalan to 2.7.2
cea5063	2020-06-04	MAPR-PIG-50 : CVE-2017-1000487 update plexus-utils to 3.0.16
1b3d2dd	2020-06-04	MAPR-PIG-49 :CVE-2018-1320: update libthrift to 0.12.0
1c9f7a3	2020-07-09	MAPR-PIG-52 : CVE-2014-3643: jersey-* to v1.13

Known Issues and Limitations

- None.

Resolved Issues

- **MAPR-PIG-25:** Pig 0.16 failed to use ORC Storage. Pig/Hive dependency issue – MapR Pig does not support MapR Hive ORC integration.

Pig 0.16.0-1912 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.16.0 changelog or the Apache Pig homepage.

Pig Version	0.16
Release Date	December 2019
Source on GitHub	https://github.com/mapr/pig/tree/0.16-mapr-1912
MapR Version Compatibility	See EEP Components and OS Support on page 5536
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

Important: The Pig-0.16 MEP-6.3.0 release is designed to work with Hive-2.3.6 MEP-6.3.0 and contains a fix that adds com.mapr.web.security.SslConfig class to Pig classpath. This fix allows Pig-0.16 MEP-6.3.0 to be successfully integrated with Hive-2.3.6 MEP-6.3.0 when HCatalog is used. If you use Hive-2.3.6 MEP-6.3.0 with older Pig-0.16 releases, this may cause an exception when HCatalog is used:

```
Caused by: java.lang.ClassNotFoundException:
com.mapr.web.security.SslConfig$SslConfigScope
        at
java.net.URLClassLoader.findClass(URLClassLoader.java:382)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:424)
        at
sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:349)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:357)
        ... 17 more
```

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
d213b20b	2019-11-19	MAPR-PIG-22: Replacing joda-time-*.jar with joda-time-2.10.3.jar
83339ac9	2019-11-18	MAPR-PIG-20: Drop support of MR 1 in Pig for MEP-6.3.0
6e9c93b5	2019-08-27	MAPR-PIG-18 : Add com.mapr.web.security.SslConfig to Pig job classpath
f832e6d71	2019-08-14	MAPR-PIG-17 : java.lang.NoClassDefFoundError: MapRDbJsonException. during Pig + Hive + Hcatalog integration
d74aea0a	2019-05-22	MAPR-PIG-10 : REGISTER statement does not work with maprfs file paths as expected

Known Issues and Limitations

- MAPR-PIG-25 – Pig 0.16 failed to use ORC Storage. Pig - Hive dependency issue – MapR Pig does not support MapR Hive ORC integration.

Resolved Issues

- None.

Pig 0.16.0-1901 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.16.0 changelog or the Apache Pig homepage.

last-updated: Apr 11, 2022

Pig Version	0.16
Release Date	February 2019
Source on GitHub	https://github.com/mapr/pig/tree/0.16-mapr-1901
MapR Version Compatibility	See EEP Components and OS Support on page 5536
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

None.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
416185c1	2019-01-02	MAPR-PIG-13: Updated zookeeper.jar version to v3.4.11
8fe7be3f	2017-12-21	MAPR-PIG-8: Hive-2.3 and Pig integration error: Could not find or load main class .opt.mapr.hive.hive-2.3.bin

Known Issues and Limitations

- None.

Resolved Issues

- None.

Pig 0.16.0-1703 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.16.0 changelog or the Apache Pig homepage.

Pig Version	0.16
Release Date	April 2017
Source on GitHub	https://github.com/mapr/pig/tree/0.16-mapr-1703
MapR Version Compatibility	See EEP Components and OS Support on page 5536
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
dbd7e839	2017-03-17	MAPR-26519: The commons-lang3 JAR is now included to allow Pig operations to complete successfully.

Pig 0.16.0-1611 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.16.0 changelog or the Apache Pig homepage.

Pig Version	0.16
Release Date	December 9, 2016
Source on GitHub	https://github.com/mapr/pig/tree/0.16-mapr-1611
MapR Version Compatibility	See EEP Components and OS Support on page 5536
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
bfd4ba9	2016-10-28	MAPR-24456: The issue that caused illustrate query to fail in Pig, when used with MRv1, is now fixed. Pig now includes different pig-core JAR files for MRv1 and Yarn and it will dynamically choose the necessary JAR based on the cluster's mode.
dfcdfbe	2016-09-13	PIG-5019: The issue that caused Pig to generate lots of warnings for UDF with warnings aggregation enabled is now fixed.
0194066	2016-09-01	MAPR-24442: The issue that caused grunt shell to not work in classic mode is now fixed. For running MRv1 in Hadoop2, Pig will not use the improved logging available with Pig 0.16.
cb921bb	2016-08-04	MAPR-24080: The issue that caused Pig jobs with Oozie to go into an indefinite loop when there was no JobTracker installed on a YARN-only cluster is now fixed.

Note: For information on the API changes in this version, see Pig 0.16.0 API on page 4014.

Pig 0.15.0 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

last-updated: Apr 11, 2022

The following Pig 0.15.0 component release notes are included in the MapR distribution for Apache Hadoop.

Pig 0.15.0-1703 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.15.0 changelog or the Apache Pig homepage.

Pig Version	0.15
Release Date	April 2017
Source on GitHub	https://github.com/mapr/pig/tree/0.15-mapr-1703
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Maven Artifacts	https://repository.mapr.com/maven/ and
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
e7b627eb	2017-03-17	MAPR-26519: The commons-lang3 JAR is now included to allow Pig operations to complete successfully.

Pig 0.15.0-1611 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.15.0 changelog or the Apache Pig homepage.

Pig Version	0.15
Release Date	December 9, 2016
Source on GitHub	https://github.com/mapr/pig/tree/0.15-mapr-1611
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • `mapr-pig-0.15.201612052156-1.noarch.rpm` • `mapr-pig_0.15.201612052156_all.deb`

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
06747e5	2016-10-28	MAPR-24456: The issue that caused illustrate query to fail in Pig, when used with MRv1, is now fixed. Pig now includes different pig-core JAR files for MRv1 and Yarn and it will dynamically choose the necessary JAR based on the cluster's mode.
1ee180c	2016-09-13	PIG-5019: The issue that caused Pig to generate lots of warnings for UDF with warnings aggregation enabled is now fixed.

Pig 0.15.0-1608 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.15.0 changelog or the Apache Pig homepage.

Pig Version	0.15
Release Date	September 1, 2016
Source on GitHub	https://github.com/mapr/pig/tree/0.15-mapr-1608
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Maven Artifacts	https://repository.mapr.com/maven/.
Package Names	The following packages are associated with this release: • `mapr-pig-0.15.201608291158-1.noarch.rpm` • `mapr-pig_0.15.201608291158_all.deb`

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
ff845f79f5	2016-07-29	MAPR-24080: The issue causing Pig jobs in Oozie to go into an infinite loop when JobTracker was not installed on the cluster is now fixed.

Pig 0.15.0-1607 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.15.0 changelog or the Apache Pig homepage.

Pig Version	0.15
Release Date	July 29, 2016
Source on GitHub	https://github.com/mapr/pig/tree/0.15-mapr-1607

MapR Version Interoperability	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-pig-0.15.201607281405-1.noarch.rpm • mapr-pig_0.15.201607281405_all.deb

Fixes

This release by MapR includes the following fix on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
3b8ba56b	2016-07-27	MAPR-24027: Backported the following: • PIG-4961: The issue causing CROSS followed by LIMIT inside a nested foreach to drop data from the result has been fixed. • PIG-4683: The issue causing nested order to break (after PIG-3591) in some cases has been fixed.
93c994f	2016-07-12	MAPR-23915: The following issues with Hive and Pig integration are now fixed: • Incorrect placement of definition for $HIVE_HOME environment variable. • Missing datanucleus JARs in the PIG classpath.

Pig 0.15.0-1603 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.15.0 changelog or the Apache Pig homepage.

Pig Version	0.15
Release Date	April 4 2016
Source on GitHub	https://github.com/mapr/pig/tree/0.15-mapr-1603
MapR Version Interoperability	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

Package Names	The following packages are associated with this release: • mapr-pig-0.15.201603241057-1.noarch.rpm • mapr-pig_0.15.201603241057_all.deb

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
3335793	2016-03-17	MAPR-22878: Pig no longer uses the Apache ZooKeeper JAR file. Instead, it uses the MapR ZooKeeper JAR file in the lib folder.

Pig 0.15.0-1602 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.15.0 changelog or the Apache Pig homepage.

Release Date	February 29, 2016
Source on GitHub	https://github.com/mapr/pig/tree/0.15-mapr-1602
MapR Version Interoperability	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-pig-0.15.201602111348-1.noarch.rpm • mapr-pig_0.15.201602111348_all.deb

New in this Release

This release of Pig 0.15.0 includes the following behavior change that is specific to MapR:

- Support for HBase 1.1

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
1e1558c	2016-02-05	MAPR-22454: Pig no longer fails when it integrates with MapR Database tables.
9ad64e7	2016-01-20	MAPR-22203: Pig no longer fails with the following error when you upload data from Hbase to Pig: `error: org.apache.pig.tools.grunt.Grunt - ERROR 1200: Pig script failed to parse`

Pig 0.15.0-1508 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.15.0 changelog or the Apache Pig homepage.

Release Date	Sept 9, 2015
Source on GitHub	https://github.com/mapr/pig.git
GitHub Release Tag	0.15-mapr-1508
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

New in This Release

This is the initial release of Pig Version 0.15.0 for MapR.

Pig 0.14.0 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following Pig 0.14.0 component release notes are included in the MapR distribution for Apache Hadoop.

Pig 0.14.0-1608 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.14.0 changelog or the Apache Pig homepage.

Pig Version	0.14
Release Date	September 1, 2016
Source on GitHub	https://github.com/mapr/pig/tree/0.14-mapr-1608
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Maven Artifacts	https://repository.mapr.com/maven/.
Package Names	The following packages are associated with this release: • `mapr-pig-0.14.201608251128-1.noarch.rpm` • `mapr-pig_0.14.201608251128_all.deb`

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
88e9f12f3	2016-08-04	MAPR-24080: The issue causing Pig jobs in Oozie to go into an infinite loop when JobTracker was not installed on the cluster is now fixed.

Pig 0.14.0-1607 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.14.0 changelog or the Apache Pig homepage.

Pig Version	0.14
Release Date	July 29, 2017
Source on GitHub	https://github.com/mapr/pig/tree/0.14-mapr-1607
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/.
Package Names	The following packages are associated with this release: • `mapr-pig-0.14.201607291005-1.noarch.rpm` • `mapr-pig_0.14.201607291005_all.deb`

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
3b8ba56	2016-07-27	MAPR-24027: Backported the following: • PIG-4961: The issue causing CROSS followed by LIMIT inside a nested foreach to drop data from the result has been fixed. • PIG-4683: The issue causing nested order to break (after PIG-3591) in some cases has been fixed.

Pig 0.14.0-1603 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.14.0 changelog or the Apache Pig homepage.

Pig Version	0.14
Release Date	April 4 2016

Source on GitHub	https://github.com/mapr/pig/tree/0.14-mapr-1603
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Maven Artifacts	https://repository.mapr.com/maven/.
Package Names	The following packages are associated with this release: • mapr-pig-0.14.201603240927-1.noarch.rpm • mapr-pig_0.14.201603240927_all.deb

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
9d577be	2016-03-17	MAPR-22878: Pig no longer uses the Apache ZooKeeper JAR file. Instead, it uses the MapR ZooKeeper JAR file in the lib folder.

Pig 0.14.0-1508 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.14.0 changelog or the Apache Pig homepage.

Release Date	Sept 9, 2015
Source on GitHub	https://github.com/mapr/pig.git
GitHub Release Tag	0.14-mapr-1508
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

New in This Release

This release of Pig 0.14.0 for MapR includes version 1.6.0 of the Parquet JAR files.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
22a06e2	2015-07-30	MAPR 18222- Pig includes the PARQUET-107 fix.

Pig 0.14.0-1504 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.14.0 changelog or the Apache Pig homepage.

Version	Pig 0.14-1504
Release Date	May 20, 2015
Source on GitHub	https://github.com/mapr/pig.git
GitHub Release Tag	0.14-mapr-1504
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

Fixes

This release from MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
8ad79bc	2015-04-01	MAPR-17874: Pig 14 fails to work with ORC tables created in Hive 1.0.
6493066	2015-03-23	MAPR-17789: Support for Hive 1.0 was added to to Pig 0.14.

Pig 0.14.0-1502 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Pig 0.14.0 changelog or the Apache Pig homepage.

Version	Pig 0.14-1502
Release Date	May 6, 2015
Source on GitHub	https://github.com/mapr/pig.git
GitHub Release Tag	• 0.14-mapr-1502 • 0.14-mapr-1502-h1 • 0.14-mapr-1502-h2
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

New in This Release

This is the initial release of Pig Version 0.14.0 for MapR.

This release adds support for ORC storage format. See Use ORC Storage with Pig in MapR's Pig documentation. Pig 0.14 and Hive 0.13 work with ORC storage.

Fixes

This release from MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
5fc4722	2015-02-10	MAPR-17046: A Pig job failed when using piggybank.jar to access Hive tables created in RCFile format.
0c66372	2015-02-05	MAPR-16221: Pig could not be installed on top of HBase 0.98.
d4f8db5	2015-01-14	MAPR-6086: The default logging directory was changed to the user home directory.

Sentry Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The release notes for Sentry component included in the MapR Converged Data Platform contains notes specific to MapR only.

Important: MapR support for Sentry is limited to Impala users.

You may also be interested in the Apache Sentry project homepage.

Sentry 1.7.0 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Sentry 1.7.0.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following release notes for the Sentry 1.7.0 component are included in the MapR distribution for Apache Hadoop.

Sentry 1.7.0 - 2101 (MEP 7.0.1) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Sentry.

Version	1.7.0
Release Date	January 2021
MapR Version Interoperability	SeeMEP Components and OS Support
Source on GitHub	https://github.com/mapr/sentry
GitHub Release Tag	1.7.0-mapr-mep-7.x-2101
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	SeePackage Names for MapR Ecosystem Packs (MEPs)

New in This Release

Sentry 1.7.0 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit	Date (YYYY-MM-DD)	Comment
700fc2b	2020-10-08	MAPR-SEN-86 : DESC command failing with Sentry 1.7
d485c97	2020-12-29	MAPR-SEN-90: Upgrading 'ant' dependency due to CVE-2020-1945
88f6020	2020-12-31	MAPR-SEN-76: CVE-2019-0205: Upgrade to version org.apache.thrift:libthrift:0.13.0
3377632	2021-01-06	MAPR-SEN-48: [Sentry-1.7.0] incorrect /conf directories in sentry HOME dir
85474d7	2021-01-11	MAPR-SEN-93: Errors during removing Sentry package on Ubuntu
a817b83	2021-01-12	MAPR-SEN-94: Errors while trying install / remove Sentry when Hive is not installed on the node

For details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- Refer to Sentry Known Issues on page 3819.

Resolved Issues

- None.

Sentry 1.7.0 - 2009 (MEP 7.0.0) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.7.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/sentry
GitHub Release Tag	1.7.0-mapr-2020
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

None.

Fixes

This MapR release includes the following new fixes since the latest MapR Sentry 1.7.0 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
1510ea0	2020-05-06	MAPR-SEN-47 : ZK updates to v3.5.6 at MEP7.0.0
9d3caba	2020-05-07	MAPR-SEN-40 : CVE-2017-18640: snakeyaml vulnerability at Sentry
b739c1fc	2020-05-07	MAPR-SEN-39: CVE-2014-0114: Vulnerability with commons-beanutils
31d5ccd	2020-05-07	MAPR-SEN-38: Fix io.netty vulnerability
48aef0c	2020-05-07	MAPR-SEN-41: CVE-2019-17571 security vulnerability for for log4j-1.2.17.jar
bc2194b	2020-05-17	MAPR-SEN-37: Fix org.codehaus.jackson vulnerability
c50e443	2020-05-17	MAPR-SEN-51 : Update Guava version to 28.2-jre
82d97e9	2020-05-17	MAPR-SEN-56 : Update curator version to 4.2.0
87eeef6	2020-05-20	MAPR-SEN-43 : Update jetty-* to ver 9.4.19.x
25eb500	2020-05-20	MAPR-SEN-57: Upgrade to version org.apache.shiro:shiro-all:1.2.5,org.apache.shiro:shiro-core:1.2.5
8b9318f2	2020-05-20	MAPR-SEN-58: Upgrade derby to version 10.12.1.1
149005a	2020-05-21	MAPR-SEN-60: Upgrade solr-solrj to version 5.0.0
131d883	2020-05-21	MAPR-SEN-61: Upgrade libthrift to version 0.12.0
5cf01f9	2020-05-21	MAPR-SEN-63: Upgrade xercesImpl to version 2.12.0
3611077	2020-06-18	MAPR-SEN-36 : Build and verify Sentry and Java 11
09639f4	2020-07-01	MAPR-SEN-66: Backport SENTRY-2427: Use Hadoop KerberosName class to derive shortName

Known Issues and Limitations

None.

Resolved Issues

None.

Sentry 1.7.0 - 2101 (MEP 6.3.2) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Sentry.

Version	1.7.0
Release Date	January 2021
MapR Version Interoperability	SeeMEP Components and OS Support
Source on GitHub	https://github.com/mapr/sentry
GitHub Release Tag	1.7.0-mapr-mep-6.x-2101
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	SeePackage Names for MapR Ecosystem Packs (MEPs)

New in This Release

Sentry 1.7.0 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit	Date (YYYY-MM-DD)	Comment
6c28247	2020-10-08	MAPR-SEN-86: DESC command failing with Sentry 1.7
77766f0	2020-12-29	MAPR-SEN-90: Upgrading 'ant' dependency due to CVE-2020-1945
7fff0ac	2020-12-31	MAPR-SEN-76: CVE-2019-0205: Upgrade to version org.apache.thrift:libthrift:0.13.0
9af5f7b	2021-01-06	MAPR-SEN-48: [Sentry-1.7.0] incorrect /conf directories in sentry HOME dir
6cbb081	2021-01-11	MAPR-SEN-93: Errors during removing Sentry package on Ubuntu
d753555	2021-01-12	MAPR-SEN-94: Errors while trying install / remove Sentry when Hive is not installed on the node

For details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- Refer to Sentry Known Issues on page 3819.

Resolved Issues

- None.

Sentry 1.7.0 - 2009 (MEP 6.3.1) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.7.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/sentry
GitHub Release Tag	1.7.0-mapr-2020
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

None.

Fixes

This MapR release includes the following new fixes since the latest MapR Sentry 1.7.0 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
2e92e4b3	2020-05-07	MAPR-SEN-40 : CVE-2017-18640: snakeyaml vulnerability at Sentry
7b59ea9	2020-05-07	MAPR-SEN-39: CVE-2014-0114: Vulnerability with commons-beanutils
767feed	2020-05-07	MAPR-SEN-38: Fix io.netty vulnerability
0ede77d	2020-05-07	MAPR-SEN-41: CVE-2019-17571 security vulnerability for for log4j-1.2.17.jar
f5499d5	2020-05-17	MAPR-SEN-37: Fix org.codehaus.jackson vulnerability
b5fa7fd	2020-05-20	MAPR-SEN-57: Upgrade to version org.apache.shiro:shiro-all:1.2.5,org.apache.shiro:shiro-core:1.2.5
0d5ef16	2020-05-20	MAPR-SEN-58: Upgrade derby to version 10.12.1.1
3938f04	2020-05-21	MAPR-SEN-60: Upgrade solr-solrj to version 5.0.0
8335bc1	2020-05-21	MAPR-SEN-61: Upgrade libthrift to version 0.12.0
38a02ce	2020-05-21	MAPR-SEN-63: Upgrade xercesImpl to version 2.12.0
43ab766	2020-07-01	MAPR-SEN-66: Backport SENTRY-2427: Use Hadoop KerberosName class to derive shortName

Known Issues and Limitations

None.

Resolved Issues

None.

Sentry 1.7.0 - 2009 (MEP 5.0.5) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.7.0
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/sentry
GitHub Release Tag	1.7.0-mapr-2020
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

None.

Fixes

This MapR release includes the following new fixes since the latest MapR Sentry 1.7.0 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
4612c9e	2020-07-01	MAPR-SEN-66: Backport SENTRY-2427: Use Hadoop KerberosName class to derive shortName
1bd0894	2020-05-07	MAPR-SEN-40: CVE-2017-18640: snakeyaml vulnerability at Sentry

Known Issues and Limitations

None.

Resolved Issues

None.

Sentry 1.7.0-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

last-updated: Apr 11, 2022

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.7.0
Release Date	February 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/sentry
GitHub Release Tag	1.7.0-mapr-1901
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

Added support of backticks in Sentry tokens.

Fixes

This MapR release includes the following new fixes since the latest MapR Sentry 1.7.0 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
8acff7b	2018-11-05	MAPR-SEN-20 : Configure Sentry JUnit tests to run locally
a00dfae	2018-11-05	MAPR-SEN-19 : Add support of backticks to Sentry
320b118	2018-01-14	MAPR-SEN-13: Can't create admin role for sentry DB model
1a2b10f	2018-01-18	MAPR-SEN-23 : Remove maprfs jars from Sentry package to avoid potential issues
01a0559	2018-01-22	MAPR-SEN-25 : Update version of the hadoop jars for Sentry MEP-6.0.1+

Known Issues and Limitations

None.

Resolved Issues

None.

Sentry 1.7.0-1901 (EEP 5.0.1, EEP 4.1.3, and EEP 3.0.5) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.7.0
Release Date	February 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/sentry/
GitHub Release Tag	1.7.0-mapr-1901
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

Added support of backticks in Sentry tokens.

Fixes

This MapR release includes the following new fixes since the latest MapR Sentry 1.7.0 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
8acff7b	2018-11-05	MAPR-SEN-20 : Configure Sentry JUnit tests to run locally
a00dfae	2018-11-05	MAPR-SEN-19 : Add support of backticks to Sentry
320b118	2018-01-14	MAPR-SEN-13: Can't create admin role for sentry DB model
20c01b3	2018-01-18	MAPR-SEN-23 : Remove maprfs jars from Sentry package to avoid potential issues
e20f771	2018-01-24	MAPR-SEN-22 : Cannot pass server privileges to an admin role

Known Issues and Limitations

None.

Resolved Issues

None.

Sentry 1.7.0-1808 (EEP 6.0.0) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.7.0
Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536

Source on GitHub	https://github.com/mapr/sentry/tree/1.7.0-mapr-1808
GitHub Release Tag	1.7.0-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

Implemented preserving configuration during package update.

Fixes

None.

Known Issues and Limitations

None.

Resolved Issues

None.

Sentry 1.7.0-1803 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	1.7.0
Release Date	March 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/sentry/
GitHub Release Tag	1.7.0-mapr-1803
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

None.

Fixes

This MapR release includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
c9764d0	2018-03-06	SEN-6 Changes to NOTICE.txt
fb881cd	2018-01-22	Update hive versions

Known Issues and Limitations

None.

Resolved Issues

None.

Sentry 1.7.0-1703 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	1.7.0
Release Date	April 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/sentry/
GitHub Release Tag	1.7.0-mapr-1703
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)
API Changes for this Version	See Sentry 1.7.0 API Changes on page 3835.

New in This Release

Sentry 1.7.0-1703 introduces the following enhancements or MapR platform-specific behavior changes:

- Sentry integration with Hive authorization framework v2
- Create simple shell for Sentry
- Upgrade Hive plugin v2 for Hive 2.1.1

Fixes

This MapR release includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
0850531	2016-08-29	MAPR-24381 - Sentry Simple Shell doesn't work on a MAPR-SASL cluster.
87d4627	2017-01-24	MAPR-25862 - Sentry 1.7.0 should be built with the hive2 profile by default.
4aab063	2017-01-30	MAPR-25936 - Sentry 1.7.0 should work with Hive 2.1.1 instead of Hive 2.0.
00707be	2017-01-30	MAPR-25941 - Tables in Impala are invisible after configuration on Sentry.
9694425	2017-01-31	MAPR-25949 - Sentry can't start.
eb4dc39	2017-02-01	MAPR-25964 - Sentry 1.7 cannot be compiled.
58e8949	2017-02-02	MAPR-25964 - Sentry 1.7 cannot be compiled, missed provided.
6532fbe	2017-03-14	MAPR-26432 [Sentry 1.7] - Failed to perform USE query through Hive Thrift.

Known Issues and Limitations

Important: MapR support for Sentry is limited to Impala users.

Resolved Issues

None.

Sentry 1.6.0 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Sentry 1.6.0.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following release notes for the Sentry 1.6.0 component are included in the MapR distribution for Apache Hadoop.

Sentry 1.6.0-1707 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Sentry 1.6.0-1707.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Important: MapR support for Sentry is limited to Impala users.

The notes below relate specifically to the MapR Distribution for Apache Hadoop.

Version	1.6.0
Release Date	July 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/sentry/
GitHub Release Tag	1.6.0-mapr-1707
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)
API Changes for this Version	None

New in This Release

No new features.

Fixes

This MapR release includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number	Description
10ef983	2017-05-26	MAPR-27524	Hue becomes unresponsive after multiple HiveServer2 requests.
f6cb28c	2017-06-09	MAPR-26432	Failed to perform USE query through Hive Thrift.

Known Issues and Limitations

MapR support for Sentry is limited to Impala users.

Resolved Issues

None.

Sentry 1.6.0-1606 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Sentry component included in the MapR Converged Data Platform.

Version	1.6.0
Release Date	June 30, 2016
Works with MapR Version	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/sentry
GitHub Release Tag	1.6.0-mapr-1606
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs) if you are running 5.2 and install EEP 1.1.1, EEP 2.0, or any subsequent EEP release. The following packages are associated with Sentry 1.6.0-1606 if you are running 5.0 or 5.1 and you install Sentry from the https://package.mapr.com/releases/ecosystem-5.x/ repository, or if you are running 5.2 and install EEP 1.0.0 or 1.1.0: • mapr-sentry-1.6.0.201602231135-1.noarch.rpm • mapr-sentry_1.6.0.201602231135_all.deb

New in this Release

This release of Apache Sentry includes support for the RELOAD command.

Fixes

GitHub Commit	Date (YYYY-MM-DD)	Comment
6b98733	2016-05-25	SENTRY-1003 - When Hive issues the RELOAD command, Sentry gets the updated aux JAR path from "hive.reloadable.aux.jars.path".
b4705c9	2016-05-25	MAPR-23468 - Added a dependency for org.datanucleus.javax.jdo.

Sentry 1.6.0-1602 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Sentry component included in the MapR Converged Data Platform.

last-updated: Apr 11, 2022

Version	1.6.0
Release Date	February 24, 2016
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/incubator-sentry/tree/1.6.0-mapr-1602
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-sentry-1.6.0.201602231135-1.noarch.rpm • mapr-sentry_1.6.0.201602231135_all.deb

New in this Release

This release of Apache Sentry includes the following behavior change that is specific to MapR:

- Database storage model is now supported. It was not supported in the Sentry 1.4.0-1412 release.

For details on the features available in the open source version of this component, see the Apache Sentry 1.6.0 changelog or the Apache Sentry project homepage.

Fixes

GitHub Commit	Date (YYYY-MM-DD)	Comment
4125954	2015-12-14	MAPR-21571: The Sentry service periodically renews the server Kerberos ticket.
cb8daf8	2015-12-08	MAPR-21241: The Sentry service can be managed by the MCS.
d83b67b	2015-12-09	MAPR-21701: The Sentry is installed with file-based mode configurations by default.

Sentry 1.4.0 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Sentry 1.4.0.

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The following release notes for the Sentry 1.4.0 component are included in the MapR distribution for Apache Hadoop.

Sentry 1.4.0-1509 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Version	1.4.0
Release Date	Oct 5, 2015
Source on GitHub	https://github.com/mapr/incubator-sentry/tree/1.4.0-mapr-1509

MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	See Maven Repository
Package Names	The following packages are associated with this release: • mapr-sentry-1.4.0.201510011040-1.noarch.rpm • mapr-sentry_1.4.0.201510011040_all.deb

New in this Release

This release of Apache Sentry includes the following behavior changes that are specific to the MapR:

- Hive home directory is now determined correctly.
- Hive is no longer automatically installed with Sentry. Sentry requires Hive. Therefore, on nodes where Sentry is installed, verify that Hive 0.13 is installed or install Hive 0.13 manually.

For details on the features available in the open source version of this component, see the Apache Sentry 1.4.0 changelog or the Apache Sentry project homepage.

Sentry 1.4.0-1501 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sentry 1.4.0 changelog or the Apache Sentry project homepage.

Version	1.4.0
Release Date	January 21, 2015
Source on GitHub	https://github.com/mapr/incubator-sentry
GitHub Release Tag	1.4.0-mapr-1501
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

Compatibility with Other Ecosystem Components

In this release, Sentry only works in combination with certain other components, as shown:

- Sentry 1.4.0 is only supported with Impala 1.4.1 and Hive 0.13.
- Sentry with Solr (or any other framework) is not supported in this release.

For more information, see MapR's Sentry documentation.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
6dee41e	2014-12-16	MAPR-16084: Running a Hive query from Hue returned an error when Hive was configured to work with Sentry.

Sentry 1.4.0-1410 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sentry 1.4.0 changelog or the Apache Sentry project homepage.

Version	1.4.0
Release Date	November 19, 2014
Source on GitHub	https://github.com/mapr/incubator-sentry
GitHub Release Tag	1.4.0-mapr-1410
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Maven Artifacts	https://repository.mapr.com/maven/

Compatibility with Other Ecosystem Components

This is the first release of Sentry in the MapR Distribution for Apache Hadoop. In this release, Sentry only works in combination with certain other components, as shown:

- Sentry 1.4.0 is only supported with Impala 1.4.1 and Hive 0.13.
- When using Hive 0.13 with MapR version 3.1.1, Hive 0.13 is not compatible with Tez.
- Sentry with Solr (or any other framework) is not supported in this release.

For more information, see the MapR's Sentry documentation.

Spark Release Notes

The release notes for Spark component (included in the MapR Data Platform) contains notes specific to MapR only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Spark 3.2.0.0 - 2201 (EEP 8.1.0) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 3.2.0.0.

The notes below relate specifically to the MapR Technologies Distribution for Apache Hadoop. For more information, you may also want to consult the open-source Spark 3.2.0 Release Notes.

These release notes contain only MapR Technologies specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	3.2.0.0

Release Date	January 2022
HPE Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	3.2.0.0-eep-810
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Beginning with EEP 6.0.0, the KeyStore and TrustStore password can be removed from `spark-defaults.conf` and set in `/opt/mapr/conf/ssl-client.xml`.
- Beginning with EEP 6.0.0, after an upgrade, the previous version's configuration files are saved in the `/opt/mapr/spark` directory.
- MapR 6.1.0 with EEP 6.0.0 and later support simplified security. If you enable security on your data-fabric cluster, HPE scripts automatically configure Spark security features.
- Beginning with Core 6.2 and EEP 7.0, Spark supports SSL for WebUI.

Hive Support

- Starting from Spark 3.1.2, Spark supports Hive 2.3.

Delta Lake Support

Spark 3.2.0 provides Delta Lake support on MapR Data Platform. See Apache Spark Feature Support on page 4027.

New in This Release

- For a complete list of new features, see the open-source Spark 3.2.0 Release Notes.

Fixes

This HPE release includes the following new fixes since the latest Spark release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
7c727c3	2021/11/04	MapR[SPARK-960] Update Hadoop in Spark-3.2.x
d53fe9f	2021/11/19	MapR [SPARK-979] Backport all needed 3.1.2 EEP commits tp 3.2 branch
e85b0ce	2021/11/22	MapR [SPARK-982] Update Spark version in warden files
5693d20	2021/11/22	MapR [SPARK-972] STS start fail due to java.lang.NoSuchMethodError
3b6cb09	2021/11/25	MapR [SPARK-981] Select from table with data storing as a local file fails

31ead44	2021/11/29	MapR [SPARK-950] Can't start spark job/services with enabled FIPS
85b3d44	2021/12/07	MapR [SPARK-952] Spark services can't start on cluster with enabled FIPS
9cfd68c	2021/12/07	MapR [SPARK-963] select from hbase table which was created via hive fails
82bfd4d	2021/12/09	MapR [SPARK-966] Streaming application with the latest offset read 1 message from mapr stream which was produced before application start
96a3e9d	2021/12/10	MapR [SPARK-964] MapRDBSourceConfig.CreateTableOption=true causes structured streaming application fail
697e7f9	2021/12/24	MapR [SPARK-985] Spark and Livy application fails if spark main package is not installed on each node
5e8401a	2021/12/28	MapR [SPARK-986] log4j-1.2.17.jar vulnerability:CVE-2019-17571
f951c10	2021/12/28	MapR [SPARK-975] Spark CVE fixes for Jan 2022 release
0c07103	2021/12/30	MapR [SPARK-921] Replace sudo command with maprexecute in Spark
6a38156	2021/12/30	MapR [SPARK-984] Select from temp view which was created under orc df fails
279f325	2022/01/11	MapR [SPARK-965] Spark Structured Streaming application fails when need to recovery from checkpoint
6060b6c	2022/01/14	MapR [SPARK-994] Update jackson-mapper-asl v1.9.13 to 1.9.13-atlassian-5
1636a6e	2022/01/17	MapR [SPARK-992] STS doesn't work on cluster with enabled FIPS
1661404	2022/01/18	MapR [SPARK-995] Write to parquet fails.
5b4c35f	2022/01/25	MapR [SPARK-1001] Update log4j v1 to the 1.3.1-mapr
a919353	2022/01/25	MapR [SPARK-1002] Spark WebUI not work on FIPS cluster
5fa1feb	2022/01/28	MapR [SPARK-1000] Spark's -Djava.library.path misses hadoop native libs

Known Issues and Limitations

- The JDBC driver for Microsoft SQL Server does not support WITH CTE query on Spark.
- When you enable the SSL in a mixed (FIPS and non-FIPS) configuration, Spark application run fails. To run Spark applications, set `spark.ssl.ui.enabled` option to `false` in `spark-defaults.conf` configuration file.

- If you are using Spark SQL with Derby database without Hive or Hive Metastore installation, you will see the Java Runtime Exception. See Apache Spark Feature Support on page 4027 for workaround. Spark 3.2.0 does not support `log4j1.2` logging on MapR Data Platform.
- MapR Data Platform does not support GPU aware scheduling feature on Spark 3.2.0. See Apache Spark Feature Support on page 4027.

Resolved Issues

- None.

Spark 3.1.2.0 - 2110 (EEP 8.0.0) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 3.1.2.0.

The notes below relate specifically to the MapR Data Platform Distribution for Apache Hadoop. For more information, you may also want to consult the open-source Spark 3.1.2 Release Notes.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	3.1.2.0
Release Date	October 2021
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	3.1.2.0-eep-800
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Beginning with EEP 6.0.0, the KeyStore and TrustStore password can be removed from `spark-defaults.conf` and set in `/opt/mapr/conf/ssl-client.xml`.
- Beginning with EEP 6.0.0, after an upgrade, the previous version's configuration files are saved in the `/opt/mapr/spark` directory.
- MapR 6.1.0 with EEP 6.0.0 and later support simplified security. If you enable security on your data-fabric cluster, HPE scripts automatically configure Spark security features.
- Beginning with Core 6.2 and EEP 7.0, Spark supports SSL for WebUI.

Hive Support

- Starting from Spark 3.1.2, Spark supports Hive 2.3.

Delta Lake Support

Starting from EEP 8.0.0, Delta Lake support is available for Apache Spark 3.1.2 on MapR Data Platform. See Apache Spark Feature Support on page 4027.

last-updated: Apr 11, 2022

New in This Release

- For a complete list of new features, see the open-source Spark 3.1.2 Release Notes.

Fixes

This HPE release includes the following new fixes since the latest Spark release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY/MM/DD)	HPE Fix Number and Description
22b57ce	2021/07/29	MapR [SPARK-811] Updating to Spark 3.x
bfd5a1a	2021/07/29	MapR [SPARK-839] Unstable problem with security keys for job on yarn-cluster mode
9045b8f	2021/07/29	[SPARK-32723][WEBUI] Upgrade to jQuery 3.5.1
c4c0808	2021/07/29	MapR [SPARK-843] Improve logging for SSL certs generation
3cf3522	2021/07/29	MapR [SPARK-847] Spark can't read data from symlink
2f6cbe1	2021/07/29	MapR [SPARK-846] Add service verifier to Spark package
f11cc5a	2021/07/29	MapR [SPARK-861] Error logs in spark-historyserver
711eb00	2021/07/29	MapR [SPARK-863] Interaction with HBase via hbase spark connector fails
bd65fe6	2021/07/29	MapR [SPARK-841] Backport SPARK-32723
213226d	2021/07/29	MapR [SPARK-851] Spark SQL transient FS error handling when writing output
6f33abe	2021/07/29	MapR [SPARK-869] fix logging location
41ac719	2021/07/29	MapR [SPARK-867] Spark Hive Example fails from simple user
b234c78	2021/07/29	MapR [SPARK-870] Can't download event logs from SHS twice
df7dbfe	2021/07/29	MapR [SPARK-871] Spark job fails from mapr-client
165644d	2021/07/29	MapR [SPARK-846] Add service verifier to Spark package - moving to proper directory
92e3441	2021/07/29	MapR [SPARK-877] Update Jenkins file to build Spark-3.x from MEP-8.0.0 private package branch
42b5f36	2021/07/29	MapR [SPARK-881] fix duplicate heade
9c75ccb	2021/07/29	MapR [SPARK-879] Add changes to examples module to build for Spark-3.x

GitHub Commit Number	Date (YYYY/MM/DD)	HPE Fix Number and Description
0282515	2021/07/29	MapR [SPARK-883] Spark-3.1.1 job submition
a87034c	2021/07/29	MapR [SPARK-885] Spark-submit fails on 8.2 and 8.3 centos
9d7de9e	2021/07/29	MapR [SPARK-888] Collect of selected result from orc table fails
b80ea8e	2021/07/29	MapR [SPARK-893] Clean deprecated kafka08 and kafka09 from pyspark code
c26d17c	2021/07/29	MapR [SPARK-891] Thrift server start fails due to unsupported Hive Metastore version
28a492a	2021/07/29	MapR [SPARK-897] Spar-3.1.1 doesn't suppot MapRSASL for Thriftserver
3b7064d	2021/07/29	MapR [SPARK-901] Spark-3.1.1 doesn't start by warden
37bc0ac	2021/07/29	MapR [SPARK-903] spark.loadFromMapRDB(tableName, schema) using v2 api fail
568f406	2021/07/29	MapR [SPARK-817] Update kafka client to v2.6.X
8b29cfa	2021/07/29	MapR [SPARK-886] MapRDB table loading fails via spark session + java api
e86c244	2021/07/29	MapR [SPARK-907] Update hadoop dependency for Spark-3.1.1
9ed986b	2021/07/29	MapR [SPARK-905] Can't connect to spark thriftserver via beeline with MAPRSASL
1422b69	2021/07/29	MapR [SPARK-904] Implement inferSchema method for MaprDBDataSource
c4a1990	2021/07/29	[EZSPA-212] Move creating of spark-env.sh script to Spark
078555a	2021/07/29	[SPARK-22769] Do not log rpc post message error when sparkEnv is already stopped
a8cb292	2021/07/29	[EZSPA-213] Add Spark-3.x to dockerfiles project
5e1ddd3	2021/08/02	MapR [SPARK-917] Porting Spark-3.1.2 to MapR
2c5d494	2021/08/09	MapR [SPARK-922] Move latest Spark commits to 3.1.2 branch
2f6b259	2021/08/10	MapR [SPARK-882] Making netcat to work on Ubuntu too
42328a6	2021/08/12	MapR [SPARK-878] remove redundant filter setting

last-updated: Apr 11, 2022

GitHub Commit Number	Date (YYYY/MM/DD)	HPE Fix Number and Description
13f8e12	2021/08/17	MapR [SPARK-919] Errors in spark DEBUG logs
605d290	2021/08/18	MapR [SPARK-927] Update Hive in Spark-3.1.2
1b188e3	2021/08/19	MapR [SPARK-923] Update Avro to 1.10.1 in Spark
6f8abad	2021/08/19	MapR [SPARK-915] CVE-2020-13956,WS-2017-3734 vulnerabilities in http-client
75d47c5	2021/08/24	MapR [SPARK-906] Spark streaming (structured and unstructured) fails with kafka 2.6.1.0
09f6deb	2021/08/27	MapR [SPARK-911] STS HA doesn't work
8252a91	2021/09/01	MapR [SPARK-928] Can't connect to spark thriftserver on kerberos cluster
dba8119	2021/09/03	MapR [SPARK-929] Spark 3.1.2 requires password when you try to remove packages
c942759	2021/09/06	MapR [SPARK-882] [Installer] Add verification scripts for spark-thrifserver and spark-historyserver
31f4b86	2021/09/06	MapR [SPARK-936] Investigate Spark warning on start of application
050309c	2021/09/06	MapR [SPARK-938] Run-example doesn't work
334bc98	2021/09/07	MapR [SPARK-919] Errors in spark DEBUG logs
354abf87	2021/09/09	[EZSPA-270] adopt mapr spark feature to work in non-mapr env
718fa44	2021/09/10	MapR [SPARK-939] Replace "mapr" to "eep" in Spark package
ac7871c	2021/09/10	MapR [SPARK-934] Spark and Livy jobs fail on core 7.0.0 with encrypted ssl password
a6b59d0	2021/09/20	MapR [SPARK-943] Spark 3 and S3 integration fails
4a04f88	2021/09/20	MapR [SPARK-945] Components can't read keyPassword
a5cb0b8	2021/09/21	MapR [SPARK-894] Hadoop artifacts should be taken from the cluster
e8c8667	2021/09/22	MapR [SPARK-946] Excessive warning messages in spark-shell
715edb7	2021/09/22	MapR [SPARK-947] Error when using Spark SQL with derby db

Known Issues

- If you are using Spark SQL with Derby database without Hive or Hive Metastore installation, you will see the Java Runtime Exception. See Apache Spark Feature Support on page 4027 for workaround. Spark 3.1.2 does not support `log4j1.2` logging on MapR Data Platform.
- MapR Data Platform does not support GPU aware scheduling feature on Spark 3.1.2. See Apache Spark Feature Support on page 4027.

Resolved Issues

- None.

Spark 2.4.7.100 - 2104 (EEP 7.1.0) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.4.7.100.

The notes below relate specifically to the MapR Data Platform Distribution for Apache Hadoop. For more information, you may also wish to consult the open-source Spark 2.4.7 Release Notes.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.4.7.100
Release Date	April 2021
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.4.7.100-mapr-2104
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Beginning with EEP 6.0.0, the keyStore and trustStore password can be removed from `spark-defaults.conf` and set in `/opt/mapr/conf/ssl-client.xml`.
- Beginning with EEP 6.0.0, after an upgrade, the previous version's configuration files are saved in the `/opt/mapr/spark` directory.
- MapR 6.1.0 with EEP 6.0.0 and later support simplified security. If you enable security on your data-fabric cluster, HPE scripts automatically configure Spark security features.
- Beginning with Core 6.2 and EEP 7.0, Spark supports SSL for WebUI.

Hive Support

This version of Spark supports integration with Hive, but has the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but is not fully compatible with Hive. See the Apache Spark documentation and the Spark documentation for details.

last-updated: Apr 11, 2022

New in This Release

- For a complete list of new features, see the open-source Spark 2.4.7 Release Notes.
- Service verifier

Fixes

This HPE release includes the following new fixes since the latest data-fabric Spark release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY/MM/DD)	HPE Fix Number and Description
ea8a479	2021/01/27	[SPARK-32723][WEBUI] Upgrade to jQuery 3.5.1
98ced49	2021/02/02	MapR [SPARK-701] CVE-2019-17571 security vulnerability for for log4j-1.2.17
a2b10a6	2021/02/08	MapR [SPARK-845] Update hadoop dependency for Spark-2.4.7
0dc3f73	2021/02/16	MapR [SPARK-843] Improve logging for SSL certs generation
ed0be85	2021/02/16	MapR [K8S-2129] Spark ignores impersonation in K8s
759896a	2021/02/17	MapR [SPARK-847] Spark can't read data from symlink
ea1a325	2021/03/01	MapR [K8S-2215] backport mapr-specific k8s commits from 2.4.5
584881f	2021/03/19	MapR [SPARK-846] Add service verifier to Spark package
127da25	2021/03/24	MapR [SPARK-856] Update Spark dependencies to latest artifacts for MEP-7.0.1 release
9d1e83d	2021/03/29	MapR [SPARK-861] Error logs in spark-historyserver
2ffdd06	2021/03/29	MapR [SPARK-859] Excessive logs for spark job
8b7abe5	2021/03/31	MapR [SPARK-863] Interaction with HBase via hbase spark connector fails
6b9d305	2021/04/05	MapR [SPARK-841] Backport SPARK-32723
ea9fc35	2021/04/08	MapR [SPARK-851] Spark SQL transient FS error handling when writing output
6e2aff7	2021/04/13	MapR [SPARK-869] fix logging location
976b271	2021/04/13	MapR [SPARK-867] Spark Hive Example fails from simple user
2bcc0c0	2021/04/13	MapR [SPARK-870] Can't download event logs from SHS twice

Known Issues

- SPARK-865: If you run a `configure.sh` command to configure HBase after Spark configuration, then you must manually copy the `hbase-site.xml` configuration file from the HBase configuration directory to the Spark configuration directory.

Resolved Issues

- None.

Spark 2.4.7.0 - 2101 (EEP 7.0.1) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.4.7.0.

The notes below relate specifically to the HPE Distribution for Apache Hadoop. For more information, you may also wish to consult the open-source Spark 2.4.7 Release Notes.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.4.7
Release Date	January 2021
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.4.7-mapr-2101
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Beginning with EEP 6.0.0, the keyStore and trustStore password can be removed from `spark-defaults.conf` and set in `/opt/mapr/conf/ssl-client.xml`.
- Beginning with EEP 6.0.0, after an upgrade, the previous version's configuration files are saved in the `/opt/mapr/spark` directory.
- MapR 6.1.0 with EEP 6.0.0 and later support simplified security. If you enable security on your data-fabric cluster, HPE scripts automatically configure Spark security features.
- Beginning with MapR 6.2 and EEP 7.0, Spark supports SSL for WebUI.

Hive Support

This version of Spark supports integration with Hive, but has the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but is not fully compatible with Hive. See the Apache Spark documentation and the MapR Spark documentation for details.

New in This Release

- For a complete list of new features, see the open-source Spark 2.4.7 Release Notes.

Fixes

This HPE release includes the following new fixes since the latest data-fabric Spark release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
e1b4639	2020/09/02	MapR [SPARK-778] Delete ssl keys for spark job during spark package remove
4999284	2020/09/02	MapR [SPARK-759] Allow PAM and MapR-Sasl to be enabled at the same time
729a9a5	2020/09/07	MapR [SPARK-796] java.lang.ClassNotFoundException: Class com.mapr.fs.MapRFileSystem not found for spark-hive integration jobs
1d4ab93	2020/09/12	MapR [SPARK-797] Do not overwrite existing warden configuration files
ed0be85	2020/09/17	MapR [SPARK-795] - extended logging for Spark Streaming and Structured Streaming
547b943	2020/10/07	MapR [K8S-1882] Error on Spark SSL key generation
35f3ff5	2020/10/21	MapR [SPARK-805] Spark standalone worker can't start with enabled SSL for UI
3f7a240	2020/10/26	MapR [SPARK-703] Fix org.codehaus.jackson vulnerability
5660d09	2020/11/03	MapR [SPARK-804] sparkContext.loadFromMapRDB throwing RuntimeException in scala code
7373f82	2020/11/19	MapR [SPARK-812] Update Ant version to avoid CVE-2020-1945
2e2127a	2020/11/22	MapR [SPARK-819] CVE-2019-0205: Update Thrift version to v0.13.0
c409736	2020/11/30	[SPARK-33405][BUILD][2.4] Upgrade commons-compress to 1.20
4507ee8	2020/11/30	MapR [SPARK-822] WS-2019-0379: update commons-codec jar for spark
c3e72fd	2020/12/09	MapR [SPARK-824] CVE-2020-11612: Netty vulnerabilities
5394b4a	2020/12/29	MapR [SPARK-832] Multiple Vulnerabilities including Spark and hive jars found
187e7f2	2021/01/05	MapR [SPARK-833] Spark thriftserver can't start after the latest commit
3dd6575	2021/01/08	MapR [SPARK-812] Update Ant version to 1.10.9 to avoid CVE-2020-1945

GitHub Commit	Date (YYYY-MM-DD)	Comment
816db59	2021/01/14	MapR [SPARK-776] IllegalArgumentException: Null user is thrown when using some methods from SparkContext
c3e4963	2021/01/15	MapR [SPARK-839] Unstable problem with security keys for job on yarn-cluster mode

Known Issues

- None.

Resolved Issues

- None.

Spark 2.4.5-2009 (EEP 7.0.0) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.4.5.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. For more information, you may also wish to consult the open-source Spark 2.4.5 Release Notes.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.4.5
Release Date	September 2020
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.4.5-mapr-2009
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Beginning with EEP 6.0.0, the keyStore and trustStore password can be removed from `spark-defaults.conf` and set in `/opt/mapr/conf/ssl-client.xml`.
- Beginning with EEP 6.0.0, after an upgrade, the previous version's configuration files are saved in the `/opt/mapr/spark` directory.
- MapR 6.1.0 with EEP 6.0.0 and later support simplified security. If you enable security on your MapR cluster, MapR scripts automatically configure Spark security features.
- Beginning with EEP 6.3.0, the Spark MapRDB JSON connector supports secondary indexes.
- Beginning with EEP 6.3.0, Spark supports configurable HTTP security headers.
- Beginning with MapR 6.2 and EEP 7.0, Spark supports SSL for WebUI.

Hive Support

This version of Spark supports integration with Hive, but has the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but is not fully compatible with Hive. See the Apache Spark documentation and the MapR Spark documentation for details.

New in This Release

- For a complete list of new features, see the open-source Spark 2.4.5 Release Notes.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
b5bc89b	2020/04/01	MapR [SPARK-695] Spark 2.4.5 porting to MapR
7c3c09b	2020/04/07	MapR [SPARK-701] CVE-2019-17571 security vulnerability for log4j-1.2.17.jar
5d35d4c	2020/04/08	MapR [SPARK-705] Spark streaming fail with NoSuchMethodError: org.apache.kafka.clients.producer.ProducerConfig.getBoolean(Ljava/lang/String;)Ljava/lang/Boolean;
7ce734b	2020/04/08	MapR [SPARK-706] HBase example execution fail with NoClassDefFoundError: org/apache/hadoop/hbase/util/Bytes
78935fe	2020/04/13	[SPARK-26966][ML] Update to JPMML 1.4.8
8123d15	2020/04/14	MapR [SPARK-714] Update bcprov-jdk15on to fix CVE issue
1f5fb4a	2020/04/28	MapR [SPARK-715] Driver https port bind fail when application starts under non-mapr user
79d29d4	2020/04/30	MapR [SPARK-733] Update OJAI version in Spark
0b8b1a9	2020/04/30	MapR [SPARK-732] Update Hive version to 2.3.7
2bdc162	2020/05/07	MapR [SPARK-735] Spark master doesn't start after installation
45b682c	2020/05/14	MapR [SPARK-719] ZK updates to v3.5.6 at MEP 7.0.0
d67eafb	2020/05/17	MapR [SPARK-739] Upgrade curator version to 4.2.0
03c7195	2020/05/19	MapR [SPARK-740] UI shows "Kafka 0.9 direct stream" even if 0.10 is used

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comment
e75abc7	2020/05/20	MapR [SPARK-737] Calling poll(1000) from paranoidPoll causes batch scheduling delay
a253f8a	2020/05/22	MapR [SPARK-738] CLONE - Check if jackson-databind-2.9.4.jar will work with Spark 2.4.X in MEP 6 and MEP7
0309667	2020/05/27	[K8S-1458] Spark 2.4.4 encounters error and does not start executors
035f4e4	2020/06/03	MapR [SPARK-688] Spark Web UI has broken styles and works wrong
8c180b4	2020/06/10	[SPARK-24421][BUILD][CORE] Accessing sun.misc.Cleaner in JDK11
8d63d69	2020/06/10	[SPARK-25946][BUILD] Upgrade ASM to 7.x to support JDK11
cf0992f	2020/06/10	[SPARK-25984][CORE][SQL] [STREAMING] Remove deprecated .newInstance(), primitive box class constructor calls
bc64754	2020/06/10	[SPARK-26507][CORE] Fix core tests for Java 11
278fc64	2020/06/10	[SPARK-26536][BUILD][TEST] Upgrade Mockito to 2.23.4
90c0cc1	2020/06/10	[SPARK-26839][SQL] Work around classloader changes in Java 9 for Hive isolation
2263dba	2020/06/10	[SPARK-26963][MLLIB] SizeEstimator can't make some JDK fields accessible in Java 9+
f632a69	2020/06/10	[SPARK-26986][ML] Add JAXB reference impl to build for Java 9+
50664cd	2020/06/10	[SPARK-27121][REPL] Resolve Scala compiler failure for Java 9+ in REPL
27d5882	2020/06/10	[SPARK-27981][CORE] Remove `Illegal reflective access` warning for `java.nio.Bits.unaligned()` in JDK9+
758a3f9	2020/06/10	[SPARK-28072][SQL] Fix IncompatibleClassChangeError in `FromUnixTime` codegen on JDK9+
5e968b8	2020/06/10	[SPARK-28736][SPARK-28735] [PYTHON][ML] Fix PySpark ML tests to pass in JDK 11
4ad4116	2020/06/10	[SPARK-28737][CORE] Update Jersey to 2.29
8863fcc	2020/06/10	[SPARK-28755][R][TESTS] Increase tolerance in 'spark.mlp' SparkR test for JDK 11
4abbd53	2020/06/10	[SPARK-28756][R] Fix checkJavaVersion to accept JDK8+

GitHub Commit	Date (YYYY-MM-DD)	Comment
30ee1cb	2020/06/10	[SPARK-29674][CORE] Update dropwizard metrics to 4.1.x for JDK 9+
97c94ec	2020/06/12	MapR [SPARK-746] Secondary index not picked
64ca771	2020/06/12	[SPARK-27467][BUILD] Upgrade Maven to 3.6.1
a976dc6	2020/06/15	MapR [SPARK-753] Update maven plugins to build with JDK11
62549fa	2020/06/15	MapR [SPARK-752] Spark 2.4.5 build hangs with JDK11
a0c6004	2020/06/16	MapR [SPARK-754] Kubernetes module build failed after changes to build with JDK11
e50347b	2020/06/16	MapR [SPARK-755] Spark build failed because of using banned dependencies
22218c8	2020/06/16	MapR [SPARK-756] skip-kafka-0-8 needs to be used for build Spark with Scala-12
60b6cde	2020/06/16	MapR [SPARK-751] Spark Jetty servers send webserver details to client
1ebaada	2020/06/18	MapR [SPARK-758] Change Core version to 4-digits
f83a849	2020/06/24	MapR [SPARK-760] DataSourceV2 ClassCastException
Ab60c7f	2020/06/25	[CORE-459] MapRDB spark connector shows incorrect results when Secondary Index enabled on fields
a3728cf	2020/07/08	MapR [SPARK-763] Spark and Hive integration fails for spark-submit
4204015	2020/07/17	MapR [SPARK-679] Spark application fail with "Cannot recover key"
49d1fa9	2020/07/17	MapR [SPARK-767] Backport SPARK-29444 into Spark 2.4.5
5b2ddba	2020/07/29	MapR [SPARK-775] Update json4s library
3b74fbb	2020/08/14	MapR [SPARK-785] Investigate Spark-2.4.5 SLF4J messages on job start
54cec38	2020/08/17	MapR [SPARK-780] Can't download event logs from SHS
d12e0db	2020/08/18	MapR [SPARK-778] Delete ssl keys for spark job during spark package remove
b206526	2020/08/27	MapR [SPARK-786] Spark thrift server fails to read from hive-maprdb json table

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comment
e13fc5a	2020/08/28	MapR [SPARK-791] Spark does not work on non-secure cluster
c932822	2020/09/01	MapR [SPARK-793] Update Zookeeper to 4 digits version

Known Issues

- MapR [SPARK-688] - The Spark Web UI does not display properly using Spark 2.4.5 in EEP 7.0.0. **Workaround:** Display the Spark job through the YARN ResourceManager UI by setting the `spark.ui.reverseProxy` property to `true`. Note that when the property is set to `true`, the Spark worker and application UI is not accessible directly. You can only access the UI through the Spark Master or proxy public URL. This behavior is described in the Apache Spark documentation.
- MapR [SPARK-742] - On a secure cluster, if `spark.ssl.ui.enabled` is not set or is set to `true`, you cannot access the Spark job through the YARN ResourceManager UI. You can access the Spark job using the URL (`https://<node.name>:4440`). **Workaround:** Set `spark.ssl.ui.enabled` to `false`. When `spark.ssl.ui.enabled` is set to `false`, you can access the Spark UI directly by using the URL (`http://<node.name>:4040`) or by using the YARN ResourceManager UI.

Resolved Issues

- None.

Spark 2.4.4.500 - 2201 (EEP 6.3.6) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.4.4.

The notes below relate specifically to the MapR Technologies Distribution for Apache Hadoop. For more information, see open-source Spark 2.4.4 Release Notes.

These release notes contain only MapR Technologies specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.4.4.500
Release Date	January 2022
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.4.4.500-mapr-636
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Beginning with EEP 6.0.0, the keyStore and trustStore password can be removed from `spark-defaults.conf` and set in `/opt/mapr/conf/ssl-client.xml`.
- Beginning with EEP 6.0.0, after an upgrade, the previous version's configuration files are saved in the `/opt/mapr/spark` directory.
- MapR 6.1.0 with EEP 6.0.0 and later support simplified security. If you enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive, but has the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but is not fully compatible with Hive. See the Apache Spark documentation and the MapR Spark documentation for details.

New in This Release

- For a complete list of new features, see the open-source Spark 2.4.4 Release Notes.

Fixes

This HPE release includes the following new fixes since the latest data-fabric Spark release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY/MM/DD)	HPE Fix Number and Description
1cb0587	2021/12/26	MapR [SPARK-986] log4j-1.2.17.jar vulnerability:CVE-2019-17571
0580236	2021/12/29	MapR [SPARK-975] Spark CVE fixes for Jan 2022 release
7bec164	2022/01/14	MapR [SPARK-994] Update jackson-mapper-asl v1.9.13 to 1.9.13-atlassian-5
b3eef67	2022/01/25	MapR [SPARK-1001] Update log4j v1 to the 1.3.1-mapr

Known Issues

- None.

Resolved Issues

- None.

Spark 2.4.4.400 - 2110 (EEP 6.3.5) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.4.4.

The notes below relate specifically to the MapR Data Platform Distribution for Apache Hadoop. For more information, you may also wish to consult the open-source Spark 2.4.4 Release Notes.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.4.4.400
Release Date	October 2021
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.4.4.400-mapr-635
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Beginning with EEP 6.0.0, the keyStore and trustStore password can be removed from `spark-defaults.conf` and set in `/opt/mapr/conf/ssl-client.xml`.
- Beginning with EEP 6.0.0, after an upgrade, the previous version's configuration files are saved in the `/opt/mapr/spark` directory.
- MapR 6.1.0 with EEP 6.0.0 and later support simplified security. If you enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive, but has the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but is not fully compatible with Hive. See the Apache Spark documentation and the MapR Spark documentation for details.

New in This Release

- For a complete list of new features, see the open-source Spark 2.4.4 Release Notes.

Fixes

This HPE release includes the following new fixes since the latest data-fabric Spark release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY/MM/DD)	HPE Fix Number and Description
ce96abb	2021/05/24	MapR [SPARK-874] DStream: Spark processes queued time=t+interval batch even if time=t batch failed
1d80bac	2021/06/02	[SPARK-30225][CORE] Correct read() behavior past EOF in NioBufferedFileInputStream
53ae06b	2021/06/04	MapR [SPARK-847] Spark can't read data from symlink

GitHub Commit Number	Date (YYYY/MM/DD)	HPE Fix Number and Description
87516c8	2021/06/09	MapR [SPARK-894] Hadoop artifacts should be taken from the cluster
35bb433	2021/06/10	MapR [SPARK-892] Customer's question about several CVE's impact
e0dd5d4	2021/08/12	MapR [SPARK-900] Introduce configuration option for symlink support
c6be19f	2021/09/15	MapR [SPARK-941] Problem with count via spark-shell
c442957	2021/09/17	MapR [SPARK-914] CVE-2020-13956, WS-2017-3734 vulnerabilities in http-client

Known Issues

- None.

Resolved Issues

- None.

Spark 2.4.4.300 - 2104 (EEP 6.3.4) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.4.4.

The notes below relate specifically to the MapR Data Platform Distribution for Apache Hadoop. For more information, you may also wish to consult the open-source Spark 2.4.4 Release Notes.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.4.4.300
Release Date	April 2021
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.4.4.300-mapr-2104
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Beginning with EEP 6.0.0, the keyStore and trustStore password can be removed from `spark-defaults.conf` and set in `/opt/mapr/conf/ssl-client.xml`.
- Beginning with EEP 6.0.0, after an upgrade, the previous version's configuration files are saved in the `/opt/mapr/spark` directory.
- MapR 6.1.0 with EEP 6.0.0 and later support simplified security. If you enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive, but has the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but is not fully compatible with Hive. See the Apache Spark documentation and the MapR Spark documentation for details.

New in This Release

- For a complete list of new features, see the open-source Spark 2.4.4 Release Notes.

Fixes

This HPE release includes the following new fixes since the latest data-fabric Spark release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY/MM/DD)	HPE Fix Number and Description
f2fae47	2021/01/08	MapR [SPARK-833] Spark thriftserver can't start after the latest commit
55646ea	2021/01/08	MapR [SPARK-812] Update Ant version to 1.10.9 to avoid CVE-2020-1945
888a055	2021/01/27	[SPARK-32723][WEBUI] Upgrade to jQuery 3.5.1
54ffbd2	2021/04/05	MapR [SPARK-841] Backport SPARK-32723
6ee1b8a	2021/04/05	MapR [SPARK-862] Excessive messages for spark processes

Known Issues

- SPARK-865: If you run a `configure.sh` command to configure HBase after Spark configuration, then you must manually copy the `hbase-site.xml` configuration file from the HBase configuration directory to the Spark configuration directory.

Resolved Issues

- None.

Spark 2.4.4.200 - 2101 (EEP 6.3.2) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.4.4.

The notes below relate specifically to the HPE Distribution for Apache Hadoop. For more information, you may also wish to consult the open-source Spark 2.4.4 Release Notes.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.4.4
Release Date	January 2021
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.

Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.4.4-mapr-2101
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Beginning with EEP 6.0.0, the keyStore and trustStore password can be removed from `spark-defaults.conf` and set in `/opt/mapr/conf/ssl-client.xml`.
- Beginning with EEP 6.0.0, after an upgrade, the previous version's configuration files are saved in the `/opt/mapr/spark` directory.
- MapR 6.1.0 with EEP 6.0.0 and later support simplified security. If you enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive, but has the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but is not fully compatible with Hive. See the Apache Spark documentation and the MapR Spark documentation for details.

New in This Release

- For a complete list of new features, see the open-source Spark 2.4.4 Release Notes.

Fixes

This HPE release includes the following new fixes since the latest data-fabric Spark release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
93c13d1	2020/09/17	MapR [SPARK-795] - extended logging for Spark Streaming and Structured Streaming
4a16ca2	2020/10/26	MapR [SPARK-703] Fix org.codehaus.jackson vulnerability
86dcb1f	2020/10/29	MapR [SPARK-790] Update XercesImpl to 2.12.0 or newer for mapr-ojai-driver
6947744	2020/11/03	MapR [SPARK-804] sparkContext.loadFromMapRDB throwing RuntimeException in scala code
d627e2b	2020/11/19	MapR [SPARK-812] Update Ant version to avoid CVE-2020-1945
39aeff0	2020/11/22	MapR [SPARK-819] CVE-2019-0205: Update Thrift version to v0.13.0
05138f5	2020/11/30	[SPARK-33405][BUILD][2.4] Upgrade commons-compress to 1.20

GitHub Commit	Date (YYYY-MM-DD)	Comment
6bf5fc0	2020/11/30	MapR [SPARK-822] WS-2019-0379: update commons-codec jar for spark
fbca5d3	2020/12/03	[SPARK-31095][BUILD][2.4] Upgrade netty-all to 4.1.47.Final
cb91f89	2020/12/09	MapR [SPARK-824] CVE-2020-11612: Netty vulnerabilities
141fd70	2020/12/21	MapR [SPARK-831] Can't download event logs from SHS for spark 2.4.4
4e96618	2020/12/29	MapR [SPARK-832] Multiple Vulnerabilities including Spark and hive jars found
7bba98f	2021/01/04	MapR [SPARK-776] IllegalArgumentException: Null user is thrown when using some methods from SparkContext

Known Issues

- None.

Resolved Issues

- None.

Spark 2.4.4-2009 (EEP 6.3.1) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.4.4.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. For more information, you may also wish to consult the open-source Spark 2.4.4 Release Notes.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.4.4
Release Date	September 2020
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.4.4-mapr-2009
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Beginning with EEP 6.0.0, the keyStore and trustStore password can be removed from `spark-defaults.conf` and set in `/opt/mapr/conf/ssl-client.xml`.
- Beginning with EEP 6.0.0, after an upgrade, the previous version's configuration files are saved in the `/opt/mapr/spark` directory.
- MapR 6.1.0 with EEP 6.0.0 and later support simplified security. If you enable security on your MapR cluster, MapR scripts automatically configure Spark security features.
- Beginning with EEP 6.3.0, the Spark MapRDB JSON connector supports secondary indexes.
- Beginning with EEP 6.3.0, Spark supports configurable HTTP security headers.
- Beginning with MapR 6.2 and EEP 7.0, Spark supports SSL for WebUI.

Hive Support

This version of Spark supports integration with Hive, but has the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but is not fully compatible with Hive. See the Apache Spark documentation and the MapR Spark documentation for details.

New in This Release

- For a complete list of new features, see the open-source Spark 2.4.4 Release Notes.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
e5a8ff7	2020/01/09	MapR [SPARK-642] Improve work with security headers
13584a9	2020/01/13	MapR [SPARK-642] Improve work with security headers
f6f14db	2020/01/15	MapR [SPARK-654] Update protobuf version to 3.X.X for Spark
db6c15f	2020/01/21	MapR [SPARK-585] Sertificates generation for cluster mode
4e5d6fd	2020/02/10	[SPARK-27992][SPARK-28881] [PYTHON][2.4] Allow Python to join with connection thread to propagate errors
79366d5	2020/02/11	[SPARK-29410][BUILD] Update commons-beanutils to 1.9.4
b05a4ae	2020/02/18	MapR [SPARK-662] Spark job is failed with updated protobuf version
2c7467b	2020/02/18	MapR [SPARK-664] commons-beanutils as dependency for mapr-spark

GitHub Commit	Date (YYYY-MM-DD)	Comment
d95867e	2020/03/03	MapR [SPARK-677] No TLS protocol logging for Spark thrift server & spark history server
5a9b5f8	2020/04/07	[SPARK-30238][SQL][2.4] hive partition pruning can only support string and integral types
add846c	2020/04/08	MapR [SPARK-691] maprdb-spark jar MetaTableImpl class reference in DBOlderClientImpl
8259253	2020/04/08	MapR [SPARK-705] Spark streaming fail with NoSuchMethodError: org.apache.kafka.clients.producer.ProducerConfig.getBoolean(Ljava/lang/String;)Ljava/lang/Boolean;
8c4e21c	2020/04/19	MapR [SPARK-717] Spark Master start fail after installation due to javax.security.auth.login.LoginException
61b7376	2020/04/27	MapR [SPARK-724] Application startup fail with java.lang.NoClassDefFoundError: com/google/protobuf/GeneratedMessageV3
eec85e1	2020/04/28	MapR [SPARK-715] Driver https port bind fail when application starts under non-mapr user
e28302b	2020/04/30	[SPARK-29445][CORE] Bump netty-all from 4.1.39.Final to 4.1.42.Final
3f8b017	2020/05/19	MapR [SPARK-740] UI shows "Kafka 0.9 direct stream" even if 0.10 is used
f014001	2020/05/20	MapR [SPARK-737] Calling poll(1000) from paranoidPoll causes batch scheduling delay
ccda955	2020/05/22	MapR [SPARK-738] CLONE - Check if jackson-databind-2.9.4.jar will work with Spark 2.4.X in MEP 6 and MEP7
5e30e36	2020/05/22	[SPARK-26966][ML] Update to JPMML 1.4.8
fbc0634	2020/05/22	MapR [SPARK-714] Update bcprov-jdk15on to fix CVE issue
3348810	2020/05/27	[K8S-1458] Spark 2.4.4 encounters error and does not start executors
2b903c6	2020/06/12	MapR [SPARK-746] Secondary index not picked
8154afe	2020/06/16	MapR [SPARK-751] Spark Jetty servers send webserver details to client
e8afcde	2020/06/25	[CORE-459] MapRDB spark connector shows incorrect results when Secondary Index enabled on fields

GitHub Commit	Date (YYYY-MM-DD)	Comment
e74a28e	2020/07/01	[SPARK-29055][CORE] Update driver/executors' storage memory when block is removed from BlockManager
79912a9	2020/07/17	MapR [SPARK-767] Backport SPARK-29444 into Spark 2.4.4
432729b	2020/08/19	MapR [SPARK-781] SSL for spark UI services
ee0107a	2020/08/28	MapR [SPARK-759] Allow PAM and MapR-Sasl to be enabled at the same time

Known Issues

- None.

Resolved Issues

- None.

Spark 2.4.4.0-1912 Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.4.4.0.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. For more information, you may also wish to consult the open-source Spark 2.4.4 Release Notes.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.4.4.0
Release Date	December 2019
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.4.4.0-mapr-630
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Beginning with EEP 6.0.0, the keyStore and trustStore password can be removed from `spark-defaults.conf` and set in `/opt/mapr/conf/ssl-client.xml`.
- Beginning with EEP 6.0.0, after an upgrade, the previous version's configuration files are saved in the `/opt/mapr/spark` directory.
- MapR 6.1.0 with EEP 6.0.0 and later support simplified security. If you enable security on your MapR cluster, MapR scripts automatically configure Spark security features.
- Beginning with EEP 6.3.0, the Spark MapRDB JSON connector supports secondary indexes.
- Beginning with EEP 6.3.0, Spark supports configurable HTTP security headers.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

- MapR Spark ACLs behave like Apache Spark ACLs. For details, see the ACL Configuration for Spark documentation.
- For a complete list of new features, see the open-source Spark 2.4.4 Release Notes.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
47147db	2019/09/20	MapR [SPARK-609] Port Apache Spark-2.4.4 changes to the MapR Spark-2.4.4 branch
b41bbe0	2019/09/25	MapR [SPARK-614] Error in sparkR while reading avro and parquet file formats
ca0c9c4	2019/10/07	MapR [SPARK-618] Update hive dependencies for spark-2.4.4 to 2.3.6 version
2827bed	2019/10/07	MapR [SPARK-619] Move absent commits from 2.4.3 branch to 2.4.4
118c6c5	2019/10/09	MapR [SPARK-620] Replace core dependency in Spark-2.4.4
c996bb0	2019/10/09	MapR [SPARK-595] Spark cannot access hs2 through zookeeper
e758a24	2019/10/11	MapR [SPARK-621] Add custom http header support. Improve work with security headers.

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comment
4000048	2019/10/16	MapR [SPARK-617 Can't use ssl via spark beeline
d3a0ec5	2019/10/22	MapR [SPARK-340] Jetty web server version at Spark should be updated to v9.4.X
c7e076e	2019/10/22	MapR [SPARK-626] Update kafka dependencies for Spark 2.4.4.0 in release MEP-6.3.0
c5cbbcc	2019/10/23	MapR [MS-925] After upgrade to MEP 6.2 (Spark 2.4.0) can no longer
5eaced8	2019/11/07	MapR [SPARK-629] Spark UI for job lose CSS styles
b0d5ee9	2019/11/11	MapR [SPARK-639] Default headers are adding two times
c99e9c9	2019/11/15	MapR [SPARK-627] SparkHistoryServer-2.4 is getting 403 Unauthorized home page for users(spark.ui.view.acls) via spark-submit

Known Issues

- MapR [SPARK-593], MapR [SPARK-558] - A Spark job can hang and the job output can be redirected to the `/opt/mapr/logs/pam.log` file if you use the spark-shell during login to the Spark Driver UI or if you try to open the Spark Web UI before it is initialized.
- MapR [SPARK-573] - A Spark job on a standalone node fails via the mapr-client. This happens because the `spark-defaults.conf` file can't be configured by Spark `configure.sh` because core `configure.sh` doesn't call it. **Workaround:** Two workarounds are possible:
  - Copy the `spark-defaults.conf` file from `/opt/mapr/spark/spark-<version>/conf/` into the same folder on the client node.
  - Run Spark `configure.sh` directly.

  The first workaround is more secure and stable, but both workarounds can be unreliable:
  - In some cases, copying the `spark-defaults.conf` file may be not enough.
  - Spark `configure.sh` is not documented for external use. In addition, Spark `configure.sh` is run implicitly by core `configure.sh`, and running it directly with the wrong commands can break the Spark configuration

Resolved Issues

- None.

Spark 2.4.0.0-1904 (EEP 6.2.0) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.4.0.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. This release of Spark has backward-compatibility changes, see the open-source Spark 2.4.0.0 Release Notes for more information.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.4.0.0
Release Date	April 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.4.0.0-mapr-1904
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Starting with EEP 6.0.0, `keyStore` and `trustStore` passwords can be removed from the `spark-defaults.conf` file and can be set in the `/opt/mapr/conf/ssl-client.xml` file.
- Starting with EEP 6.0.0, after an upgrade, configuration files of previous versions are saved in the `/opt/mapr/spark` directory.
- The MapR 6.1 and EEP 6.0.0 release introduces "Simplified Security". If you are using these versions and enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

- For a complete list of all new features, refer to the open source documentation.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.3.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
4bdca6c	2019-02-25	MapR [SPARK-427] Update kafka in Spark-2.4.0 to the 1.1.1-mapr
0ccea10	2019-02-25	MapR [SPARK-434] Move absent commits from 2.3.2 branch
0af5795	2019-02-25	MapR [SPARK-442] Spark build fails beacuse of the wrong tests in spark-streaming-kafka-10 module

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comment
d40b974	2019-02-25	MapR [SPARK-446] Spark configure.sh doesn't start/stop Spark services
d42400a	2019-02-25	MapR [SPARK-430] PID files should be under /opt/mapr/pid
b7eec10	2019-02-25	MapR [SPARK-221] Investigate possibility to move creating of the spark-env.sh from private-pkg to configure.sh
399d5b8	2019-02-25	MapR [SPARK-287] Move logic of creating /apps/spark folder from installer's scripts to the configure.sh
0170f29	2019-02-25	[SPARK-449] Kafka offset commit issue fixed
2497c80	2019-02-25	MapR [SPARK-417] impersonation fixes for spark executor. Impersonation is moved from HadoopRDD.compute() method to org.apache.spark.executor.Executor.run() method
e1d14ed	2019-02-25	MapR [SPARK-456] Spark shell can't be started
1cc194b	2019-02-26	[SPARK-466] SparkR errors fixed
9c4cf43	2019-02-26	[SPARK-379] Fix Spark version for Avro and Kubernetes integration tests
4436a8a	2019-02-26	MapR [SPARK-464] Can't submit spark 2.4 jobs from mapr-client
b14e1a6	2019-02-27	MapR [SPARK-465] Error messages after update of spark 2.4
c9fa510	2019-02-28	MapR [K8S-637][K8S] Add configure.sh configuration in spark-defaults.conf for job runtime
11e3daf	2019-02-28	MapR [SPARK-481] Cannot run spark configure.sh on Client node
4a740fb	2019-03-01	MapR [SPARK-486][K8S] Fix sasl encryption error on Kubernetes
30f88de	2019-03-07	MapR [SPARK-416] CVE-2018-1320 vulnerability in Apache Thrift
a3f0109	2019-03-08	MapR [SPARK-496] Spark HS UI doesn't work
f60e8a4	2019-03-08	MapR [SPARK-482] Spark streaming app fails to start by UnknownTopicOrPartitionException with checkpoint
71f5db9	2019-03-15	MapR [SPARK-514] Recovery from checkpoint is broken
ba9e107	2019-03-18	MapR [SPARK-515] Move configuring spark-env.sh back to the private-pkg
9fbdc61	2019-03-19	MapR [SPARK-515][K8S] Remove configure.sh call for k8s

GitHub Commit	Date (YYYY-MM-DD)	Comment
cbbd78f	2019/03/19	MapR [SPARK-492] Spark 2.4.0.0 configure.sh has error messages
fce6079	2019/03/19	SPARK-463] MAPR_MAVEN_REPO variable for specifying mapR repository
100aff7	2019/03/22	MapR [SPARK-494] Spark - Distribute Notice.txt across components starting with MEP 6.2
a4e4259	2019/03/25	MapR [SPARK-460] Spark Metrics for CollectD Configuration for collecting Spark metrics
7615273	2019/03/26	MapR [SPARK-510] nonmapr "admin" users not able to view other user logs in SHS
80edc50	2019/03/26	[SPARK-508] MapR-DB OJAI Connector for Spark isNull condition returns incorrect result
dfc0022	2019/03/28	MapR [SPARK-462] Spark and SparkHistoryServer allow week ciphers, which can allow man in the middle attack
baf607e	2019/03/28	MapR [SPARK-461] Stop graph after jobs completion to prevent 'java.lang.IllegalStateException: No active subscriptions'
d48945f	2019/04/04	MapR [SPARK-516] Spark jobs failure using yarn mode on kerberos fixed
1c793f8	2019/04/11	MapR [SPARK-531] Remove duplicating entries from classpath in ClasspathFilter
6a39ff6	2019/04/11	SPARK-444] Fix of hive version for spark dev branches
c5aeb67	2019/04/15	Spark 2.4.0 backport 2.4.1
2ae047f	2019/04/19	SPARK-539 Workaround for absent MapRDBJsonSplit class
94eb0f1	2019/04/20	K8S-853: Enable spark metrics for external tenant
c7abaf8	2019/04/22	MapR [SPARK-536] PySpark streaming package for kafka-0-10 added
ef70d34	2019/04/22	MapR [SPARK-540] Include 'avro' artifacts
f08108e	2019/04/23	MapR [K8S-893] Hide plain text password from logs
28ddfe9e	2019/05/17	MapR [SPARK-541] Avoid duplication of the first unexpired record

The following tickets are back-ported from Spark 2.4.1:

- SPARK-26709 - OptimizeMetadataOnlyQuery does not correctly handle the files with zero record

last-updated: Apr 11, 2022

- SPARK-26080 - Unable to run worker.py on Windows
- SPARK-26873 - FileFormatWriter creates inconsistent MR job IDs
- SPARK-26745 - Non-parsing Dataset.count() optimization causes inconsistent results for JSON inputs with empty lines
- SPARK-26677 - Incorrect results of not(eqNullSafe) when data read from Parquet file
- SPARK-26708 - Incorrect result caused by inconsistency between a SQL cache's cached RDD and its physical plan
- SPARK-26267 - Kafka source may reprocess data
- SPARK-26706 - Fix Cast$mayTruncate for bytes
- SPARK-26078 - WHERE .. IN fails to filter rows when used in combination with UNION
- SPARK-26233 - Incorrect decimal value with java beans and first/last/max... functions
- SPARK-27097 - Avoid embedding platform-dependent offsets literally in whole-stage generated code
- SPARK-26188 - Spark 2.4.0 Partitioning behavior breaks backwards compatibility
- SPARK-25921 - Python worker reuse causes Barrier tasks to run without BarrierTaskContext

Known Issues

- `pyspark.sql.utils.AnalysisException` - Python OJAI connector failure caused by incorrect resolution of python user-defined function calls by Spark SQL parser.

  The same SQL expressions from SELECT clause and GROUP BY clause resolves to different expression IDs.

  Sample SQL query that leads to `pyspark.sql.utils.AnalysisException`, the `stringtodate1(yelping_since)` expression is used in SELECT and GROUP BY, `stringtodate1` is python user-defined function:

  ```
  SELECT business_id, stringtodate1(yelping_since) AS startyear,
  avg(stars) AS avgstars FROM temp_table_name GROUP BY business_id,
  stringtodate1(yelping_since)
  ```

  Workaround: `stringtodate1(yelping_since)` expression in GROUP BY is replaced with alias `startyear`.

  ```
  SELECT business_id, stringtodate1(yelping_since) AS startyear, avg(stars)
  AS avgstars FROM temp_table_name GROUP BY business_id, startyear
  ```

Resolved Issues

- None.

Spark 2.3.3.0-1904 (EEP 6.1.1 and EEP 6.0.2) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.3.3.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. This release of Spark has backward-compatibility changes, see the open-source Spark 2.3.3 Release Notes for more information.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.3.3
Release Date	April 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.3.3-mapr-1904
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Starting with EEP 6.0.0, `keyStore` and `trustStore` passwords can be removed from the `spark-defaults.conf` file and can be set in the `/opt/mapr/conf/ssl-client.xml` file.
- Starting with EEP 6.0.0, after an upgrade, configuration files of previous versions are saved in the `/opt/mapr/spark` directory.
- The MapR 6.1 and EEP 6.0.0 release introduces "Simplified Security". If you are using these versions and enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

- For a complete list of all new features, refer to the open source documentation.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.3.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
63acd83	2019/01/23	MapR [SPARK-374] Spark Hive example fails when we submit job from another(simple) cluster user
9a634a7	2019/01/23	MapR [SPARK-396] Interface change of sendToKafka
2b2c934	2019/01/25	MapR [SPARK-357] consumer groups are prepeneded with a "service_" prefix

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comment
0ab4b07	2019/01/30	MapR [SPARK-429] Changes in maprdb connector are the cause of broken backward compatibility
9421abc	2019/02/13	MapR [SPARK-417] impersonation fixes for spark executor. Impersonation is moved from HadoopRDD.compute() method to org.apache.spark.executor.Executor.run() method
684235c	2019/02/15	[SPARK-449] Kafka offset commit issue fixed
524de52	2019/02/27	MapR [SPARK-465] Error messages after update of spark 2.4
9d04094	2019/03/07	CVE-2018-1320 vulnerability in Apache Thrift
f3296aa	2019/03/08	MapR [SPARK-482] Spark streaming app fails to start by UnknownTopicOrPartitionException with checkpoint
39f7af7	2019/03/12	MapR [SPARK-445] Messages loss fixed by reverting [MAPR-32290]
ffda9a5	2019/03/13	[SPARK-25471][PYTHON][TEST] Fix pyspark-sql test error when using Python 3.6 and Pandas 0.23
78ec0b2	2019/03/14	MapR [SPARK-499] version is changed to 2.3.3
54e66b3	2019/03/19	MapR [SPARK-463] MAPR_MAVEN_REPO variable for specifying mapR repository
2b37b41	2019/03/26	MapR [SPARK-510] nonmapr "admin" users not able to view other user logs in SHS
3bf2145	2019/03/26	[SPARK-508] MapR-DB OJAI Connector for Spark isNull condition returns incorrect result
f0745b8	2019/03/26	MapR [SPARK-482] Spark streaming app fails to start by UnknownTopicOrPartitionException with checkpoint
648fb33	2019/03/28	MapR [SPARK-462] Spark and SparkHistoryServer allow week ciphers, which can allow man in the middle attack
50abc9d	2019/03/28	MapR [SPARK-461] Stop graph after jobs completion to prevent 'java.lang.IllegalStateException: No active subscriptions'
0bb97c7	2019/04/04	MapR [SPARK-516] Spark jobs failure using yarn mode on kerberos fixed
a8e8a19	2019/04/11	MapR [SPARK-444] Fix of hive version for spark dev branches

GitHub Commit	Date (YYYY-MM-DD)	Comment
e2e9ea0	2019/04/19	SPARK-539 Workaround for absent MapRDBJsonSplit class
2bdf60b	2019/04/22	MapR [SPARK-536] PySpark streaming package for kafka-0-10 added
65ad08a	2019/05/17	MapR [SPARK-541] Avoid duplication of the first unexpired record

Known Issues

- None.

Resolved Issues

- None.

Spark 2.2.1 - 2101 (EEP 5.0.6) Release Notes

The notes below relate specifically to the MapR Data Platform distribution for Apache Hadoop. You may also be interested in the open source Spark 2.2.1 Release Notes.

Spark Version	2.2.1
Release Date	January, 2021
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.2.1-mapr-2101
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Package Names for MapR Ecosystem Packs (MEPs)

Important Notes

- Although Spark 2.2 can connect to Hive Metastore 2.1, features of Hive that were added after Hive 1.2 are not supported by Spark.

  As of Spark 2.2.1 and EEP 5.0, Spark uses Kafka-1.0.1.

- Spark Yarn and standalone modes are only supported on clusters in MRv2 (YARN) mode. Spark Yarn and standalone modes not supported on clusters in MRv1 (classic) mode.

- Core 6.0 and EEP 5.0 introduce "Simplified Security." If you are using these versions and enable security in your cluster, scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.

- Spark-SQL is supported, but not fully compatible with Hive. See the Apache Spark documentation and the HPE Spark documentation for details.

last-updated: Apr 11, 2022

Fixes

This release includes the following new patches since the latest HPE Spark 2.2.1 release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
5c6b8d6	2020/11/04	MapR [SPARK-804] sparkContext.loadFromMapRDB throwing RuntimeException in scala code
aecb512	2020/11/23	MapR [SPARK-798] Structured Streaming rewinds offset to 0 upon re-subscription
db7098c	2021/01/11	MapR [SPARK-834] Backport SPARK-795 into Spark 2.2.1 (MEP 5.x)

Known Issues and Limitations

None.

Resolved Issues

None.

Spark 2.2.1-2009 (EEP 5.0.5) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.2.1.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. For more information, you may also wish to consult the open-source Spark 2.2.1 release notes.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.2.1
Release Date	September 2020
MapR Version Interoperability	See Component Versions for Released EEPs on page 5586 and EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.2.1-mapr-2009
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Although Spark 2.2 can connect to Hive Metastore 2.1, Hive features added after Hive 1.2 are not supported by Spark.
- Beginning in Spark 2.2.1 and EEP 5.0. Kafka-1.0.1 is using by Spark.
- Spark Yarn and Standalone modes are only supported on clusters in MRv2 (YARN) mode. It is not supported on clusters in MRv1 (classic) mode.
- MapR 6.0 and EEP 5.0 introduce "Simplified Security." If you are using these versions and enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive, but has the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but is not fully compatible with Hive. See the Apache Spark documentation and the MapR Spark documentation for details.

New in This Release

- For a complete list of new features, see the open-source Spark 2.2.1 release notes.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
ee0107a	2020/02/25	MapR [SPARK-675] SSL truststore password should NOT be logged
aeeb51b	2020/03/10	MapR [SPARK-677] No TLS protocol logging for Spark thrift server & spark history server
510ab00	2020/03/23	MapR [SPARK-694] Duplicate py4j zip files in python/lib causing exceptions
5979337	2020/04/30	MapR [SPARK-704] Backporting netty update from SPARK-22324
692f67d	2020/08/20	MapR [SPARK-787] HCP usage should be turn off by default

Known Issues

- None.

Resolved Issues

- None.

Spark 2.2.1-1912 (EEP 5.0.4) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.2.1.

last-updated: Apr 11, 2022

The notes below relate specifically to the MapR Distribution for Apache Hadoop. This release of Spark has backward-compatibility changes, see the open-source Spark 2.2.1 Release Notes for more information.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.2.1
Release Date	December 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.2.1-mapr-1912
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Spark 2.2 can connect to Hive Metastore 2.1, but features of Hive added after Hive 1.2 are not supported by Spark.
- Starting from Spark 2.2.1 and EEP 5.0.0, Spark uses Kafka version 1.0.1.
- Spark Yarn and Standalone modes are supported only on clusters in MRv2 (YARN) mode. They are not supported on clusters in MRv1 (classic) mode.
- MapR 6.0 and EEP 5.0 and later introduce security by default. If you are using these versions and enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

None.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.2.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
e77ddc4	2019/06/04	MapR [SPARK-545] PySpark streaming package for kafka-0-9 fixed
3bd05f3	2019/06/06	MapR [SPARK-541] Avoid duplication of the first unexpired record

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comment
fa252d8	2019/06/14	MapR [SPARK-333] Render application UI init page if driver is not up
d4fde38	2019/07/02	[SPARK-24002][SQL] Task not serializable caused by org.apache.parquet.io.api.Binary$ByteBufferBackedBinary.getBytes
90499d5	2019/07/31	MapR [SPARK-592] Add possibility to use start-thriftserver.sh script with 2304 port
41e68c4	2019/10/15	MapR [SPARK-595] Spark cannot access hs2 through zookeeper
c8111ff	2019/11/12	MapR [SPARK-575] Warning messages in spark workspace after the second attempt to login to job's UI
e17c039	2019/11/12	MapR [SPARK-641] backport SPARK-21357 into mapr-spark-2.2.1

Known Issues

- None.

Resolved Issues

- None.

Spark 2.2.1-1904 (EEP 5.0.3) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.2.1.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. This release of Spark has backward-compatibility changes, see the open-source Spark 2.2.1 Release Notes for more information.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.3.3
Release Date	April 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.2.1-mapr-1904
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Starting with EEP 6.0.0, `keyStore` and `trustStore` passwords can be removed from the `spark-defaults.conf` file and can be set in the `/opt/mapr/conf/ssl-client.xml` file.
- Starting with EEP 6.0.0, after an upgrade, configuration files of previous versions are saved in the `/opt/mapr/spark` directory.
- The MapR 6.1 and EEP 6.0.0 release introduces "Simplified Security". If you are using these versions and enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

- For a complete list of all new features, refer to the open source documentation.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.3.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
41854fb	2019/01/23	MapR [SPARK-396] Interface change of sendToKafka
d6c12f5	2019/02/05	[SPARK-383] Avoid using 'SimpleChannelInboundHandler' to prevent 'TypeParameterMatcher' generation
26388c1	2019/02/07	[SPARK-10878][CORE] Fix race condition when multiple clients resolvesartifacts at the same time
a2f38c9	2019/02/15	[SPARK-449] Kafka offset commit issue fixed
ef8f5cd	2019/03/07	CVE-2018-1320 vulnerability in Apache Thrift
314e8e4	2019/03/12	MapR [SPARK-445] Messages loss
4a9322d	2019/03/15	MapR [SPARK-514] Recovery from checkpoint is broken
14618bb	2019/03/19	SPARK-463] MAPR_MAVEN_REPO variable for specifying mapR repository
b9e7086	2019/03/26	[SPARK-508] MapR-DB OJAI Connector for Spark isNull condition returns incorrect result

GitHub Commit	Date (YYYY-MM-DD)	Comment
924aa51	2019/03/28	MapR [SPARK-510] nonmapr "admin" users not able to view other user logs in SHS
2ca0af1	2019/03/28	MapR [SPARK-461] Stop graph after jobs completion to prevent 'java.lang.IllegalStateException: No active subscriptions'
0607a96	2019/04/04	[SPARK-21642][CORE] Use FQDN for DRIVER_HOST_ADDRESS instead of ip address
ff48dbc	2019/04/11	MapR [SPARK-444] Fix of hive version for spark dev branches

Known Issues

- None.

Resolved Issues

- None.

Spark 2.3.2.0-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.3.2.0.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. This release of Spark has backward-compatibility changes, see the open-source Spark 2.3.2.0 Release Notes for more information.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.3.2.0
Release Date	February 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.3.2-mapr-1901
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Starting with EEP 6.0.0, `keyStore` and `trustStore` passwords can be removed from the `spark-defaults.conf` file and can be set in the `/opt/mapr/conf/ssl-client.xml` file.
- Starting with EEP 6.0.0, after an upgrade, configuration files of previous versions are saved in the `/opt/mapr/spark` directory.
- The MapR 6.1 and EEP 6.0.0 release introduces "Simplified Security". If you are using these versions and enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

For a complete list of all new features, refer to the open source documentation.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.3.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
fddc84f	2018/10/01	MapR [SPARK-325] Add examples for work with the MapR-DB JSON connector into the Spark project
a2b88ef	2018/10/16	MapR [SPARK-331] Remove snapshot versions of MapR dependencies from Spark
4f380b4	2018/10/23	[MAPR-32263] Seek called on unsubscribed partitions
e8d59b9	2018/10/24	[MAPR-32290] Spark processing offsets when messages are already TTL in the first batch
58b3115	2018/11/14	MapR [SPARK-352] Spark shell fails with "NoClassDefFoundError: org/apache/hadoop/fs/FSDataInputStream" if java is not available in PATH
fcee5ab	2018/11/15	MapR [SPARK-350] Deprecate Spark Kafka-09 package
a2a3a0f	2018/11/19	MapR [SPARK-326] Investigate possibility of writing Java example for the MapR-DB OJAI connector
da4e0b3	2018/11/27	MapR [SPARK-356] Merge MapR changes from kafka-09 package into the kafka-10
3608c7b	2018/11/29	MapR [SPARK-319] Fix for SparkR version check
2aaa6a5	2018/12/03	MapR [SPARK-349] Update OJAI client to v3 for Spark MapR-DB JSON connector
71f6153	2018/12/10	MapR [SPARK-137] Analyze the warning during compilation of OJAI connector
f62df7a	2018/12/13	MapR [SPARK-369] Spark 2.3.2 fails with error related to zookeeper

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comment
a2bf94f	2018/12/18	[MAPR-26258] hbasecontext.HBaseDistributedScan Example fails
cd75768	2018/12/27	MapR [SPARK-372] Support 4 digits version scheme for Spark
182a7b1	2019/01/08	[MSPARK-24355] Spark external shuffle server improvement to better handle block fetch requests.
6b28052	2019/01/09	MapR [SPARK-381] Kafka version changed to '1.1.1-mapr-SNAPSHOT'
fec2162	2019/01/11	MapR [SPARK-374] Spark Hive example fails when we submit job from another(simple) cluster user
609c0a8	2019/01/11	MapR [SPARK-363] Hive requests fail with ClassNotFoundException
e6a8a6b	2019/01/17	MapR [SPARK-410] All submitted jobs fail for Spark 2.3.2.0
f032694	2019/01/18	MapR [SPARK-363] Hive version changed to '1.2.0-mapr-spark-MEP-6.0.0'
2686b34	2019/01/21	MapR [SPARK-373] Unexpected behavior during job running in standalone cluster mode
b991202	2019/01/22	MapR [SPARK-419] Update hive-maprdb-json-handler jar for spark 2.3.2.0
63acd83	2019/01/23	MapR [SPARK-374] Spark Hive example fails when we submit job from another (simple) cluster user

Known Issues

- None.

Resolved Issues

- None.

Spark 2.3.1-1808 (EEP 6.0.0) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.3.1-1808.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. This release of Spark has backward-compatibility changes, see the open-source Spark 2.3.1 Release Notes for more information.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.3.1
Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536.

Source on GitHub	https://github.com/mapr/spark/tree/2.3.1-mapr-1808
GitHub Release Tag	2.3.1-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Starting with EEP 6.0.0, `keyStore` and `trustStore` passwords can be removed from the `spark-defaults.conf` file and can be set in the `/opt/mapr/conf/ssl-client.xml` file.
- Starting with EEP 6.0.0, after an upgrade, configuration files of previous versions are saved in the `/opt/mapr/spark` directory.
- The MapR 6.1 and EEP 6.0.0 release introduces "Simplified Security". If you are using these versions and enable security on your MapR cluster, MapR scripts automatically configure Spark security features.
- The encryption algorithms used to Configure SSL Encryption for Spark on YARN on page 4037 are no longer available for your web service to pick up. You need to remove the `spark.ssl.enabledAlgorithms TLS_RSA_WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_256_CBC_SHA` line to let parties negotiate the matching ciphers.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

For a complete list of all new features, refer to the open source documentation.

The following features of Spark 2.3.1 are **NOT officially supported**:

- Continuous Processing in Structured Streaming
- Stream-Stream Joins in Structured Streaming
- Spark on Kubernetes support

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.3.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
484df64	2018/05/03	[MAPR-31305] Spark History server NOT loading applications submitted by users other than "mapr"

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comment
1542dfc	2018/05/04	MapR [SPARK-181] Kafka 0.10 Structured Streaming unit tests fixed
c08d3c7	2018/05/07	MapR [SPARK-210] Rename sprk-defaults.conf to spark-defaults.conf.template
7fe6261	2018/05/11	MapR [SPARK-227] KafkaUtils.createDirectStream fails with kafka-09
563e240	2018/05/11	MapR [SPARK-76] spark configure.sh should not restart the service every time it is run
0a9f1ca	2018/05/15	MapR [SPARK-213] Loading of data for parquet files bug fixed
f8660f2	2018/05/17	MapR [SPARK-220] SparkR fails with UDF functions bug fixed
a0f0d55	2018/05/22	[SPARK-24062][THRIFT SERVER] Fix SASL encryption cannot enabled issue in thrift server
d570fcb	2018/05/23	MapR [SPARK-209] Implement saving of user configurations during ecosystem package update
44318f1	2018/05/25	MapR [SPARK-226] Spark - pySpark Security Vulnerability
769754f	2018/05/29	MapR [SPARK-244] Provide ability to use MapR-Negotiation authentication for Spark HistoryServer
4db1e9a	2018/05/30	MapR [SPARK-216] Spark thriftserver fails when work with hive-maprdb json table
527522e	2018/05/30	MapR [SPARK-214] Hive-2.1 properties cannot be read from a hive-site.xml as Spark uses Hive-1.2
b9a5c43	2018/05/31	MapR [SPARK-248] MapRDBTableScanRDD fails to convert to Scala Dataframe when using where clause
a0524c3	2018/06/01	MapR [SPARK-255] Installer fresh install 610/600 secure fails to start "mapr-spark-thriftserver" and "mapr-spark-historyserver"
a30ec42	2018/06/05	MapR [SPARK-256] Spark does not work in Yarn mode
a0abaaa	2018/06/09	MapR [SPARK-260] fix EC option handling

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comment
ad91e9a	2018/06/12	MapR [SPARK-259] Spark application does not finish correctly
9be83a0	2018/06/13	MapR [SPARK-261] Use mapr-security-web for getting passwords
b296795	2018/06/13	[MAPR-31632] RM UI showing broken page for Spark jobs
cc5c12e	2018/06/14	MapR [SPARK-263] Add possibility to use keyPassword which is different from keyStorePassword
528e034	2018/06/18	MapR [SPARK-266] Spark jobs cannot finish correctly, when there is an error during job running
63c159b	2018/06/21	MapR [SPARK-272] Use only client passwords from ssl-client.xml
917558f	2018/06/26	MapR [SPARK-273] Update Hive-1.2 dependencies in Spark
bbb05f9	2018/06/26	MapR [SPARK-276] Update zookeeper dependency to v.3.4.11 for spark 2.3.1
ce5a14b	2018/07/02	MapR [SPARK-279] Cannot connect to spark thrift server with new spark and hive packages
e316246	2018/07/02	MapR [SPARK-278] Spark submit fails for jobs with python
8dc3e24	2018/07/09	MapR [SPARK-282] Remove maprfs and hadoop jars from mapr spark package
aeb5e3a	2018/07/10	[SPARK-212] SparkHiveExample fails when we run it twice
b444403	2018/07/13	[SPARK-130] MapR Database connector - NPE while saving Pair RDD with "null" values
3ca1f51	2018/07/16	MapR [SPARK-281] Spark configure.sh -R is ignoring custom security and overriding hive-site.xml
ac12360	2018/07/24	MapR [SPARK-277] Spark thriftserver fails when we try inserting for hive-maprdb json table
925c9fb	2018/07/27	MapR [SPARK-296] Structured Streaming memory leak
940f23d	2018/08/03	MapR [SPARK-297] Added unit test for empty value conversion

GitHub Commit	Date (YYYY-MM-DD)	Comment
67cb089	2018/08/08	[SPARK-302] Local privilege escalation
17e3c3b	2018/08/10	[SPARK-301] Error while submitting job in Standalone cluster mode on MapR secure cluster
832411e	2018/08/14	[SPARK-306] Kafka clients 1.0.1 present in jars directory for Spark 2.3.1
c1ee416	2018/08/15	[MAPR-32014] Spark Consumer fails with java.lang.AssertionError

Known Issues

- You cannot connect to a Spark Thrift Server on a Kerberos-secured cluster as Kerberos and SSL are not compatible.

  Workaround: Modify the `hive.server2.use.SSL` to `false` in the `hive-site.xml` file.

- When you install a secure (MapR-SASL) cluster using the MapR Installer, the `configure.sh` script configures Hive after Spark. As a result, Spark copies the wrong `hive-site.xml` file and the Spark and Hive integration may not work correctly and you may have problems connecting to Spark beeline.

  Workaround: Check the `hive-site.xml` file in the Spark home directory, and, if needed, rerun the `configure.sh` script or copy the `hive-site.xml` file from your Hive home directory and restart services.

Resolved Issues

None.

Spark 2.2.1-1901 (EEP 5.0.2) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.2.1.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. This release of Spark has backward-compatibility changes, see the open-source Spark 2.1.0 Release Notes for more information.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.2.1
Release Date	February 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.2.1-mapr-1901
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Spark 2.2 can connect to Hive Metastore 2.1. But, features of Hive added after Hive 1.2 are not supported by Spark.
- Starting from Spark 2.2.1 and EEP 5.0.0 Kafka version is updated to 1.0.1.
- Spark Yarn and Standalone modes are supported only on clusters in MRv2 (YARN) mode. It is not supported on clusters in MRv1 (classic) mode.
- MapR 6.0 and EEP 5.0 and later introduce security by default. If you are using these versions and enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

None.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.2.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
e444b4a	2018/10/01	MapR [SPARK-316] Backporting request for SPARK-22062 and SPARK-21475
fc8076b	2018/10/01	[SPARK-22033][CORE] BufferHolder, other size checks should account for the specific VM array size limitations
f1dd6d9	2018/10/17	[MAPR-32263] Seek called on unsubscribed partitions
874aefc	2018/10/24	[MAPR-32290] Spark processing offsets when messages are already ttl in first batch
7ecab19	2018/11/19	[SPARK-357] consumer groups are prepeneded with a "service_" prefix
9cf9c83	2018/12/21	[MAPR-26258] hbasecontext.HBaseDistributedScan Example fails
f0bb7de	2019/01/04	MapR [SPARK-390] Fix for hive version
b346a9f	2019/01/04	MapR [SPARK-311] Spark beeline uses default ssl truststore instead of mapr ssl truststore

GitHub Commit	Date (YYYY-MM-DD)	Comment
22d8192	2019/01/08	MapR [SPARK-382] Codegen issue fixed
66ada4a	2019/01/08	MapR [SPARK-319] Fix for sparkR version check
70fc922	2019/01/22	MapR [SPARK-419] Update hive-maprdb-json-handler jar for spark 2.2.1

Known Issues

- None.

Resolved Issues

- None.

Spark 2.2.1-1808 (EEP 5.0.1) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.2.1-1808.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 2.2.1 Release Notes.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.3.1
Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark/tree/2.2.1-mapr-1808
GitHub Release Tag	2.2.1-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Spark 2.2 can connect to Hive Metastore 2.1. But, features of Hive added after Hive 1.2 are not supported by Spark.
- Starting from Spark 2.2.1 and EEP 5.0.0 Kafka version is updated to 1.0.1.
- MapR 6.0 and EEP 5.0 and later introduce security by default. If you are using these versions and enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.

last-updated: Apr 11, 2022

- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.2.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
6adbb83	2018/04/25	[MAPR-31202] Spark History Server bug fixed. Redundant 'jsr311-api' artifact excluded
bbfaa66	2018/05/03	[MAPR-31305] Spark History server NOT loading applications submitted by users other than 'mapr'
e2953fc	2018/05/10	MapR [SPARK-227] KafkaUtils.createDirectStream fails with kafka-09
f747793	2018/05/22	MapR [SPARK-244] Added impersonation for history server
1b45342	2018/05/26	MapR [SPARK-226] Spark - pySpark Security Vulnerability
ff78e8c	2018/05/30	MapR [SPARK-216] Spark thriftserver fails when work with hive-maprdb json table
9859ca9	2018/05/30	MapR [SPARK-214] Hive-2.1 poperties can't be read from a hive-site.xml as Spark uses Hive-1.2
eb8710e	2018/05/31	Mapr [SPARK-248] MapRDBTableScanRDD fails to convert to Scala Dataframe when using where clause
2dc24ef	2018/06/13	[MAPR-31632] RM UI showing broken page for Spark jobs
aa624b7	2018/07/18	[WEBUI] Avoid possibility of script in query param keys
c5af6d1	2018/08/02	MapR [SPARK-300] Update hive dependencies for spark 2.2.1
759cdaf	2018/08/02	MapR [SPARK-297] Empty values are loaded as non-null
eedbccc	2018/08/02	MapR [32014] Spark Consumer fails with java.lang.AssertionError
f697fdd	2018/08/03	MapR [SPARK-297] Added unit test for empty value conversion
90e2f7c	2018/08/07	MapR [SPARK-281] Spark configure.sh -R is ignoring custom security and overriding hive-site.xml
f1ca279	2018/08/08	[SPARK-302] Local privilege escalation

GitHub Commit	Date (YYYY-MM-DD)	Comment
1d8577d	2018/08/19	[MAPR-32167] - SparkSQL queries fails with org.apache.spark.sql.catalyst.errors.package$TreeNodeException after upgrade
7c0017b	2018/08/28	[SPARK-16986][WEB-UI] Converter Started, Completed and Last Updated to client time zone in history page
fe84d4b	2018/08/30	MapR [SPARK-279] Can't connect to spark thrift server with new Spark and Hive packages

Known Issues

- You cannot connect to a Spark Thrift Server on a Kerberos-secured cluster as Kerberos and SSL are not compatible.

  Workaround: Modify the `hive.server2.use.SSL` to `false` in the `hive-site.xml` file.

- When you install a secure (MapR-SASL) cluster using the MapR Installer, the `configure.sh` script configures Hive after Spark. As a result, Spark copies the wrong `hive-site.xml` file and the Spark and Hive integration may not work correctly and you may have problems connecting to Spark beeline.

  Workaround: Check the `hive-site.xml` file in the Spark home directory, and, if needed, rerun the `configure.sh` script or copy the `hive-site.xml` file from your Hive home directory and restart services.

- Spark versions up to and including 2.3.0 have the following security vulnerability:
  - CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Resolved Issues

None.

Spark 2.1.0-1904 (EEP 4.1.4) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.1.0.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. This release of Spark has backward-compatibility changes, see the open-source Spark 2.1.0 Release Notes for more information.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.1.0
Release Date	April 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.1.0-mapr-1904
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Spark 2.2 can connect to Hive Metastore 2.1. But, features of Hive added after Hive 1.2 are not supported by Spark.
- Spark Yarn and Standalone modes are supported only on clusters in MRv2 (YARN) mode. They are not supported on clusters in MRv1 (classic) mode.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

- For a complete list of all new features, refer to the open source documentation.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.3.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
a51f07e	2019/03/28	MapR [SPARK-461] Stop graph after jobs completion to prevent "java.lang.IllegalStateException: No active subscriptions"

Known Issues

- None.

Resolved Issues

- None.

Spark 2.1.0-1901 (EEP 4.1.3 and EEP 3.0.5) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.1.0-1901.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 2.1.0 Release Notes.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.1.0
Release Date	February 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.1.0-mapr-1901

Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Spark 2.2 can connect to Hive Metastore 2.1. But, features of Hive added after Hive 1.2 are not supported by Spark.
- Spark Yarn and Standalone modes are supported only on clusters in MRv2 (YARN) mode. They are not supported on clusters in MRv1 (classic) mode.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.2.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
e6a9733	2018/11/01	[SPARK-19263] DAGScheduler should avoid sending conflicting task set
664b59f	2018/11/01	MapR [SPARK-266] Spark jobs can't finish correctly, when there is an error during job running
7c714d7	2019/01/22	MapR [SPARK-419] Update hive-maprdb-json-handler jar for spark

Known Issues

- None.

Resolved Issues

- None.

Spark 2.1.0-1808 (EEP 3.0.4 and EEP 4.1.2) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.1.0-1808.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 2.1.0 Release Notes.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Spark Version	2.1.0

last-updated: Apr 11, 2022

Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark/tree/2.1.0-mapr-1808
GitHub Release Tag	2.1.0-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Spark 2.2 can connect to Hive Metastore 2.1. But, features of Hive added after Hive 1.2 are not supported by Spark.
- Spark Yarn and Standalone modes are supported only on clusters in MRv2 (YARN) mode. They are not supported on clusters in MRv1 (classic) mode.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.2.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
6fb5a8d	2018/04/25	[MAPR-31202] Spark History Server bug fixed. Redundant 'jsr311-api' artifact excluded
40d7890	2018/05/02	[MAPR-31274] Spark SQL Thrift Server cannot access hive-maprdbjson table
244a07c	2018/05/26	MapR [SPARK-226] Spark - pySpark Security Vulnerability
43d2314	2018/05/30	MapR [SPARK-214] Hive-2.1 poperties can't be read from a hive-site.xml as Spark uses Hive-1.2
a6b5a92	2018/06/13	[MAPR-31632] RM UI showing broken page for Spark jobs
232dbad	2018/07/18	[WEBUI] Avoid possibility of script in query param keys
455a796	2018/08/02	MapR [SPARK-300] Update Hive dependencies for spark 2.1.0
ec8bd03	2018/08/08	[SPARK-302] Local privilege escalation

GitHub Commit	Date (YYYY-MM-DD)	Comment
2a1db27	2018/08/28	[SPARK-16986][WEB-UI] Converter Started, Completed and Last Updated to client time zone in history page
55195ae	2018/08/30	MapR [SPARK-279] Cannot connect to Spark Thrift Server with new Spark and Hive packages

Known Issues

- You cannot connect to a Spark Thrift Server on a Kerberos-secured cluster as Kerberos and SSL are not compatible.

  Workaround: Modify the `hive.server2.use.SSL` to `false` in the `hive-site.xml` file.

- When you install a secure (MapR-SASL) cluster using the MapR Installer, the `configure.sh` script configures Hive after Spark. As a result, Spark copies the wrong `hive-site.xml` file and the Spark and Hive integration may not work correctly and you may have problems connecting to Spark beeline.

  Workaround: Check the `hive-site.xml` file in the Spark home directory, and, if needed, rerun the `configure.sh` script or copy the `hive-site.xml` file from your Hive home directory and restart services.

- Spark versions up to and including 2.3.0 have the following security vulnerability:
  - CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Resolved Issues

None.

Spark 2.2.1-1803 (EEP 5.0.0) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.2.1-1803.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 2.2.1 Release Notes.

Spark Version	2.2.1
Release Date	March 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.2.1-mapr-1803
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Spark 2.2 can connect to Hive Metastore 2.1. But, features of Hive added after Hive 1.2 are not supported by Spark.
- Starting from Spark 2.2.1 and EEP 5.0.0 Kafka is updated to 1.0.1.
- MapR 6.0 and EEP 5.0 and later introduce built-in security. If you are using these versions and enable security on your MapR cluster, MapR scripts automatically configure Spark security features.
- Spark Master port was changed to 8580.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

- Support for Structured Streaming. See Structured Spark Streaming.
- Structured Streaming MapR Database OJAI Sink. See Structured Spark Streaming.
- PAM for Spark Web UIs on secure clusters. See PAM Authentication for Spark.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.2.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
5430a1d	2018/01/22	MapR [SPARK-16] Spark 2.2.1 porting
b3a4ffa	2018/01/22	[MAPR-30228] Spark example job failed with "/opt/mapr/conf/ssl_keystore (Permission denied)" error on Spark 2.1.X EEP 4.0 - MapR SASL security enabled
aeb9e9d	2018/01/22	MapR [SPARK-144] Add insertToMapRDB method for rdd for Java API
489e21d	2018/01/23	MapR [SPARK-16] Change Spark version in Warden files and configure.sh
548e492	2018/02/01	MapR [SPARK-143] Added PAM for Spark UIs for secure clusters
2499aba	2018/02/06	MapR [SPARK-135] Spark 2.2 with MapR Streams (Kafka 1.0)
1b37b08	2018/02/07	MapR [SPARK-21] Structured Streaming MapR Database Sink created

GitHub Commit	Date (YYYY-MM-DD)	Comment
2d9e466	2018/02/07	[MAPR-30583] InMemoryFileIndex changed to getFileBlockLocations in a parallel way
1a2e864	2018/02/08	MapR [SPARK-152] Incorrect date string parsing fixed
cca410c	2018/02/15	MapR [SPARK-153] Exception in spark job with configured labels on yarn-client mode
6f83937	2018/02/20	MapR [SPARK-159] Added possibility to configure secure ports as part of security by default
f341e85	2018/02/20	MapR [SPARK-155] Changed Spark Master port from 8080
f417a68	2018/02/26	MapR [SPARK-161] Include Kafka Structured streaming jar to Spark package
77ed36f	2018/02/28	MapR [SPARK-164] Update Kafka version to 1.0.1-mapr in Spark Kafka Producer module
5a390c1	2018/03/07	MapR [SPARK-170] StackOverflowException in equals method in DBMapValue
45cea4a	2018/03/19	MapR [SPARK-154] Spark R NoSuchElementException during start
3725eed	2018/03/21	MapR [SPARK-191] Incorrect work of MapR Database Sink 'complete' and 'update' modes fixed
844642e	2018/03/23	MapR [SPARK-143] Spark History Server does not require login for secured-by-default clusters
71cd6d2	2018/03/25	MapR [SPARK-194] Redirect to Spark History server
1c093b1	2018/03/26	MapR [SPARK-188] Could not connect to thrift server via spark beeline on kerberos cluster
4eff5df	2018/04/02	[SPARK-198] Update hadoop dependency version to 2.7.0-mapr-1803 for Spark 2.2.1
d324bfa	2018/04/04	[SPARK-205] Change Kafka dependencies versions

Known Issues

Spark versions up to and including 2.3.0 have the following security vulnerability:

- CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Resolved Issues

None.

last-updated: Apr 11, 2022

Spark 2.1.0-1803 (EEP 4.1.1) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.1.0-1803.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 2.1.0 Release Notes.

Spark Version	2.1.0
Release Date	March 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.1.0-mapr-1803
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Spark 2.2 can connect to Hive Metastore 2.1. But, features of Hive added after Hive 1.2 are not supported by Spark.
- Spark Yarn and Standalone modes are supported only on clusters in MRv2 (YARN) mode. It is not supported on clusters in MRv1 (classic) mode.
- MapR 6.0 and EEP 4.1 and later introduce built-in security. If you are using these versions and enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

- PAM for Spark Web UIs on secure clusters.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.2.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
0a08b10	2018/01/18	[MAPR-30536] Spark SQL queries on Map column failed after upgrade
f160e85	2018/02/01	MapR [SPARK-143] Added PAM for Spark UIs for secure clusters

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comment
46da505	2018/02/07	[MAPR-30583] PartitioningAwareFileIndex changed to getFileBlockLocations in a parallel way
bf7296a	2018/02/08	MapR [SPARK-152] Incorrect date string parsing fixed
4c01f15	2018/03/01	[SPARK-21181] Release byteBuffers to suppress netty error messages
9473a5f	2018/03/07	MapR [SPARK-171] StackOverflowException in equals method in DBMapValue
3b5d608	2018/03/19	MapR [SPARK-190] Spark R NoSuchElementException during start
3e3f07e	2018/03/23	MapR [SPARK-143] Spark History Server does not require login for secured-by-default clusters
44f7810	2018/03/25	MapR [SPARK-194] Redirect to Spark History server

Known Issues

Spark versions up to and including 2.3.0 have the following security vulnerability:

- CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Resolved Issues

Bug 25770: Removed invalid error message displayed at times during the running of Spark application:

```
ERROR MapRFileSystem: Failed to delete path, error: No such file or
directory
          (Tickets: MapR-28620, MapR-25770, MFS-1897)
```

Spark 2.1.0-1803 (EEP 3.0.3) Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.1.0-1803.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 2.1.0 Release Notes.

Spark Version	2.1.0
Release Date	March 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.1.0-mapr-1803
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Spark 2.2 can connect to Hive Metastore 2.1. But, features of Hive added after Hive 1.2 are not supported by Spark.
- Spark Yarn and Standalone modes are supported only on clusters in MRv2 (YARN) mode. It is not supported on clusters in MRv1 (classic) mode.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

- None.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.2.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
0a08b10	2018/01/18	[MAPR-30536] Spark SQL queries on Map column failed after upgrade
46da505	2018/02/07	[MAPR-30583] PartitioningAwareFileIndex changed to getFileBlockLocations in a parallel way
bf7296a	2018/02/08	MapR [SPARK-152] Incorrect date string parsing fixed
4c01f15	2018/03/01	[SPARK-21181] Release byteBuffers to suppress netty error messages
9473a5f	2018/03/07	MapR [SPARK-171] StackOverflowException in equals method in DBMapValue
3b5d608	2018/03/19	MapR [SPARK-190] Spark R NoSuchElementException during start

Known Issues

- The following error is displayed at times during the running of Spark application:

```
ERROR MapRFileSystem: Failed to delete path, error: No such file or
directory
(Tickets: MapR-28620, MapR-25770, MFS-1897)
```

- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Resolved Issues

None.

Spark 2.1.0-1801 Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.1.0-1801.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 2.1.0 Release Notes.

Spark Version	2.1.0
Release Date	February 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.1.0-mapr-1801
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Full support of MapR Streams is available only on MapR 5.2 and later clusters.
- Spark 2.1 can connect to Hive Metastore 2.1. But, features of Hive added after Hive 1.2 are not supported by Spark.
- Spark Standalone and Spark on YARN can only run on clusters in MRv2 (YARN) mode. They are not supported on clusters in MRv1 (classic) mode.
- MapR 6.0 and EEP 4.0 and later introduce built-in security. If you are using these versions and enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

- Support for Java and Python APIs for MapR Database OJAI connector. See MapR Database OJAI Connector for Apache Spark.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.1.0 release. For details, refer to the commit log for this project in GitHub.

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comment
5430a1d	2017/10/31	MapR [SPARK-107] idField information is lost in MapRDBDataFrameWriterFunctions. saveToMapRDB
68c211b	2017/11/16	MapR [SPARK-113] Hit java.lang.UnsupportedOperationException: empty.reduceLeft during loadFromMapRDB
a325770	2017/11/27	MapR [SPARK-125] Enable handling default value of idFieldPath parameter
737e2ac	2017/11/28	[SPARK-18827][CORE] Fix cannot read broadcast on disk
f02a1dc	2017/11/28	[SPARK-19104][BACKPORT-2.1] [SQL] Lambda variables in ExternalMapToCatalyst is made global
594d5d4	2017/11/29	MapR [SPARK-121] Spark OJAI JAVA: Read to Dataset functionality implementation
2a8a6c1	2017/11/29	MapR [SPARK-128] MapR Database connector - Fix wrong handle of null fields when nullable is false
06c6597	2017/12/05	MapR [SPARK-131] Exception when trying to save JSON table with Binary _id field
b273661	2017/12/05	MapR [SPARK-118] Spark OJAI Python: Read implementation
b8adcd0	2017/12/05	MapR [SPARK-117] Spark OJAI Python: Save functionality implementation
ef88f8a	2017/12/13	MapR [SPARK-118] Spark OJAI Python: Move MapR Database Connector class importing in order to fix MapR [ZEP-101] interpreter issue
3d7e193	2017/12/13	MapR [SPARK-118] Spark OJAI Python: Missed DataFrame import while moving imports in order to fix MapR [ZEP-101] interpreter issue
7e3e1e7	2017/12/14	MapR [SPARK-121] Spark OJAI JAVA: Update functionality removed
5f2dd1d	2017/12/26	MapR [SPARK-140] Change the option name "tableName" to "tablePath" in the Spark/MapR Database connectors
c7f2f8a	2017/12/28	MapR [SPARK-139] Remove "update" related APIs from connector
496f040	2017/12/28	[SPARK-21321][SPARK CORE] Spark very verbose on shutdown
0a08b10	2018/01/18	[MAPR-30536] Spark SQL queries on Map column fails after upgrade

Known Issues

- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Resolved Issues

Users logged in with a normal user account (not mapr or root) can run spark jobs on the cluster without disabling Spark SSL.

Spark 2.1.0-1710 Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.1.0-1710.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 2.1.0 Release Notes.

Spark Version	2.1.0
Release Date	November 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.1.0-mapr-1710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

Important:

- Full support of MapR Streams is available only on MapR 5.2 and later clusters.
- Spark 2.1 can connect to Hive Metastore 2.1. But, features of Hive added after Hive 1.2 are not supported by Spark.
- Spark Standalone and Spark on YARN can only run on clusters in MRv2 (YARN) mode. They are not supported on clusters in MRv1 (classic) mode.
- MapR 6.0 and EEP 4.0 introduce "Simplified Security". If you are using these versions and enable security on your MapR cluster, MapR scripts automatically configure Spark security features.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

- Simplified Security - Starting in the MapR 6.0 and EEP 4.0 releases, you can use the "Enable Security" check box in the installer to enable security for the core platform and the installed ecosystem components. Alternatively, running `configure.sh -R` enables Spark security features if you have enabled security on your MapR cluster. See Security with Spark Standalone on page 4032 and Security with Spark on YARN on page 4036 for more information.

last-updated: Apr 11, 2022

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.1.0 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
f1a4e96	2017/08/03	[SPARK-16845][SQL] Fix code generation to prevent `GeneratedClass$SpecificOrdering` from growing beyond 64 KB
13def70	2017/08/07	[MAPR-28339] Fix race condition resulting in delete failures, when Spark SQL queries use "SaveAsTable" API
c64da80	2017/08/09	[MAPR-20331][SQL] Enhance Hive partition pruning predicate pushdown
b27ea82	2017/08/17	[MAPR-28705] Fix late arriving message
c170554	2017/08/17	[MAPR-18971][CORE] Upgrade Netty to 4.0.43.Final version
0de12ba	2017/08/18	[MAPR-28659] Fix issue where executor threads in Spark executor are stuck in a lock state
6a33e6f	2017/08/28	[MAPR-19307][PYSPARK] Ensure user conf is propagated to SparkContext
85004b1	2017/08/29	[MAPR-28460] Fix impersonation when data read from MapR Database via Spark-Hive
a75bbe8	2017/08/29	[SPARK-39] Remove ambiguous dependencies from Spark classpath
c5a87b0	2017/09/06	[MAPR-29052] Use waitForConsumerAssignment() instead of consumer.poll(0) to avoid initialization error in MapR Event Store For Apache Kafka client
cf96fdd	2017/09/11	[MAPR-29106] - Fix unsafe deserialization in Apache Spark launcher API
7942de3	2017/09/18	[SPARK-18991][CORE] Change ContextCleaner.referenceBuffer to use ConcurrentHashMap to make it faster
3d44f64	2017/09/18	[SPARK-45] Move Spark-OJAI connector code to Spark github repo
e2a4e2a	2017/09/25	[SPARK-20358][CORE] Fix executors failing stage on interrupted exception by cancelled tasks
e39c4b7	2017/10/03	[SPARK-69] Fix license problem when reading from JSON and writing to MapR Database
d444627	2017/10/06	[MAPR-29014]Fix message offset for MapR Core 6.0

Known Issues

- **MAPR-17271:** On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Resolved Issues

None.

Spark 2.1.0-1707 Release Notes

This section provides reference information, including new features, patches, and known issues for Spark 2.1.0-1707.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 2.1.0 Release Notes.

Spark Version	2.1.0
Release Date	August 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.1.0-mapr-1707
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

Note:

- Full support of MapR Streams is available only on MapR 5.2 and later clusters.
- Spark 2.1 can connect to Hive Metastore 2.1. But, features of Hive added after Hive 1.2 are not supported by Spark.
- Spark Standalone and Spark on YARN can only run on clusters in MRv2 (YARN) mode. They are not supported on clusters in MRv1 (classic) mode.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

New in This Release

Spark 2.1.0-1707 introduces the following enhancement:

- Spark on Mesos

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.1.0 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
40dca4e	2017/07/28	[MAPR-28441] - Fix Spark Streaming's handling of zero offsets from Kafka 0.9
99daf6b	2017/06/30	[SPARK-19182][DSTREAM] Optimize the lock in `StreamingJobProgressListener` to not block UI when generating streaming jobs.
8e3b9ed	2017/06/27	Revert earlier fix for MAPR-25770.
52b06b9	2017/06/23	[MAPR-27845] Fix the manner in which Spark determines Hive's security configuration.
6917681	2017/06/14	[MAPR-27840] Fix wrong type casting when importing data from Oracle.
17f311e	2017/06/06	[SPARK-20393][WEBU UI] Strengthen Spark to prevent XSS vulnerabilities.
e10e660	2017/05/30	Revert [SPARK-16736][CORE][SQL] to avoid superfluous filesystem calls.
d8f8657	2017/05/29	[SPARK-18949][SQL] [BACKPORT-2.1] Add `recoverPartitions` API to `Catalog` interface.
7733a1c	2017/05/29	[SPARK-19459][SQL][BRANCH-2.1] Support nested char and varchar fields in ORC.
f31976e	2017/05/22	[MAPR-27519] Improve performance of calculating web UI counters for Kafka-streaming.
6a3b683	2017/05/22	[SPARK-19276][CORE] Expose `FetchFailure` exceptions hidden by user exceptions.
333371c	2017/05/22	[SPARK-19597][CORE] Add a test case for task deserialization errors.
377e2ea	2017/05/22	[SPARK-17931] Eliminate unnecessary task serialization.
0788b14	2017/05/22	[SPARK-18662] Move resource managers to their own sub-directories.
fb5fca1	2017/05/22	Fix Mesos build breakage for Scala 2.10.
e05a1e9	2017/05/22	[SPARK-17062][MESOS] Add conf option to Mesos disfixer.
e48f72e	2017/05/22	[SPARK-18836][CORE] Improve performance by serializing a single copy of the task metrics in the `DAGScheduler`.
807ba4d	2017/05/22	[SPARK-18761][CORE] Introduce a "task reaper" to oversee killing of tasks in executors.

GitHub Commit	Date (YYYY-MM-DD)	Comment
54413bd	2017/05/22	[SPARK-20359][SQL] Avoid unnecessary execution in `EliminateOuterJoin` that can lead to NPE.
c919bdb	2017/05/22	[SPARK-19893][SQL] Avoid running DataFrame set operations on map types.
bd90640	2017/05/22	[SPARK-18863][SQL] Return an error if a subquery's output contains non-aggregate expressions without GROUP BY.
b898e28	2017/05/22	[SPARK-20280][CORE] Fix `FileStatusCache Weigher` to avoid integer overflow.
7c3b1b2	2017/05/22	[SPARK-19748][SQL] Fix refresh of an `InMemoryFileIndex` with `FileStatusCache`. Correct the order of operations.
58f2250	2017/05/22	[SQL] Improve the readability of partition handling code.
b3430f7	2017/05/22	[SPARK-20059][YARN] Use the correct classloader for `HBaseCredentialProvider`.
285be99	2017/05/19	[SPARK-20043][ML] Fix the `DecisionTreeModel` so the `ImpurityCalculator` builder handles uppercase impurity type Gini.
9c60a4d	2017/05/19	[SPARK-20125][SQL] Fix conversion of an Option type to a DataSet, when the Option contains a map type.
6663ca6	2017/05/19	[SPARK-18717][SQL] Fix code generation when mapping to an immutable Scala Map.
d602458	2017/05/19	[SPARK-20086][SQL] Fix `CollapseWindow` so it does not collapse dependent adjacent windows.
4755b36	2017/05/19	[SPARK-19925][SPARKR] Fix SparkR `spark.getSparkFiles` to avoid failures when called on executors.
dac1c4a	2017/05/19	[SPARK-19237][SPARKR][CORE] Fix `spark-submit` on Windows to handle the case where Java is not installed.
f48a43a	2017/05/19	SPARK-20017][SQL] Fix the `str_to_map` and `explode` functions to avoid NPEs. Change the nullability of the `StringToMap` function from false to true.

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comment
6e5245d	2017/05/19	[SPARK-19980][SQL] [BACKPORT-2.1] Fix DataSet transformations on POJOs to preserve nulls. Add NULL checks in the Bean serializer.
bbd0c4d	2017/05/19	[SPARK-19872] [PYTHON] Fix `UnicodeDecodeError` in PySpark when reading from a text file with repartition. Use the correct deserializer for RDD construction for coalesce and repartition.
20579df	2017/05/19	[SPARK-19887][SQL] Fix handling of dynamic partition keys when persisting tables.
ff91608	2017/05/19	[SPARK-19611][SQL] Fix breakages for Hive tables backed by case sensitive data files. Introduce configurable table schema inference.
e9984d0	2017/05/19	[SPARK-19082][SQL] Fix config option `ignoreCorruptFiles` for Parquet files.
014e909	2017/05/19	[SPARK-19857][YARN] Correct calculation of next credential update time.
15fd019	2017/05/19	[SPARK-19765][SPARK-18549] [SPARK-19093][SPARK-19736] [BACKPORT-2.1][SQL] Backport cache related fixes from Spark 2.2 to Spark 2.1.
c689b5c	2017/05/19	[SPARK-18703][SPARK-18675][SQL] [BACKPORT-2.1] Fix CTAS for Hive serde table so it works for all Hive versions. Drop staging directories and data files that were not dropped until JVM termination.
4cf5e41	2017/05/19	[SPARK-14772][PYTHON][ML] Fix Python ML `Params.copy` method to match Scala implementation.
f8d3846	2017/05/19	[SPARK-19691][SQL][BRANCH-2.1] Fix `ClassCastException` when calculating percentile of decimal column.
5aa4a2d	2017/05/19	[SPARK-19500] [SQL] Fix failure in radix sort when attempting to spill the aggregated hash map.
20806a8	2017/05/19	[SPARK-19399][SPARKR] [BACKPORT-2.1] Fix tests broken by the introduction of R `coalesce` API for DataFrame and Column.
a4bedf1	2017/05/19	[SPARK-19399][SPARKR] Add R `coalesce` API for DataFrame and Column.
fc9e7b0	2017/05/19	[SPARK-18788][SPARKR] Add `getNumPartitions` API to SparkR.

GitHub Commit	Date (YYYY-MM-DD)	Comment
642e7bb	2017/05/19	[SPARK-18335][SPARKR] Extend `createDataFrame` to support a `numPartitions` parameter.
4ec94f5	2017/05/19	[SPARK-19342][SPARKR] Fix `collect` method for timestamp columns so it does not incorrectly covert to numeric.
d639208	2017/05/19	[SPARK-19543] Fix `from_json` when the input row is empty.
5948815	2017/05/19	[SPARK-19509][SQL] Fix Grouping Sets to handle nullable grouping columns.
f446906	2017/05/19	[SPARK-19472][SQL] Fix parser error when trying to resolve nested CASE WHEN statement with parenthesis. The statement was mistaken for a function call.
889860a	2017/05/19	[SPARK-19406][SQL] Fix function `to_json` to respect user-provided options.
2370cdf	2017/05/19	[SPARK-19396][DOC] Support case-insensitive JDBC options.
242b33c	2017/05/19	[SPARK-19324][SPARKR] Fix SparkR so it does not remove Spark JVM stdout output.
823d5e8	2017/05/19	[SPARK-19338][SQL] Include UDF names in explain output.
768a10b	2017/05/19	[SPARK-19231][SPARKR] Add error handling for download and untar of Spark releases.
d1f9ed5	2017/05/19	[SPARK-19129][SQL] Disallow ALTER TABLE drop partition with an empty partition value.
98d8d9c	2017/05/19	[SPARK-19180] [SQL] Fix incorrect offset in `OffHeapColumn`.
79ff854	2017/05/19	[SPARK-19092][SQL] [BACKPORT-2.1] Fix `save()` API in the `DataFrameWriter` to avoid a scan of all the saved files.
c44f274	2017/05/19	[SPARK-19130][SPARKR] Support setting columns to implicit literal values in SparkR.
148167b	2017/05/16	[MAPR-26414] Fix Spark History Server memory leak.
1a9b364	2017/05/15	Update dependencies after ECO-1703 release.
3554f31	2017/05/04	[SPARK-33] Fix streaming example.

GitHub Commit	Date (YYYY-MM-DD)	Comment
3ae224b	2017/04/28	[SPARK-19019][PYTHON] [BRANCH-2.0] Fix hijacked `collections.namedtuple`. Port cloudpickle changes needed for PySpark to work with Python 3.6.0.
4584170	2017/04/28	[SPARK-19146][CORE] Drop more elements when `stageData.taskData.size` > retainedTasks.
a259c8e	2017/04/28	[MAPR-26287] Remove unnecessary code from `hadoop-version-picker.sh`.
c0c94e5	2017/04/28	[MAPR-26414] Fix Spark History Server memory leak.

Known Issues

- **MAPR-17271:** On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Resolved Issues

None.

Spark 2.1.0-1703 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 2.1.0 Release Notes.

Spark Version	2.1.0
Release Date	April 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.1.0-mapr-1703
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)
API Changes for this Version	See Spark API Changes on page 4123.

Note: For some important Spark limitations, See "Known Issues and Limitations" later in this release note.

New in This Release

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
f4bf0f5	2017/03/16	[MAPR-26060] Fixed case when mapr-streams make gaps in offsets (#97).
b6f643d	2017/03/09	Ported features from kafka 10 to kafka 9
b2d468e	2017/03/09	Merge remote-tracking branch 'origin/branch-2.1.0-mapr' into branch-2.1.0-mapr.
8aba33a	2017/03/06	Merge pull request #95 from mapr/spark-2.1.1-critical-backport.
c64db71	2017/03/06	[SPARK-18589][SQL] Fix Python UDF accessing attributes from both side of join.
417eca2	2017/03/06	[SPARK-19120] Refresh Metadata Cache After Loading Hive Tables.
0422b78	2017/03/06	[SPARK-18700][SQL] Add StripedLock for each table's relation in cache.
a45edcc	2017/03/06	[SPARK-19129][SQL] SessionCatalog: Disallow empty part col values in partition spec.
b6529d8	2017/03/06	[SPARK-19520][STREAMING] Do not encrypt data written to the WAL.
edfa296	2017/03/06	[SPARK-19750][UI][BRANCH-2.1] Fix redirect issue from http to https.
0c13e47	2017/03/06	[SPARK-19652][UI] Do auth checks for REST API access (branch-2.1).
2e3fcd3	2017/03/06	[SPARK-19220][UI] Make redirection to HTTPS apply to all URIs. (branch-2.1).
bfb75e5	2017/03/06	[SPARK-19766][SQL] Constant alias columns in INNER JOIN should not be folded by FoldablePropagation rule.
611e920	2017/03/01	[MAPR-26289][SPARK-2.1] Streaming general improvements (#93).
519f6f6	2017/02/27	Merge pull request #92 from mapr/mapr-26258.
9841429	2017/02/27	Set default HBase version to 1.1.8.
8c85366	2017/02/27	[MAPR-26258] hbasecontext.HBaseDistributedScan Example fails.
82a01e7	2017/02/13	Changes from Kafka10 package were ported to Kafka09 package.

GitHub Commit	Date (YYYY-MM-DD)	Comment
7577dd7	2017/02/09	Merge pull request #88 from mapr/mapr-26076-spark-2.1.0.
a90ea6e	2017/02/09	[SPARK-15844][CORE] HistoryServer doesn't come up if spark.authenticate = true.
3a83ddb	2017/02/08	Merge pull request #87 from mapr/mapr-26053.
608e920	2017/02/08	[MAPR-26053] Include MapR Classes to the default value of spark.sql.hive.metastore.sharedPrefixes.
5fca03a	2017/01/23	Merge pull request #85 from mapr/mapr-24068.
33830be	2017/01/23	[MAPR-24068] YARN throws exception when label expression set.
e4263dc	2017/01/17	Merge pull request #84 from mapr/mapr-25807.
c9a53dc	2017/01/17	[MAPR-25807] Spark-Warehouse path computes incorrectly.
f7b6fcc	2017/01/16	Merge pull request #83 from mapr/thrift-maprsasl-spark-2.1.0.
ad8a592	2017/01/16	Add MapR-SASL support for Thrift Server.
a0d8c09	2017/01/12	Adding scala library.
a5f1bb2	2017/01/12	[MAPR-25713] Spark might try to load MapR Class Loader multiple times and fail.
6683ffc	2017/01/12	[SPARK-18528][SQL] Fix a bug to initialise an iterator of aggregation buffer.
ed5b22f	2017/01/12	[MAPR-25311] Bump Spark dependencies after ECO-1611 release.
effc5ba	2017/01/12	[MINOR] Fix spark-jars.sh script.
fa6f142	2017/01/12	[MAPR-24603] Could not launch beeline shell after starting Spark thrift server.
fc17f1a	2017/01/12	fixed syntax error in V09DirectKafkaWordCount example (#75).
c7de39f	2017/01/12	Spark 2.0.1 MAPR-streams Python API (#73).
e338b71	2017/01/12	[MAPR-24415] SPARK_JAVA_OPTS is deprecated (#71).
de237dc	2017/01/12	Kafka streaming producer added. (#66).
adb91d4	2017/01/12	Fixed Scala Style for SparkHiveExample.

GitHub Commit	Date (YYYY-MM-DD)	Comment
5e4ba56	2017/01/12	[MAPR-24491] HBase classpath might contain Hive libraries.
6935a9a	2017/01/12	Minor fix for previous commit.
14902af	2017/01/12	Added script for MAPR-24374.
ae730d2	2017/01/12	Some minor changes to spark-defaults.conf.
0d2545c	2017/01/12	Changed default HBase version to 1.1.1 in compatibility.version.
dfda5f3	2017/01/12	Streaming example was refactored.
583a764	2017/01/12	[MAPR-24470] HiveFromSpark test fails in yarn-cluster mode.
f03efbe	2017/01/12	Changed Hive execution version to 1.2.0.
e2dc96b	2017/01/12	Added spark streaming integration with kafka 0.9 and mapr-streams.
b6d6609	2017/01/12	Added MapR Repo.
6e2f22e	2017/01/12	[MAPR-23559] Spark PID in /opt/mapr/pid.
4bbbfc1	2017/01/12	[MAPR-22940] Failed to connect Spark beeline (after Spark thrift server is started) on Kerberos cluster.
979a663	2017/01/12	Remove Hive jars from generated classpath for Hive.
3bc9863	2017/01/12	Fix hardcoded Hive library path.
8801758	2017/01/12	[MAPR-23203] Remove derby jars from generated Hive classpath.
4e9bd0f	2017/01/12	[MAPR-18865] Unable to submit Spark apps from Windows client.
12d25e8	2017/01/12	Skip maven clean task on the parent module.
9ef2527	2017/01/12	New: Issue with running Hive commands in Spark.
66002ec	2017/01/12	Spark warden.services.conf should have dependency on CLDB.
ddcbed7	2017/01/12	Remove DFS shuffle settings.
b7cc4bb	2017/01/12	Fix bugs in the logic to avoid SSH for localhost.
1495ce6	2017/01/12	Copy every file in the conf directory into the distribution package.
647579b	2017/01/12	Create spark-defaults.conf for MapR.
f05c39b	2017/01/12	Avoid SSH to localhost when stopping secondary instances.
500a83c	2017/01/12	Add htrace jar to Spark classpath for hbase 0.98.

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comment
e19480e	2017/01/12	Support hbase classpath computation in util script.
8d8db1e	2017/01/12	Created ext-util.
1b5e0d3	2017/01/12	Adding external conf and scripts.
de9bdaf	2017/01/12	Enable SPARK_HIVE mode while building.
f64b395	2017/01/12	Build Spark on MapR.
de78691	2017/01/12	Spark Master failed to start in HA mode.
d9eef52	2017/01/12	The datanucleus jar in Spark need to be updated for Bug 21228.
89e1555	2017/01/12	Change dependencies to MapR and bump Hadoop version.
9e41ce0	2017/01/12	Change Spark version.

Known Issues and Limitations

- Spark 2.1 does not support Spark Structured Streaming.
- Full support of MapR Event Store For Apache Kafka is available only on clusters with MapR 5.2 and later.
- Spark 2.1 is able to connect to Hive Metastore 2.1, but features of Hive that were added after Hive 1.2 are not supported by Spark.
- Spark is not able to submit jobs to YARN when the cluster is in "classic" mode, even if YARN is installed and configured.
- **MAPR-17271:** On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
- **MAPR-26254:** Spark Standalone is not fully supported on Kerberos-secured clusters.
- **MAPR-26039:** Spark does not propagate mapr_sec_enabled variable to Driver.
- **MAPR-25770:** MapR-FS logs ERROR when Spark is trying to delete an already-deleted file.
- Filter push-down is not supported with MapR Database.
- The MapR Database Binary Connector for Apache Spark supports MapR Database binary tables except for the "bulk load" operation (SPARK-7).
- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Resolved Issues

None.

Spark 2.0.1-1707 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Spark 2.0.1-1707.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 2.0.1 Release Notes.

Spark Version	2.0.1
Release Date	August 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.0.1-mapr-1707
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

Important:

- Full support of MapR Streams is available only on MapR 5.2 and later clusters.
- You cannot submit Spark jobs in YARN mode when the cluster is running in MRv1(classic) mode. This applies even if you have installed and configured YARN in your cluster.
- When integrating Hive with Spark 2.0.1-1707, use Hive 1.2.-1707, which contains the fix for MAPR-26310.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 2.0.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
bcb1640	2017/05/22	[MAPR-27519] Improve performance of calculating web UI counters for Kafka-streaming.
ab0a3f2	2017/05/16	[SPARK-19019][PYTHON] [BRANCH-2.0] Fix hijacked `collections.namedtuple`. Port cloudpickle changes needed for PySpark to work with Python 3.6.0.
9faac4a	2017/05/05	[MAPR-26414] Fix Spark History Server memory leak.
46b1913	2017/05/04	[SPARK-33] Fix streaming example.
ab1f040	2017/05/04	[SPARK-19146][CORE] Drop more elements when stageData.taskData.size > retainedTasks.
5e61eb7	2017/05/04	[MAPR-26287] Remove unnecessary code from hadoop-version-picker.sh.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
2b16ad3	2017/05/04	[MAPR-26414] Fix Spark History Server memory leak.

Known Issues and Limitations

- **MAPR-17271:** On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
- **MAPR-25052:** Spark Thrift Server does not start on clusters secured by MapR-SASL.
- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Resolved Issues

None.

Spark 1.6.1-1707 Release Notes

This section provides reference information, including new features, fixes, known issues, and limitations for Spark 1.6.1-1707.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 1.6.1 Release Notes.

Spark Version	1.6.1
Release Date	August 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	1.6.1-mapr-1707
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

Important:

- To integrate Spark 1.6.1 with MapR Streams, you must install the latest Kafka 0.9.0.0 package.
- Full support of MapR Streams is available only on MapR 5.2 and later clusters.
- When integrating Hive with Spark 2.0.1-1707, use Hive 1.2.-1707, which contains the fix for MAPR-26310.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 1.6.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	MapR Fix Number and Description
c352e23	2017/05/16	[MAPR-27339] Fix failures when Spark jobs write to Hive tables.
0fdd46e	2017/05/15	[SPARK-16664][SQL] Fix persist calls on DataFrames with more than 200 columns.

Known Issues and Limitations

- **MAPR-17271:** On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
- **MAPR-19761:** On a secure cluster, MapR software does not support the Spark SQL Thrift JDBC server. When the cluster is secure, the Spark Thrift server will not start.
- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Resolved Issues

None.

Spark 2.0.1-1703 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 2.0.1 Release Notes.

Spark Version	2.0.1
Release Date	April 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.1.0-mapr-1703
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)
API Changes for this Version	See Spark API Changes on page 4123.

Note: For some important Spark limitations, See "Known Issues and Limitations" later in this release note.

New in This Release

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark release. In addition, Spark 2.0.1-1703 includes backports of all the fixes contained in Apache Spark 2.0.2. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
b5fdf9e	**2017/03/01**	Merge pull request #94 from mapr/mapr-26289-spark-2.0.1.
f75cad8	2017/03/01	Set default poll timeout to 120s.
1cf7251	2017/03/01	Added include-kafka-09 profile to Assembly.
c9c6030	2017/02/24	[MAPR-26060] Fixed case when mapr-streams make gaps in offsets (#91).
36debc8	2017/02/09	Merge pull request #89 from mapr/mapr-26076-spark-2.0.1.
ed262d0	2017/02/09	[SPARK-15844][CORE] HistoryServer doesn't come up if spark.authenticate = true.
674f9bd	2017/02/08	Merge pull request #86 from mapr/spark-2.0.2-porting.
529e51b	2017/02/08	Fixed version for Kafka 0.10 SQL.
e680ec2	2017/02/06	[SPARK-18283][STRUCTURED STREAMING][KAFKA] Added test to check whether default starting offset in latest.
a68148e	2017/02/06	[SPARK-18125][SQL][BRANCH-2.0] Fix a compilation error in codegen due to splitExpression.
316f706	2017/02/06	[SPARK-17849][SQL] Fix NPE problem when using grouping sets.
01f3743	2017/02/06	[SPARK-17693][SQL] [BACKPORT-2.0] Fixed Insert Failure To Data Source Tables when the Schema has the Comment Field.
a996282	2017/02/06	[SPARK-17981][SPARK-17957][SQL] [BACKPORT-2.0] Fix Incorrect Nullability Setting to False in FilterExec.
6d9dee4	2017/02/06	[SPARK-18189][SQL][FOLLOWUP] Move test from ReplSuite to prevent java.lang.ClassCircularityError.
cdd189c	2017/02/06	[SPARK-17337][SPARK-16804][SQL] [BRANCH-2.0] Backport subquery related PRs.
681a839	2017/02/06	[SPARK-18200][GRAPHX] [FOLLOW-UP] Support zero as an initial capacity in OpenHashSet.
cb68e70	2017/02/06	[SPARK-18200][GRAPHX] Support zero as an initial capacity in OpenHashSet.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
42d7574	2017/02/06	[SPARK-18111][SQL] Wrong approximate quantile answer when multiple records have the minimum value(for branch 2.0).
95aeff9	2017/02/06	[SPARK-18160][CORE][YARN] spark.files & spark.jars should not be passed to driver in yarn mode.
37fcf10	2017/02/06	[SPARK-16796][WEB UI] Mask spark.authenticate.secret on Spark environ.
b1723aa	2017/02/06	[SPARK-18133][BRANCH-2.0] [EXAMPLES][ML] Python ML Pipeline Exampl.
a7be955	2017/02/06	[SPARK-18144][SQL] logging StreamingQueryListener$QueryStartedEvent.
724a6e3	2017/02/06	[SPARK-18114][HOTFIX] Fix line-too-long style error from backport of SPARK-18114.
2f1aaa1	2017/02/06	[SPARK-18148][SQL] Misleading Error Message for Aggregation Without Window/GroupBy.
992d65f	2017/02/06	[SPARK-18189][SQL] Fix serialization issue in KeyValueGroupedDataset.
f481615	2017/02/06	[SPARK-18114][MESOS] Fix mesos cluster scheduler generage command option error.
07d3ffe	2017/02/06	[SPARK-18030][TESTS] Fix flaky FileStreamSourceSuite by not deleting the files.
5250480	2017/02/06	[SPARK-18143][SQL] Ignore Structured Streaming event logs to avoid breaking history server (branch 2.0).
bdf4511	2017/02/06	[SPARK-16312][FOLLOW-UP] [STREAMING][KAFKA][DOC] Add java code snippet for Kafka 0.10 integration doc.
ecd62ed	2017/02/06	[SPARK-18164][SQL] ForeachSink should fail the Spark job if `process` throws exception.
6cab38c	2017/02/06	[SPARK-16963][SQL] Fix test "StreamExecution metadata garbage collection".
19d27ad	2017/02/06	[SPARK-17813][SQL][KAFKA] Maximum data per trigger.
6c079b9	2017/02/06	[SPARK-18132] Fix checkstyle.
9c149f4	2017/02/06	[SPARK-18009][SQL] Fix ClassCastException while calling toLocalIterator() on dataframe produced by RunnableCommand.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
597b754	2017/02/06	[SPARK-16963][STREAMING][SQL] Changes to Source trait and related implementation classes.
38745a9	2017/02/06	[SPARK-13747][SQL] Fix concurrent executions in ForkJoinPool for SQL (branch 2.0).
aa8c453	2017/02/06	[SPARK-18104][DOC] Don't build KafkaSource doc.
6f62a53	2017/02/06	[SPARK-18063][SQL] Failed to infer constraints over multiple aliases.
a031493	2017/02/06	[SPARK-16304] LinkageError should not crash Spark executor.
3b01f41	2017/02/06	[SPARK-17733][SQL] InferFiltersFromConstraints rule never terminates for query.
67484f3	2017/02/06	[SPARK-18022][SQL] java.lang.NullPointerException instead of real exception when saving DF to MySQL.
0002f56	2017/02/06	[SPARK-16988][SPARK SHELL] spark history server log needs to be fixed to show https url when ssl is enabled.
b50e511	2017/02/06	[SPARK-18070][SQL] binary operator should not consider nullability when comparing input types.
be401c8	2017/02/06	[SPARK-17624][SQL][STREAMING][TEST] Fixed flaky StateStoreSuite.maintenance.
c03b30f	2017/02/06	[SPARK-18044][STREAMING] FileStreamSource should not infer partitions in every batch.
86e6db7	2017/02/06	[SPARK-17153][SQL] Should read partition data when reading new files in filestream without globbing.
62ecfdd	2017/02/06	[SPARK-18058][SQL][BRANCH-2.0]Comparing column types ignoring Nullability in Union and SetOperation.
7d291d4	2017/02/06	[SPARKR][BRANCH-2.0] R merge API doc and example fix.
38c59da	2017/02/06	[SPARK-17123][SQL][BRANCH-2.0] Use type-widened encoder for DataFrame for set operations.
453a44c	2017/02/06	[SPARK-17698][SQL] Join predicates should not contain filter clauses.
0ed97fe	2017/02/06	[SPARK-17986][ML] SQLTransformer should remove temporary tables.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
1ac5708	2017/02/06	[SPARK-16606][MINOR] Tiny follow-up to , to correct more instances of the same log message typo.
8049e1d	2017/02/06	[STREAMING][KAFKA][DOC] clarify kafka settings needed for larger batches.
1b55321	2017/02/06	[SPARK-17812][SQL][KAFKA] Assign and specific startingOffsets for structured stream.
f1fc622	2017/02/06	[SPARK-17929][CORE] Fix deadlock when CoarseGrainedSchedulerBackend reset.
a922ca4	2017/02/06	[SPARK-17926][SQL][STREAMING] Added json for statuses.
290ac5b	2017/02/06	[SPARK-17811] SparkR cannot parallelize data.frame with NA or NULL in Date columns.
a94a716	2017/02/06	[SPARK-18034] Upgrade to MiMa 0.1.11 to fix flakiness
1db928e	2017/02/06	[SPARKR] fix warnings
bbd260f	2017/02/06	[SPARK-17999][KAFKA][SQL] Add getPreferredLocations for KafkaSourceRDD.
c4816ab	2017/02/06	[SPARK-18003][SPARK CORE] Fix bug of RDD zipWithIndex & zipWithUniqueId index value overflowing.
9c22c9d	2017/02/06	[SPARK-17989][SQL] Check ascendingOrder type in sort_array function rather than throwing ClassCastException.
ae60c75	2017/02/06	[SPARK-18001][DOCUMENT] fix broke link to SparkDataFrame.
f2b58bf	2017/02/06	[SPARK-17711][TEST-HADOOP2.2] Fix hadoop2.2 compilation error.
003b20c	2017/02/06	[SPARK-17731][SQL][STREAMING] [FOLLOWUP] Refactored StreamingQueryListener APIs for branch-2.0.
9ad2ee7	2017/02/06	[SPARK-17841][STREAMING] [KAFKA] drain commitQueue.
efcc529	2017/02/06	[MINOR][DOC] Add more built-in sources in sql-programming-guide.md.
edbe6a6	2017/02/06	[SPARK-17711] Compress rolled executor log.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
28d9c60	2017/02/06	[SPARK-17751][SQL] [BACKPORT-2.0] Remove spark.sql.eagerAnalysis and Output the Plan if Existed in AnalysisException.
b8b951a	2017/02/06	[SQL][STREAMING][TEST] Follow up to remove Option.contains for Scala 2.10 compatibility.
78e5c84	2017/02/06	[SQL][STREAMING][TEST] Fix flaky tests in StreamingQueryListenerSuite.
3fbcb1f	2017/02/06	[SPARK-17731][SQL][STREAMING] Metrics for structured streaming for branch-2.0.
1a14c88	2017/02/06	Fix example of tf_idf with minDocFreq.
1bf46c0	2017/02/06	[SPARK-17892][SQL][2.0] Do Not Optimize Query in CTAS More Than Once #15048.
ea7ccbe	2017/02/06	[MINOR][SQL] Add prettyName for current_database function.
e627ac0	2017/02/06	[SPARK-17819][SQL][BRANCH-2.0] Support default database in connection URIs for Spark Thrift Server.
e97b8cc	2017/02/06	[SPARK-17953][DOCUMENTATION] Fix typo in SparkSession scaladoc.
beeb656	2017/02/06	[SPARK-17863][SQL] should not add column into Distinct.
3d6ab95	2017/02/06	[SPARK-17834][SQL] Fetch the earliest offsets manually in KafkaSource instead of counting on KafkaConsumer.
00239e8	2017/02/06	minor doc fix for Row.scala.
9957c50	2017/02/06	[SPARK-17876] Write StructuredStreaming WAL to a stream instead of materializing all at once.
be58a9b	2017/02/06	[SPARK-16827][BRANCH-2.0] Avoid reporting spill metrics as shuffle metrics.
b064786	2017/02/06	[SPARK-17782][STREAMING] [KAFKA] alternative eliminate race condition of poll twice.
eb73c46	2017/02/06	[SPARK-17790][SPARKR] Support for parallelizing R data.frame larger than 2GB.
8a5a689	2017/02/06	[SPARK-17884][SQL] To resolve Null pointer exception when casting from empty string to interval type.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
4fb6c0c	2017/02/06	[SPARK-17808][PYSPARK] Upgraded version of Pyrolite to 4.13.
dccbe82	2017/02/06	[SPARK-17853][STREAMING] [KAFKA][DOC] make it clear that reusing group.id is bad.
22078b0	2017/02/06	[SPARK-17880][DOC] The url linking to `AccumulatorV2` in the document is incorrect.
904dc7b	2017/02/06	Fix hadoop.version in building-spark.md.
7c94cc5	2017/02/06	[SPARK-17816][CORE] [BRANCH-2.0] Fix ConcurrentModificationException issue in BlockStatusesAccumulator.
50d4eac	2017/02/06	[SPARK-17346][SQL][TESTS] Fix the flaky topic deletion in KafkaSourceStressSuite.
ea25634	2017/02/06	[SPARK-17738][TEST] Fix flaky test in ColumnTypeSuite.
95a7871	2017/02/06	[SPARK-17417][CORE] Fix # of partitions for Reliable RDD checkpointing.
784dd2f	2017/02/06	[SPARK-17832][SQL] TableIdentifier.quotedString creates un-parseable names when name contains a backtick.
dcdca00	2017/02/06	[SPARK-17806] [SQL] fix bug in join key rewritten in HashJoin.
f36c03b	2017/02/06	[SPARK-17782][STREAMING] [BUILD] Add Kafka 0.10 project to build modules.
eb75678	2017/02/06	[SPARK-17346][SQL][TEST-MAVEN] Add Kafka source for Structured Streaming (branch 2.0).
c46948e	2017/02/06	[SPARK-17805][PYSPARK] Fix in sqlContext.read.text when pass in list of paths.
cad3e53	2017/02/06	[SPARK-17612][SQL][BRANCH-2.0] Support `DESCRIBE table PARTITION` SQL syntax.
87e573f	2017/02/06	[SPARK-17792][ML] L-BFGS solver for linear regression does not accept general numeric label column types.
e1cdf30	2017/02/06	[SPARK-17750][SQL] [BACKPORT-2.0] Fix CREATE VIEW with INTERVAL arithmetic.
08a30d9	2017/02/06	[SPARK-17803][TESTS] Upgrade docker-client dependency.
4a48d45	2017/02/06	[SPARK-17780][SQL] Report Throwable to user in StreamExecution.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
67ee7ad	2017/02/06	[SPARK-17798][SQL] Remove redundant Experimental annotations in sql.streaming.
85d0dc1	2017/02/06	[SPARK-17643] Remove comparable requirement from Offset (backport for branch-2.0).
a255661	2017/02/06	[SPARK-17758][SQL] Last returns wrong result in case of empty partition.
07a30cb	2017/02/06	[SPARK-17778][TESTS] Mock SparkContext to reduce memory usage of BlockManagerSuite.
230b501	2017/02/06	[SPARK-17773][BRANCH-2.0] Input/ Output] Add VoidObjectInspector.
8ae27fb	2017/02/06	[SPARK-17549][SQL] Only collect table size stat in driver for cached relation.
3fa5485	2017/02/06	[SPARKR][DOC] minor formatting and output cleanup for R vignettes.
13595fc	2017/02/06	[SPARK-17559][MLLIB] persist edges if their storage level is non in PeriodicGraphCheckpointer.
75d7369	2017/02/06	[SPARK-17112][SQL] "select null" via JDBC triggers IllegalArgumentException in Thriftserver.
159c854	2017/02/06	[SPARK-17753][SQL] Allow a complex expression as the input a value based case statement.
ca37182	2017/02/06	[SPARK-17587][PYTHON][MLLIB] SparseVector __getitem__ should follow __getitem__ contract.
825c9e3	2017/02/06	[SPARK-17736][DOCUMENTATION] [SPARKR] Update R README for rmarkdown,…
258b068	2017/02/06	[MINOR][DOC] Add an up-to-date description for default serialization during shuffling.
92cd75c	2017/02/06	Updated the following PR with minor changes to allow cherry-pick to branch-2.0.
60d2ac2	2017/02/06	[SPARK-17721][MLLIB][ML] Fix for multiplying transposed SparseMatrix with SparseVector.
e6d1fbe	2017/02/06	[SPARK-17672] Spark 2.0 history server web Ui takes too long for a single application.
90df14b	2017/02/06	[SPARK-17712][SQL] Fix invalid pushdown of data-independent filters beneath aggregates.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
7120a46	2017/02/06	[SPARK-16343][SQL] Improve the PushDownPredicate rule to pushdown predicates correctly in non-deterministic condition.
539f476	2017/02/06	[MINOR][DOCS] Fix th doc. of spark-streaming with kinesis.
27de1d4	2017/01/05	Merge pull request #81 from mapr/mapr-25713.
8ea6501	2017/01/05	[MAPR-25713] Spark might try to load MapR Class Loader multiple times and fail.
7e9e5f4	2016/12/26	Merge pull request #80 from mapr/mapr-25638.
965975c	2016/12/26	[SPARK-18528][SQL] Fix a bug to initialise an iterator of aggregation buffer.
96b1fea	2016/12/12	Merge pull request #79 from mapr/mapr-25311.
c5f682b	2016/12/12	[MAPR-25311] Bump Spark dependencies after ECO-1611 release.

Known Issues and Limitations

- Spark 2.0.1 does not support Spark Structured Streaming.
- Full support of MapR Event Store For Apache Kafka is available only on clusters with MapR 5.2 and later.
- Spark is not able to submit jobs to YARN when the cluster is in "classic" mode, even if YARN is installed and configured.
- **MAPR-17271:** On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
- **MAPR-25052:** Spark Thrift Server does not start on clusters secured by MapR-SASL.
- **MAPR-26039:** Spark does not propagate mapr_sec_enabled variable to Driver.
- Spark versions up to and including 2.3.0 have the following security vulnerability:CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Resolved Issues

None.

Spark 1.6.1-1703 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 1.6.1 Release Notes.

Spark Version	1.6.1
Release Date	April 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536.

Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	1.6.1-mapr-1703
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

Note: For some important Spark limitations, See "Known Issues and Limitations" later in this release note.

New in This Release

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive-on-Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive. For details, see the Apache Spark documentation and the MapR Spark documentation.

Fixes

This MapR release includes the following new fixes since the latest MapR Spark 1.6.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	MapR Fix Number and Description
9275605	2017/03/16	Fixed gap handling.
dbf3b16	2017/03/10	Removed overridden methods count, countApprox, isEmpty from kafkaRDD.
3014abc	2017/03/07	[MAPR-26040] Kafka streaming module was refactored (#96).
62def95	2017/03/06	[BUILD][1.6] Fix compilation.
e092b35	2017/02/23	Merge pull request #90 from mapr/spark-1.6.3-porting.
04cbcb0	2017/02/23	[SPARK-17696][SPARK-12330] [CORE] Partial backport of to branch-1.6.
9be6167	2017/02/13	[SPARK-14357][CORE] Properly handle the root cause being a commit denied exception.
566a5a9	2017/02/13	[SPARK-16182][CORE] Utils.scala -- terminateProcess() should call Process.destroyForcibly() if and only if Process.destroy() fails.
31843fc	2017/02/13	[SPARK-17618] Fix invalid comparisons between UnsafeRow and other row formats.
2781e0b	2017/02/13	[SPARK-8428][SPARK-13850] Fix integer overflows in TimSort.

Known Issues and Limitations

- To integrate Spark 1.6.1 with MapR Event Store For Apache Kafka, you must install the latest Kafka 0.9.0.0 package.

- Full support of MapR Event Store For Apache Kafka is available only on clusters with MapR 5.2 and later.
- MAPR-17271: On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
- MAPR-19761: On a secure cluster, MapR software does not support the Spark SQL Thrift JDBC server. When the cluster is secure, the Spark Thrift server will not start.
- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Resolved Issues

None.

Spark 2.0.1-1611 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 2.0.1 Release Notes.

Spark Version	2.0.1
Release Date	December 9, 2016
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	2.0.1-mapr-1611
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive on Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive; for details, see the Apache Spark documentation and the MapR Spark documentation.

Note: For the API changes in this version, see Spark 2.0.1 API.

Fixes

This MapR release includes the following new fixes on top of the Apache Spark 2.0.1 release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
bed0ba9	2016-11-14	[MINOR] Fixed the spark-jars.sh script.
287a9b2	2016-11-09	[MINOR] Removed the unused config template.
b05ddad	2016-11-09	[MAPR-24603] Could not launch the Beeline shell after starting the Spark Thrift Server.
b64908b	2016-11-07	Fixed a syntax error in V09DirectKafkaWordCount example (#75).
b6788d0	2016-11-01	Spark 2.0.1 MapR-Streams Python API (#73).

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
b1dff21	2016-10-31	[MAPR-24415] SPARK_JAVA_OPTS was deprecated (#71).
361effd	2016-10-26	[MAPR-24863] Added spark-defaults template for installer (#69).
a1e0492	2016-10-20	[MAPR-25002] FileNotFoundException during SparkHiveExample (#68).
7357889	2016-10-12	Added the Kafka streaming producer (#66).
d0c05d4	2016-10-07	[SPARK-17707][WEBUI] The Web UI prevented the spark-submit application from finishing.
23f9305	2016-09-08	[SPARK-15487][WEBUI] Spark Master UI to reverse proxy Application and Workers UI.
d99c601	2016-10-05	Fixed Scala style for SparkHiveExample.
fb327ad	2016-10-05	Changed the Kafka 0.9 version to 2.0.1.
cd374ce	2016-10-05	Changed version to 2.0.1-mapr-SNAPSHOT.
73a298a	2016-09-20	[MAPR-24491] The HBase classpath might contain Hive libraries.
f010164	2016-09-14	Minor fix for previous commit.
f3ec15a	2016-09-14	Added script for MAPR-24374.
4292173	2016-09-13	Some minor changes to spark-defaults.conf.
36e847d	2016-09-13	Changed the default HBase version to 1.1.1 in compatibility.version.
d721e09	2016-09-12	Refactored a streaming example.
9aba5a1	2016-09-12	[MAPR-24470] HiveFromSpark test failed in yarn-cluster mode.
70809c9	2016-09-08	Changed the Hive execution version to 1.2.0.
acabece	2016-09-01	Added spark streaming integration with Kafka 0.9 and Mapr-Streams.
641faa3	2016-08-31	Added MapR Repo.
1a2e25e	2016-08-22	[MAPR-23559] Spark PID in /opt/mapr/pid.
690788c	2016-05-25	[MAPR-22940] Failed to connect Spark Beeline (after Spark Thrift Server is started) on Kerberos cluster.
fe2c1ce	2016-05-03	Removed Hive jars from generated classpath for Hive.
85c45d1	2016-05-03	Fixed hardcoded Hive library path.
1124353	2016-04-29	[MAPR-23203] Removed derby jars from the generated Hive classpath.
3bc0cd4	2016-04-13	[MAPR-23068] Spark samples failed in Hue 3.9.
506bce9	2016-03-16	[MAPR-18865] Unable to submit Spark apps from Windows client.
9c2fda0	2015-10-20	Skip maven clean task on the parent module.
b699ada	2015-07-30	New: Issue with running Hive commands in Spark.
49d106a	2015-07-28	Spark warden.services.conf should have a dependency on the CLDB.
d7425bd	2015-05-20	Removed DFS shuffle settings.
5ed864c	2015-04-19	Fixed bugs in the logic to avoid SSH for localhost.
ad12fcf	2015-04-14	Copied every file in the conf directory into the distribution package.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
4b15fb8	2015-04-10	Created spark-defaults.conf for MapR.
e9e4ab1	2015-04-08	Avoided SSH to localhost when stopping secondary instances.
c8405cd	2015-02-25	Added htrace jar to Spark classpath for HBase 0.98.
c502093	2015-02-25	Added the Scala library.
254aecf	2015-02-25	Supported HBase classpath computation in util script.
f6319e4	2015-02-24	Created ext-util.
c850a32	2015-02-24	Added external conf and scripts.
93279f6	2015-02-06	Enabled SPARK_HIVE mode while building.
c851ac8	2015-10-12	Built Spark on MapR.
7e1b86c	2015-11-25	Spark Master failed to start in HA mode.
bcf7665	2015-02-04	Updated the datanucleus jar in Spark for Bug 21228.
fbaf081	2015-08-22	Changed the dependencies for MapR and increased the Hadoop version from 2.2.0 to 2.7.0.

Known Issues and Limitations

Known issues:

- **MAPR-17271:** On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
- **MAPR-25052:**The Spark Thrift Server does not start on clusters secured by MapR-SASL.
- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Limitations:

- Spark 2.0.1 Standalone mode is supported only on clusters in MRv2 (YARN) mode.
- Full support of MapR Event Store For Apache Kafka is available only on clusters with MapR 5.2 and later.

Resolved Issues

None.

Spark 1.6.1-1611 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open-source Spark 1.6.1 Release Notes.

Spark Version	1.6.1
Release Date	December 9, 2016
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	1.6.1-mapr-1611
Package Names	The following packages are associated with this release:

	• mapr-spark-1.6.1.201612010646-1.noarch.rpm • mapr-spark-master-1.6.1.201612010646-1.noarch.rpm • mapr-spark-historyserver-1.6.1.201612010646-1.noarch.rpm • mapr-spark_1.6.1.201612010646_all.deb • mapr-spark-master_1.6.1.201612010646_all.deb • mapr-spark-historyserver_1.6.1.201612010646_all.deb

New in This Release

This version of Spark supports integration with Hive and MapR Event Store For Apache Kafka. However, note the following exceptions:

- Hive on Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive; for details, see the Apache Spark documentation and the MapR Spark documentation.
- If you want to integrate Spark 1.6.1 with MapR Event Store For Apache Kafka, you must install the latest Kafka 0.9.0.0 package.
- Full support of MapR Event Store For Apache Kafka is available only on MapR 5.2 clusters.

Fixes

This MapR release includes the following new fixes since the previous release of MapR Spark 1.6.1. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
50fbc6b	2016-09-23	[SPARK-17210][SPARKR] The sparkr.zip archive is not distributed to executors when running sparkr in RStudio.
bba4dfa	2016-09-29	MAPR-24602: Spark RStudio jobs failed with a java.net.SocketTimeoutException.
e837cec	2016-10-10	[MAPR-24635] The Spark History Server takes a long time to display the stage details.
847bc6f	2016-10-24	[MAPR-24870] Reverted changes that were made in MAPR-23243.
3edeef6	2016-10-28	[MAPR-24415] SPARK_JAVA_OPTS is deprecated.
175c9b5	2016-11-04	[MAPR-13522] [CORE] Executor should kill itself when it is unable to send a heartbeat to the driver more than *n* times.
92e042e	2016-11-04	[MAPR-13522] [CORE] Fixed the exit log place for heartbeat.
26990db	2016-11-11	[MAPR-25223] SparkR did not work.

Known Issues and Limitations

- **MAPR-17271:** On secure clusters, the MapR Control System (MCS) does not display links for Spark Master and Spark History Server.
- **MAPR-19761:** On a secure cluster, MapR does not support the Spark SQL Thrift JDBC server. When the cluster is secure, the Spark Thrift Server will not start.

- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Resolved Issues

None.

Spark 1.6.1-1609 Release Notes

The notes below relate specifically to the MapR Converged Data Platform. You may also be interested in the open source Spark 1.6.1 Release Notes

Spark Version	1.6.1
Release Date	September 30, 2016
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Spark Support Matrix on page 5636.
Source on GitHub	https://github.com/mapr/spark/tree/1.6.1-mapr-1609
GitHub Release Tag	1.6.1-mapr-1609
Package Names	The following packages are associated with this release: • `mapr-spark-1.6.1.201609271200-1.noarch.rpm` • `mapr-spark_1.6.1.201609271200_all.deb` • `mapr-spark-historyserver-1.6.1.201609271200-1.noarch.rpm` • `mapr-spark-historyserver_1.6.1.201609271200_all.deb` • `mapr-spark-master-1.6.1.201609271200-1.noarch.rpm` • `mapr-spark-master_1.6.1.201609271200_all.deb`

Important Notes

- If you want to integrate Spark 1.6.1-1609 with MapR Event Store For Apache Kafka, you must install the Kafka 0.9.0-1607 package.
- Full support of MapR Event Store For Apache Kafka is available only on MapR 5.2 clusters.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive on Spark is not supported.
- Spark-SQL is supported, but it is not fully compatible with Hive; see the Apache Spark documentation and MapR Spark documentation for details.

Fixes

This release by MapR includes the following new fixes since the previous release of MapR Spark 1.6.1. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comments
2b4ae57	2016-09-19	[MAPR-24603] This fix fixes the issue where a beeline shell could not be launched after starting Spark.

last-updated: Apr 11, 2022

GitHub Commit	Date (YYYY-MM-DD)	Comments
b2d53e7	2016-09-20	[MAPR-24491] This fix fixes the issue where the HBase classpath might contain Hive libraries.
b9de0fb	2016-09-26	[MAPR-24678] This fix fixes the issue where PySpark is unable to consume from MapR Event Store For Apache Kafka.

Known Issues

- **MAPR-17271**: On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
- **MAPR-19761:** On a secure cluster, MapR does not support the Spark SQL Thrift JDBC server. When the cluster is secure, the Spark Thrift server will not start.
- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Spark 1.6.1-1608 Release Notes

The notes below relate specifically to the MapR Converged Data Platform. You may also be interested in the open source Spark 1.6.1 Release Notes

Spark Version	1.6.1
Release Date	September 1, 2016
MapR Version Interoperability	See the Interoperability Matrices on page 5519, Ecosystem Support Matrix (Pre-5.2 releases) on page 5625, and Spark Support Matrix on page 5636.
Source on GitHub	https://github.com/mapr/spark/tree/1.6.1-mapr-1608
Package Names	The following packages are associated with this release: • `mapr-spark-master-1.6.1.201608302253-1.noarch.rpm` • `mapr-spark-master_1.6.1.201608302253_all.deb`

Important Notes

If you want to integrate Spark 1.6.1-1608 with MapR Event Store For Apache Kafka, you must install the Kafka 0.9.0-1607 package.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive on Spark is not supported.
- Spark-SQL is supported but it is not fully compatible with Hive; see the Apache Spark documentation and MapR's Spark documentation for details.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comments
3024b71	2016-08-01	MAPR-24129: Backported SPARK-16796 so that passwords are now masked on the Environment page of the Spark History Server UI.
7755848	2016-08-16	[MARP-24264] The following fixes have been backported to resolve the Spark vulnerability to code injection in the `toCommentSafeString` method: • [SPARK-11352][SQL][BRANCH-1.5] Escape */ in the generated comments. • [SPARK-12138][SQL] Escape \u in the generated comments of codegen • [SPARK-15165][SQL] Codegen can break because toCommentSafeString is not actually safe For more information, see Spark Vulnerability with the toCommentSafeString Method.
04c451a	2016-08-18	[MAPR-24292] MapR Event Store For Apache Kafka consumer no longer fails to read the first message when the message contains a single string.
12df59b	2016-08-25	[MAPR-24263] The Spark assembly and example JAR files no longer include a dependency on MapR-FS libraries.

Known Issues

- **MAPR-17271**: On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
- **MAPR-19761:** On a secure cluster, MapR does not support the Spark SQL Thrift JDBC server. When the cluster is secure, the Spark Thrift server will not start.
- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Spark 1.6.1-1607 Release Notes

The notes below relate specifically to the MapR Converged Data Platform. You may also be interested in the open source Spark 1.6.1 Release Notes

Spark Version	1.6.1
Release Date	July 29, 2016
MapR Version Interoperability	See the Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Spark Support Matrix on page 5636.
Source on GitHub	https://github.com/mapr/spark/tree/1.6.1-mapr-1607

Package Names	The following packages are associated with this release: • mapr-spark-1.6.1.201607242143-1.noarch.rpm • mapr-spark_1.6.1.201607242143_all.deb • mapr-spark-historyserver-1.6.1.201607242143-1.noarch.rpm • mapr-spark-historyserver_1.6.1.201607242143_all.deb • mapr-spark-master-1.6.1.201607242143-1.noarch.rpm • mapr-spark-master_1.6.1.201607242143_all.deb

New in This Release

This release of Apache Spark includes the following behavior change that is specific to MapR:

Poll Time for Consuming MapR Event Store For Apache Kafka	When Spark consumes MapR Event Store For Apache Kafka messages, the default poll time is 1000 milliseconds. Previously, the default was 100 milliseconds.

Important Notes

If you want to integrate Spark 1.6.1-1607 with MapR Event Store For Apache Kafka, you must install the Kafka 0.9.0-1607 package.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
c0bb193	2016-06-08	MAPR-23559: Spark now stores PID files in the following directory: `/opt/mapr/pid`
42d163f	2016-06-08	MAPR-22541: Spark now adds the working directory to the CLASSPATH.
4d048420	2016-06-24	MAPR-23612: Spark no longer hangs due to an incorrect offset configuration for MapR Event Store For Apache Kafka.
941e206	2016-06-30	MAPR-23122: Spark Streaming uses `streams.consumer.zerooffset.on.eof` to calculate the offset for MapR Event Store For Apache Kafka.
25621e4	2016-07-04	MAPR-22940: Spark Thrift Server is now able to start on a node where Hive is not running. However, when HiveServer2 uses Kerberos authentication, the Spark Thrift Server must run on the same node as HiveServer2. Otherwise, beeline will not be able to connect to the Spark Thirft Server.

GitHub Commit	Date (YYYY-MM-DD)	Comment
2a3abdb	2016-07-13	MAPR-23854: Spark is now able to retrieve messages from MapR Event Store For Apache Kafka.
6d8d5d6	2016-07-19	MAPR-24011: Backported SPARK-14699 and SPARK-13352 to improve Spark performance.
4df099e	2016-07-19	MAPR-24005: Backported Spark 14699 so that Spark standalone Pi jobs no longer generate "Executor lost" errors.

Known Issues

- **MAPR-17271**: On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
- **MAPR-19761:** On a secure cluster, MapR does not support the Spark SQL Thrift JDBC server. When the cluster is secure, the Spark Thrift server will not start.
- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Spark 1.6.1-1605 Release Notes

The notes below relate specifically to the MapR Converged Data Platform. You may also be interested in the open source Spark 1.6.1 Release Notes

Spark Version	1.6.1
Release Date	June 6, 2016
MapR Version Interoperability	See the Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Spark Support Matrix on page 5636.
Source on GitHub	https://github.com/mapr/spark/tree/1.6.1-mapr-1605
Package Names	The following packages are associated with this release: • mapr-spark-1.6.1.201605311547-1.noarch.rpm • mapr-spark_1.6.1.201605311547_all.deb • mapr-spark-historyserver-1.6.1.201605311547-1.noarch.rpm • mapr-spark-historyserver_1.6.1.201605311547_all.deb • mapr-spark-master-1.6.1.201605311547-1.noarch.rpm • mapr-spark-master_1.6.1.201605311547_all.deb

New in this Release

This release of Apache Spark for MapR includes the following behavior changes:

- On clusters that run Spark on YARN, SASL encryption between all SparkWorker and SparkMaster nodes is enabled by default.
- SSL encryption between all SparkWorker and SparkMaster nodes is disabled by default.

last-updated: Apr 11, 2022

For details on the features available in the open source version of this component, see the Apache Spark documentation.

Important Notes

This release of Apache Spark for MapR includes the following shared library dependencies:

- Mesos version: 0.21.1
- HBase version: 0.98.12-mapr-1506
- Flume version 1.6.0
- Zookeeper version: 3.4.5-mapr-1503
- Hive version: 1.2.0-mapr-1603
- Hive parquet version: 1.6.0
- Kafka version: 0.9.0.0
- Kafka clients version: 0.9.0.0-mapr-1602

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
151ee01	2016-05-17	MAPR-23355: On clusters that run Spark on YARN, authentication between all SparkWorker and SparkMaster nodes is configured to use Spark SASL encryption by default.
e80140d	2016-05-18	MAPR-23243: datanucleus.schema.autoCreateAll is programatically set to true when Hive uses the embedded derby metastore.
c0741cb	2016-05-18	MAPR-23298: Spark SSL encryption is now disabled by default.

Known Issues

- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Spark 1.6.1-1604 Release Notes

The notes below relate specifically to the MapR Converged Data Platform. You may also be interested in the open source Spark 1.6.1 Release Notes

Spark Version	1.6.1
Release Date	May 4, 2016
MapR Version Interoperability	See the Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Spark Support Matrix on page 5636.
Source on GitHub	https://github.com/mapr/spark/tree/1.6.1-mapr-1604

last-updated: Apr 11, 2022

Package Names	The following packages are associated with this release: • mapr-spark-1.6.1.201605031407-1.noarch.rpm • mapr-spark_1.6.1.201605031407_all.deb • mapr-spark-historyserver-1.6.1.201605031407-1.noarch.rpm • mapr-spark-historyserver_1.6.1.201605031407_all.deb • mapr-spark-master-1.6.1.201605031407-1.noarch.rpm • mapr-spark-master_1.6.1.201605031407_all.deb

New in this Release

This is the initial release of Spark Version 1.6.1 for MapR.

For details on the features available in the open source version of this component, see the Apache Spark documentation.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
d0b2386	2016-04-13	Backported SPARK-13599 to remove transitive groovy dependencies from spark-hive and spark-hiveserver.
9ded88e	2016-04-13	MAPR-23068: Spark now uses Py4J version 0.8.2.1.
a37d88b	2016-04-13	MAPR-23093: The Spark KafkaProducerExample no longer fails when it is submitted in yarn-client mode on a secure cluster that uses Kerberos authentication.
6f4ca35	2016-04-28	MAPR-23203: HiveFromSpark example no longer fails with a `datanucleus.schema.autoCreateTables` error when it is submitted in yarn-cluster mode.

Known Issues

- **MAPR-17271**: On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
- **MAPR-19761:** On a secure cluster, MapR does not support the Spark SQL Thrift JDBC server. When the cluster is secure, the Spark Thrift server will not start.
- **MAPR-22940/SPARK-11851:** On clusters that use Kerberos authentication, Spark Thrift Server is unable to connect to beeline.
- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

last-updated: Apr 11, 2022

Spark 1.5.2-1608 Release Notes

The notes below relate specifically to the MapR Converged Data Platform. You may also be interested in the open source Spark 1.5.2 Release Notes.

Spark Version	1.5.2
Release Date	September 1, 2016
MapR Version Interoperability	See the Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Spark Support Matrix on page 5636.
Source on GitHub	https://github.com/mapr/spark/tree/1.5.2-mapr-1608
Package Names	The following packages are associated with this release: • `mapr-spark-master-1.5.2.201608302219-1.noarch.rpm` • `mapr-spark-master_1.5.2.201608302219_all.deb`

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive on Spark is not supported.
- Spark-SQL is supported but it is not fully compatible with Hive; see the Apache Spark documentation and MapR's Spark documentation for details.

Important Notes

This release of Apache Spark for MapR includes the following shared library dependencies:

- Mesos version: 0.21.1
- HBase version: 0.98.12-mapr-1506
- Flume version 1.6.0
- Zookeeper version: 3.4.5-mapr-1503
- Hive version: 1.2.0-mapr-1603
- Hive parquet version: 1.6.0
- Kafka version: 0.9.0.0
- Kafka clients version: 0.9.0.0-mapr-1602

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comments
ac11283	2016-08-02	MAPR-24129: Backported SPARK-16796 so that passwords are now masked on the Environment page of the Spark History Server UI.
0bda0b9	2016-08-18	MARP-24264: The following fixes have been backported to resolve the Spark vulnerability to code injection in the `toCommentSafeString` method:

GitHub Commit	Date (YYYY-MM-DD)	Comments
		• [SPARK-11352][SQL][BRANCH-1.5] Escape */ in the generated comments. • [SPARK-12138][SQL] Escape \u in the generated comments of codegen • [SPARK-15165][SQL] Codegen can break because toCommentSafeString is not actually safe For more information, see Spark Vulnerability with the toCommentSafeString Method.
fc72384	2016-08-22	MAPR-24318: Backported SPARK -1191 and SPARK-11311 so that Spark is now able to create UDF's using the Hive Thrift Service.
681dcf4	2016-08-24	MAPR-24263: The Spark assembly and example JAR files no longer include a dependency on MapR-FS libraries.

Known Issues

- **MAPR-17271:** On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
- **MAPR-19761**: On a secure cluster, MapR does not support the Spark SQL Thrift JDBC server. When the cluster is secure, the Spark Thrift server will not start.
- **MAPR-22940/SPARK-11851**: On clusters that use Kerberos authentication, Spark Thrift Server is unable to connect to beeline.
- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Spark 1.5.2-1605 Release Notes

The notes below relate specifically to the MapR Converged Data Platform. You may also be interested in the open source Spark 1.5.2 Release Notes.

Spark Version	1.5.2
Release Date	June 6, 2016
MapR Version Interoperability	See the Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Spark Support Matrix on page 5636.
Source on GitHub	https://github.com/mapr/spark/tree/1.5.2-mapr-1605
Package Names	The following packages are associated with this release: • mapr-spark-master-1.5.2.201606021619-1.noarch.rpm • mapr-spark-master_1.5.2.201606021619_all.deb • mapr-spark-historyserver-1.5.2.201606021619-1.noarch.rpm • mapr-spark-historyserver_1.5.2.201606021619_all.deb

	• mapr-spark-master-1.5.2.201606021619-1.noarch.rpm • mapr-spark-master_1.5.2.201606021619_all.deb

New in this Release

This release of Apache Spark for MapR includes the following behavior change:

- SSL encryption between all SparkWorker and SparkMaster nodes is disabled by default.

For details on the features available in the open source version of this component, see the Apache Spark documentation.

Important Notes

This release of Apache Spark for MapR includes the following shared library dependencies:

- Mesos version: 0.21.1
- HBase version: 0.98.12-mapr-1506
- Flume version 1.6.0
- Zookeeper version: 3.4.5-mapr-1503
- Hive version: 1.2.0-mapr-1603
- Hive parquet version: 1.6.0
- Kafka version: 0.9.0.0
- Kafka clients version: 0.9.0.0-mapr-1602

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
4578d0e	2016-06-31	MAPR-23298: Spark SSL encryption is now disabled by default.

Known Issues

- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Spark 1.5.2-1603 Release Notes

The notes below relate specifically to the MapR Converged Data Platform. You may also be interested in the open source Spark 1.5.2 Release Notes.

Spark Version	1.5.2
Release Date	April 4, 2016
MapR Version Interoperability	See the Ecosystem Support Matrix (Pre-5.2 releases) on page 5625 and Spark Support Matrix on page 5636.

Source on GitHub	https://github.com/mapr/spark/tree/1.5.2-mapr-1603
Package Names	The following packages are associated with this release: • mapr-spark-1.5.2.201604040801-1.noarch.rpm • mapr-spark_1.5.2.201604040801_all.deb • mapr-spark-historyserver-1.5.2.201604040801-1.noarch.rpm • mapr-spark-historyserver_1.5.2.201604040801_all.deb • mapr-spark-master-1.5.2.201604040801-1.noarch.rpm • mapr-spark-master_1.5.2.201604040801_all.deb

New in This Release

This release of Apache Spark for MapR includes the following new features or behavior changes:

- You can submit Spark jobs from a Windows client node.
- On a secure cluster that runs Spark on YARN, SSL encryption between SparkWorker and SparkMaster nodes is automatically enabled.
- On a secure cluster that runs Spark on YARN, authentication between SparkWorker and SparkMaster nodes is automatically enabled.

For details on the features available in the open source version of this component, see the Apache Spark documentation.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
03b314	2016-03-04	MAPR-21699: When you run Spark on YARN on a secure cluster, encryption and authentication now are enabled by default.
fbfaf1	2016-03-16	MAPR-18865: `spark-env.cmd` script is now included in the `conf` directory so that it can determine the hadoop classpath.
d9a0e3	2016-03-16	[MAPR-21699] When you run Spark on YARN on a secure cluster, SSL encryption is automatically enabled for Akka, broadcast, and MapR filesystem connections.

Known Issues

- **MAPR-17271**: On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.

last-updated: Apr 11, 2022

- **MAPR-19761:** On a secure cluster, MapR does not support the Spark SQL Thrift JDBC server. When the cluster is secure, the Spark Thrift server will not start.
- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Spark 1.5.2-1602 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open source Spark 1.5.2 Release Notes.

Spark Version	1.5.2
Release Date	February 29, 2016
MapR Version Interoperability	See the Spark Support Matrix on page 5636 and the Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.
Source on GitHub	https://github.com/mapr/spark/tree/1.5.2-mapr-1602
Package Names	The following packages are associated with this release: • mapr-spark-1.5.2.201602261506-1.noarch.rpm • mapr-spark_1.5.2.201602261506_all.deb • mapr-spark-historyserver-1.5.2.201602261506-1.noarch.rpm • mapr-spark-historyserver_1.5.2.201602261506_all.deb • mapr-spark-master-1.5.2.201602261506-1.noarch.rpm • mapr-spark-master_1.5.2.201602261506_all.deb

New in This Release

This release of Apache Spark for MapR includes the following feature:

- Support for MapR Event Store For Apache Kafka through Kafka 0.9.0 API. This is a beta feature.

For details on the features available in the open source version of this component, see the Apache Spark documentation.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
784ad2e	2016-01-13	MAPR-22122: Spark now supports the Kafka 0.9.0 Streaming Consumer API.

Known Issues

- **MAPR-17271**: On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.

- **MAPR-19761:** On a secure cluster, MapR does not support the Spark SQL Thrift JDBC server. When the cluster is secure, the Spark Thrift server will not start.
- **MAPR-22522:** Spark jobs fail when Hive uses Kerberos authentication on a secure cluster that also uses Kerberos authentication.

  Workaround: Add "-Dhadoop.login=hybrid" to SPARK_SUBMIT_OPTS in spark-env.sh ($SPARK_HOME/conf/spark-env.sh). For example:

```
 if [ "$MAPR_SECURITY_STATUS" = "true" ]; then
SPARK_SUBMIT_OPTS="$SPARK_SUBMIT_OPTS -Dmapr_sec_enabled=true -Dhadoop.log
in=hybrid"
fi
```

- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Spark 1.5.2-1512 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open source Spark 1.5.2 Release Notes.

Spark Version	1.5.2
Release Date	December 21, 2015
MapR Version Interoperability	See Spark Support Matrix on page 5636.
Source on GitHub	https://github.com/mapr/spark
Package Names	The following packages are associated with this release: • `mapr-spark-1.5.2.201512161339-1.noarch.rpm` • `mapr-spark_1.5.2.201512161339_all.deb` • `mapr-spark-historyserver-1.5.2.201512161339-1.noarch.rpm` • `mapr-spark-historyserver_1.5.2.201512161339_all.deb` • `mapr-spark-master-1.5.2.201512161339-1.noarch.rpm` • `mapr-spark-master_1.5.2.201512161339_all.deb`

New in This Release

This release of Apache Spark for MapR includes the following features:

- Support for SparkR (R on Spark)

For details on the features available in the open source version of this component, see the Apache Spark documentation.

Hive Support

This version of Spark supports integration with Hive. However, note the following exceptions:

- Hive on Spark is not supported.

- Spark-SQL is supported but it is not fully compatible with Hive; see the Apache Spark documentation and the MapR Spark documentation for details.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
a7dad34	2015-11-25	MAPR-21570: The Spark Master no longer fails to start when it is configured to be highly available.
cdd328a	2015-11-19	MAPR-21525: The HBase version is now set to 0.98.12 in the `/opt/mapr/spark/spark-1.5.2/mapr-util/compatibility.version` file.
0d4c58e	2015-11-02	MAPR-21243: With Spark on YARN, `spark.sql.hive.metastore.sharedPrefixes` is now set automatically based on the mode that is used to submit the job.

Known Issues

- **MAPR-17271**: On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
- **MAPR-19761:** On a secure cluster, MapR does not support the Spark SQL Thrift JDBC server. When the cluster is secure, the Spark Thrift server will not start.
- **MAPR-20263:** On a secure cluster, MapR does not support submitting jobs that interacts with Hive Metastore on yarn-cluster mode. When the cluster is secure, jobs will not complete successfully.
- Spark versions up to and including 2.3.0 have the following security vulnerability: CVE-2018-1334 Apache Spark local privilege escalation vulnerability

Spark and Spark on YARN 1.4.1-1508 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open source Spark 1.4.1 Release Notes.

Spark Version	1.4.1
Release Date	Sept 9, 2015
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	1.4.1-mapr-1508
MapR Version Interoperability	See Spark Support Matrix on page 5636.

New in This Release

This is the initial MapR release of Spark 1.4.1.

Hive Support

- This version of Spark supports Spark-SQL with Hive 0.13. However, Hive on Spark is not supported.
- Spark-SQL is not fully compatible with Hive; see the Apache Spark documentation for details.

Known Issues

MAPR-17271:	On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.
MAPR-19761:	On a secure cluster, MapR does not support the Spark SQL Thrift JDBC server. When the cluster is secure, the Spark Thrift server will not start.

Spark and Spark on YARN 1.3.1-1505-r1 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the open source Spark 1.3.1 Release Notes.

Version	Spark 1.3.1-1505
Release Date	June 2, 2015
Source on GitHub	https://github.com/mapr/spark
GitHub Release Tag	1.3.1-mapr-1505-r1
MapR Version Compatibility	See Ecosystem Support Matrix (Pre-5.2 releases) on page 5625.

New in This Release

This is the initial MapR release of Spark 1.3.1.

For instructions on installing or upgrading to Spark 1.3.1, see the MapR's Spark documentation.

Hive Support

This version of Spark supports Spark-SQL with Hive 0.13. Other versions of Hive, including Hive 1.0, are not supported with Spark-SQL.

Note: Spark-SQL is not fully compatible with Hive; see the Apache Spark documentation for details.

Event Logging in Spark 1.3.1

In both Spark 1.2.1 and Spark 1.3.1, event logging is enabled by default. However, Spark 1.3.1 also checks that the event directory is present on MapR filesystem (where the logs are written):

`maprfs:///apps/spark`

Create this directory whether or not the History Server is enabled. Alternatively, disable event logging by setting `spark.eventLog.enabled` to `false`.

If you are using MapR Version 4.0.x, applications that are run by a non-mapr user may not be visible in the History Server UI. To work around this problem, manually update the file ownership under `/apps/spark` to `mapr:mapr` for those applications:

```
hadoop fs -chown -R mapr:mapr /apps/spark/app-XYZ
```

This workaround is not required if you are using MapR Version 4.1.

Known Issues

MAPR-17271:	On secure clusters, the MapR Control System (MCS) does not display links for Spark-Master and Spark-HistoryServer.

last-updated: Apr 11, 2022

MAPR-15970:	If RM HA is set up on the cluster and the AM for the YARN job runs on a node other than the RM node, the AM link in the RM UI returns an error and does not bring up the AM UI. See the MapR's Spark documentation for a workaround.

Fixes

This release from MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
53ad194	2015-05-27	**MAPR-17554:** Spark services can now be stopped during an uninstall.
2c2b216	2015-05-20	**MAPR-18750:** DFS shuffle-related properties are no longer placed in the spark-defaults.conf file by default.
2c2b216	2015-05-20	**MAPR-18360:** When Spark is uninstalled, only secnodary instances running on localhost are stopped.
ed8b996	2015-05-27	**MAPR-18793:** Spark-SQL now works with the Hive Metastore on a secure MapR cluster.
feb4cce		
7c71cf2	2015-05-27	**MAPR-18794:** Spark users can now access the YARN logs via https when MapR security is enabled.
979a73f	2015-06-01	**MAPR-18912:** Hadoop 2.5 did not contain a curator jar required by Spark Master HA.

Sqoop Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The release notes for the Sqoop component (included in the MapR Converged Data Platform) contain notes specific to MapR only. More details are available on the Apache Sqoop Project page.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Sqoop 1.4.7 - 2110 (EEP 8.0.0) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Data Platform Distribution for Apache Sqoop. You may also be interested in the Apache Sqoop changelog and the Apache Sqoop home page.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes.

Version	1.4.7

Release Date	October 2021
HPE Version Interoperability	See EEP Components and OS Support
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.7.100-eep-800
Maven Artifacts	http://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP (MEP) and OS to view the list of package names.

New in This Release

Sqoop 1.4.7 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
e226e3a	2021-05-27	SQOOP-105: Fix FileSystemCounters counter variable in the ImportJobBase class

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

.

Sqoop 1.4.7 - 2104 (EEP 7.1.0) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Sqoop. You may also be interested in the Apache Sqoop 1.4.7 changelog or the Apache Sqoop project homepage.

Version	1.4.7
Release Date	April 2021
HPE Version Interoperability	See MEP Components and OS Support
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.7.0-mapr-710
Maven Artifacts	http://repository.mapr.com/maven/
Package Names	SeePackage Names for MapR Ecosystem Packs (MEPs)

New in This Release

Sqoop 1.4.7 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- Bug fixes and updates but no significant new features.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
c1c73b3c	2021-03-31	SQOOP-103: Change driver class to 'com.mysql.cj.jdbc.Driver'
04b6d16b	2021-03-30	SQOOP-102: Sqoop shuffles columns names for export to mysql
b39ded9b	2021-03-24	SQOOP-88: Moving Sqoop to 4-digit version
bd41899b	2021-03-24	SQOOP-101: Updated HBase dependency to 1.4.13.0-mapr-SNAPSHOT
4b316fc6	2021-03-22	SQOOP-98 - cleaned avro dependencies
8a4d9d97	2021-03-22	SQOOP-100: Updated Hadoop dependencies for MEP 7.1.0 release
6f995942	2021-03-17	SQOOP-99: Updated Sqoop dependencies to use jackson v2.X
25fdf52e	2021-03-09	SQOOP-97: Updated Parquet to 1.11.0
85d20320	2021-03-09	SQOOP-95: Fix Hive import as parquet file job
6f7fa989	2021-03-09	Backported SQOOP-3329: Remove Kite dependency from the Sqoop project
97dc41a9	2021-03-09	Backported SQOOP-3338: Document Parquet support
7aa36ed6	2021-03-09	Backported SQOOP-3335: Add Hive support to the new Parquet writing implementation (Szabolcs Vasas via Boglarka Egyed)
d637ad6c	2021-03-09	Backported SQOOP-3328: Implement an alternative solution for Parquet reading and writing(Szabolcs Vasas via Boglarka Egyed)
8f210cb1	2021-03-09	Backported SQOOP-3319: Extract code using Kite into separate classes (Szabolcs Vasas via Boglarka Egyed)
324caa98	2021-03-09	Backported SQOOP-3318: Remove Kite dependency from test cases (Szabolcs Vasas via Boglarka Egyed)

last-updated: Apr 11, 2022

6e3d1f89	2021-03-09	Backported SQOOP-3273: Removing com.cloudera.sqoop packages (Szabolcs Vasas via Anna Szonyi)
3981197c	2021-03-09	Backported SQOOP-90: Update Avro to 1.9.2 version

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- For Avro format, you must use `-Dmapreduce.job.user.classpath.first parameter=true` in the command when you import into the MapR File System.

.

Sqoop 1.4.7 - 2101 (EEP 7.0.1) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Sqoop. You may also be interested in the Apache Sqoop 1.4.7 changelog or the Apache Sqoop project homepage.

Version	1.4.7
Release Date	January 2021
HPE Version Interoperability	MEP Components and OS Support
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.7-mapr-701
Maven Artifacts	http://repository.mapr.com/maven/
Package Names	SeePackage Names for MapR Ecosystem Packs (MEPs)

New in This Release

Sqoop 1.4.7 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
5ec4037	2020-10-12	MAPRSQOOP-87: Reduce excessive mutations
271b5d7	2020-12-31	MAPRSQOOP-89: Updated common-compress to 1.20

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- For Avro format, you must use `-Dmapreduce.job.user.classpath.first parameter=true` in the command when you import into the filesystem.

Resolved Issues

- None

.

Sqoop 1.4.7 - 2009 (EEP 7.0.0) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.4.7 changelog or the Apache Sqoop project home page.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.4.7
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.7-mapr-mep-7.x-2009
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- Stop bundling the TDCH jars.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
79c4038	2020-03-26	MAPR-SQOOP-62 Parse hive database name from hive table name for compatibility
76e512e 9777a43	2020-04-01 2020-04-24	MAPR-SQOOP-61 Reduce excessive write permissions for sqoop files
deebcf3 7a4fd1b	2020-05-06 2020-05-08	MAPR-SQOOP-57 Remove teradata libraries from distribution
997459a	2020-05-11	MAPR-SQOOP-75 Update Maprfs and HBase, exclude unused Fasterxml Jackson

Commit	Date (YYYY-MM-DD)	Comment
af87490	2020-07-13	MAPR-SQOOP-82 Use static tools-list instead of running Sqoop Java Task
f073e25	2020-08-04	MAPR-SQOOP-83 Backport SQOOP-3267 to Sqoop 1.4.7
51a50fc	2020-08-18	MAPR-SQOOP-85 Exclude slf4j from distribution

Known Issues and Limitations

- Use `-Dmapreduce.job.user.classpath.first=true` in your command for import to MapR filesystem in Avro format.

Sqoop 1.4.7 - 2201 (EEP 6.3.6) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Technologies Distribution for Apache Sqoop. You may also be interested in the Apache Sqoop changelog and the Apache Sqoop home page.

These release notes contain only MapR Technologies specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes.

Version	1.4.7
Release Date	January 2022
HPE Version Interoperability	See EEP Components and OS Support
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.7-mapr-636
Maven Artifacts	http://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP (MEP) and OS to view the list of package names.

New in This Release

Sqoop 1.4.7 - 2201 introduces the following enhancements or MapR Technologies platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Date (YYYY-MM-DD)	HPE Fix Number and Description
6b4f6367	2022-01-25	SQOOP-122: Updated log4j v1 to the 1.3.1-mapr
c5d4b45	2022-01-20	SQOOP-121: Updated Log4j due CVE-2019-17571
daceb9eb	2022-01-04	SQOOP-120: Updated commons-io to 2.7 version

ebba765d	2022-01-04	SQOOP-111: Excluded unused dependencies with vulnerabilities
959b29ff	2022-01-04	SQOOP-114: Updated postgresql to 42.2.13 version
e442c2d9	2022-01-04	SQOOP-113: Updated accumulo-master to 2.0.1 version
72f9ff91	2022-01-04	SQOOP-112: Updated commons-compress to 1.21 version
a08ce310	2021-12-21	SQOOP-102: Sqoop shuffles columns names for export to mysql

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- For Avro format, you must use `-Dmapreduce.job.user.classpath.first parameter=true` in the command when you import data into the filesystem.

Resolved Issues

- None.

.

Sqoop 1.4.7 - 2110 (EEP 6.3.5) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Data Platform Distribution for Apache Sqoop. You may also be interested in the Apache Sqoop changelog and the Apache Sqoop home page.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes.

Version	1.4.7
Release Date	October 2021
HPE Version Interoperability	See EEP Components and OS Support
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.7-mapr-635
Maven Artifacts	http://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP (MEP) and OS to view the list of package names.

New in This Release

Sqoop 1.4.7 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
a7616d2	2021-07-21	SQOOP-105: Fix FileSystemCounters counter variable in the ImportJobBase class
267940a	2021-06-03	SQOOP-107: Updated jackson-mapper-asl to 1.9.13-atlassian-5

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- For Avro format, you must use `-Dmapreduce.job.user.classpath.first parameter=true` in the command when you import data into the filesystem.

Resolved Issues

- None.

.

Sqoop 1.4.7 - 2101 (EEP 6.3.2) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Sqoop. You may also be interested in the Apache Sqoop 1.4.7 changelog or the Apache Sqoop project homepage.

Version	1.4.7
Release Date	January 2021
HPE Version Interoperability	MEP Components and OS Support
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.7-mapr-632
Maven Artifacts	http://repository.mapr.com/maven/
Package Names	SeePackage Names for MapR Ecosystem Packs (MEPs)

New in This Release

Sqoop 1.4.7 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

Commit	Date (YYYY-MM-DD)	Comment
5ec4037	2020-10-12	MAPRSQOOP-87: Reduce excessive mutations
271b5d7	2020-12-31	MAPRSQOOP-89: Updated common-compress to 1.20

For complete details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- For Avro format, you must use `-Dmapreduce.job.user.classpath.first parameter=true` in the command when you import into the filesystem.

Resolved Issues

- None.

Sqoop 1.4.7 - 2009 (EEP 6.3.1) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.4.7 changelog or the Apache Sqoop project home page.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.4.7
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.7-mapr-2009
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
79c4038	2020-03-26	MAPR-SQOOP-62 Parse hive database name from hive table name for compatibility
76e512e 9777a43	2020-04-01 2020-04-24	MAPR-SQOOP-61 Reduce excessive write permissions for sqoop files
997459a	2020-05-11	MAPR-SQOOP-75 Update Maprfs and HBase, exclude unused Fasterxml Jackson
af87490	2020-07-13	MAPR-SQOOP-82 Use static tools-list instead of running Sqoop Java Task
f073e25	2020-08-04	MAPR-SQOOP-83 Backport SQOOP-3267 to Sqoop 1.4.7

Commit	Date (YYYY-MM-DD)	Comment
51a50fc	2020-08-18	MAPR-SQOOP-85 Exclude slf4j from distribution

Known Issues and Limitations

- Use `-Dmapreduce.job.user.classpath.first=true` in your command for import to MapR file system in Avro format.

Sqoop 1.4.7-1904 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.4.7 changelog or the Apache Sqoop project home page.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.4.7
Release Date	April 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.7-mapr-1904
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names. The following packages are associated with this release: • mapr-sqoop-1.4.7.<>.noarch.rpm • mapr-sqoop_1.4.7.<>all.deb

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
2af0268	2019-01-31	SQOOP-52: Sqoop incremental import into parquet file fails with "Directory not empty" error

Known Issues and Limitations

- Use `-Dmapreduce.job.user.classpath.first=true` in your command for import to MapR filesystem in Avro format.

Resolved Issues

- None.

Sqoop 1.4.7-1808 (EEP 6.0.0) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.4.7 changelog or the Apache Sqoop project home page.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.4.7
Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/sqoop/tree/1.4.7-mapr-1808
GitHub Release Tag	1.4.7-mapr-1807
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names. The following packages are associated with this release: • mapr-sqoop-1.4.7.<>.noarch.rpm • mapr-sqoop_1.4.7.<>all.deb

New in This Release

- Backported Apache Sqoop 1.4.7.
- TDCH updates:
  - TDCH version is updated to v1.5.4.
  - Added time datatype support.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
a74b6bd	2018-04-17	MAPR-SQOOP-40: Updated TDCH version at Sqoop 1.
5087bab	2018-06-26	MAPR-SQOOP-49: Sqoop job create failed with NPE.

Known Issues and Limitations

Use `-Dmapreduce.job.user.classpath.first=true` in your command for import to MapR filesystem in Avro format.

Resolved Issues

None

Sqoop 1.4.6-1904 (EEP 5.0.3 and EEP 4.1.4) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.4.6 changelog or the Apache Sqoop project home page.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.4.6
Release Date	April 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.6-mapr-1904
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
94a1a1d4	2019-05-20	SQOOP-52: Sqoop incremental import into parquet file fails with "Directory not empty" error
7a959be	2019-05-15	SQOOP-56: SQOOP HCAT import is not working

Known Issues and Limitations

- None.

Resolved Issues

- None.

Sqoop 1.4.6-1904 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.4.6 changelog or the Apache Sqoop project home page.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.4.6
Release Date	April 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.6-mapr-1904
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names. The following packages are associated with this release: • mapr-sqoop-1.4.6.<>.noarch.rpm • mapr-sqoop_1.4.6.<>all.deb

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
94a1a1d4	2019-05-20	SQOOP-52: Sqoop incremental import into parquet file fails with "Directory not empty" error
7a959be	2019-05-15	SQOOP-56: SQOOP HCAT import is not working

Known Issues and Limitations

- None.

Resolved Issues

- None.

Sqoop 1.4.6-1808 (EEP 3.0.4, EEP 4.1.2, and EEP 5.0.1) Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.4.6 changelog or the Apache Sqoop project home page.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	1.4.6

Release Date	September 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/sqoop/tree/1.4.6-mapr-1808
GitHub Release Tag	1.4.6-mapr-1808
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names. The following packages are associated with this release: • mapr-sqoop- 1.4.6.<>.noarch.rpm • mapr-sqoop_1.4.6.<>all.deb

New in This Release

- TDCH updates:
  - Added support for time datatype.
  - TDCH version is updated to v1.5.4.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
0ae8ff0	2018-06-27	MAPR-31814: Fixed dataset URI for parquet file

Known Issues and Limitations

None

Resolved Issues

None

Sqoop 1.4.6-1803 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.4.6 changelog or the Apache Sqoop project home page.

Version	1.4.6
Release Date	March 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.6-mapr-1803
Maven Artifacts	https://repository.mapr.com/maven/

Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names. The following packages are associated with this release: • mapr-sqoop- 1.4.6.<>.noarch.rpm • mapr-sqoop_1.4.6.<>all.deb

New in This Release

- None

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
789a1ac	2018-01-29	MAPR-30627: Unset hive.metastore.sasl.enabled property for non-secure cluster

Known Issues and Limitations

None

Resolved Issues

None

Sqoop 1.4.6-1710 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.4.6 changelog or the Apache Sqoop project home page.

Version	1.4.6
Release Date	November 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.6-mapr-1710
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- The release includes backports of the following issues from the Apache release:
  - SQOOP-2400: hive.metastore.sasl.enabled should be set to true for Oozie integration
  - SQOOP-2103: Not able define Decimal(n,p) data type in map-column-hive option

- SQOOP-2406:Add support for secure mode when importing Parquet files into Hive
- SQOOP-2597: Missing method AvroSchemaGenerator.generate()
- SQOOP-2372: Import all tables as parquet will throw NPE
- SQOOP-3071: Fix OracleManager to apply localTimeZone correctly in case of Date objects too
- SQOOP-2723: Oracle connector not working with lowercase columns
- SQOOP-2783: Query import with parquet fails on incompatible schema
- SQOOP-2737: Cannot import table from Oracle with column with spaces in name
- SQOOP-2909: Oracle related ImportTest fails after SQOOP-2737
- SQOOP-3066: Introduce an option + env variable to enable/disable SQOOP-2737 feature
- SQOOP-2863: Properly escape column names for generated INSERT statements
- SQOOP-2911: Fix failing HCatalogExportTest caused by SQOOP-2863
- SQOOP-2971: OraOop does not close connections properly
- SQOOP-3033: Sqoop option --skip-dist-cache is not saved as a parameter when saving Sqoop Job
- SQOOP-3152: --map-column-hive to support DECIMAL(xx,xx)
- SQOOP-3157: Improve regex introduced in [SQOOP-3152] thus not causing column mapping and AVRO issues
- SQOOP-3123: Import from oracle using oraoop with map-column-java to avro fails if special characters encounter in table name or column name
- SQOOP-3159: Sqoop (export + --table) with Oracle table_name having '$' fails with error (ORA-00942 or java.lang.NoClassDefFoundError)
- SQOOP-3158: Columns added to Mysql after initial sqoop import, export back to table with same schema fails

- Support for a new `--ignore-alias` option, which imports data using column names from the source table rather than the default column alias names.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
f443f5f	2017-07-10	MAPR-28281: Fixes codegen to use column names instead of column labels
8dd25f2	2017-08-04	MAPR-28662: Fixes loss in precision due to Sqoop treating Number data type in Oracle as Double in Hive
c26a90d	2017-08-10	MAPR-28773: Fixes error in `import-all-tables` when using `--as-parquetfile` option

last-updated: Apr 11, 2022

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
8e83fd1	2017-08-10	MAPR-28783: Fixes Hive connection problem when using `import-all-tables` tool

Known Issues and Limitations

None

Resolved Issues

None

Sqoop 1.4.6-1707 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.4.6 changelog or the Apache Sqoop project home page.

Version	1.4.6
Release Date	August 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.6-mapr-1707
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

- New import and export options in the MapR Connector for Teradata. See MapR Connector Import and Export Options on page 4140.
- Support for TDCH 1.5.2.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
c687f59	2017-05-23	MAPR-SQOOP-7: Adds support for Teradata database to sqoop create-hive-table tools
a0b709e	2017-05-24	MAPR-SQOOP-9: Updates TDCH version to 1.5.2
6918d71	2017-06-02	MAPR-27525: Adds missing tez jars in sqoop classpath when running sqoop-Hive on tez import job

Known Issues and Limitations

None.

Resolved Issues

None.

Sqoop 1.4.6-1703 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.4.6 changelog or the Apache Sqoop project home page.

Version	1.4.6
Release Date	April 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.6-mapr-1703
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
cda64ea	2016-12-13	MAPR-25491: Sqoop import fails after TCP connection reset if split by datetime column.

Known Issues and Limitations

None.

Resolved Issues

None.

Sqoop 1.4.6-1611 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.4.6 changelog or the Apache Sqoop project home page.

Version	1.4.6

last-updated: Apr 11, 2022

Release Date	December 9, 2016
MapR Version Interoperability	See EEP Components and OS Support on page 5536 and Ecosystem Support Matrix (Pre-5.2 releases) on page 5625
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.4.6-mapr-1611
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

This version of Sqoop includes the MapR Connector for Teradata (powered by the Teradata Connector for Hadoop). The MapR Connector for Teradata is supported for use only on MapR 5.2 EEP 2.0 or later.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
000e138	2016-12-05	Added Sqoop Teradata Connector for Hadoop (TDCH) dependencies.

Known Issues and Limitations

None.

Resolved Issues

None.

Sqoop1.4.6-1609 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Sqoop component included in the MapR Distribution for Apache Hadoop.

Version	1.4.6
Release Date	September 30, 2016
MapR Version Interoperability	See Interoperability Matrices on page 5519
Source on GitHub	https://github.com/mapr/sqoop/tree/1.4.6-mapr-1609
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • `mapr-sqoop-1.4.6.201609271407-1.noarch.rpm` • `mapr-sqoop_1.4.6.201609271407_all.deb`

Fixes

This release from MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
55a3bdd	2016-08-15	SQOOP-2783: (MAPR-24254) The issue causing query import into parquet to fail has been fixed by changing the default avro schema name to `AutoGeneratedSchema`.
208fd2c	2016-08-15	SQOOP-2582: (MAPR-24254) The issue causing query import into parquet to fail has been fixed.

Sqoop1.4.6-1607 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Sqoop component included in the MapR Distribution for Apache Hadoop.

Version	1.4.6
Release Date	July 29, 2016
MapR Version Interoperability	See Interoperability Matrices on page 5519
Source on GitHub	https://github.com/mapr/sqoop/tree/1.4.6-mapr-1607
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • `mapr-sqoop-1.4.6.201607271059-1.noarch.rpm` • `mapr-sqoop_1.4.6.201607271059_all.deb`

Fixes

This release from MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
28786c5	2016-06-24	MAPR-23789: Sqoop will no longer lose parts of data during import from MySQL, if data type of primary key in MySQL table is double.

Sqoop 1.4.6-1601 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Below are release notes for the Sqoop component included in the MapR Distribution for Apache Hadoop.

Version	1.4.6
Release Date	February 1, 2016
MapR Version Interoperability	See Interoperability Matrices on page 5519
Source on GitHub	https://github.com/mapr/sqoop/tree/1.4.6-mapr-1601
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release:

	• mapr-sqoop-1.4.6.201601281607-1.noarch.rpm • mapr-sqoop_1.4.6.201601281609_all.deb

For details on the features available in the open source version of this component, see the Apache Sqoop 1.4.6 changelog or the Apache Sqoop homepage.

Fixes

This release from MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
a7a609b	2015-01-20	MAPR-22204: Sqoop 1.4.6 no longer fails to import data to HBase 1.1.1.
f410696	2015-11-09	MAPR-21284: Sqoop 1.4.6 no longer throws an NullPointerException while importing from mysql using the --query option.
17957b4	2015-10-23	MAPR-20263: On a secure cluster, Sqoop 1.4.6 no longer fails to import data with the parquet format into Hive.

Sqoop 1.4.6-1509 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Version	1.4.6
Release Date	Oct 5, 2015
Source on GitHub	https://github.com/mapr/sqoop/tree/1.4.6-mapr-1509
MapR Version Compatibility	See Interoperability Matrices on page 5519.
Maven Artifacts	Maven Repository
Package Names	The following packages are associated with this release: • mapr-sqoop-1.4.6.201510011045-1.noarch.rpm • mapr-sqoop_1.4.6.201510011045_all.deb

New in this Release

This release of Sqoop includes the following new behavior changes that are specific to the MapR:

- Kite now includes support for MapR- FS.

For details on the features available in the open source version of this component, see the Apache Sqoop 1.4.6 changelog or the Apache Sqoop homepage.

Fixes

This release from MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
1a05f49	2015-09-04	Sqoop now uses kite version 1.1.0.

Commit	Date (YYYY-MM-DD)	Comment
91405f0	2015-08-05	MAPR-19680: Sqoop no longer throws the following exception when you import a MySQL table with parquet format to the MapR-FS: `org.kitesdk.data.DatasetNotFoundException: Unknown dataset URI pattern`

Known Issues

MAPR-20263:	Sqoop 1.4.6 fails to import data in parquet format to Hive on a secure MapR cluster.

Sqoop 1.4.6-1506 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.4.6 changelog or the Apache Sqoop homepage.

Version	1.4.6
Release Date	July 10, 2015
Source on GitHub	https://github.com/mapr/sqoop.git
MapR Version Compatibility	See Interoperability Matrices on page 5519.
GitHub Release Tag	mapr-1.4.6-1506
Maven Artifacts	https://repository.mapr.com/maven/

New in This Release

This is the initial release of Sqoop 1.4.6 for MapR. This release adds support for the following connectors:

- Netezza Connector
- JDBD-based Connector for Cubrid

Sqoop2 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The release notes for the Sqoop2 component included in the MapR Converged Data Platform contain notes specific to MapR only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Sqoop2 1.99.7-1803 Release Notes

last-updated: Apr 11, 2022

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.99.7 changelog or the Apache Sqoop project home page.

Version	1.99.7
Release Date	March 2018
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.99.7-mapr-1803
Maven Artifacts	See the Maven Repository and the Maven Artifacts for MapR.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
0bae356	2017-12- 03	MAPR-SQOOP- 33: Export mapruserticket in restart script on a secure cluster

Known Issues and Limitations

- Integration between Sqoop 1.99.7 and Oracle database is not supported.
- The Kafka connector is not supported.

Resolved Issues

None

Sqoop2 1.99.7-1710 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.99.7 changelog or the Apache Sqoop project home page.

Version	1.99.7
Release Date	November 2017
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.99.7-mapr-1710

Maven Artifacts	See the Maven Repository and the Maven Artifacts for MapR.
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- Simplified Security - Starting in the MapR 6.0 and EEP 4.0 releases, you can use the "Enable Security" check box in the installer to enable security for the core platform and the installed ecosystem components. Alternatively, you can use the configure.sh -secure command to enable security for the core and the ecosystem components. See Configuring Sqoop2 on page 3680 for details on how this impacts Sqoop2.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
86f4fc0	2017-04-14	MAPR-SQOOP-11: Fixes Sqoop 2 client so it does not return a "Permission denied" warning if not running as admin user
654b594	2017-07-04	MAPR-SQOOP-10: Adds configuration changes to support default security

Known Issues and Limitations

- Integration between Sqoop 1.99.7 and Oracle database is not supported.
- The Kafka connector is not supported.

Resolved Issues

None

Sqoop2 1.99.7-1611 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Sqoop 1.99.7 changelog or the Apache Sqoop project home page.

Version	1.99.7
Release Date	December 9, 2016
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.99.7-mapr-1611
Maven Artifacts	See the Maven Repository and the Maven Artifacts for MapR.
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

This release of Sqoop2 for MapR includes the following new features or behavior changes:

SSL Encryption	You can configure SSL to enable encrypted communications between the Sqoop2 server and its clients.
Repository Encryption	You can configure the Sqoop2 repository to encrypt password data.
Connectors	The following connectors are supported: `ftp-connector`, `kite-connector`, and `sftp-connector`. The `hdfs-connector` supports the parquet data format.
Location of the sqoop.properties file	The sqoop.properties file is now in the following directory: `/opt/mapr/sqoop/sqoop-<version>/conf/`. In Sqoop 1.99.6, the `sqoop.properties` file was in the following directory: `/opt/mapr/sqoop/sqoop-<version>/server/conf/`.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
6a23544	2016-09-29	MAPR-24576: Kite Connector now supports MapR-FS.
fc2e624	2016-10-31	MAPR-25097: Jobs no longer fail in classic mode.

Known Issues and Limitations

- Integration between Sqoop 1.99.7 and Oracle database is not supported.
- The Kafka connector is not supported.
- The Kite connector cannot be used to connect to Hive on secure clusters.

Resolved Issues

None.

Sqoop2 1.99.6-1607 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Converged Data Platform. You may also be interested in the the Apache Sqoop 1.99.6 changelog or the Apache Sqoop project homepage.

Version	1.99.6
Release Date	July 29, 2016
Source on GitHub	https://github.com/mapr/sqoop/tree/1.99.6-mapr-1607
MapR Version Compatibility	See Interoperability Matrices on page 5519.
Maven Artifacts	https://repository.mapr.com/maven/

last-updated: Apr 11, 2022

Package Names	The following packages are associated with this release: • mapr-sqoop2-server-2.0.0.201607271151-1.noarch.rpm • mapr-sqoop2-server_2.0.0.201607271151_all.deb • mapr-sqoop2-client-2.0.0.201607271151-1.noarch.rpm • mapr-sqoop2-client_2.0.0.201607271151_all.deb

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYYY-MM-DD)	Comment
9e8e0d6	2016-07-05	MAPR-23887: The link to Sqoop2 in the MCS no longer fails with the following error: `Service: sqoop2 does not have reachable URL`.

Sqoop2 1.99.6-1507 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the the Apache Sqoop 1.99.6 changelog or the Apache Sqoop project homepage.

Version	1.99.6
Release Date	August 5, 2015
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.99.6-mapr-1507
MapR Version Compatibility	See Interoperability Matrices on page 5519.
Maven Artifacts	https://repository.mapr.com/maven/

New in this Release

This release adds support for MapR-SASL.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
4fe3883	2015-05-20	Sqoop2 automatically detects the hadoop version and sets the hadoop configuration directory in the sqoop.properties file.
7c25f5c	2015-07-23	MAPR-19634: Sqoop2 now supports MapR-SASL.

Sqoop2 1.99.3-1409 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the the Apache Sqoop project homepage.

Version	1.99.3
Release Date	September 30, 2014
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.99.3-mapr-1409
MapR Version Compatibility	See Interoperability Matrices on page 5519.
Maven Artifacts	See MapR Maven Repository for Sqoop2 Artifacts

New in this Release

Sqoop2 1.99.3-1409 supports MapR version 4.0.1, which includes Hadoop 1.x (JobTracker and TaskTracker architecture) and Hadoop 2.x (ResourceManager and NodeManager architecture).

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
aff8146	2014-09-26	`sqoop.sh` script add common.loader property to `catalina.properties`.
e88bbd0	2014-09-24	Added `catalina.loader` for MapR 4.0.1.
ff6b77d	2014-09-09	BUG-14580: Fixed Sqoop2 export to MSSQL.

Sqoop2 1.99.3-1405 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the the Apache Sqoop 1.99.3 changelog or the Apache Sqoop project homepage.

Version	1.99.3
Release Date	June 11, 2014
Source on GitHub	https://github.com/mapr/sqoop
GitHub Release Tag	1.99.3-mapr-1405
MapR Version Compatibility	See Interoperability Matrices on page 5519.
Maven Artifacts	See MapR Maven Repository for Sqoop2 Artifacts

New in this Release

Sqoop2 addresses security concerns by introducing Admin and Operator roles. In Sqoop2, create access for Connections is restricted to Admins and execute access is limited to Operators.

Here are some other differences between Sqoop and Sqoop2:

- Connectors are no longer restricted to the JDBC model - they can define their own vocabulary
- Sqoop2 offers a REST API for operation and management, which integrates better with external systems such as Oozie
- Sqoop2 works with Hue, an interactive web-based UI
- Sqoop2 is installed and configured server-side, while Sqoop is installed and configured client-side.

For a comprehensive list of differences between Sqoop and Sqoop2, see MapR's Sqoop2 documentation.

Storm Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The release notes for Storm component (included in the MapR Converged Data Platform) contains notes specific to MapR only.

Note: Apache Storm is supported on pre-6.0 clusters but is not supported on MapR 6.0 or later. To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Storm 0.10.0-1703 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Storm homepage.

Note: This release of Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

Version	0.10.0
Release Date	April 2017
MapR Version Interoperability	See the Interoperability Matrix.
Source on GitHub	https://github.com/mapr/incubator-storm/
GitHub Release Tag	0.10.0-mapr-1703
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

This version supports integration with Hive 2.1.

last-updated: Apr 11, 2022

Fixes

For details on the features available in the open-source version of this component, see the Apache Storm homepage and the Apache Storm 0.10.2 changelog.

GitHub Commit Number	Date (YYYY-MM-DD)	MapR Fix Number and Description
b3abc36	2017-02-24	[MAPR-26080] Storm 0.10.0 - Hive 2.1 on Tez integration issue.

Known Issues and Limitations

Apache Storm is supported on pre-6.0 clusters but is not supported on MapR 6.0 or later.

Resolved Issues

None.

Storm 0.10.0-1611 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Storm homepage.

Note: This release of Storm 0.10.0 includes backports of all the patches included in Apache Storm 0.10.2.

Version	0.10.0
Release Date	December 9, 2016
MapR Version Interoperability	See the Interoperability Matrix.
Source on GitHub	https://github.com/mapr/incubator-storm/
GitHub Release Tag	0.10.0-mapr-1611
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

No new features.

Fixes

For details on the features available in the open-source version of this component, see the Apache Storm homepage and the Apache Storm 0.10.2 changelog.

Known Issues and Limitations

None.

Resolved Issues

None.

Storm 0.10.0-1609 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Storm Version	0.10.0
Release Date	September 30, 2016
MapR Version Interoperability	See Interoperability Matrices on page 5519
Source on GitHub	https://github.com/mapr/incubator-storm/tree/0.10.0-mapr-1609
GitHub Release Tag	0.10.0-mapr-1609
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-storm-0.10.0.201607121432-1.noarch.rpm • mapr-storm-nimbus-0.10.0.201607121432-1.noarch.rpm • mapr-storm-supervisor-0.10.0.201607121432-1.noarch.rpm • mapr-storm-ui-0.10.0.201607121432-1.noarch.rpm • mapr-storm-nimbus_0.10.0.201607121432_all.deb • mapr-storm-supervisor_0.10.0.201607121432_all.deb • mapr-storm-ui_0.10.0.201607121432_all.deb • mapr-storm_0.10.0.201607121432_all.deb

Fixes

GitHub Commit	Data (YYYY-MM-DD)	Comment
05110a1	2016-09-07	MAPR-24472: With this fix, MapR-related dependencies are added into the Storm `/lib` directory.
bf7965a	2016-09-13	MAPR-24554: With this fix, Maven artifacts are updated to the actual ecosystem version.
aab55be	2016-09-13	MAPR-24472: This fix removes creation of the `$MARLIN_CLASSPATH`, which concatenates to the `storm` classpath. The fix also adds the `json` library and removes some of the `slf4j` and the `maprfs` library.

Storm 0.10.0-1607 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Storm Version	0.10.0
Release Date	July 29, 2016
MapR Version Interoperability	See Interoperability Matrices on page 5519
Source on GitHub	https://github.com/mapr/incubator-storm/tree/0.10.0-mapr-1607
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release:

	• mapr-storm-0.10.0.201607121432-1.noarch.rpm • mapr-storm_0.10.0.201607121432_all.deb • mapr-storm-nimbus-0.10.0.201607121432-1.noarch.rpm • mapr-storm-nimbus_0.10.0.201607121432_all.deb • mapr-storm-supervisor-0.10.0.201607121432-1.noarch.rpm • mapr-storm-supervisor_0.10.0.201607121432_all.deb • mapr-storm-ui-0.10.0.201607121432-1.noarch.rpm • mapr-storm-ui_0.10.0.201607121432_all.deb

New in this Release

This release of Storm 0.10.0 includes backports of all the patches included in Storm 0.10.1.

For details on the features available in the open source version of this component, see Apache Storm homepage and Apache Storm 0.10.1 changelog.

Storm 0.10.0-1602 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Storm Version	0.10.0
Release Date	February 29, 2016
MapR Version Interoperability	See Interoperability Matrices on page 5519
Source on GitHub	https://github.com/mapr/incubator-storm/tree/0.10.0-mapr-1602
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-storm-0.10.0.201602252228-1.noarch.rpm • mapr-storm_0.10.0.201602252228_all.deb • mapr-storm-nimbus-0.10.0.201602252228-1.noarch.rpm • mapr-storm-nimbus_0.10.0.201602252228_all.deb • mapr-storm-supervisor-0.10.0.201602252228-1.noarch.rpm • mapr-storm-supervisor_0.10.0.201602252228_all.deb • mapr-storm-ui-0.10.0.201602252228-1.noarch.rpm • mapr-storm-ui_0.10.0.201602252228_all.deb

New in this Release

This initial release of Storm 0.10.0 and it includes the following behavior change that is specific to MapR:

- Support for MapR Event Store For Apache Kafka through the Kafka 0.9 API.
- Examples were updated to support Kerberos authentication and multi-node clusters.

last-updated: Apr 11, 2022

For details on the features available in the open source version of this component, see the Apache Storm 0.10.0 changelog or the Apache Storm homepage.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

GitHub Commit	Date (YYY-MM-DD)	Comment
554c968	2016-01-11	MAPR-22227: Using the Kafka 0.9 API, Storm now supports a non-autocommit spout for MapR Event Store For Apache Kafka.
1cf0ed21	2015-12-25	MAPR-22227: Storm now includes a MapR Event Store For Apache Kafka example.
40cbaca9	2015-12-28	A Hive and JDBC (MySQL) example is now included in the Storm package.

Known Issues

- MAPR-22311: Storm does not securely communicate with Zookeeper on a secure cluster that uses MapR-SASL authentication.

Storm 0.9.4-1509 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

Version	0.9.4
Release Date	Oct 5, 2015
Source on GitHub	https://github.com/mapr/incubator-storm/tree/0.9.4-mapr-1509
MapR Version Compatibility	See Interoperability Matrices on page 5519
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	The following packages are associated with this release: • mapr-storm-0.9.4.201510010912-1.noarch.rpm • mapr-storm_0.9.4.201510010912_all.deb • mapr-storm-nimbus-0.9.4.201510010912-1.noarch.rpm • mapr-storm-nimbus_0.9.4.201510010912_all.deb • mapr-storm-supervisor-0.9.4.201510010912-1.noarch.rpm • mapr-storm-supervisor_0.9.4.201510010912_all.deb • mapr-storm-ui-0.9.4.201510010912-1.noarch.rpm • mapr-storm-ui_0.9.4.201510010912_all.deb

New in this Release

This release of Storm includes the following behavior change that is specific to MapR:

last-updated: Apr 11, 2022

- When you install the Storm packages on a node, dependent packages on that node are also installed or upgraded to match the Storm package that you installed.

For details on the features available in the open source version of this component, see the Apache Storm 0.9.4 changelog or the Apache Storm homepage.

Storm 0.9.4-1507 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Storm 0.9.5 changelog or the Apache Storm homepage.

Version	0.9.4
Release Date	August 5, 2015
Source on GitHub	https://github.com/mapr/incubator-storm.git
GitHub Release Tag	0.9.4-mapr-1507
MapR Version Compatibility	See Interoperability Matrices on page 5519
Maven Artifacts	See MapR Maven Repository for Sqoop2 Artifacts

Fixes

This release from MapR includes the following fixes on the base Apache release. These fixes were back-ported from Storm Version 0.9.5. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
3f2e0e3	2015-06-11	STORM-745: Fixed storm.cmd to evaluate 'shift' correctly with 'storm jar'.
7991e72	2015-06-11	STORM-796: Added support for "error" command in ShellSpout.
a3187c8	2015-06-11	STORM-790: When a task is null in transfer-fn, Storm no longer lets the worker process die. Instead, it logs a "task is null" message.
a3187c8	2015-06-11	STORM-130: Supervisor no longer gets killed due to java.io.FileNotFoundException: `File '../stormconf.ser' does not exist.`

Storm 0.9.4-1504 Release Notes

Important: This component is deprecated. HPE recommends using an alternate product. For more information, see Discontinued Ecosystem Components on page 5584.

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Storm 0.9.4 changelog or the Apache Storm homepage.

Version	0.9.4
Release Date	May 6, 2015
Source on GitHub	https://github.com/mapr/incubator-storm

GitHub Release Tag	0.9.4-mapr-1504
MapR Version Compatibility	See Interoperability Matrices on page 5519
Maven Artifacts	See MapR Maven Repository for Storm Artifacts

New in This Release

This is the first release of Storm 0.9.4 in the MapR Distribution for Apache Hadoop.

For information about using Storm and installation instructions, see Storm (Versions 0.9.3-1501 and 0.9.4). For information about upgrading to Storm 0.9.4, see MapR's Storm documentation.

Tez Release Notes

The release notes for the Tez component contain notes specific to MapR software only.

Note: To identify the EEP to which a specific release note belongs, see EEP Release Notes on page 5658. To see which operating systems support the ecosystem components in a specific EEP, see EEP Components and OS Support on page 5536. To view release notes for prior MapR releases, see Previous Versions on page 6578.

Tez 0.9.2 - 2201 (EEP 8.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Tez. You may also be interested in the Apache Tez changelog and the Apache Tez home page.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.9.2
Release Date	January 2022
Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Not applicable
GitHub Release Tag	0.9.2.400-eep-810
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP (MEP) and OS to view the list of package names.

New in This Release

Tez 0.9.2 - 2201 is a defect-repair release.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
f5c03c004	2022-01-25	EEP-TEZ-215: Upgrade Log4J version to '1.3.1-mapr'
32dfb6f5a	2021-12-23	EEP-TEZ-208: CVE-2019-10744, CVE-2020-8203, CVE-2021-23337 : lodash-*.js

1cadfeed3	2021-12-23	EEP-TEZ-210: CVE-2021-25329: tomcat-catalina-9.0.36.jar & CVE-2020-13934, CVE-2020-17527, CVE-2021-25122, CVE-2021-41079: tomcat-coyote-9.0.36.jar
24fe8da63	2021-12-23	EEP-TEZ-213: log4j-1.2.17.jar vulnerability: CVE-2019-17571
6f5bfee01	2021-12-17	EEP-TEZ-209: High: WS-2020-0408 ; 3 Medium: CVE-2021-21290: netty*.jar
e0f57eec9	2021-12-10	EEP-TEZ-207: CVE-2019-10172, CVE-2019-10202: jackson-mapper-asl-1.9.13.jar
56d130d53	2021-12-06	EEP-TEZ-212: Upgrade Jetty to 9.4.44.v20210927 to sync with Hadoop
21d777f71	2021-11-23	EEP-TEZ-211: Hive on Tez job failed for core 7 with FIPS enabled. Can't find HmacSHA1 algorithm.
97da9f716	2021-11-23	EEP-TEZ-205: Update hadoop version to 2.7.6.200-eep-810-SNAPSHOT
27b2b96f3	2021-10-08	MAPR-TEZ-203: Backport Apache Jira TEZ-3951 and add timeout logic for cancellation

For details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

None.

Resolved Issues

None.

Tez 0.9.2 - 2110 (EEP 8.0.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Tez. You may also be interested in the Apache Tez changelog and the Apache Tez home page.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.9.2
Release Date	October 2021
Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	https://github.com/mapr/tez
GitHub Release Tag	0.9.2.300-eep-800
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP (MEP) and OS to view the list of package names.

New in This Release

Tez 0.9.2 - 2110 introduces the following enhancements or HPE platform-specific behavior changes:

None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
c4e33daed	2021-09-03	MAPR-TEZ-201: Update the maven artifact version strings to eep.
c98eef1fd	2021-08-17	MAPR-TEZ-200: Change project version from 0.9.2-mapr-SNAPSHOT to 0.9.2.0-mapr-SNAPSHOT for development artifacts.
c2264efc9	2021-08-09	MAPR-TEZ-199: Update Hadoop version to 2.7.6.0-mapr-720-SNAPSHOT.
93a750b7e	2021-07-26	MAPR-TEZ-172: Add Hbase jars to maprfs:/apps/tez/tez-0.9/hbase during Tez configuration via configure.sh.

For details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

None.

Note: In the previous release note (Tez 0.9.2 - 2104 (EEP 7.1.0) Release Notes on page 6483), you can see that MAPR-TEZ-172 was part of the known issues. In this release, MAPR-TEZ-172 has been fixed, and the commit reference is 93a750b7e.

Resolved Issues

None.

Tez 0.9.2 - 2104 (EEP 7.1.0) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Tez. You may also be interested in the Apache Tez changelog and the Apache Tez home page.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.9.2
Release Date	April 2021
Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Not Applicable
GitHub Release Tag	0.9.2.200-mapr-710
Maven Artifacts	https://repository.mapr.com/maven/

Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

Tez 0.9.2 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- Additional logs for configuration.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
e2fb3f5d3	2021-03-31	MAPR-TEZ-194: Sync Jetty version with MAPRHADOOP-2.7.5.0-mapr-710.
eafbec476	2021-03-25	MAPR-TEZ-193 : Update Hadoop version to 2.7.5.0-mapr-710.
29030a603	2021-02-24	MAPR-TEZ-189: Fixing the unit test org.apache.tez.common.TestTezCommonUtils.
3b1a23627	2021-02-24	MAPR-TEZ-191: Fixing the unit test org.apache.hadoop.mapred.split.TestGroupedSplits.
2f270a759	2021-02-12	MAPR-TEZ-187 : Update Hadoop version to 2.7.4.0-mapr-710.
e9be5ed6f	2021-02-12	MAPR-TEZ-188: Upgrade async-http-client which uses deprecated Netty header methods.
29a9cbcc4	2021-02-12	MAPR-TEZ-99: Add logging for configure.sh in Tez.
f8f8f1662	2021-02-09	TEZ-185: Cherry picking TEZ-3929 - Upgrade Jersey to 1.19 (Eric Wohlstadter via jeagles).
2a7eaf824	2021-01-26	IN-2782: Tez UI is not configured correctly via UI installer on secure cluster.
faa6e4caf	2021-01-26	MAPR-TEZ-183: Fixing the warning “/opt/mapr/tez/tez-0.9/conf.*_*’: No such file or directory” during the execution of the Tez configure.sh script.

For details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- HIVE-947: If you add `tez.history.logging.service.class` and `tez.tez-ui.history-url.base` properties to `tez-site.xml` file, Hive applications will fail in EEP 7.1.0. To fix this issue, remove these properties from the `tez-site.xml` configuration file.
- TEZ-172: HBase + Hive + Tez integration in EEP 7.1.0 you may face this exception “Caused by: java.lang.ClassNotFoundException”

- TEZ-172: An HBase + Hive + Tez integration in EEP 7.1.0 may result in the following exception:

```
Caused by: java.lang.ClassNotFoundException:
org.apache.hadoop.hbase.client.mapr.BaseTableMappingRules
    at java.base/
jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:5
81)
    at java.base/
jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(ClassLoaders.jav
a:178)
    at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:522)
    ... 39 more
```

  This exception is due to a new Tez classloader implemented in the Tez project. To fix this issue, put these additional JARs into the `/apps/tez/tez-0.9` filesystem folder.

  Run the following commands before you run the HBase + Hive + Tez integration in EEP 7.1.0:

```
hadoop fs -mkdir /apps/tez/tez-0.9/hbase
hadoop fs -put /opt/mapr/hbase/hbase-1.4.13/lib/* /apps/tez/tez-0.9/hbase/
```

  Add the following property to `/opt/mapr/tez/tez-0.9/conf/tez-site.xml`:

```
<property>
<name>tez.lib.uris</name>
<value>${fs.defaultFS}/apps/tez/tez-0.9,${fs.defaultFS}/apps/tez/tez-0.9/
lib,${fs.defaultFS}/apps/tez/tez-0.9/hbase/</value>
 </property>
```

  This information is based on the assumption that Hive version is 2.3, Hbase version is 1.4.13, Tez version is 0.9, Hadoop version is 2.7.5.0, Zookeeper version is 3.5.6.0, and ecosystem release is 2104.

Resolved Issues

- None.

Tez 0.9.2 - 2101 (EEP 7.0.1) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Tez. You may also be interested in the Apache Tez changelog and the Apache Tez home page.

Version	0.9.2
Release Date	January 2021
HPE Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Not Applicable
GitHub Release Tag	0.9.2.100-mapr-701
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your MEP and OS to view the list of package names.

New in This Release

Tez 0.9.2 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
ec9528e	2020-12-11	MAPR-TEZ-125: Fix io.netty vulnerability
1a225f6	2020-11-23	MAPR-TEZ-175: Replacing maprcli command with Yarn API to get RM url address
4f3d2f3 d7ff176	2020-11-05	MAPR-TEZ-150: Set configuration back-ups limit
b1b6c87	2020-10-23	MAPR-TEZ-153: Fixed commons-beanutils-core vulnerability CVE-2014-0114
9f37a2b	2020-10-04	MAPR-TEZ-165: Corrected path to logs of Tez-UI in warden file
451d8de	2020-10-02	MAPR-TEZ-173: reduced Tez package size

For details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- TEZ-172: HBase + Hive + Tez integration in MEP-7.0.1 you may face this exception “Caused by: java.lang.ClassNotFoundException”

- TEZ-172: An HBase + Hive + Tez integration in MEP 7.0.1 may result in the following exception:

```
Caused by: java.lang.ClassNotFoundException:
org.apache.hadoop.hbase.client.mapr.BaseTableMappingRules
    at java.base/
jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:5
81)
    at java.base/
jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(ClassLoaders.jav
a:178)
    at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:522)
    ... 39 more
```

  This exception is due to a new Tez classloader implemented in the Tez project. To fix this issue, put these additional JARs into the `/apps/tez/tez-0.9` filesystem folder.

  Run the following commands before you run the HBase + Hive + Tez integration in MEP-7.0.1:

```
hadoop fs -mkdir /apps/tez/tez-0.9/hbase
hadoop fs -put /opt/mapr/hbase/hbase-1.4.12/lib/* /apps/tez/tez-0.9/hbase/
```

  Add the following property to `/opt/mapr/tez/tez-0.9/conf/tez-site.xml`:

```
<property>
<name>tez.lib.uris</name>
<value>${fs.defaultFS}/apps/tez/tez-0.9,${fs.defaultFS}/apps/tez/tez-0.9/
lib,${fs.defaultFS}/apps/tez/tez-0.9/hbase/</value>
 </property>
```

  This information is based on the assumption that Hive version is 2.3, Hbase version is 1.4.12, Tez version is 0.9, Hadoop version is 2.7.4, Zookeeper version is 3.5.6.0, and ecosystem release is 2101.

Resolved Issues

- None.

Tez 0.9.2-2009 (EEP 7.0.0) Release Notes

The notes below relate specifically to the data-fabric distribution for Apache Tez. You may also be interested in the Apache Tez changelog and the Apache Tez home page.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.9.2
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Not Applicable
GitHub Release Tag	0.9.2-mapr-2009
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- Tomcat server updated to v9.0.36.

- Java 11 support

Fixes

This data-fabric release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	Data Fabric Fix Number and Description
3e82ada	2019-07-11	MAPR-TEZ-76 : Make Tez-UI server manageable by MCS
873df67	2019-07-08	MAPR-TEZ-95 : Web Application Potentially Vulnerable to Clickjacking
3c3e375	2020-01-20	MAPR-TEZ-97: Protobuf upgrade for 6.2 to Protobuf v3.11.1 in Tez
2f60580	2020-04-30	MAPR-TEZ-126: Vulnerability with commons-beanutils
B9a91d6 4f3f002	2020-05-04 2020-06-09	MAPR-TEZ-113: Add security headers for TezUI.
1c4095d	2020-05-12	MAPR-TEZ-127: Update moment*.js version to 2.25.2
9391f29	2020-05-14	MAPR-TEZ-138: Update Guava version to 28.2-jre
027fa33	2020-05-25	MAPR-TEZ-132: Updated 3d-patry libraries.
4e7d752	2020-06-09	MAPR-TEZ-139: Disable AJP connector
5b26d36	2020-06-09	MAPR-TEZ-125: Update netty to version 4.1.50.Final
835e5ab	2020-06-09	MAPR-TEZ-143 : Error while running rez job: Unrecognized VM option 'PrintGCTimeStamps'
f91729e	2020-06-22	MAPR-TEZ-147: Update Tomcat to 9.0.36 version
438c78d	2020-07-08	MAPR-TEZ-149: Fixed redirection from root url
6932a41	2020-06-09	MAPR-TEZ-142: Changed value of Content-Security-Policy header for fixing TezUI
a5a35a3	2020-08-10	MAPR-TEZ-164: Fixed TezUI status script
a11af9c	2020-08-25	MAPR-TEZ-159: Configure server.xml for TezUI
72658ba	2020-09-03	MAPR-TEZ-170 : Update ZK version to 3.5.6.0
db7e74b	2020-09-03	TEZ-4223 - Adding new jars or resources after the first DAG runs does not work.
ef1dc7d	2020-09-03	TEZ-4228 : TezClassLoader should be used in TezChild and for Configuration objects
c011227a	2020-06-09	TEZ-3860. JDK9: ReflectionUtils may not use URLClassLoader
bafc5b7	2019-07-04	TEZ-4057: Fix Unsorted broadcast shuffle umasks

Known Issues and Limitations

- TEZ-172: If you use HBase + Hive + Tez integration in MEP-7.0.0, you may encounter this exception:

```
Caused by: java.lang.ClassNotFoundException:
org.apache.hadoop.hbase.client.mapr.BaseTableMappingRules
    at java.base/
jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:5
81)
```

```
    at java.base/
jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(ClassLoaders.jav
a:178)
    at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:522)
    ... 39 more
```

This exception happens because of the new Tez classloader implemented in the Tez project.
Workaround: To fix the issue, you need to put these additional jars into the data-fabric file-system folder `/apps/tez/tez-0.9`.

This is the list of JARs and operations you need to execute before running HBase + Hive + Tez integration in MEP-7.0.0:

```
hadoop fs -mkdir /apps/tez/tez-0.9/hbase
hadoop fs -put /opt/mapr/hbase/hbase-1.4.12/lib/* /apps/tez/tez-0.9/hbase/
```

Add the following property to `/opt/mapr/tez/tez-0.9/conf/tez-site.xml`:

```
<property>
<name>tez.lib.uris</name>
<value>${fs.defaultFS}/apps/tez/tez-0.9,${fs.defaultFS}/apps/tez/tez-0.9/
lib,${fs.defaultFS}/apps/tez/tez-0.9/hbase/</value>
</property>
```

This example assumes the following software versions:

- Hive version is 2.3
- HBase version is 1.4.12
- Tez version is 0.9
- Hadoop version is 2.7.4
- Zookeeper version is 3.5.6.0
- MEP release is 2009

Resolved Issues

- None.

Tez 0.9.1 - 2201 (EEP 6.3.6) Release Notes

The notes below relate specifically to the MapR Data Platform Distribution for Apache Tez. You may also be interested in theApache Tez changelog and the Apache Tez home page.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes.

Version	0.9.1
Release Date	January 2022
HPE Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Not Applicable
GitHub Release Tag	0.9.1-mapr-2201
Maven Artifacts	https://repository.mapr.com/maven/

Package Names	Navigate tohttps://package.mapr.hpe.com/releases/MEP/ and select your MEP and OS to view the list of package names.

New in This Release

Tez 0.9.1 - 2201 is a defect-repair release.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
c71dc37e4	2022-01-25	EEP-TEZ-215: Upgrade Log4J version to '1.3.1-mapr'
c9d2c75c9	2021-12-23	EEP-TEZ-208: CVE-2019-10744, CVE-2020-8203, CVE-2021-23337 : lodash-*.js
edc36b95b	2021-12-23	EEP-TEZ-210: CVE-2021-25329: tomcat-catalina-9.0.36.jar & CVE-2020-13934, CVE-2020-17527, CVE-2021-25122, CVE-2021-41079: tomcat-coyote-9.0.36.jar
c0cc5bae8	2021-12-23	EEP-TEZ-213: log4j-1.2.17.jar vulnerability: CVE-2019-17571
239132474	2021-12-17	EEP-TEZ-209: High: WS-2020-0408 ; 3 Medium: CVE-2021-21290: netty*.jar
c8411eb7e	2021-12-10	EEP-TEZ-207: CVE-2019-10172, CVE-2019-10202: jackson-mapper-asl-1.9.13.jar
6ee9d869e	2021-10-08	MAPR-TEZ-203: Backport Apache Jira TEZ-3951 and add timeout logic for cancellation
198d0b67b	2021-10-08	TEZ-3953: Restore ABI-compat for DAGClient for TEZ-3951 (Sergey Shelukhin via Gopal V)
ac5c1e9a4	2021-10-08	TEZ-3951. TezClient wait too long for the DAGClient for prewarm; tries to shut down the wrong DAG (Sergey Shelukhin via Harish Jaiprakash)
4176270ed	2021-10-08	TEZ-3943. TezClient leaks DAGClient for prewarm (Sergey Shelukhin via jlowe)
14d1b5b77	2021-10-08	TEZ-3892: getClient API for TezClient (Eric Wohlstadter via Gopal V)

For details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Tez 0.9.1 - 2104 (EEP 6.3.4) Release Notes

The notes below relate specifically to the MapR Data Platform Distribution for Apache Tez. You may also be interested in theApache Tez changelog and the Apache Tez home page.

These release notes contain only HPE-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes.

Version	0.9.1
Release Date	April 2021
HPE Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Not Applicable
GitHub Release Tag	0.9.1-mapr-2104
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate tohttps://package.mapr.hpe.com/releases/MEP/ and select your MEP and OS to view the list of package names.

New in This Release

Tez 0.9.1 - 2104 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Data (YYYY-MM-DD)	HPE Fix Number and Description
a1d77867e	2021-03-30	MAPR-TEZ-154: (CVE) Update httpclient to version 4.5.3.
2d9ad8317	2021-02-12	MAPR-TEZ-188: Upgrade async-http-client which uses deprecated Netty header methods.
9a5c14d94	2021-02-12	MAPR-TEZ-99: Add logging for configure.sh in Tez.
593ac26bb	2021-02-09	MAPR-TEZ-186 : Exclude jersey 1.9 to avoid conflict with MapR core patch.
c1e923989	2021-02-09	TEZ-185: Cherry picking TEZ-3929 - Upgrade Jersey to 1.19 (Eric Wohlstadter via jeagles).
4fd588216	2021-01-26	IN-2782: Tez UI is not configured correctly via UI installer on secure cluster.
4ccb9cd3d	2021-01-26	MAPR-TEZ-183: Fixing the warning “/opt/mapr/tez/tez-0.9/conf.*_*’: No such file or directory” during the execution of the Tez configure.sh script.

last-updated: Apr 11, 2022

For details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Tez 0.9.1 - 2101 (EEP 6.3.2) Release Notes

The notes below relate specifically to the HPE Ezmeral Data Fabric Distribution for Apache Tez. You may also be interested in theApache Tez changelog and the Apache Tez home page.

Version	0.9.1
Release Date	January 2021
HPE Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Not Applicable
GitHub Release Tag	0.9.1-mapr-2101
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate tohttps://package.mapr.hpe.com/releases/MEP/ and select your MEP and OS to view the list of package names.

New in This Release

Tez 0.9.1 - 2101 introduces the following enhancements or HPE platform-specific behavior changes:

- None.

Fixes

This HPE release includes the following fixes on the base release:

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
abf2801	2020-12-11	MAPR-TEZ-125: Fix io.netty vulnerability
a0049d4	2020-11-23	MAPR-TEZ-175: Replacing maprcli with Yarn API to get RM url address in tez-ui
12f57a0 4fb40ea	2020-11-05	MAPR-TEZ-150: Set configuration back-ups limit
6139f92	2020-10-23	MAPR-TEZ-153: Fixed commons-beanutils-core vulnerability CVE-2014-0114
0cf3f13	2020-10-02	MAPR-TEZ-173: reduced Tez package size

For details, refer to the commit log for this project in GitHub.

Known Issues and Limitations

- None.

Resolved Issues

- None.

Tez 0.9.1-2009 (EEP 6.3.1) Release Notes

The notes below relate specifically to the data-fabric distribution for Apache Tez. You may also be interested in the Apache Tez changelog and the Apache Tez home page.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.9.1
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Not Applicable
GitHub Release Tag	0.9.1-mapr-2009
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- Tomcat server updated to v9.0.36.

Fixes

This data-fabric release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	Data Fabric Fix Number and Description
3949420	2020-04-30	MAPR-TEZ-126: Vulnerability with commons-beanutils
4ad13dc 1a3a1e8	2020-05-04 2020-07-08	MAPR-TEZ-113: Add security headers for TezUI.
c112408	2020-05-12	MAPR-TEZ-127: Update moment*.js version to 2.25.2
88f6937	2020-05-25	MAPR-TEZ-132: Updated 3d-patry libraries.
b900108	2020-05-25	MAPR-TEZ-139: Disable AJP connector
5f03f95	2020-05-25	MAPR-TEZ-125: Update netty to version 4.1.50.Final
118aa78	2020-06-22	MAPR-TEZ-147: Update Tomcat to 9.0.36 version
1576c16	2020-07-08	MAPR-TEZ-149: Fixed redirection from root url
42ac06a	2020-07-20	MAPR-TEZ-142: Changed value of Content-Security-Policy header for fixing TezUI

last-updated: Apr 11, 2022

C5d0fcd 2468a39 8538854	2020-08-25 2020-08-26 2020-09-03	MAPR-TEZ-159: Configure server.xml for TezUI

Known Issues and Limitations

- None

Resolved Issues

- None.

Tez 0.9.1-1912 (EEP 6.3.0) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Tez changelog and the Apache Tez home page.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.9.1
Release Date	December 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Not Applicable
GitHub Release Tag	0.9.1-mapr-1912
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
7e2c008	2019-07-08	MAPR-TEZ-95: Web Application Potentially Vulnerable to Clickjacking
5bececf	2019-07-05	MAPR-TEZ-100: Add to configure.sh configuration of tez-site.xml permission policy
23b4937	2019-06-25	MAPR-TEZ-75: Tez-UI Tomcat is deployed with examples

Known Issues and Limitations

- None.

Resolved Issues

- None.

Tez 0.9.1-1904 (EEP 6.2.0, EEP 6.1.1, and EEP 6.0.2) Release Notes

The release notes for Apache Tez 0.9.1-1904 relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Tez changelog and the Apache Tez homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.9.1
Release Date	April 2019
Source on GitHub	Not Applicable
GitHub Release Tag	0.9.1-mapr-1904
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Maven Artifacts	See Maven Artifacts for MapR on page 4155
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Minor changes backported from Apache Tez 0.9.2 which fix bugs and increase stability.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date	Comment
76882f3	2018-09-20	TEZ-3982: DAGAppMaster and tasks should not report negative or invalid progress
447f09e	2018-07-07	TEZ-3963: Possible InflaterInputStream leaked in TezCommonUtils and related classes
2cf9b40	2018-07-12	TEZ-3964: Inflater not closed in some places
40d42ba	2018-09-19	TEZ-3972: Tez DAG can hang when a single task fails to fetch
e744258	2018-09-25	TEZ-3981: UnorderedPartitionedKVWriter.getInitialMemoryRequirement may return negative memory
289c919	2018-10-11	TEZ-3990: The number of shuffle penalties for a host/inputAttemptIdentifier should be capped
f39dab4	2018-08-10	TEZ-3974: Correctness regression of TEZ-955 in TEZ-2937

Known Issues and Limitations

- None.

Resolved Issues

- None.

Tez 0.9.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes

last-updated: Apr 11, 2022

The release notes for Apache Tez 0.9.1-1901 (MapR Ecosystem Pack 6.1.0) relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Tez changelog and the Apache Tez homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.9.1
Release Date	January 2019
Source on GitHub	Not Applicable
GitHub Release Tag	0.9.1-mapr-1901
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Maven Artifacts	See Maven Artifacts for MapR on page 4155
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Added Shuffle SSL encryption configuration of Tez with `/opt/mapr/server/configure.sh -R`.
- Hadoop jars are excluded from Tez libraries.

Changes in Security with Default Configuration

Added the encryption properties to the `tez-site.xml` configuration file by default on a secured cluster:

Property	Value
`tez.runtime.shuffle.ssl.enable`	true
`tez.runtime.shuffle.keep-alive.enabled`	true

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date	Comment
7fe4dff	2018-12-13	MAPR-TEZ-53: Add Shuffle SSL encryption by default for secure cluster
c761f356	2018-12-24	MAPR-TEZ-54: Remove hadoop-yarn jars from Tez package

Known Issues and Limitations

None.

Resolved Issues

None.

Tez 0.9.1-1808 (EEP 6.0.0) Release Notes

The release notes for Apache Tez 0.9.1-1808 (MapR Ecosystem Pack 6.0.0) relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Tez changelog and the Apache Tez homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.9.1
Release Date	September 2018
Source on GitHub	Not Applicable
GitHub Release Tag	0.9.1-mapr-1808
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Maven Artifacts	See Maven Artifacts for MapR on page 4155
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

- Implemented preserving configuration during package update. For more information, see Pre-Upgrade Steps for Tez on page 349.
- Implemented RM HA for the new Tez UI.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date	Comment
38cf581	2018-07-06	MAPR-TEZ-44: org.apache.hadoop.hive.ql.exec.mr.MapRedTask expecting non-static method org.apache.hadoop.yarn.client.RMProxy
d8faa09	2018-06-14	MAPR-TEZ-39: Cannot obtain application information using link (Application Tracking URL) in TEZ UI on TEZ-0.9
175083e	2018-06-14	MAPR-TEZ-37: Tomcat configurations example missing quotes for TEZ-0.9
ec5cf16	2018-06-07	MAPR-TEZ-33: Incorrect log link in DAG information for Tez 0.9
224211d	2018-05-24	MAPR-TEZ-28: Implement RM HA for new Tez UI
881eaa5	2018-04-20	MAPR-TEZ-31: Change hadoop-common dependency to the latest version

last-updated: Apr 11, 2022

Known Issues and Limitations

None.

Resolved Issues

None.

Tez 0.8.4-2009 (EEP 5.0.5) Release Notes

The notes below relate specifically to the data-fabric distribution for Apache Tez. You may also be interested in the Apache Tez changelog and the Apache Tez home page.

These release notes contain only data-fabric-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.8.4
Release Date	September 2020
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Not Applicable
GitHub Release Tag	0.8.4-mapr-2009
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- None

Fixes

This data-fabric release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	Data Fabric Fix Number and Description
5b3a9d3	2020-01-25	MAPR-TEZ-147: Update Tomcat to version 9.0.36
910310b	2020-07-03	MAPR-TEZ-158: Update bower version to 1.8.8

Known Issues and Limitations

- None

Resolved Issues

- None.

Tez 0.8.4-1912 (EEP 5.0.4) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Tez changelog and the Apache Tez home page.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.8.4
Release Date	December 2019
MapR Version Interoperability	See EEP Components and OS Support on page 5536
Source on GitHub	Not Applicable
GitHub Release Tag	0.8.4-mapr-1912
Maven Artifacts	https://repository.mapr.com/maven/
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names.

New in This Release

- None.

Fixes

This MapR release includes the following fixes on the base Apache release. For details, refer to the commit log for this project in GitHub.

GitHub Commit Number	Data (YYYY-MM-DD)	MapR Fix Number and Description
d2c4fb1	2019-07-26	MAPR-TEZ-80: Application logs are not available if we redirected to TEZ UI from RM UI
9f66b1e	2019-07-08	MAPR-TEZ-95: Web Application Potentially Vulnerable to Clickjacking
5e167ea	2019-06-25	MAPR-TEZ-75: Tez-UI Tomcat is deployed with examples

Known Issues and Limitations

- None.

Resolved Issues

- None.

Tez 0.8.4-1901 (EEP 4.1.3 and EEP 5.0.2) Release Notes

The release notes for Apache Tez 0.8.4-1901 (MapR Ecosystem Pack 6.0.0) relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Tez changelog and the Apache Tez homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.8.4
Release Date	February 2019
Source on GitHub	Not Applicable
GitHub Release Tag	0.8.4-mapr-1901

MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Maven Artifacts	See Maven Artifacts for MapR on page 4155
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Hadoop jars are excluded from Tez libraries.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
fbc855e	2018-12-26	MAPR-TEZ-54: Remove hadoop-yarn jars from Tez package

Known Issues and Limitations

None.

Resolved Issues

None.

Tez 0.8.4-1808 (EEP 4.1.2 and EEP 5.0.1) Release Notes

The release notes for Apache Tez 0.8.4-1808 (MapR Ecosystem Pack 6.0.0) relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Tez changelog and the Apache Tez homepage.

These release notes contain only MapR-specific information and are not necessarily cumulative in nature. For information about how to use the release notes, see Ecosystem Component Release Notes on page 5658.

Version	0.8.4
Release Date	September 2018
Source on GitHub	Not Applicable
GitHub Release Tag	0.8.4-mapr-1808
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Maven Artifacts	See Maven Artifacts for MapR on page 4155
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Implemented preserving of configuration during package update.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
9ced85e	2018-05-13	MAPR-TEZ-29: Cannot obtain application information via link (Application Tracking URL) in the TEZ UI
d93168d	2018-03-23	MAPR-TEZ-20: Incorrect link to a task log in the TEZ UI

Known Issues and Limitations

None.

Resolved Issues

None.

Tez 0.8.4-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Tez changelog and the Apache Tez homepage.

Version	0.8.4
Release Date	March 2018
Source on GitHub	N/A
GitHub Release Tag	0.8.4-mapr-1803
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Maven Artifacts	See Maven Artifacts for MapR on page 4155
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

Added support for RM HA in the Tez user interface.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
d25f5db	2018-03-18	MAPR-TEZ-15: TIMELINE_BASE_URL in configs.js remains unchanged after installing Tez via the Installer
5debdd3	2018-02-09	MAPR-TEZ-11: Cannot obtain log information via Tez UI on MapR-secure cluster

Commit	Date (YYYY-MM-DD)	Comment
3aa1b7b	2018-02-19	MAPR-TEZ-7: Tomcat configuration example was missing quotes
2211002	2018-01-30	MAPR-TEZ-13: Cannot obtain real job progress via TEZ UI while RM HA is enabled
e09f228	2018-03-12	MAPR-TEZ-9: Tez should clean up all working dirs and stop Tomcat service (and remove dir) after removing Tez from a cluster
47dceb8	2018-03-14	MAPR-TEZ-12: Warnings during removing

Known Issues and Limitations

None.

Resolved Issues

None.

Tez 0.8.4-1803 (EEP 3.0.3) Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Tez changelog and the Apache Tez homepage.

Version	0.8.4
Release Date	March 2018
Source on GitHub	N/A
GitHub Release Tag	0.8.4-mapr-1803
MapR Version Interoperability	See EEP Components and OS Support on page 5536.
Maven Artifacts	See Maven Artifacts for MapR on page 4155
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

None.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
47dceb8	2018-03-14	MAPR-TEZ-12: Warnings during removing

Known Issues and Limitations

None.

Resolved Issues

None.

Tez 0.8-1710 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Tez changelog and the Apache Tez homepage.

Version	0.8
Release Date	November 2017
Source on GitHub	N/A
Maven Artifacts	See Maven Artifacts for MapR on page 4155
Package Names	Navigate to https://package.mapr.hpe.com/releases/MEP/ and select your EEP and OS to view the list of package names

New in This Release

The Tez user interface relies on the timeline server, which serves as a backing store for the application data generated during the lifetime of a YARN application. The Tez user interface uses the timeline server to display both a live and historical view of the Tez application inside a Tez web application.

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
f2fb43b	2017-10-05	MAPR-TEZ-5: Add dependency on apache-tomcat to Tez package

Known Issues and Limitations

Tez 0.8 is supported only for Hive 2.1 use cases.

Resolved Issues

None.

Tez 0.8-1703 Release Notes

The notes below relate specifically to the MapR Distribution for Apache Hadoop. You may also be interested in the Apache Tez changelog and the Apache Tez homepage.

Version	0.8
Release Date	April 2017
Source on GitHub	https://github.com/mapr/private-tez/tree/branch-0.8-mapr-1703
Maven Artifacts	See Maven Artifacts for MapR on page 4155
Package Names	See Package Names for MapR Ecosystem Packs (EEPs)

New in This Release

There are no new features in this release.

last-updated: Apr 11, 2022

Fixes

This release by MapR includes the following fixes on the base Apache release. For complete details, refer to the commit log for this project in GitHub.

Commit	Date (YYYY-MM-DD)	Comment
8c1bf5a	2017-03-15	MAPR-26499: The error when building tez-0.8 is now fixed.
1d6a461	2017-03-13	MAPR-26033: The issue that caused Hive action on Tez in Oozie to fail with exception is now fixed.
b7d98ac	2017-02-07	MAPR-26038: The version for all artifacts now include "-mapr" suffix.
9ef001b	2017-01-06	MAPR-25723: For DAG submission with already existing resource and mismatched sizes, you will get WARN message now instead of a runtime exception and the job will continue to run.
19b6014	2017-01-11	MAPR-25714: Tez will now use the correct dependency maprfs.jar file.

Known Issues and Limitations

Tez 0.8 is supported only for Hive 2.1 use cases.

Resolved Issues

None.

MapR Ecosystem Pack (EEP) Reference

This section contains links to information that is specific to a given EEP.

Note that the *MapR Ecosystem Pack (MEP)* has been renamed as the *Ezmeral Ecosystem Pack (EEP)*. For more information about MapR Data Platform terminology, see Documentation Enhancements in What's New in Version 6.1.0 on page 34.

EEP 8.1.0 Reference Information

This section contains links to release notes and other reference information for EEP 8.1.0.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 8.1.0 on page 4177
Listed are all Maven artifacts for EEP 8.1.0 components.

Related reference

MapR Ecosystem Pack 8.1.0 Release Notes on page 5658
This topic contains information about the components included in MapR Ecosystem Pack 8.1.0.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

EEP 8.1.0 Components and OS Support on page 5536
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 8.1.0 and shows the operating system support for each component.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 8.1.0
Summarizes the new features and product updates in MapR Ecosystem Pack (EEP) 8.1.0.

EEP 8.1.0 can be used with core 6.2.0 and core 7.0.0. For more information about EEP and core version support, see EEP Support and Lifecycle Status on page 5531.

EEP 8.1.0 Versions and Features

EEP 8.1.0 introduces a new component, Airflow, to the HPE Ezmeral MapR Ecosystem Pack and provides significant updates to Spark. Other components received minor updates. The following table summarizes the significant version updates in EEP 8.1.0:

Component	EEP 8.0.0 Version	EEP 8.1.0 Version
Airflow	N/A	2.2.1.0
Data Access Gateway	3.0.0.0	4.0.0.0
Spark	3.1.2	3.2.0

To compare the versions of various components in different EEPs, see Component Versions for Released EEPs on page 5586.

FIPS Support

When used with release 7.0.0, most EEP 8.1.0 components support the Federal Information Processing Standard (FIPS) 140-2 Level 1. The following table summarizes EEP 8.1.0 component support for FIPS:

EEP 8.1.0 Component	FIPS Support
Airflow	No
Data Access Gateway	Yes
Drill	Yes
HBase	Yes*
Hive	Yes
HTTPFS	Yes
Hue	No
Kafka REST	Yes
Kafka Schema Registry	Yes
Kafka Connect HDFS	Yes
Kafka Connect JDBC	Yes
KSQL	Yes
Kafka Streams	Yes
Livy	Yes**
Oozie	Yes

last-updated: Apr 11, 2022

EEP 8.1.0 Component	FIPS Support
Spark	Yes**
Tez	Yes
YARN	Yes

*HBase cannot be used in a mixed (FIPS and non-FIPS) configuration. For example, a non-FIPS client node cannot communicate with a FIPS server node.

**In a mixed (FIPS and non-FIPS) configuration, there is a known issue related to Spark and Livy applications when the Spark UI is enabled. See the Spark and Livy release notes.

For release note information, see the MapR Ecosystem Pack 8.1.0 Release Notes on page 5658.

Discontinued Components

S3 Gateway, Oozie, and Data Science Refinery (DSR) were added to the list of discontinued components. For more information, see Discontinued Ecosystem Components on page 5584.

Terminology Changes

Beginning with EEP 8.0.0, the MapR Data Platform product documentation includes the following terminology changes:

Old Name	New Name
Ecosystem Pack (MEP)[1]	Ezmeral Ecosystem Pack (EEP)
HPE Ezmeral Data Fabric XD Distributed File and Object Store	MapR XD Distributed File and Object Store
Object Store with S3-Compatible API	S3 Gateway

[1]In some areas, *MEP* continues to be used instead of EEP. To minimize issues for longtime MEP users, the package repository for released EEPs continues to use the *MEP* abbreviation in the directory names. See https://package.mapr.com/releases/MEP/. In addition, *MEP* remains in some documentation URLs to ensure that bookmarks and links to the URLs continue to work.

For more information about data-fabric terminology, see Documentation Enhancements in What's New in Version 6.1.0 on page 34.

Support for Ubuntu 18.04 and 20.04 (But Not Ubuntu 16.04)

EEP 8.1.0 can be used with core 6.2.0 on Ubuntu 18.04 and 20.04 but is not supported with core 6.2.0 on Ubuntu 16.04. For a list of the operating systems that each EEP can support, see EEP Components and OS Support on page 5536. For a list of the operating systems that different versions of core can support, see Operating System Support Matrix on page 5522.

Installer Support for EEP 8.1.0

Installer 1.17.0.x supports EEP 8.1.0 and previously released EEPs on core 6.2.0 only. Installer 1.17.0.x cannot be used on core 7.0.0. For a list of the EEPs that are supported by different versions of the Installer, see MapR Installer EEP Support on page 5602.

Installer 1.17.0.x cannot be used with older versions of Ubuntu. For more information, see Selecting an Installer Version to Use on page 5402.

last-updated: Apr 11, 2022

EEP Upgrades

If your cluster is currently running EEP 5.x or 6.x, you can upgrade to MapR Ecosystem Pack 6.3.5. If your cluster is running EEP 7.0.x or 7.1.x, you can upgrade to MapR Ecosystem Pack 7.1.1 or 8.x.x.

For information about upgrading EEPs, see:

- Checking the EEP Version on page 5413
- EEP Support and Lifecycle Status on page 5531
- Upgrading MapR Ecosystem Packs on page 335

For information about upgrading to core 7.0.0 and EEP 8.1.0, see:

- Installation Notes (Release 7.0.0)
- Upgrade Notes (Release 7.0.0)

EEP 8.1.0 Ecosystem Components and Release Notes

For a list of the EEP 8.1.0 components and their release notes, see MapR Ecosystem Pack 8.1.0 Release Notes on page 5658.

Version Change for Hive JAR Artifacts

Beginning with EEP 8.1.0, JAR artifacts for Hive use four digits instead of three digits. For more information, see Hive 2.3.9.0 - 2201 (EEP 8.1.0) Release Notes on page 5883.

Availability of EEP 6.3.6

EEP 6.3.6 was released at the same time as EEP 8.1.0 to provide defect repair for EEP 6.3.x users. For more information, see EEP 6.3.6 Reference Information on page 6522.

API Server and Web Server Packages for EEP 8.1.0

EEP 8.1.0 can be used with release 7.0.0 and with release 6.2.0. However, the API server (`mapr-apiserver`) and web server (`mapr-webserver`) packages that you must apply are different depending on the core release version. And the packages for release 7.0.0 and release 6.2.0 reside in different locations. Use the following table to determine which packages to use:

For release	Use the API server and web server packages in the . . .
7.0.0	Releases repository for core 7.0.0: `http://package.mapr.hpe.com/releases/v7.0.0/`
6.2.0	EEP repository for EEP 8.1.0: `http://package.mapr.hpe.com/releases/MEP/MEP-8.1.0/`

For more information about the API server and web server packages, see Setting Up the Control System on page 423.

Related concepts

EEP 8.1.0 Reference Information on page 6504
This section contains links to release notes and other reference information for EEP 8.1.0.

EEP 6.3.6 Reference Information on page 6522
This section contains links to release notes and other reference information for EEP 6.3.6.

EEP 8.x.y Ecosystem JDK / JRE Support
Summarizes JDK and JRE build and run information for EEP 8.x.y data-fabric ecosystem components.

The "Different from Open-Source Equivalent" column highlights that while some open-source components are built with JDK 8, the data-fabric component is built with JDK 11 and will only run on JRE 11.

Ecosystem Components	Built Using JDK Version	Runs on JRE or JDK Version	Different from Open-Source Equivalent?
AsyncHBase 1.8.2.0	8	8-11	Yes (OSS version compiles with Java 6)
Data Access Gateway 3.0.0.0	11	11	N/A
Drill 1.16.1.300	11	11	Yes
Flume 1.9.0.200	11	11	Yes
Hadoop 2.7.6.100	11	11	Yes
HBase 1.4.13.100	11	11	Yes
Hive 2.3.9	11	11	Yes
HttpFS 1.1.0.100	11	11	Yes
Hue 4.6.0.200	Not a Java component		
Kafka Rest 6.0.0.0	11	11	Yes
Kafka Schema Registry 6.0.0.0	11	11	Yes
Kafka Connect HDFS 10.0.0.0	11	11	Yes
Kafka Connect JDBC 10.0.1.0	11	11	Yes
KSQL 6.0.0.0	11	11	Yes
Livy 0.7.0.100	11	11	Yes
Oozie 5.2.1.100	11	11	No
Pig 0.17.0.100	11	11	Yes
S3 Gateway 2.2.0.0	Not a Java component		
Spark 3.1.2.0	11	11	Yes
Sqoop 1.4.7	11	11	No
Tez 0.9.2	11	11	Yes
MapR Monitoring Components			
Collectd 5.12.0.300	11	11	Yes
Elasticsearch 6.8.8.400	12	11	No
Fluentd 1.10.3.300	N/A	N/A	N/A
Grafana 7.5.10.300	N/A	N/A	N/A
Kibana 6.8.8.400	N/A	N/A	N/A
OpenTSDB 2.4.1.300	11	11	Yes

EEP 8.0.0 Reference Information

This section contains links to release notes and other reference information for EEP 8.0.0.

last-updated: Apr 11, 2022

 **Notice:** HPE recommends using EEP 8.1.0 instead of EEP 8.0.0. For more information about EEP 8.1.0, see EEP 8.1.0 Reference Information on page 6504.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 8.0.0 on page 4214
Listed are all Maven artifacts for EEP 8.0.0 components.

Related reference

MapR Ecosystem Pack 8.0.0 Release Notes on page 5660
This topic contains information about the components included in MapR Ecosystem Pack 8.0.0.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

EEP 8.0.0 Components and OS Support on page 5537
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 8.0.0 and shows the operating system support for each component.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 8.0.0
Summarizes the new features and product updates in MapR Ecosystem Pack (EEP) 8.0.0.

 **Notice:** HPE recommends using EEP 8.1.0 instead of EEP 8.0.0. For more information about EEP 8.1.0, see EEP 8.1.0 Reference Information on page 6504.

EEP 8.0.0 can be used only with core 6.2.0. For more information about EEP and core version support, see EEP Support and Lifecycle Status on page 5531.

EEP 8.0.0 Versions and Features

The following component versions changed significantly for EEP 8.0.0:

Component Group	Component	EEP 7.1.0 Version	EEP 8.0.0 Version
Hadoop	Hadoop	2.7.5.0	2.7.6.100
Kafka	Kafka Connect	5.1.2.200	10.0.0.0
	Kafka REST	5.1.2.200	6.0.0.0
	Kafka Schema Registry	5.1.2.200	6.0.0.0
	Kafka Streams	2.1.1.200	2.6.1.0
Monitoring	Collectd	5.10.0.0	5.12.0.300
	Grafana	7.5.2.200	7.5.10.300
	Open TSDB	2.4.0	2.4.1.300
S3 Gateway	S3 Gateway	2.1.0.0	2.2.0.0
Spark	Spark	2.4.7.100	3.1.2.0

To compare the versions of various components in different EEPs, see Component Versions for Released EEPs on page 5586.

Hadoop Updates

last-updated: Apr 11, 2022

Hadoop 2.7.6.100 includes numerous fixes and enhancements contained in the Apache Hadoop base release. For more information, see Hadoop 2.7.6.100 - 2110 (EEP 8.0.0) Release Notes on page 5836.

Kafka and Streams Updates

EEP 8.0.0 delivers the following Kafka and Streams improvements:

- Kafka and Streams
  - MirrorMaker 2 support.
  - Consumers no longer read any topic data when the Consumer application calls `consumer.poll` and the timeout (`request.timeout.ms`) is set to 0. Previously, Consumers read one message.
- Kafka Schema Registry
  - Support for JSON Schema and Protobuf formats in addition to Avro.
- Kafka REST Proxy
  - REST Proxy API v3 HTTP Methods and URIs support. For more information, see API v3 HTTP Methods and URIs on page 3890. Note that REST Proxy API v1 HTTP Methods and URIs are no longer supported in Kafka Rest 6.0.0.

For more information, see the Kafka release notes in MapR Ecosystem Pack 8.0.0 Release Notes on page 5660.

Monitoring (Collectd, Grafana, Open TSDB) Updates

Monitoring updates for EEP 8.0.0 keep the components up to date with recent open-source releases. Open TSDB added a fourth digit to its version to be consistent with other data-fabric component versions. For more information, see Monitoring Components - EEP 8.0.0 Release Notes on page 6235.

S3 Gateway

The S3 Gateway, formerly called the *Object Store with S3-Compatible API*, includes various MinIO and LDAP updates. For more information, see S3 Gateway 2.2.0.0 - 2110 (EEP 8.0.0) Release Notes on page 6260.

Spark Updates

- EEP 8.0.0 updates the Spark version to 3.x. For more information see Spark 3.1.2.0 - 2110 (EEP 8.0.0) Release Notes on page 6347 and Spark 3.1.2 Release Notes.
- Delta Lake support is available for Spark 3.1.2 on MapR Data Platform. See Apache Spark Feature Support on page 4027.
- For information about upgrading to Spark 3.x, see this page.

Terminology Changes

Beginning with EEP 8.0.0, the MapR Data Platform product documentation includes the following terminology changes:

Old Name	New Name
Ecosystem Pack (MEP)[1]	Ezmeral Ecosystem Pack (EEP)
HPE Ezmeral Data Fabric XD Distributed File and Object Store	MapR XD Distributed File and Object Store
Object Store with S3-Compatible API	S3 Gateway

[1]In some areas, *MEP* continues to be used instead of EEP. To minimize issues for longtime MEP users, the package repository for released EEPs continues to use the *MEP* abbreviation in the directory names.

last-updated: Apr 11, 2022

See https://package.mapr.com/releases/MEP/. In addition, *MEP* remains in some documentation URLs to ensure that bookmarks and links to the URLs continue to work.

For more information about data-fabric terminology, see Documentation Enhancements in What's New in Version 6.1.0 on page 34.

Deprecated and Discontinued Ecosystem Components

Pig, Flume, and Sqoop are now *deprecated*, meaning that these components will be updated only for critical security flaws and will be discontinued within six months.

Impala and Sentry are *discontinued*, meaning that they are removed from EEP 8.0.0 and later for core 6.2.0. No maintenance is provided for discontinued components.

For more information, see Discontinued Ecosystem Components on page 5584 and Understand the EEP Lifecycle on page 5528.

Maven Artifact Version String

Beginning with EEP 8.0.0, *eep* replaces *mapr* in the Maven artifact version string. For example:

Old Version String

```
<groupId>org.apache.hive</groupId>
<artifactId>hive</artifactId>
<version>2.3.8-mapr-2104</version>
```

New Version String

```
<groupId>org.apache.hive</groupId>
<artifactId>hive</artifactId>
<version>2.3.9-eep-2110</version>
```

This change applies to EEP 8.0.0 and later EEPs and does not apply to previously published Maven artifacts.

Support for Ubuntu 18.04 (But Not Ubuntu 16.04)

EEP 8.0.0 can be used with core 6.2.0 on Ubuntu 18.04 but is not supported with core 6.2.0 on Ubuntu 16.04. For a list of the operating systems that each EEP can support, see EEP Components and OS Support on page 5536. For a list of the operating systems that different versions of core can support, see Operating System Support Matrix on page 5522.

Maintenance EEPs

At the release of EEP 8.0.0, EEPs 7.1.1 and 6.3.5 were released as maintenance EEPs. Maintenance EEPs provide defect repair and an upgrade path for previously released EEPs.

EEP 7.1.1 is identical to EEP 7.1.0 except for changes to the monitoring (Spyglass) components. For a list of monitoring fixes in EEP 7.1.1, see Monitoring Components - EEP 7.1.1 Release Notes on page 6236. To compare ecosystem component versions, see Component Versions for Released EEPs on page 5586. For reference information about specific EEPs, see MapR Ecosystem Pack (EEP) Reference on page 6504.

Installer Support for EEP 8.0.0

Installer 1.17.0.0 supports EEP 8.0.0 and maintenance EEPs 7.1.1 and 6.3.5, as well as previously released EEPs. For a list of the EEPs that are supported by different versions of the Installer, see MapR Installer EEP Support on page 5602.

Installer 1.17.0.0 cannot be used with older versions of Ubuntu. For more information, see Selecting an Installer Version to Use on page 5402.

last-updated: Apr 11, 2022

EEP Upgrades

The EEP 8.0.0 release includes EEP 8.0.0, EEP 7.1.1, and EEP 6.3.5, but no other EEP revisions. If your cluster is currently running EEP 5.x or 6.x, you can upgrade to MapR Ecosystem Pack 6.3.5. If your cluster is running EEP 7.0.x or 7.1.x, you can upgrade to MapR Ecosystem Pack 7.1.1 or 8.0.0.

For more information about upgrading EEPs, see:

- Checking the EEP Version on page 5413
- EEP Support and Lifecycle Status on page 5531
- Upgrading MapR Ecosystem Packs on page 335

For information about upgrading to core 6.2.0 and EEP 8.0.0, see Installation and Upgrade Notes (MapR 6.1.0) on page 39.

EEP 8.0.0 Ecosystem Components and Release Notes

For a list of the EEP 8.0.0 components and their release notes, see MapR Ecosystem Pack 8.0.0 Release Notes on page 5660.

Related concepts

EEP 8.0.0 Reference Information on page 6508
This section contains links to release notes and other reference information for EEP 8.0.0.

EEP 7.1.1 Reference Information

This section contains links to release notes and other reference information for EEP 7.1.1.

EEP 7.1.1 is identical to EEP 7.1.0 except for changes to the monitoring (Spyglass) components. For a list of monitoring fixes in EEP 7.1.1, see Monitoring Components - EEP 7.1.1 Release Notes on page 6236.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 7.1.1 on page 4233
Maven artifacts for EEP 7.1.1 are unchanged from the Maven artifacts for EEP 7.1.0.

EEP 7.x.y Ecosystem JDK / JRE Support on page 6521
Summarizes JDK and JRE build and run information for EEP 7.x.y data-fabric ecosystem components.

Related reference

MapR Ecosystem Pack 7.1.1 Release Notes on page 5662
This topic contains information about the components included in MapR Ecosystem Pack 7.1.1.

EEP 7.1.1 Components and OS Support on page 5538
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 7.1.1 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 7.1.0 Reference Information

This section contains links to release notes and other reference information for EEP 7.1.0.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 7.1.0 on page 4233
Listed are all Maven artifacts for EEP 7.1.0 components.

EEP 7.x.y Ecosystem JDK / JRE Support on page 6521
Summarizes JDK and JRE build and run information for EEP 7.x.y data-fabric ecosystem components.

Related reference

MapR Ecosystem Pack 7.1.0 Release Notes on page 5664
This topic contains information about the components included in MapR Ecosystem Pack 7.1.0.

EEP 7.1.0 Components and OS Support on page 5539
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 7.1.0 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 7.1.0

Summarizes the new features in MapR Ecosystem Pack (EEP) 7.1.0.

EEP 7.1.0 provides defect repair and new features as described later on this page.

EEP 7.1.0 can be used with core 6.2.0. For more information about EEP and core version support, see EEP Support and Lifecycle Status on page 5531.

SLES Support

EEP 7.1.0 is supported on SLES 15 SP2 and core 6.2.0. For the certified operating systems, see Operating System Support Matrix on page 5522.

Control System Packages

Beginning with EEP 7.1.0, the `mapr-apiserver` and `mapr-webserver` packages are available in the EEP repository. For users of earlier EEPs, such as EEP 7.0.1, older versions of these packages continue to be available in the core 6.2.0.0 repository. However, users of EEP 7.1.0 on SLES 15 SP2 must obtain the `mapr-apiserver` and `mapr-webserver` packages from the EEP 7.1.0 repository because the core 6.2.0.7 repository does not contain the packages. In the event that the packages are present in both the core and EEP repositories that you have enabled for a node, the package manager will install the latest version of each package.

For Control System installation information, see Setting Up the Control System on page 423.

ObjectStore 2.1.0 Support

EEP 7.1.0 supports ObjectStore 2.1.0 on core 6.2.0.

Service Verifier

Certain ecosystem components in EEP 7.1.0 include a service verifier. See Using Service Verification on page 5454 for additional information.

last-updated: Apr 11, 2022

Installer Support for EEP 7.1.0

Installer 1.16.0.0 supports EEP 7.1.0 and maintenance EEPs 6.3.4 and 5.0.7, as well as previously released EEPs. For more information, see Release History for EEPs on page 5623 and MapR Installer Updates on page 5481.

EEP Upgrades

The EEP 7.1.0 release includes EEP 6.3.4 and EEP 5.0.7, but no other EEP revisions. If your cluster is currently running EEP 5.x or 6.x, you can upgrade to EEP 5.0.7 or EEP 6.3.4. If your cluster is running EEP 7.0.x, you can upgrade to EEP 7.1.0.

For more information about upgrading, see:

- Checking the EEP Version on page 5413
- EEP Support and Lifecycle Status on page 5531
- Upgrading MapR Ecosystem Packs on page 335

EEP 7.1.0 Ecosystem Components

This section lists the components that have been updated for EEP 7.1.0.

Drill 1.16.1.200	EEP 7.1.0 includes support for Drill 1.16.1.200. For a list of updates, see Drill 1.16.1.200-2104 (MEP 7.1.0) Release Notes.
Hadoop 2.7.5.0	EEP 7.1.0 includes support for Hadoop 2.7.5.0. For a list of updates, see Hadoop 2.7.5.0 - 2104 (MEP 7.1.0) Release Notes.
HBase 1.4.13.0	EEP 7.1.0 includes support for Apache HBase 1.4.13.0. For information about new features, patches, and known issues, see HBase 1.4.13.0 - 2104 (MEP 7.1.0) Release Notes. For HBase documentation, see HBase on page 3382.
Hive 2.3.8	EEP 7.1.0 includes support for Hive 2.3.8. For a list of new features and updates, see Hive 2.3.8 - 2104 (MEP 7.1.0) Release Notes.
HTTPFS 1.1.0.0	EEP 7.1.0 includes support for HTTPFS 1.1.0.0. For a list of new features and updates, see HttpFS 1.1.0.0 - 2104 (MEP 7.1.0) Release Notes.
Hue 4.6.0.0	EEP 7.1.0 includes support for Hue 4.6.0.0. For a list of new features and updates, see Hue 4.6.0.0 - 2104 (MEP 7.1.0) Release Notes.
Impala 2.12.0.500	EEP 7.1.0 includes support for Impala 2.12.0.500. The `mapr-impala` packages display a new version and provide support for SLES 15 SP2, but are otherwise unchanged from EEP 7.0.1. For a list of the EEP 7.0.1 features and updates, see Impala 2.12.0.400 - 2101 (EEP 7.0.1) Release Notes on page 6139.

last-updated: Apr 11, 2022

Livy 0.7.0.0	EEP 7.1.0 includes support for Livy Livy 0.7.0.0. For a list of new features and updates, see Livy 0.7.0.0 - 2104 (MEP 7.1.0) Release Notes.
Kafka Streams 2.1.1.200	EEP 7.1.0 includes support for Kafka Streams 2.1.1.200. For a list of updates, see Kafka Streams 2.1.1.200 - 2104 (MEP 7.1.0) Release Notes.
KSQL 5.1.2.200	EEP 7.1.0 includes support for KSQL 5.1.2.200. For a list of updates, see KSQL 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes.
Kafka Connect HDFS 5.1.2.200	EEP 7.1.0 includes support for Kafka Connect HDFS 5.1.2.200. For a list of updates, see Kafka Connect HDFS 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes.
Kafka REST Proxy 5.1.2.200	EEP 7.1.0 includes support for Kafka REST Proxy 5.1.2.200. For a list of updates, see Kafka REST Proxy 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes.
Kafka Schema Registry 5.1.2.200	EEP 7.1.0 includes support for Kafka Schema Registry 5.1.2.200. For a list of updates, see Kafka Schema Registry 5.1.2.200 - 2104 (MEP 7.1.0) Release Notes.
Monitoring	EEP 7.1.0 includes support for new versions of the monitoring components (Collectd, Elasticsearch, Fluentd, Grafana, Kibana, and OpenTSDB). For release-note information, see Monitoring Components - MEP 7.1.0 Release Notes.
Object Store with S3 Compatible API 2.1.0.0	EEP 7.1.0 includes support for Object Store with S3 Compatible API 2.1.0.0. For release-note information, see Object Store with S3-Compatible API 2.1.0.0 - 2104 (MEP 7.1.0) Release Notes.
Oozie 5.2.1.0	EEP 7.1.0 includes support for Oozie 5.2.1.0. For a list of updates, see Oozie 5.2.1.0 - 2104 (MEP 7.1.0) Release Notes.
Spark 2.4.7.100	EEP 7.1.0 includes support for Spark 2.4.7.100. For a list of new features and updates, see Spark 2.4.7.100 - 2104 (MEP 7.1.0) Release Notes.
Sqoop 1.4.7	EEP 7.1.0 includes support for Sqoop 1.4.7. For a list of updates, see Sqoop 1.4.7 - 2104 (MEP 7.1.0) Release Notes.
Tez 0.9.2	EEP 7.1.0 includes support for Tez 0.9.2. For a list of updates, see Tez 0.9.2 - 2104 (MEP 7.1.0) Release Notes.

last-updated: Apr 11, 2022

EEP 7.0.1 Reference Information

This section contains links to release notes and other reference information for EEP 7.0.1.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 7.0.1 on page 4272
Listed are all Maven artifacts for EEP 7.0.1 components.

EEP 7.x.y Ecosystem JDK / JRE Support on page 6521
Summarizes JDK and JRE build and run information for EEP 7.x.y data-fabric ecosystem components.

Related reference

MapR Ecosystem Pack 7.0.1 Release Notes on page 5666
This topic contains information about the components included in MapR Ecosystem Pack 7.0.1.

EEP 7.0.1 Components and OS Support on page 5540
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 7.0.1 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 7.0.1

Summarizes the new features in MapR Ecosystem Pack (EEP) 7.0.1.

EEP 7.0.1 provides defect repair and new features as described later on this page.

EEP 7.0.1 can be used with core 6.2.0. For more information about EEP and core version support, see EEP Support and Lifecycle Status on page 5531.

No SLES Support

EEP 7.0.x is not supported on SLES because release 6.2.0 is not currently certified on a SLES release. For the certified operating systems, see Operating System Support Matrix on page 5522.

Installer Support for EEP 7.0.1

Installer 1.15.0.0 supports EEP 7.0.1 and maintenance EEPs 6.3.2 and 5.0.6, as well as previously released EEPs. For more information, see Release History for EEPs on page 5623 and MapR Installer Updates on page 5481.

EEP Upgrades

The EEP 7.0.1 release includes EEP 6.3.2 and EEP 5.0.6, but no other EEP revisions. If your cluster is currently running EEP 5.x or 6.x, you can upgrade to EEP 5.0.6 or EEP 6.3.2. If your cluster is running EEP 7.0.0, you can upgrade to EEP 7.0.1. However, no upgrades to core 6.2.0 are currently supported.

For more information about upgrading, see:

- Checking the EEP Version on page 5413
- EEP Support and Lifecycle Status on page 5531
- Upgrading MapR Ecosystem Packs on page 335

EEP 7.0.1 Ecosystem Components

This section lists the components that have been updated for EEP 7.0.1.

Drill 1.16.1.100	EEP 7.0.1 includes support for Drill 1.16.1.100. For a list of updates, see Drill 1.16.1.100-2101 (EEP 7.0.1) Release Notes on page 5751.
Flume 1.9.0.100	EEP 7.0.1 includes support for Flume 1.9.0.100. For a list of updates, see Flume 1.9.0.100-2101 (EEP 7.0.1) Release Notes on page 5820.
HBase 1.4.12.100	EEP 7.0.1 includes support for Apache HBase 1.4.12.100. For information about new features, patches, and known issues, see HBase 1.4.12.100 - 2101 (EEP 7.0.1) Release Notes on page 5860. For HBase documentation, see HBase on page 3382.
Hive 2.3.7	EEP 7.0.1 includes support for Hive 2.3.7. For a list of new features and updates, see Hive 2.3.7 - 2101 (EEP 7.0.1) Release Notes on page 5902.
HTTPFS 1.0	EEP 7.0.1 includes support for HTTPFS 1.0. For a list of new features and updates, see HttpFS 1.0 - 2101 (EEP 7.0.1) Release Notes on page 6054.
Impala 2.12.0.400	EEP 7.0.1 includes support for Impala 2.12.0.400. For a list of updates, see Impala 2.12.0.400 - 2101 (EEP 7.0.1) Release Notes on page 6139.
Kafka Schema Registry 5.1.2.100	EEP 7.0.1 includes support for Kafka Schema Registry 5.1.2.100. For a list of updates, see Kafka Schema Registry 5.1.2.100 - 2101 (MEP 7.0.1) Release Notes on page 6231.
KSQL 5.1.2.100	EEP 7.0.1 includes support for KSQL 5.1.2.100. For a list of updates, see KSQL 5.1.2.100 - 2101 (MEP 7.0.1) Release Notes on page 6187.
Monitoring	EEP 7.0.1 includes support for new versions of the monitoring components (Collectd, Elasticsearch, Fluentd, Grafana, Kibana, and OpenTSDB). For release-note information, see MapR Monitoring Components - EEP 7.0.1 Release Notes on page 6239.
Oozie 5.2.0.100	EEP 7.0.1 includes support for Oozie 5.2.0.100. For a list of updates, see Oozie 5.2.0.100 - 2101 (EEP 7.0.1) Release Notes on page 6272.
Sentry 1.7.0	EEP 7.0.1 includes support for Sentry 1.7.0. For a list of updates, see Sentry 1.7.0 - 2101 (MEP 7.0.1) Release Notes on page 6330.

last-updated: Apr 11, 2022

Spark 2.4.7.0	EEP 7.0.1 includes support for Spark 2.4.7.0. For a list of new features and updates, see Spark 2.4.7.0 - 2101 (EEP 7.0.1) Release Notes on page 6353.
Sqoop 1.4.7	EEP 7.0.1 includes support for Sqoop 1.4.7. For a list of updates, see Sqoop 1.4.7 - 2101 (EEP 7.0.1) Release Notes on page 6451.
Tez 0.9.2	EEP 7.0.1 includes support for Tez 0.9.2. For a list of updates, see Tez 0.9.2 - 2101 (EEP 7.0.1) Release Notes on page 6485.

EEP 7.0.0 Reference Information

This section contains links to release notes and other reference information for EEP 7.0.0.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 7.0.0 on page 4311
Listed are all Maven artifacts for EEP 7.0.0 components.

Related reference

MapR Ecosystem Pack 7.0.0 Release Notes on page 5668
This topic contains information about the components included in MapR Ecosystem Pack 7.0.0.

EEP 7.0.0 Components and OS Support on page 5541
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 7.0.0 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 7.0.0

Summarizes the new features in MapR Ecosystem Pack (EEP) 7.0.0.

EEP 7.0.0 provides defect repair and new features as described later on this page.

EEP 7.0.0 can be used with core 6.2.0. For more information about EEP and core version support, see EEP Support and Lifecycle Status on page 5531.

Core Support for Red Hat / CentOS 8.1 and Ubuntu 16.04 and 18.04

The EEP 7.0.0 release adds support for new versions of Red Hat / CentOS and Ubuntu. For a complete list of supported operating systems, see Operating System Support Matrix on page 5522.

Installer Support for EEP 7.0.0

Installer 1.14.0.0 supports EEP 7.0.0 and maintenance EEPs 6.3.1 and 5.0.5, as well as previously released EEPs. For more information, see Release History for EEPs on page 5623 and MapR Installer Updates on page 5481.

EEP Upgrades

The EEP 7.0.0 release includes EEP 6.3.1 and EEP 5.0.5, but no other EEP revisions. If your cluster is currently running EEP 5.x or 6.x, you can upgrade to EEP 5.0.5 or EEP 6.3.1. However, no upgrades to core 6.2.0 or EEP 7.0.0 are currently supported.

For more information about upgrading, see:

- Checking the EEP Version on page 5413
- EEP Support and Lifecycle Status on page 5531
- Upgrading MapR Ecosystem Packs on page 335

EEP 7.0.0 Ecosystem Components

AsynchHBase 1.8.2	EEP 7.0.0 includes support for AsynchHBase 1.8.2. For release-note information, see AsyncHBase 1.8.2-2009 Release Notes on page 5739.
Data Access Gateway3.0.0.0	EEP 7.0.0 includes support for data Access Gateway 3.0.0.0. For release-note information, see Data Access Gateway 3.0 Release Notes on page 5744.
Drill 1.16.1	EEP 7.0.0 includes support for Drill 1.16.1. For a list of updates, see Drill 1.16.1.0-2009 (EEP 7.0.0) Release Notes on page 5753.
Flume 1.9.0.0	EEP 7.0.0 includes support for Flume 1.9.0.0. For a list of updates, see Flume 1.9.0.0-2009 Release Notes on page 5821.
Hadoop 2.7.4.0	Beginning with release 6.2.0 and EEP 7.0.0, Hadoop and YARN are removed from core and provided as a collection of ecosystem components. For release note information, see Hadoop 2.7.4.0-2009 (EEP 7.0.0) Release Notes on page 5843.
HBase 1.4.12.0	EEP 7.0.0 includes support for Apache HBase 1.4.12.0. For information about new features, patches, and known issues, see HBase 1.4.12.0-2009 (EEP 7.0.0) Release Notes on page 5861. For HBase documentation, see HBase on page 3382.
Hive 2.3	EEP 7.0.0 includes support for Hive 2.3. For a list of new features and updates, see Hive 2.3.7-2009 (EEP 7.0.0) Release Notes on page 5906.
Hue 4.6.0.0 with Livy 0.5.0	EEP 7.0.0 includes support for Hue 4.6.0.0 with Livy 0.5.0. For a list of new features and updates, see Hue 4.6.0.0 - 2009 (EEP 7.0.0) Release Notes on page 6075.

last-updated: Apr 11, 2022

Impala 2.12.0.200	EEP 7.0.0 includes support for Impala 2.12.0.200. For a list of updates, see Impala 2.12.0.200 - 2009 Release Notes on page 6140.
Kafka Schema Registry 5.1.2.0	Kafka Schema Registry 5.1.2.0 is released for general availability. For a list of updates, see Kafka Schema Registry 5.1.2.0 - 2009 (MEP 7.0.0) Release Notes on page 6232.
KSQL 5.1.2.0	EEP 7.0.0 includes support for KSQL 5.1.2.0. This release includes the following security enhancements: • You can specify a custom SSL truststore file location through properties set from the command line or in the `ksql-server.properties` file. • Authentication, encryption (SSL/TLS), and impersonation are supported between the KSQL client and KSQL server. • For non-secure clusters, impersonation and authentication are disabled for Kafka REST and Kafka Connect, by default. To configure the custom truststore, see: KSQL Security on page 3845. For a list of updates, see KSQL 5.1.2.0 - 2009 (MEP 7.0.0) Release Notes on page 6188.
Monitoring	EEP 7.0.0 includes support for new versions of the monitoring components (Collectd, Elasticsearch, Fluentd, Grafana, Kibana, and OpenTSDB). In addition, new certificates added to release 6.2.0 simplify the installation of log monitoring. For a summary of the installation changes, see Installation and Upgrade Notes (MapR 6.1.0) on page 39. For release-note information, see MapR Monitoring Components - EEP 7.0.0 Release Notes on page 6240.
S3 Gateway 2.0.0.0	EEP 7.0.0 includes support for the 2.0.0.0 release of the S3 Gateway. For a list of new features, see S3 Gateway 2.0.0.0 - 2009 (EEP 7.0.0) Release Notes on page 6262.
Oozie 5.2.0.0	EEP 7.0.0 includes support for Oozie 5.2.0.0. For a list of updates, see Oozie 5.2.0.0 - 2009 (EEP 7.0.0) Release Notes on page 6274.
Pig 0.17.0.0	EEP 7.0.0 includes support for Pig 0.17.0.0. For a list of updates, see Pig 0.17.0.0 - (EEP 7.0.0) 2009 Release Notes on page 6315.
Sentry 1.7.0	EEP 7.0.0 includes support for Sentry 1.7.0. For a list of updates, see Sentry 1.7.0 - 2009 (MEP 7.0.0) Release Notes on page 6331.
Spark 2.4.5	EEP 7.0.0 includes support for Spark 2.4.5.0.

last-updated: Apr 11, 2022

	For a list of new features and updates, see Spark 2.4.5-2009 (EEP 7.0.0) Release Notes on page 6355.
Sqoop 1.4.7	EEP 7.0.0 includes support for Sqoop 1.4.7. For a list of updates, see Sqoop 1.4.7 - 2009 (EEP 7.0.0) Release Notes on page 6452.

Deprecated Components and Features

Metering	EEP 7.0.0 does not support metering.
Myriad	Myriad is not supported in EEP 7.0.0 and later. References to Myriad have been removed from the release 6.2.0 documentation.
Sqoop2	Sqoop2 is not supported in EEP 7.0.0 and later. References to Sqoop2 have been removed from the release 6.2.0 documentation.
Sqoop TDCH	The Connector for Teradata (Sqoop TDCH) is not supported in EEP 7.0.0 and later. References to the connector have been removed from the release 6.2.0 documentation.

EEP 7.x.y Ecosystem JDK / JRE Support

Summarizes JDK and JRE build and run information for EEP 7.x.y data-fabric ecosystem components.

The "Different from Open-Source Equivalent" column highlights that while some open-source components are built with JDK 8, the data-fabric component is built with JDK 11 and will only run on JRE 11.

Ecosystem Components	Built Using JDK Version	Runs on JRE or JDK Version	Different from Open-Source Equivalent?
AsyncHBase 1.8.2.0	8	8-11	Yes (OSS version compiles with Java 6)
Drill 1.16.1.0	11	11	Yes
Flume 1.9.0.0	11	11	Yes
HBase 1.4.12.0	11	11	Yes
Hive 2.3.7	11	11	Yes
HttpFS 1.0	11	11	Yes
Hue 4.6.0.0	Not a Java component		
Impala 2.12.0.200	Not a Java component		
KSQL 5.1.2.0	11	11	Yes
Kafka Rest 5.1.2	11	11	Yes
Schema Registry 5.1.2	11	11	Yes
Kafka Connect HDFS/ JDBC	11	11	Yes
MapR Data Access Gateway 3.0	11	11	N/A
MapR Object Store 2.0.0.0	Not a Java component		
Oozie 5.2.0.0	11	11	No

last-updated: Apr 11, 2022

Ecosystem Components	Built Using JDK Version	Runs on JRE or JDK Version	Different from Open-Source Equivalent?
Pig 0.17.0.0	11	11	Yes
Sentry 1.7.x	11	11	Yes
Spark 2.4.5.0	11	11	Yes
Sqoop 1.4.7	11	11	No
Hadoop 2.7.4.0	11	11	Yes
Livy 0.5.0	11	11	Yes
Tez 0.9.2	11	11	Yes
MapR Monitoring Components			
Collectd 5.10.0.0	11	11	Yes
Elasticsearch 6.8.8.0	12	11	No
Fluentd 1.10.3.0	N/A	N/A	N/A
Grafana 6.7.2.0	N/A	N/A	N/A
Kibana 6.8.8.0	N/A	N/A	N/A
OpenTSDB 2.4.0.x	11	11	Yes

EEP 6.3.6 Reference Information

This section contains links to release notes and other reference information for EEP 6.3.6.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 6.3.6 on page 4351
Listed are all Maven artifacts for EEP 6.3.6 components.

Related reference

MapR Ecosystem Pack 6.3.6 Release Notes on page 5670
This topic contains information about the components included in MapR Ecosystem Pack 6.3.6.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

EEP 6.3.6 Components and OS Support on page 5542
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.3.6 and shows the operating system support for each component.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 6.3.5 Reference Information

This section contains links to release notes and other reference information for EEP 6.3.5.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 6.3.5 on page 4380
Listed are all Maven artifacts for EEP 6.3.5 components.

Related reference

MapR Ecosystem Pack 6.3.5 Release Notes on page 5673
This topic contains information about the components included in MapR Ecosystem Pack 6.3.5.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

EEP 6.3.5 Components and OS Support on page 5544
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.3.5 and shows the operating system support for each component.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 6.3.4 Reference Information

This section contains links to release notes and other reference information for EEP 6.3.4.

CAUTION: MEP 6.3.4 requires a core patch to resolve a Warden defect. The defect is fixed in `mapr-patch-6.1.0.20180926230239.GA-20210512163609.x86_64` and later. Before upgrading to MEP 6.3.4, you must apply the patch. See Patches and Documentation for MapR 6.1.1 on page 96.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 6.3.4 on page 4404
Listed are all Maven artifacts for EEP 6.3.4 components.

Related reference

MapR Ecosystem Pack 6.3.4 Release Notes on page 5675
This topic contains information about the components included in MapR Ecosystem Pack 6.3.4.

EEP 6.3.4 Components and OS Support on page 5545
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.3.4 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 6.3.3 Reference Information

This section contains links to release notes and other reference information for EEP 6.3.3.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 6.3.3 on page 4431
Listed are all Maven artifacts for EEP 6.3.3 components.

Related reference

MapR Ecosystem Pack 6.3.3 Release Notes on page 5677
This topic contains information about the components included in MapR Ecosystem Pack 6.3.3.

EEP 6.3.3 Components and OS Support on page 5546

last-updated: Apr 11, 2022

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.3.3 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 6.3.3
Summarizes the new features in MapR Ecosystem Pack (EEP) 6.3.3.

EEP 6.3.3 provides defect repair but no new features.

EEP 6.3.3 can be used with MapR core 6.1.1 and with 6.1.0 if the latest patches are applied to core 6.1.0. For more information about EEP and MapR core version support, see EEP Support and Lifecycle Status on page 5531.

MapR Support for SUSE 12 SP5

The EEP 6.3.3 release adds MapR core support for SUSE 12 SP5. For a complete list of supported operating systems, see Operating System Support Matrix on page 5522.

EEP 6.3.3 Ecosystem Components

Except for changes to the monitoring components, EEP 6.3.3 components are unchanged from EEP 6.3.2. The monitoring components and librdkafka have been natively compiled in SUSE 12 to work with SUSE 12 SP5. For monitoring changes, see:

- MapR Monitoring Components - EEP 6.3.3 Release Notes on page 6245

MapR Installer Support for EEP 6.3.0	Mapr Installer 1.15.0.1 supports EEP 6.3.3 and release 6.1.1. For more information, see MapR Installer Updates on page 5481.

EEP 6.3.2 Reference Information

This section contains links to release notes and other reference information for EEP 6.3.2.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 6.3.2 on page 4431
Listed are all Maven artifacts for EEP 6.3.2 components.

Related reference

MapR Ecosystem Pack 6.3.2 Release Notes on page 5679
This topic contains information about the components included in MapR Ecosystem Pack 6.3.2.

EEP 6.3.2 Components and OS Support on page 5547
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.3.2 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

last-updated: Apr 11, 2022

EEP 6.3.1 Reference Information

This section contains links to release notes and other reference information for EEP 6.3.1.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 6.3.1 on page 4459
Listed are all Maven artifacts for EEP 6.3.1 components.

Related reference

MapR Ecosystem Pack 6.3.1 Release Notes on page 5681
This topic contains information about the components included in MapR Ecosystem Pack 6.3.1.

EEP 6.3.1 Components and OS Support on page 5548
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.3.1 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 6.3.0 Reference Information

This section contains links to release notes and other reference information for EEP 6.3.0.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 6.3.0 on page 4487
Listed are all Maven artifacts for EEP 6.3.0 components.

Related reference

MapR Ecosystem Pack 6.3.0 Release Notes on page 5683
This topic contains information about the components included in MapR Ecosystem Pack 6.3.0.

EEP 6.3.0 Components and OS Support on page 5549
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.3.0 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 6.3.0

Summarizes the new features in MapR Ecosystem Pack (EEP) 6.3.0.

EEP 6.3.0 provides defect repair and new features as described later on this page.

EEP 6.x can be used with MapR Core 6.1.0. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

MapR Core Support for Red Hat / CentOS 7.7 and Ubuntu 18.04

The EEP 6.3.0 release adds MapR Core support for new versions of Red Hat / CentOS and Ubuntu. For a complete list of supported operating systems, see Operating System Support Matrix on page 5522.

last-updated: Apr 11, 2022

Upgrades Recommended for EEP 4.x and EEP 6.x Installations

The EEP 6.3.0 release includes EEP 6.3.0 and EEP 5.0.4, but no other EEP revisions. Specifically, the EEP 6.3.0 release does not include updates for the following EEPs:

- EEP 4.0.x on MapR 6.0.1
- EEP 4.1.x on MapR 6.0.1
- EEP 6.0.x on MapR 6.1.0
- EEP 6.1.x on MapR 6.1.0
- EEP 6.2.x on MapR 6.1.0

If your cluster is currently running one of these EEPs and you want to upgrade to obtain new features or defect repair, you must upgrade to EEP 5.0.4 on MapR 6.01 or EEP 6.3.0 on MapR 6.1.0. For more information about upgrading, see:

- Checking the EEP Version on page 5413
- EEP Support and Lifecycle Status on page 5531
- Upgrading MapR Ecosystem Packs on page 335

Java OJAI Thin Client

EEP 6.3.0 adds a Java OJAI "Thin" Client to the list of lightweight client applications that use the MapR Data Access Gateway to send requests to the MapR data platform. These client applications, which include C#, Go, Python, and Node.js, provide lightweight libraries that support the OJAI API and function as an alternative to the Java OJAI client.

For more information, see these topics:

- Using the Java OJAI Thin Client on page 2670
- Understanding the MapR Data Access Gateway on page 750
- MapR Data Access Gateway 2.0 Release Notes on page 5745

EEP 6.3.0 Ecosystem Components

HBase Support	EEP 6.3.0 includes support for Apache HBase 1.1.13. Previously, the last HBase version supported with MapR Core was HBase 1.1.8. For information about new features, patches, and known issues, see HBase 1.1.13.0-1912 (EEP 6.3.0) Release Notes on page 5871. For Apache HBase new-feature and bug-fix information, see these Apache pages: • What's new in HBase 1.1.9 • What's new in HBase 1.1.10 • What's new in HBase 1.1.11 • What's new in HBase 1.1.12 • What's new in HBase 1.1.13 For upgrade considerations, see Considerations for Upgrading to HBase 1.1.13 on page 337.

	To configure MapR-SASL security, see Configure MapR-SASL Security (Authentication and Encryption) for HBase on page 3382. For MapR HBase documentation, see HBase on page 3382.
Drill 1.16.0.10	EEP 6.3.0 includes support for Drill 1.16.0.10. For a list of updates, see Drill 1.16.0.10-1912 (EEP 6.3.0) Release Notes on page 5762.
Hive 2.3.6	EEP 6.3.0 includes support for Hive 2.3.6. For a list of new features and updates, see Hive 2.3.6-1912 (EEP 6.3.0) Release Notes on page 5936.
Hue 4.3.0 with Livy 0.5.0	EEP 6.3.0 includes support for Hue 4.3.0 with Livy 0.5.0. For a list of new features and updates, see Hue 4.3.0.100-1912 (EEP 6.3.0) Release Notes on page 6085.
Impala 2.12.0.100	EEP 6.3.0 includes support for Impala 2.12.0.100. For a list of updates, see Impala 2.12.0.100-1912 Release Notes on page 6141.
Oozie 5.1.0.300	EEP 6.3.0 includes support for Oozie 5.1.0.300. For a list of updates, see Oozie 5.1.0.300-1912 (EEP 6.3.0) Release Notes on page 6281.
Spark 2.4.4.0	EEP 6.3.0 includes support for Spark 2.4.4.0. For a list of new features and updates, see Spark 2.4.4.0-1912 Release Notes on page 6368.
MapR Installer Support for EEP 6.3.0	Mapr Installer 1.13.0.0 supports EEP 6.3.0 and maintenance EEP 5.0.4, as well as previously released EEPs. For more information, see Release History for EEPs on page 5623 and MapR Installer Updates on page 5481.

EEP 6.2.0 Reference Information

This section contains links to release notes and other reference information for EEP 6.2.0.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 6.2.0 on page 4515
Listed are all Maven artifacts for EEP 6.2.0 components.

Related reference

MapR Ecosystem Pack 6.2.0 Release Notes on page 5684
This topic contains information about the components included with the MapR Ecosystem Pack 6.2.0.

EEP 6.2.0 Components and OS Support on page 5550
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.2.0 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 6.2.0
Provides a summary of the new functionality in MapR Ecosystem Pack (EEP) 6.2.0.

EEP 6.2.0 provides support for the following new features.

EEP 6.x can be used with MapR Core 6.1.0. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

EEP 6.2.0 Ecosystem Components

Hive Updates	EEP 6.2.0 includes support for MapR Database JSON projection push down without secondary indexes. The metrics report file `/tmp/hive_report.json` is split: `/tmp/hiveserver2_report.json` and `/tmp/hivemetastore_report.json` for HiveServer2 and Hive Metastore, respectively.
New Hue Version	EEP 6.2.0 includes support for Hue 4.3.0. For more information, see Hue 4.3.0-1904 (EEP 6.2.0) Release Notes on page 6087.
Oozie Updates	EEP 6.2.0 includes Oozie support for Spark artifacts version 2.4.0. Jetty version is upgraded to 9.3.25.v20180904. For more information, see Oozie 5.1.0.200-1904 (EEP 6.2.0) Release Notes on page 6282.
New Spark Version	EEP 6.2.0 includes support for Spark 2.4.0.0. For more information, see Spark 2.4.0.0-1904 (EEP 6.2.0) Release Notes on page 6370.
Drill 1.16.0.0	EEP 6.2.0 includes support for Drill 1.16.0.0. Drill 1.16.0.0 includes several new features, including new SQL commands that generate table statistics and define schema for text files, multiple Drill Web UI enhancements, and format plugins for LTSV files and SYSLOG. For a complete list of features and updates, see Drill 1.16.0.0-1904 (EEP 6.2.0) Release Notes on page 5763
MapR Installer Support for EEP 6.2.0	Mapr Installer 1.12.0.0 supports EEP 6.2.0 and maintenance EEPs 6.1.1, 6.0.2, 5.0.3, and 4.1.4. For more information, see Release History for EEPs on page 5623 and MapR Installer Updates on page 5481.

EEP 6.1.1 Reference Information

This section contains links to release notes and other reference information for EEP 6.1.1.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 6.1.1 on page 4547
Listed are all Maven artifacts for EEP 6.1.1 components.

Related reference

MapR Ecosystem Pack 6.1.1 Release Notes on page 5686
This topic contains information about the components included with the MapR Ecosystem Pack 6.1.1.

EEP 6.1.1 Components and OS Support on page 5551
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.1.1 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 6.1.0 Reference Information

This section contains links to release notes and other reference information for EEP 6.1.0.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 6.1.0 on page 4578
Listed are all Maven artifacts for EEP 6.1.0 components.

Related reference

MapR Ecosystem Pack 6.1.0 Release Notes on page 5688
This topic contains information about the components included with the MapR Ecosystem Pack 6.1.0.

EEP 6.1.0 Components and OS Support on page 5552
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.1.0 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 6.1.0

Provides a summary of the new functionality in MapR Ecosystem Pack (EEP) 6.1.0.

EEP 6.1.0 provides support for new features in MapR Database JSON.

EEP 6.x can be used with MapR Core 6.1.0. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

New Features in MapR Database JSON

C# and Go OJAI Clients	With EEP 6.1.0, you can use the C# and Go OJAI clients to write MapR Database JSON applications. Python and Node.js clients also are supported. These clients provide lightweight libraries that support the OJAI API and alternatives to the Java OJAI client. You can connect to MapR Database JSON from middleware components, and add, update, and query documents in a MapR Database JSON table. All

clients tailor their features to developers of those respective languages.

The following topics provide more information about these clients:

- Using the C# OJAI Client on page 2688
- Using the Go OJAI Client on page 2692
- Understanding the MapR Data Access Gateway on page 750
- MapR Data Access Gateway 2.0 Release Notes on page 5745

New Features in Apache Kafka

Support for Kafka Schema Registry

Kafka Schema Registry provides a RESTful interface for storing and retrieving Avro schemas.

For more information, see Kafka Schema Registry on page 3941.

New Features in Flume

Support for SSL By Default On Secure Clusters for Flume Avro Source and Sink

For more information, see Flume Thrift Security Parameters on page 3369.

New Features in Oozie

New Oozie Version

EEP 6.1.0 includes support for Oozie 5.1.0.0. See Oozie on page 3989 and Oozie 5.1.0.0-1901 (EEP 6.1.0) Release Notes on page 6284 for more information.

New Features in Tez

Support for Shuffle SSL Encryption Configuration of Tez

For more information, see Tez 0.9.1-1901 (EEP 6.1.0 and EEP 6.0.1) Release Notes on page 6495.

4-Digit Versions

Transition to 4-Digit Versions Begins with EEP 6.1.0

With EEP 6.1.0, version numbers for some ecosystem components and MapR tools use four – rather than three – digits. In future releases, more components and tools will transition to four digits as new versions of the components are introduced. Four-digit versions enable greater precision and flexibility in the delivery of packages, patches, and Maven artifacts for MapR software. The transition to four-digit versions will happen gradually. In EEP 6.1.0, the following components use four-digit versions:

- MapR Installer 1.11.0.0
- Drill 1.15.0.0
- Oozie 5.1.0.0

- Spark 2.3.2.0
- Collectd 5.8.1.0
- Elasticsearch 6.5.3.0
- Fluentd 1.3.2.0
- Grafana 5.4.2.0
- Kibana 6.5.3.0

For more information about component versions, see:

- MapR Ecosystem Pack 6.1.0 Release Notes on page 5688
- Component Versions for Released EEPs on page 5586

EEP 6.0.2 Reference Information

This section contains links to release notes and other reference information for EEP 6.0.2.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 6.0.2 on page 4616
Listed are all Maven artifacts for EEP 6.0.2 components.

Related reference

MapR Ecosystem Pack 6.0.2 Release Notes on page 5690
This topic contains information about the components included with the MapR Ecosystem Pack 6.0.2.

EEP 6.0.2 Components and OS Support on page 5553
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.0.2 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 6.0.1 Reference Information

This section contains links to release notes and other reference information for EEP 6.0.1.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 6.0.1 on page 4647
Listed are all Maven artifacts for EEP 6.0.1 components.

Related reference

MapR Ecosystem Pack 6.0.1 Release Notes on page 5692
This topic contains information about the components included with the MapR Ecosystem Pack 6.0.1.

EEP 6.0.1 Components and OS Support on page 5554
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.0.1 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 6.0.0 Reference Information

This section contains links to release notes and other reference information for EEP 6.0.0.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 6.0.0 on page 4685
Listed are all Maven artifacts for EEP 6.0.0 components.

Related reference

MapR Ecosystem Pack 6.0.0 Release Notes on page 5694
This topic contains information about the components included with the MapR Ecosystem Pack 6.0.0.

EEP 6.0.0 Components and OS Support on page 5555
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 6.0.0 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 6.0.0

Provides a summary of the new functionality in MapR Ecosystem Pack (EEP) 6.0.0.

EEP 6.0.0 provides new features in MapR Database JSON; Amazon S3 API; and Apache Hadoop YARN, Kafka, Spark, Hive, Tez, Hue, Livy, Flume, Oozie, and Sqoop.

EEP 6.x can be used with MapR Core 6.1.0. For more information about EEP and MapR Core version support, see EEP Support and Lifecycle Status on page 5531.

New Features in MapR Database JSON

Node.js and Python OJAI Clients	With EEP 6.0.0, you can use the Node.js and Python OJAI clients to write MapR Database JSON applications. These clients provide lightweight libraries that support the OJAI API and alternatives to the Java OJAI client. You can connect to MapR Database JSON from middleware components, and add, update, and query documents in a MapR Database JSON table. Both clients tailor their features to developers of those respective languages. The following topics provide more information about these clients: • Using the Node.js OJAI Client on page 2673 • Using the Python OJAI Client on page 2678

last-updated: Apr 11, 2022

New Features in Apache Kafka

Support for Apache Kafka Streams

Kafka Streams is a programming library that enables you to create Java or Scala streaming applications and, specifically, building streaming applications that transform input topics into output topics.

Kafka Streams enables you to build moderately complex operational streaming applications faster by offloading common functions such as failure recovery, joins and enrichment, and aggregations and windowing.

See Kafka Streams on page 3854 and Kafka Streams 1.1-1808 Release Notes on page 6182 for more information.

Support for Apache Kafka KSQL

Streaming SQL for Apache Kafka (KSQL) is an open source streaming SQL engine that implements continuous, interactive queries. KSQL enables you to query, read, write, and process data in real time and scale, using SQL commands. KSQL interacts directly with the Kafka Streams API, removing the requirement of building a Java application. See KSQL on page 3843 and KSQL 4.1.1-1808 Release Notes on page 6191 for more information.

New Ecosystem Component

Support for S3 Gateway

The S3 Gateway (S3 gateway) provides you with a REST interface compatible with the Amazon S3 API standard to store and retrieve data from the MapR platform object store in the form of files. For more information, see S3 Gateway on page 3959.

New Features in YARN

YARN Resource Calculation Based on Labels

EEP 6.0.0 implements correct steady and instantaneous fair shares, headroom, and maximum resource calculation for queues with label-based scheduling (LBS). For more information, see YARN Resource Calculation Based on Labels on page 1278.

Support for Azure Data Lake Store

You can use Azure Data Lake Store (ADLS) as an input source or an output destination for all applications. For more information, see Support for ADLS on page 40.

New Features in Spark

New Spark Version

EEP 6.0.0 includes support for Spark 2.3.1. For more information, see Spark 2.3.1-1808 (EEP 6.0.0) Release Notes on page 6385.

New Features in Hive and Tez

New Hive and Tez Versions

EEP 6.0.0 includes support for Hive 2.3 and Tez 0.9. For more information, see Hive and Tez Integration on page 3502. You can also refer to Hive 2.3.3-1808 (EEP 6.0.0) Release Notes on page 5953 and Tez 0.9.1-1808 (EEP 6.0.0) Release Notes on page 6496.

last-updated: Apr 11, 2022

Support for UPDATE, INSERT INTO, and MERGE statements in Hive	• You can use the UPDATE statement to update primitive, complex, and complex nested data types in MapR Database JSON tables, using the Hive connector. For more information, see Understanding the UPDATE Statement on page 3469. • You can use the INSERT INTO statement to insert or overwrite rows in nested MapR Database JSON tables, using the Hive connector. For more information, see Understanding the INSERT INTO Statement on page 3473. • You can use the MERGE statement to efficiently perform record-level INSERT and UPDATE operations within Hive tables. For more information, see Understanding the MERGE Statement on page 3481.
Support for Splitting Hive Log Files	You can split Hive log files into HiveServer2 and Metestore log files by process ID. For more information, see Splitting Hive Logs into HiveServer2 and Metastore logs by Process ID on page 3606.
Support for SQL Standards-Based Hive Authorization	You can configure SQL standards-based authorization to enable fine-grained access control with SQL commands. For more information, see SQL Standards-Based Hive Authorization on page 3453.
Support for Auto-Generated PEM files for WebHCat REST API on a MapR-SASL cluster	For more information, see Requirements for Using Automatically Generated PEM Files on page 3498.
Support for Configuring JDBC Connection String with SSL Encryption Enabled or Disabled	For more information, see Configuring JDBC Connection String with SSL Encryption Enabled or Disabled on page 3521.
Support for Zero RM HA for the new Tez UI	For more information, see Hive-on-Tez User Interface on page 3506.

New Features in Hue and Livy

New Hue and Livy Versions	EEP 6.0.0 includes support for Hue 4.2.0 and Livy 0.5.0. For more information, see Hue 4.2.0-1808 (EEP 6.0.0) Release Notes on page 6093.
New Features for Livy Security	For more information, see Configure Livy with Security on page 3839.

New Feature in Flume

Support for SSL By Default On Secure Clusters for Flume Thrift Source and Sink	For more information, see Flume Thrift Security Parameters on page 3369.

New Feature in Oozie

Support for Encrypting the Oozie Database User Password	For more information, see Encrypt the Oozie Database User Password on page 3997.

last-updated: Apr 11, 2022

New Features in Sqoop

New Sqoop Version	EEP 6.0.0 includes support for Sqoop 1.4.7. The Teradata Connector for Hadoop (TDCH) version is updated to v1.5.4. For more information, see Sqoop 1.4.7-1808 (EEP 6.0.0) Release Notes on page 6458.

New Features in Upgrade

Support for Preserving User Configuration	With EEP 6.0.0 you can preserve user configuration in the Hive, Tez, Hue, Impala, Sentry, Oozie, Sqoop, Kafka, Spark, Livy, and HttP-FS components. For more information, see Preparing to Upgrade the MapR Ecosystem Pack on page 335.

EEP 5.0.7 Reference Information

This section contains links to release notes and other reference information for EEP 5.0.7.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 5.0.7 on page 4721
There are no changes to the Maven artifacts for EEP 5.0.7 components.

Related reference

MapR Ecosystem Pack 5.0.7 Release Notes on page 5695
This topic contains information about the components included with the MapR Ecosystem Pack 5.0.7.

EEP 5.0.7 Components and OS Support on page 5556
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.7 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 5.0.6 Reference Information

This section contains links to release notes and other reference information for EEP 5.0.6.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 5.0.6 on page 4721
Listed are all Maven artifacts for EEP 5.0.6 components.

Related reference

MapR Ecosystem Pack 5.0.6 Release Notes on page 5697
This topic contains information about the components included with the MapR Ecosystem Pack 5.0.6.

EEP 5.0.6 Components and OS Support on page 5557
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.6 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586

last-updated: Apr 11, 2022

The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 5.0.5 Reference Information

This section contains links to release notes and other reference information for EEP 5.0.5.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Maven Artifacts for EEP 5.0.5 on page 4735
Listed are all Maven artifacts for EEP 5.0.5 components.

Related reference

MapR Ecosystem Pack 5.0.5 Release Notes on page 5699
This topic contains information about the components included with the MapR Ecosystem Pack 5.0.5.

EEP 5.0.5 Components and OS Support on page 5558
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.5 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 5.0.4 Reference Information

This section contains links to release notes and other reference information for EEP 5.0.4.

Related concepts

Maven Artifacts for EEP 5.0.4 on page 4759
Listed are all Maven artifacts for EEP 5.0.4 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 5.0.4 Release Notes on page 5700
This topic contains information about the components included with the MapR Ecosystem Pack 5.0.4.

EEP 5.0.4 Components and OS Support on page 5559
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.4 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 5.0.3 Reference Information

This section contains links to release notes and other reference information for EEP 5.0.3.

Related concepts

Maven Artifacts for EEP 5.0.3 on page 4779

Listed are all Maven artifacts for EEP 5.0.3 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 5.0.3 Release Notes on page 5702
This topic contains information about the components included with the MapR Ecosystem Pack 5.0.3.

EEP 5.0.3 Components and OS Support on page 5560
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.3 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 5.0.2 Reference Information

This section contains links to release notes and other reference information for EEP 5.0.2.

Related concepts

Maven Artifacts for EEP 5.0.2 on page 4807
Listed are all Maven artifacts for EEP 5.0.2 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 5.0.2 Release Notes on page 5704
This topic contains information about the components included with the MapR Ecosystem Pack 5.0.2.

EEP 5.0.2 Components and OS Support on page 5561
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.2 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 5.0.1 Reference Information

This section contains links to release notes and other reference information for EEP 5.0.1.

Related concepts

Maven Artifacts for EEP 5.0.1 on page 4832
Listed are all Maven artifacts for EEP 5.0.1 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 5.0.1 Release Notes on page 5705
This topic contains information about the components included with the MapR Ecosystem Pack 5.0.1.

EEP 5.0.1 Components and OS Support on page 5562
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.1 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 5.0.0 Reference Information

This section contains links to release notes and other reference information for EEP 5.0.0.

Related concepts

Maven Artifacts for EEP 5.0.0 on page 4853
Listed are all Maven artifacts for EEP 5.0.0 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 5.0.0 Release Notes on page 5707
This topic contains information about the components included with the MapR Ecosystem Pack 5.0.0.

EEP 5.0.0 Components and OS Support on page 5563
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 5.0.0 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 5.0.0

Provides a summary of the new functionality in EEP 5.0.0.

EEP 5.0.0 includes support for MapR 6.0.1 and these new features:

New Features and Additions

MapR Database JSON REST API	The MapR Database JSON REST API allows you to use HTTP calls to perform basic operations on MapR Database JSON tables. For more information, see Using the MapR Database JSON REST API on page 2696.
Apache Spark 2.2.1 with support for Structured Streaming	Apache Spark Structured Streaming is a new stream processing API that makes creating real-time analytic applications easier than ever by providing a functional, SQL-like API. For more information, see Structured Streaming in Spark on page 3952.
Support for PAM Authentication for Apache Spark	PAM authentication for Apache Spark is supported on secure clusters. For more information, see PAM Authentication for Spark on page 4133.
New Drill Version	EEP 5.0.0 includes support for Drill 1.13. For more information, see Drill 1.13-1803 Release Notes on page 5784.
Support for RM HA in Tez User Interface	EEP 4.1.1 and EEP 5.0.0 supports RM HA in the Tez user interface. For more information, see Tez 0.8.4-1803 (EEP 4.1.1 and EEP 5.0.0) Release Notes on page 6501.

last-updated: Apr 11, 2022

New Kafka REST Proxy 4.0.0 for MapR	For more information, see Kafka REST Proxy on page 3865.
New Kafka Connect 4.0.0 for MapR	For more information, see Kafka Connect on page 3903.
New Features of Impala 2.10	For more information, see New Features in Impala 2.10 for MapR on page 3691.

EEP 4.1.4 Reference Information

This section contains links to release notes and other reference information for EEP 4.1.4.

Related concepts

Maven Artifacts for EEP 4.1.4 on page 4889
Listed are all Maven artifacts for EEP 4.1.4 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 4.1.4 Release Notes on page 5708
This topic contains information about the components included with the MapR Ecosystem Pack 4.1.4.

EEP 4.1.4 Components and OS Support on page 5564
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 4.1.4 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 4.1.3 Reference Information

This section contains links to release notes and other reference information for EEP 4.1.3.

Related concepts

Maven Artifacts for EEP 4.1.3 on page 4911
Listed are all Maven artifacts for EEP 4.1.3 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 4.1.3 Release Notes on page 5710
This topic contains information about the components included with the MapR Ecosystem Pack 4.1.3.

EEP 4.1.3 Components and OS Support on page 5565
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 4.1.3 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 4.1.2 Reference Information

This section contains links to release notes and other reference information for EEP 4.1.2.

Related concepts

Maven Artifacts for EEP 4.1.2 on page 4936
Listed are all Maven artifacts for EEP 4.1.2 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 4.1.2 Release Notes on page 5711
This topic contains information about the components included with the MapR Ecosystem Pack 4.1.2.

EEP 4.1.2 Components and OS Support on page 5566
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 4.1.2 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 4.1.1 Reference Information

This section contains links to release notes and other reference information for EEP 4.1.1.

Related concepts

Maven Artifacts for EEP 4.1.1 on page 4957
Listed are all Maven artifacts for EEP 4.1.1 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 4.1.1 Release Notes on page 5713
This topic contains information about the components included with the MapR Ecosystem Pack 4.1.1.

EEP 4.1.1 Components and OS Support on page 5567
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 4.1.1 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 4.1.0 Reference Information

This section contains links to release notes and other reference information for EEP 4.1.0.

Related concepts

Maven Artifacts for EEP 4.1.0 on page 4982
Listed are all Maven artifacts for EEP 4.1.0 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 4.1.0 Release Notes on page 5714
This topic contains information about the components included with the MapR Ecosystem Pack 4.1.0.

Component Versions for Released EEPs on page 5586

The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

EEP 4.1.0 Components and OS Support on page 5568
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 4.1.0 and shows the operating system support for each component.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 4.1.0
Provides a summary of the new functionality in EEP 4.1.0.

EEP 4.1.0 includes support for MapR 6.0 and these new features:

New Features and Additions

Support for Java and Python APIs for MapR Database OJAI Connector for Apache Spark.	For more information, see Understanding the MapR Database OJAI Connector for Spark on page 4050.
Support for saving an Apache Spark Dataset to a MapR Database table.	For more information, see Saving an Apache Spark Dataset to a MapR Database JSON Table on page 4097.
Support for insertToMapRDB API to insert an RDD, Dataframe, or Dataset into a MapR Database table.	For more information, see Saving Data to a MapR Database JSON Table on page 4088.
New Drill Version	EEP 4.1.0 includes support for Drill 1.12. For more information, see Drill 1.12.0-1801 Release Notes on page 5790.

EEP 4.0.0 Reference Information

This section contains links to release notes and other reference information for EEP 4.0.0.

Related concepts

Maven Artifacts for EEP 4.0.0 on page 4992
Listed are all Maven artifacts for EEP 4.0.0 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 4.0.0 Release Notes on page 5715
This topic contains information about the components included with the MapR Ecosystem Pack 4.0.0.

EEP 4.0.0 Components and OS Support on page 5569
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 4.0.0 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 4.0.0
Provides a summary of the new functionality in EEP 4.0.0.

EEP 4.0.0 includes support for MapR 6.0 and these new features:

New Features and Additions

Enhanced Security	For more information, see Built-in Security in MapR.
Zeppelin Ecosystem Component	For more information, see Zeppelin on MapR on page 3033.
Enhancements to Parallelism in Drill 1.11	In previous versions of Drill, parallelism correlated with the number of tablets (table partitions) that MapR Database split a table into, and each tablet stored up to 4 GB of data. One minor fragment ran for each tablet. As of MapR 6.0 and Drill 1.11, parallelism correlates with the amount of data instead of the number of tablets. One minor fragment is created for approximately every 128 MB of data. For example, Drill creates 32 minor fragments for 4 GB of data. The planner.slice_target option determines the number of minor fragments that run in parallel. For more information, see: • maprdb Format Plugin for Drill on page 3255 • Modifying Query Planning Options

EEP 3.0.5 Reference Information

This section contains links to release notes and other reference information for EEP 3.0.5.

Related concepts

Maven Artifacts for EEP 3.0.5 on page 5028
Listed are all Maven artifacts for EEP 3.0.5 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 3.0.5 Release Notes on page 5717
This topic contains information about the components included with the MapR Ecosystem Pack 3.0.5.

EEP 3.0.5 Components and OS Support on page 5570
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 3.0.5 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 3.0.4 Reference Information

This section contains links to release notes and other reference information for EEP 3.0.4.

Related concepts

Maven Artifacts for EEP 3.0.4 on page 5046
Listed are all Maven artifacts for EEP 3.0.4 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 3.0.4 Release Notes on page 5718
This topic contains information about the components included with the MapR Ecosystem Pack 3.0.4.

EEP 3.0.4 Components and OS Support on page 5571
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 3.0.4 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 3.0.3 Reference Information

This section contains links to release notes and other reference information for EEP 3.0.3.

Related concepts

Maven Artifacts for EEP 3.0.3 on page 5063
Listed are all Maven artifacts for EEP 3.0.3 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 3.0.3 Release Notes on page 5720
This topic contains information about the components included with the MapR Ecosystem Pack 3.0.3.

EEP 3.0.3 Components and OS Support on page 5572
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 3.0.3 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 3.0.2 Reference Information

This section contains links to release notes and other reference information for EEP 3.0.2.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 3.0.2 Release Notes on page 5721
This topic contains information about the components included with the MapR Ecosystem Pack 3.0.2.

EEP 3.0.2 Components and OS Support on page 5573
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 3.0.2 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 3.0.1 Reference Information

This section contains links to release notes and other reference information for EEP 3.0.1.

last-updated: Apr 11, 2022

Related concepts

Maven Artifacts for EEP 3.0.1 on page 5089
Listed are all Maven artifacts for EEP 3.0.1 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

EEP 3.0.1 Components and OS Support on page 5574
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 3.0.1 and shows the operating system support for each component.

MapR Ecosystem Pack 3.0.1 Release Notes on page 5723
This topic contains information about the components included with the MapR Ecosystem Pack 3.0.1.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 3.0.1

Provides a summary of the new functionality in EEP 3.0.1.

EEP 3.0.1 is a maintenance EEP that includes these new features:

- Support for SLES 12 SP2. For more information, see the Operating System Support Matrix.
- Support for Spark on Mesos.

EEP 3.0 Reference Information

This section contains links to release notes and other reference information for EEP 3.0.

Related concepts

Maven Artifacts for EEP 3.0.0 on page 5109
Listed are all Maven artifacts for EEP 3.0.0 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 3.0 Release Notes on page 5724
This topic contains information about the components included with the MapR Ecosystem Pack 3.0.

EEP 3.0 Components and OS Support on page 5575
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 3.0 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

What's New in EEP 3.0

Provides a summary of the new functionality in EEP 3.0.

EEP 3.0 provides a series of stability and security fixes for Spark and improves the speed of ETL and batch processing with a faster version of Hive.

New Features and Additions

MapR Database OJAI Connector for Apache Spark

The MapR Database OJAI Connector for Apache Spark is a new API that makes it easier to build real-time or batch pipelines between your data and MapR Database and leverage Spark within the pipeline. This feature includes:

- Two new APIs that allow you to load data from a MapR Database JSON table to a Spark RDD or save a Spark RDD to a MapR Database JSON table
- A custom partitioner that allows you to partition data for better performance
- Data locality: when the connector reads data from MapR Database, it uses the data locality feature of MapR Database to spawn the Spark executors

For more information, see Understanding the MapR Database OJAI Connector for Spark on page 4050.

MapR Database Binary Connector for Apache Spark

The new MapR Database Binary Connector for Apache Spark allows you to write applications that consume HBase binary tables and use them in Spark. Features include:

- Writing directly to HBase HFiles for bulk insertion into HBase
- Spark SQL can draw on tables that are represented in HBase

For more information, see MapR Database Binary Connector for Apache Spark on page 4101.

MapR Event Store For Apache Kafka C Applications (librdkafka)

As of MapR maintenance release 5.2.1, you can develop C applications for MapR Event Store For Apache Kafka. The MapR Event Store For Apache Kafka C Client is a distribution of librdkafka that integrates with MapR Streams.

For more information, see MapR Event Store For Apache Kafka C Applications on page 2795.

MapR Event Store For Apache Kafka Python Applications

As of MapR 5.2.1, you can create Python applications for MapR Event Store For Apache Kafka using the MapR Streams Python client. The Streams Python client is a binding for librdkafka and contains support for high-level consumers.

For more information, see MapR Event Store For Apache Kafka Python Applications on page 2998.

Key Upgrades

Apache Spark 2.1.0

Spark 2.1 in the MapR converged data platform brings improvements in enterprise-ready stability and security, including:

- More than 1200 fixes on the Spark 2.x line

last-updated: Apr 11, 2022

- MapR-SASL support for encrypted Thrift-server connections
- Scalable partition handling
- Stable data-type APIs

For more information, see Apache Spark Feature Support on page 4027.

Apache Hive 2.1.1

EEP 3.0 provides a faster version of Hive to improve the speed of data-processing tasks, to reduce latency for interactive queries, and to increase throughput for batch queries. Key improvements include:

- 2x faster ETL through an enhanced cost-based optimizer (CBO), faster type conversions, and dynamic partition pruning
- New HiveServer UI with new diagnostics and monitoring tools
- Dynamically partitioned hash joins, which provide unsorted inputs in order to eliminate the sorting step.
- Vectorized query execution that greatly reduces the CPU usage for typical query operations, like scans, filters, aggregates, and joins

For more information, see Hive on page 3405.

Apache Drill 1.10

Continuing on the iterative releases, Drill 1.10 is another important milestone for Apache Drill. Numerous enhancements have been added to this release for BI tool integration, end-to-end security, performance, and usability enhancements. Highlights of this release include:

- Tableau native connectivity
- Support for Kerberos and MapR-SASL authentication between the client and Drillbit
- Support for the CREATE TEMPORARY TABLE AS (CTTAS) command
- Ability to query data with Hue 3.12 (experimental only)
- Improved compatibility with Hive/Spark-generated Parquet files

For more information, see the Drill Introduction.

EEP 2.0.3 Reference Information

This section contains links to release notes and other reference information for EEP 2.0.3.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 2.0.3 Release Notes on page 5725
This topic contains information about the components included with the MapR Ecosystem Pack 2.0.3.

last-updated: Apr 11, 2022

EEP 2.0.3 Components and OS Support on page 5576
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 2.0.3 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 2.0.2 Reference Information

This section contains links to release notes and other reference information for EEP 2.0.2.

Related concepts

Maven Artifacts for EEP 2.0.2 on page 5147
Listed are all Maven artifacts for EEP 2.0.2 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

EEP 2.0.2 Components and OS Support on page 5577
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 2.0.2 and shows the operating system support for each component.

MapR Ecosystem Pack 2.0.2 Release Notes on page 5727
This topic contains information about the components included with the MapR Ecosystem Pack 2.0.2.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 2.0.1 Reference Information

This section contains links to release notes and other reference information for EEP 2.0.1.

Related concepts

Maven Artifacts for EEP 2.0.1 on page 5151
Listed are all Maven artifacts for EEP 2.0.1 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 2.0.1 Release Notes on page 5728
This topic contains information about the components included with the MapR Ecosystem Pack 2.0.1.

EEP 2.0.1 Components and OS Support on page 5577
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 2.0.1 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

last-updated: Apr 11, 2022

EEP 2.0 Reference Information

This section contains links to release notes and other reference information for EEP 2.0.

Related concepts

Maven Artifacts for EEP 2.0.0 on page 5158
Listed are all Maven artifacts for EEP 2.0.0 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 2.0 Release Notes on page 5729
This topic contains information about the components included with the MapR Ecosystem Pack 2.0.

EEP 2.0 Components and OS Support on page 5578
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 2.0 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 1.1.4 Reference Information

This section contains links to release notes and other reference information for EEP 1.1.4.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 1.1.4 Release Notes on page 5731
This topic contains information about the components included with the MapR Ecosystem Pack 1.1.4.

EEP 1.1.4 Components and OS Support on page 5579
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 1.1.4 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 1.1.3 Reference Information

This section contains links to release notes and other reference information for EEP 1.1.3.

Related concepts

Maven Artifacts for EEP 1.1.3 on page 5192
Listed are all Maven artifacts for EEP 1.1.3 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 1.1.3 Release Notes on page 5732
This topic contains information about the components included with the MapR Ecosystem Pack 1.1.3.

EEP 1.1.3 Components and OS Support on page 5580

This topic lists the ecosystem and MapR-monitoring components that are included in EEP 1.1.3 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 1.1.2 Reference Information

This section contains links to release notes and other reference information for EEP 1.1.2.

Related concepts

Maven Artifacts for EEP 1.1.2 on page 5196
Listed are all Maven artifacts for EEP 1.1.2 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 1.1.2 Release Notes on page 5733
This topic contains information about the components included with the MapR Ecosystem Pack 1.1.2.

EEP 1.1.2 Components and OS Support on page 5581
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 1.1.2 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 1.1.1 Reference Information

This section contains links to release notes and other reference information for EEP 1.1.1.

Related concepts

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 1.1.1 Release Notes on page 5735
This topic contains information about the components included with the MapR Ecosystem Pack 1.1.1.

EEP 1.1.1 Components and OS Support on page 5582
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 1.1.1 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

EEP 1.1.0 Reference Information

This section contains links to release notes and other reference information for EEP 1.1.0.

last-updated: Apr 11, 2022

Related concepts

Maven Artifacts for EEP 1.1.0 on page 5207
Listed are all Maven artifacts for EEP 1.1.0 components.

Package Names for MapR Ecosystem Packs (EEPs) on page 5737
This page describes how to view the the package names for each MapR Ecosystem Pack (EEP) release.

Related reference

MapR Ecosystem Pack 1.1.0 Release Notes on page 5736
This topic contains information about the components included with the MapR Ecosystem Pack 1.1.0.

EEP 1.1 Components and OS Support on page 5583
This topic lists the ecosystem and MapR-monitoring components that are included in EEP 1.1 and shows the operating system support for each component.

Component Versions for Released EEPs on page 5586
The published MapR Ecosystem Packs (EEPs) contain different component versions with different features. Comparing the component versions can help you make decisions about installing or upgrading data-fabric software.

Release History for EEPs on page 5623
This section shows the original release dates for all MapR Ecosystem Packs (EEPs).

MapR Data Fabric for Kubernetes Release Notes

This section contains release notes for the MapR Data Fabric for Kubernetes.

MapR CSI Storage Plugin Release Notes

This section contains release notes for the Container Storage Interface (CSI) Storage Plugin.

MapR Container Storage Interface (CSI) Storage Plugin Release 1.2.x (FUSE POSIX)
These notes describe release 1.2.x of the MapR Container Storage Interface (CSI) Storage Plugin for FUSE POSIX.

You may also be interested in the Kubernetes Release Notes. For the latest 1.2.x version, see the `mapr-csi` github repository.

Version	1.2.x
Release Date	November 2020
MapR Version Interoperability	Compatible with release 6.1.0 and later.
Kubernetes Compatibility	Kubernetes 1.17.0 and later.*
OpenShift Compatibility	4.4 and later.
CSI Driver Downloads	See Downloads (CSI) on page 225 for more information.
Documentation	• Overview: MapR Container Storage Interface (CSI) Storage Plugin Overview on page 666 • Installation: Installing, Uninstalling, and Upgrading the MapR Container Storage Interface (CSI) Storage Plugin on page 228 • Supported Versions: CSI Version Compatibility on page 5596
Related Resources	https://www.hpe.com/us/en/resource-library.html

***** Kubernetes alpha features are not supported.

New in this Release

This release of the MapR Container Storage Interface (CSI) Storage Plugin increments the version of the `csi-kdfplugin` to 1.2.x. Release 1.2.x includes support for:

- Volume cloning for dynamic provisioning. For more information, see CSI Volume Cloning.
- Snapshot restore for dynamic provisioning. For more information, see Snapshot & Restore Feature.
- Dynamic and static provisioning of raw block volumes. For more information, see Raw Block Volumes on page 670.

You can access the `csi-kdfplugin` by installing the custom resource definition (CRD) using the `csi-maprkdf-v<version>.yaml` file. Or you can build your own container and point to the plugin on the Docker hub at `maprtech/csi-kdfplugin:<version>`. For installation information, see Installing, Uninstalling, and Upgrading the MapR Container Storage Interface (CSI) Storage Plugin on page 228.

Patches

None.

Limitations

Note the following limitations:

- CSI Driver version 1.2.x does not support coexistence with the FlexVolume Driver on the same Kubernetes cluster.
- All nodes in the Kubernetes cluster must use the same Linux OS. Configuration files are available to support the following Linux distributions:
  - CentOS
  - RHEL (use CentOS configuration file)
  - Ubuntu
- The Container POSIX client package is included by default when you install the MapR Container Storage Interface (CSI) Storage Plugin. The Basic, Container, or Platinum POSIX client can be enabled by specifying a parameter in the pod spec.
- The CSI Driver does not include support for inline volumes in pods. It supports only PersistentVolumes.

Known Issues

Note the following known issues:

- Snapshot restore fails if the snapshot contains symlinks to other files in the directory.

Resolved Issues

Issue	Description
CSI-30	Enable memory profiling for fuse process w/ trackMemory : true option
CSI-241	Support volume clone for dynamic provisioning
CSI-242	Support snapshot restore for dynamic provisioning
CSI-243	Support for OpenShift 4.4, 4.5, 4.6+ & Kubernetes 1.17, 1.18, 1.19+

last-updated: Apr 11, 2022

Issue	Description
CSI-248	Retain fuse logs after pod delete w/ retainLogs: true option
REL-301	Update kdfplugin image w/ 6.2 release bits on centos8
CSI-254	Option 'numrpcthreads' added to configure Number of Client RPC threads (default:1, max:4).
CSI-258	DF client fixes & updates
CSI-259	Reduce verbose logging on CSI logfiles
CSI-262	[BETA] Support ticket-based authentication to apiserver. You can use MAPR_CLUSTER_TICKET instead of MAPR_CLUSTER_USER and MAPR_CLUSTER_PASSWORD. See REST Secrets on page 3168.
BDP-2631	Update to livenessprobe v2.2.0 image to remove level5 messages

MapR Container Storage Interface (CSI) Storage Plugin Release 1.0 (Loopback NFS)
These notes describe release 1.0.x of the MapR Container Storage Interface (CSI) Storage Plugin for Loopback NFS.

You may also be interested in the Kubernetes Release Notes. For the latest 1.0.x version, see the `mapr-csi` github repository.

Version	1.0.x
Release Date	November 2020
MapR Version Interoperability	Compatible with release 6.1.0 and later.
Kubernetes Compatibility	Kubernetes 1.17.0 and later.*
OpenShift Compatibility	4.4 and later.
CSI Driver Downloads	See Downloads (CSI) on page 225 for more information.
Documentation	• Overview: MapR Container Storage Interface (CSI) Storage Plugin Overview on page 666 • Installation: Installing, Uninstalling, and Upgrading the MapR Container Storage Interface (CSI) Storage Plugin on page 228 • Supported Versions: CSI Version Compatibility on page 5596
Related Resources	https://www.hpe.com/us/en/resource-library.html

***** Kubernetes alpha features are not supported.

New in this Release

This first release of the Container Storage Interface (CSI) Storage Plugin for NFS includes `.yaml` configuration files that can be installed onto a Kubernetes cluster. Once installed, these containers provide an NFS-based CSI Driver for the file-system volume plug-in and a Kubernetes Dynamic Volume Provisioner that permit static and dynamic provisioning of data-fabric storage from Kubernetes.

You can access the `csi-nfsplugin` by installing the custom resource definition (CRD) using the `csi-maprnfskdf-v<version>.yaml` file. Or you can build your own container and point to the plugin on the Docker hub at `maprtech/csi-nfsplugin:<version>`. For installation information, see Installing, Uninstalling, and Upgrading the MapR Container Storage Interface (CSI) Storage Plugin on page 228.

Release 1.0.x also includes support for dynamic and static provisioning of raw block volumes. For more information, see Raw Block Volumes on page 670.

Patches

None.

Limitations

Note the following limitations:

- CSI Driver version 1.0.x does not support coexistence with the FlexVolume Driver on the same Kubernetes cluster.
- All nodes in the Kubernetes cluster must use the same Linux OS. Configuration files are available to support the following Linux distributions:
  - CentOS
  - RHEL (use CentOS configuration file)
  - Ubuntu
- The CSI Driver does not include support for inline volumes in pods. It supports only PersistentVolumes.

Known Issues

Note the following known issues:

- Snapshot restore fails if the snapshot contains symlinks to other files in the directory.

Resolved Issues

Issue	Description
CSI-30	Enable memory profiling for fuse process w/ trackMemory : true option
CSI-241	Support volume clone for dynamic provisioning
CSI-242	Support snapshot restore for dynamic provisioning
CSI-243	Support for OpenShift 4.4, 4.5, 4.6+ & Kubernetes 1.17, 1.18, 1.19+
CSI-248	Retain fuse logs after pod delete w/ retainLogs: true option
REL-301	Update kdfplugin image w/ 6.2 release bits on centos8
CSI-254	Option 'numrpcthreads' added to configure Number of Client RPC threads (default:1, max:4).
CSI-258	DF client fixes & updates
CSI-259	Reduce verbose logging on CSI logfiles
CSI-262	[BETA] Support ticket-based authentication to apiserver. You can use MAPR_CLUSTER_TICKET instead of MAPR_CLUSTER_USER and MAPR_CLUSTER_PASSWORD. See REST Secrets on page 3168.
BDP-2631	Update to livenessprobe v2.2.0 image to remove level5 messages

MapR Container Storage Interface (CSI) Storage Plugin Release 1.1.0

These notes describe Release 1.1.0 of the MapR Container Storage Interface (CSI) Storage Plugin.

You may also be interested in the Kubernetes Release Notes.

Version	1.1.0

last-updated: Apr 11, 2022

Release Date	August 2020
MapR Version Interoperability	Compatible with MapR 6.1.0 and later.
Kubernetes Compatibility	Kubernetes 1.16.0 and later.*
OpenShift Compatibility	4.2 and 4.3.
CSI Driver Downloads	See Downloads (CSI) on page 225 for more information.
Documentation	• Overview: MapR Container Storage Interface (CSI) Storage Plugin Overview on page 666 • Installation: Installing, Uninstalling, and Upgrading the MapR Container Storage Interface (CSI) Storage Plugin on page 228 • Supported Versions: CSI Version Compatibility on page 5596
Related Resources	https://www.hpe.com/us/en/resource-library.html

***** Kubernetes alpha features are not supported.

New in this Release

This release of the MapR Container Storage Interface (CSI) Storage Plugin increments the version of the `csi-kdfplugin` to 1.1.0. Release 1.1.0 includes support for all three MapR POSIX licenses (Basic, Container, and Platinum) and allows users to pass custom startup parameters to the FUSE process.

Release 1.1.0 also includes support for volume expansion for dynamic provisioning. For more information, see Example: Volume Expansion for Dynamic Provisioning Using MapR Container Storage Interface (CSI) Storage Plugin on page 3126.

You can access the new `csi-kdfplugin` by installing the custom resource definition using the `csi-maprkdf-v1.1.0.yaml` file. Or you can build your own container and point to the plugin on the Docker hub at `maprtech/csi-kdfplugin:1.1.0`. For installation information, see Installing, Uninstalling, and Upgrading the MapR Container Storage Interface (CSI) Storage Plugin on page 228.

Patches

None.

Limitations

Note the following limitations:

- CSI Driver version 1.1.0 does not support coexistence with the FlexVolume Driver on the same Kubernetes cluster.
- All nodes in the Kubernetes cluster must use the same Linux OS. Configuration files are available to support the following Linux distributions:
  - CentOS
  - Red Hat (use CentOS configuration file)
  - Ubuntu

- The Container POSIX client package is included by default when you install the MapR Container Storage Interface (CSI) Storage Plugin. The Basic, Container, or Platinum POSIX client can be enabled by specifying a parameter in the pod spec. Only the FUSE-based POSIX client is supported. NFSv3 and NFSv4 are not supported.
- The CSI Driver does not include support for inline volumes in pods. It supports only PersistentVolumes.

Known Issues

Note the following known issues:

- On nodeplugin pod restart or upgrade scenario, the existing POSIX client(s) running in the CSI Driver container are killed. The workaround is to move/stop the container workload using MapR CSI Storage Plugin, restart/update the MapR CSI Storage Plugin and start using the MapR CSI Storage Plugin again.
- On Provisioner restart, Provisioner loses the information about the REST server where volume or snapshot should be deleted for existing volume and snapshots provisioned. The administrator must manually remove the volume and/or snapshot for provisioned volumes from the MapR Data Platform.
- Provisioned snapshot information is written to the provisioner log, but not available in the Kubernetes objects such as volumeSnapshots, VolumesnapshotContents etc.
- If you want read-only behavior, specify `readOnly` in the `volumeAttributes`. For example, the following is supported:

```
csi:
    nodePublishSecretRef:
      name: "mapr-ticket-secret"
      namespace: "test-csi"
    driver: com.mapr.csi-kdf
    volumeHandle: pv-securepv-test-read-only-id
    volumeAttributes:
      volumePath: "/user/root"
      cluster: "clusterA"
      cldbHosts: "10.10.10.210"
      securityType: "secure"
      readOnly: "true"
```

The following is not supported:

```
csi:
    nodePublishSecretRef:
      name: "mapr-ticket-secret"
      namespace: "test-csi"
    driver: com.mapr.csi-kdf
    volumeHandle: pv-securepv-test-read-only-id
    readOnly: true
    volumeAttributes:
      volumePath: "/user/root"
      cluster: "clusterA"
      cldbHosts: "10.10.10.210"
      securityType: "secure"
```

Resolved Issues

Issue	Description
K8S-844	[csi-driver] mapr k8s log rotation policy and clearing of unused mounts (more fixes)

Issue	Description
K8S-1199	Ship debugging tools with POSIX client in K8S world
K8S-1405	Support CSI driver for OpenShift 4.3
K8S-1694	[csi-driver] Implemement CSI Spec v1.3 for CSI Driver
K8S-1695	[csi-driver] Implement VolumeExpansion RPC
K8S-1696	[csi-driver] Update all side container images (support k8s 1.15+)

MapR Container Storage Interface (CSI) Storage Plugin Release 1.0.2
These notes describe Release 1.0.2 of the MapR Container Storage Interface (CSI) Storage Plugin.

You may also be interested in the Kubernetes Release Notes.

Version	1.0.2
Release Date	March 2020
MapR Version Interoperability	Compatible with MapR 6.1.0 or later.
Kubernetes Compatibility	Kubernetes 1.13.0 and later.*
OpenShift Compatibility	4.1 and 4.2
CSI Driver Downloads	See Downloads (CSI) on page 225 for more information.
Documentation	MapR Container Storage Interface (CSI) Storage Plugin Overview on page 666
Related Resources	https://www.hpe.com/us/en/resource-library.html

***** Kubernetes alpha features are not supported.

New in this Release

This release of the MapR Container Storage Interface (CSI) Storage Plugin increments the version of the `csi-kdfplugin` to 1.0.2. Release 1.0.2 adds support for all three MapR POSIX licenses (Basic, Container, and Platinum) and allows users to pass custom startup parameters to the FUSE process.

For new-feature information, see Example: Mounting a PersistentVolume for Static Provisioning on page 3113.

You can access the new `csi-kdfplugin` by installing the custom resource definition using the `csi-maprkdf-v1.0.2.yaml` file. Or you can build your own container and point to the plugin on the Docker hub at `maprtech/csi-kdfplugin:1.0.2`. For installation information, see Installing, Uninstalling, and Upgrading the MapR Container Storage Interface (CSI) Storage Plugin on page 228.

Patches

None.

Limitations

Note the following limitations:

- CSI Driver version 1.0 does not support coexistence with the FlexVolume Driver on the same Kubernetes cluster.
- All nodes in the Kubernetes cluster must use the same Linux OS. Configuration files are available to support the following Linux distributions:
  - CentOS

  - Red Hat (use CentOS configuration file)
  - Ubuntu
- The Container POSIX client package is included by default when you install the MapR Container Storage Interface (CSI) Storage Plugin. The Basic or Platinum POSIX client can be enabled by specifying a parameter in the Pod spec. Only the FUSE-based POSIX client is supported. NFSv3 and NFSv4 are not supported.
- The CSI Driver does not include support for inline volumes in pods. It supports only PersistentVolumes.

Known Issues

Note the following known issues:

- On nodeplugin Pod restart or upgrade scenario, the existing POSIX client(s) running in the CSI Driver container are killed. The workaround is to move/stop the container workload using MapR CSI Storage Plugin, restart/update the MapR CSI Storage Plugin and start using the MapR CSI Storage Plugin again.
- On Provisioner restart, Provisioner loses the information about the REST server where volume or snapshot should be deleted for existing volume and snapshots provisioned. The administrator must manually remove the volume and/or snapshot for provisioned volumes from the MapR Data Platform.
- Provisioned snapshot information is written to the provisioner log, but not available in the Kubernetes objects such as volumeSnapshots, VolumesnapshotContents etc.
- If you want read-only behavior, specify `readOnly` in the `volumeAttributes`. For example, the following is supported:

```
csi:
    nodePublishSecretRef:
      name: "mapr-ticket-secret"
      namespace: "test-csi"
    driver: com.mapr.csi-kdf
    volumeHandle: pv-securepv-test-read-only-id
    volumeAttributes:
      volumePath: "/user/root"
      cluster: "clusterA"
      cldbHosts: "10.10.10.210"
      securityType: "secure"
      readOnly: "true"
```

  The following is not supported:

```
csi:
    nodePublishSecretRef:
      name: "mapr-ticket-secret"
      namespace: "test-csi"
    driver: com.mapr.csi-kdf
    volumeHandle: pv-securepv-test-read-only-id
    readOnly: true
    volumeAttributes:
      volumePath: "/user/root"
      cluster: "clusterA"
      cldbHosts: "10.10.10.210"
      securityType: "secure"
```

Resolved Issues

Issue	Description
K8S-844	MapR K8S log rotation policy and clearing of unused mounts
K8S-1068	Ability to modify the fuse.conf per container basis.
K8S-1208	PODs fail to mount MapR with /opt/mapr/k8s/hostname empty file error.

MapR Container Storage Interface (CSI) Storage Plugin Release 1.0

These notes describe the first release of the MapR Container Storage Interface (CSI) Storage Plugin.

You may also be interested in the Kubernetes Release Notes.

Version	1.0
Release Date	February 2019
MapR Version Interoperability	Compatible with MapR 6.1.0 or later.
Kubernetes Compatibility	Kubernetes 1.13.0 and later.*
OpenShift Compatibility	4.1 and 4.2
CSI Driver Downloads	See Downloads (CSI) on page 225 for more information.
Documentation	MapR Container Storage Interface (CSI) Storage Plugin Overview on page 666
Related Resources	https://www.hpe.com/us/en/resource-library.html

***** Kubernetes alpha features are not supported.

New in this Release

This first release of the MapR Container Storage Interface (CSI) Storage Plugin includes `.yaml` configuration files that can be installed onto a Kubernetes cluster. Once installed, these containers provide a CSI Driver for the MapR File System volume plug-in and a Kubernetes Dynamic Volume Provisioner that permit static and dynamic provisioning of MapR storage from Kubernetes.

Fixes

None.

Limitations

Note the following limitations:

- CSI Driver version 1.0 does not support coexistence with the FlexVolume Driver on the same Kubernetes cluster.
- All nodes in the Kubernetes cluster must use the same Linux OS. Configuration files are available to support the following Linux distributions:
  - CentOS
  - Red Hat (use CentOS configuration file)
  - Ubuntu
- The Basic POSIX client package is included by default when you install the MapR Container Storage Interface (CSI) Storage Plugin. The Platinum POSIX client can be enabled by specifying a parameter in the Pod spec. Only the FUSE-based POSIX client is supported. NFSv3 and NFSv4 are not supported.

- The CSI Driver does not include support for inline volumes in pods. It only supports PersistentVolumes.

Known Issues

Note the following known issues:

- On nodeplugin Pod restart or upgrade scenario, the existing POSIX client(s) running in the CSI Driver container are killed. The workaround is to move/stop the container workload using MapR CSI Storage Plugin, restart/update the MapR CSI Storage Plugin and start using the MapR CSI Storage Plugin again.
- On Provisioner restart, Provisioner loses the information about the REST server where volume or snapshot should be deleted for existing volume and snapshots provisioned. The administrator must manually remove the volume and/or snapshot for provisioned volumes from the MapR Data Platform.
- Provisioned snapshot information is written to the provisioner log, but not available in the Kubernetes objects such as volumeSnapshots, VolumesnapshotContents etc.
- If you want read-only behavior, specify `readOnly` in the `volumeAttributes`. For example, the following is supported:

```
csi:
    nodePublishSecretRef:
      name: "mapr-ticket-secret"
      namespace: "test-csi"
    driver: com.mapr.csi-kdf
    volumeHandle: pv-securepv-test-read-only-id
    volumeAttributes:
      volumePath: "/user/root"
      cluster: "clusterA"
      cldbHosts: "10.10.10.210"
      securityType: "secure"
      readOnly: "true"
```

The following is not supported:

```
csi:
    nodePublishSecretRef:
      name: "mapr-ticket-secret"
      namespace: "test-csi"
    driver: com.mapr.csi-kdf
    volumeHandle: pv-securepv-test-read-only-id
    readOnly: true
    volumeAttributes:
      volumePath: "/user/root"
      cluster: "clusterA"
      cldbHosts: "10.10.10.210"
      securityType: "secure"
```

Resolved Issues

None.

MapR Data Fabric for Kubernetes FlexVolume Driver Release Notes

This section contains release notes for the MapR Data Fabric for Kubernetes FlexVolume Driver.

MapR Data Fabric for Kubernetes Release 1.1.0

These notes describe version 1.1.0 of the MapR Data Fabric for Kubernetes.

You may also be interested in the Kubernetes documentation.

Version	1.1.0
Release Date	December 2018
MapR Version Interoperability	Compatible with MapR 5.2.2 or later.  **Note:** If your installation requires MapR and Kubernetes software to coexist on the same nodes, you must use one of these versions: • Version 1.0.1 with MapR 6.0.1 or later • Version 1.1.0 with MapR 6.1.0 or later
OS Compatibility	The operating system (OS) on a node where the volume plug-in is installed must be a supported OS for the MapR version. For a list of supported OS versions, see Operating System Support Matrix on page 5522.
Kubernetes Compatibility	Kubernetes 1.9 or later.  **Note:** Kubernetes alpha features are not supported.
MapR Software Downloads	MapR installation (`.yaml`) files are located here: https://package.mapr.com/tools/KubernetesDataFabric
Source on GitHub	This repository contains Docker images, installation files, and examples: https://github.com/mapr/KubernetesDataFabric
Docker Hub	Docker containers for the MapR installation files are located here: • https://hub.docker.com/r/maprtech/kdf-provisioner/ • https://hub.docker.com/r/maprtech/kdf-plugin/
Documentation	MapR Data Fabric for Kubernetes FlexVolume Driver Overview on page 671
Related Resources	https://mapr.com/solutions/data-fabric/kubernetes/

New in This Release

Version 1.1.0 of the MapR Data Fabric for Kubernetes includes a new plug-in and provisioner and uses the updated version of the FUSE POSIX client included in MapR 6.1.0. Version 1.1.0 can be used on cluster nodes that:

- Are installed with MapR software only (MapR 5.2.2 or later)
- Have both MapR software (MapR 6.1.0 or later) and Kubernetes software

This release of the MapR Data Fabric for Kubernetes includes a set of Docker containers and their respective `.yaml` configuration files that can be installed onto a Kubernetes cluster. Once installed, these containers provide a Kubernetes FlexVolume Driver for MapR File System and a Kubernetes Dynamic Volume Provisioner that permit static and dynamic provisioning of MapR storage from Kubernetes.

To upgrade a previously installed version of the plug-in and provisioner to version 1.1.0, see Upgrading the MapR Data Fabric for Kubernetes on page 248.

Fixes

None.

Known Issues and Limitations

Note these limitations:

- Installations that require MapR and Kubernetes software to coexist on the same nodes must use one of the following:
  - Version 1.0.1 with MapR 6.0.1 or later
  - Version 1.1.0 with MapR 6.1.0 or later
- All nodes in the Kubernetes cluster must use the same Linux OS. Configuration files are available to support these Linux distributions:
  - CentOS
  - RedHat (use CentOS configuration file)
  - SSE (use CentOS configuration file)
  - Ubuntu
- Docker for Mac with Kubernetes is not supported as a development platform for containers used with the MapR Data Fabric for Kubernetes.
- Volume plug-in files are supported for:
  - CentOS
  - Ubuntu
  - Microsoft Azure AKS
  - Red Hat OpenShift**
  - Google Kubernetes Engine (GKE)
- Amazon EKS is not supported.
- The Basic POSIX client package is included by default when you install the MapR Data Fabric for Kubernetes. The Platinum POSIX client can be enabled by specifying a parameter in the Pod spec. Only the POSIX client is supported. NFSv3 is not supported.

**OpenShift Origin is supported because it supports Kubernetes 1.9. The OpenShift Container Platform (formerly known as OpenShift Enterprise) can be used only if it supports Kubernetes 1.9 or later.

Resolved Issues

Issue	Description
K8S-310	Node restart causes the KDF driver to not update the volume mount.
K8S-315	Version 1.1.0 uses the FUSE POSIX client included in MapR 6.1.0.
K8S-332	On an upgrade from a previous version of the volume plug-in, POSIX can fail with the following error in the POSIX log file: `Create/Attach to stats shared memory failed.`
K8S-353	Kubelet restart causes the existing volume mount to fail.

Issue	Description
K8S-362	Version 1.1.0 provides compatibility with MapR 6.1.0 or earlier volume attributes, as provided in the storage class.
K8S-399	Version 1.1.0 provides ReadWriteMany support with the KDF volume driver.

MapR Data Fabric for Kubernetes Release 1.0.2
These notes describe version 1.0.2 of the MapR Data Fabric for Kubernetes.

You may also be interested in the Kubernetes documentation.

Version	1.0.2
Release Date	July 2018
MapR Version Interoperability	Compatible with MapR 5.2.2 or later.  **Note:** Version 1.0.2 does *not* support the coexistence of MapR and Kubernetes software on the same nodes. If your installation requires MapR and Kubernetes software to coexist on the same nodes, see the MapR Data Fabric for Kubernetes release notes to identify the latest version that supports coexistence.
OS Compatibility	The operating system (OS) on a node where the volume plug-in is installed must be a supported OS for the MapR version. For a list of supported OS versions, see Operating System Support Matrix on page 5522.
Kubernetes Compatibility	Kubernetes 1.9 or later.  **Note:** Kubernetes alpha features are not supported.
MapR Software Downloads	MapR installation (`.yaml`) files are located here: https://package.mapr.hpe.com/tools/KubernetesDataFabric
Source on GitHub	This repository contains Docker images, installation files, and examples: https://github.com/mapr/KubernetesDataFabric
Docker Hub	Docker containers for the MapR installation files are located here: • https://hub.docker.com/r/maprtech/kdf-provisioner/ • https://hub.docker.com/r/maprtech/kdf-plugin/
Documentation	MapR Data Fabric for Kubernetes FlexVolume Driver Overview on page 671
Related Resources	https://mapr.com/solutions/data-fabric/kubernetes/

New in This Release

Version 1.0.2 of the MapR Data Fabric for Kubernetes can only be used on cluster nodes that are installed with MapR software (MapR 5.2.2 or later). Version 1.0.2 cannot be used on cluster nodes having both MapR and Kubernetes software.

This release of the MapR Data Fabric for Kubernetes includes a set of Docker containers and their respective `.yaml` configuration files that can be installed onto a Kubernetes cluster. Once installed, these containers provide a Kubernetes FlexVolume Driver for MapR File System and a Kubernetes Dynamic Volume Provisioner that permit static and dynamic provisioning of MapR storage from Kubernetes.

Fixes

None.

Known Issues and Limitations

Note these limitations:

- Version 1.0.2 does *not* support the coexistence of MapR and Kubernetes software on the same nodes. If your installation requires MapR and Kubernetes software to coexist on the same nodes, see the MapR Data Fabric for Kubernetes release notes to identify the latest version that supports coexistence.
- All nodes in the Kubernetes cluster must use the same Linux OS. Configuration files are available to support these Linux distributions:
  - CentOS
  - RedHat (use CentOS configuration file)
  - SSE (use CentOS configuration file)
  - Ubuntu
- Docker for Mac with Kubernetes is not supported as a development platform for containers used with the MapR Data Fabric for Kubernetes.
- Volume plug-in files are supported for:
  - CentOS
  - Ubuntu
  - Microsoft Azure AKS
  - Red Hat OpenShift**
  - Google Kubernetes Engine (GKE)
- Amazon EKS is not supported.
- The Basic POSIX client package is included by default when you install the MapR Data Fabric for Kubernetes. The Platinum POSIX client can be enabled by specifying a parameter in the Pod spec. Only the POSIX client is supported. NFSv3 is not supported.

**OpenShift Origin is supported because it supports Kubernetes 1.9. The OpenShift Container Platform (formerly known as OpenShift Enterprise) can be used only if it supports Kubernetes 1.9.

Resolved Issues

Issue	Description
K8S-139	Version 1.0.2 includes a fix for UID/GID handling in secure, non-impersonated environments. Before Version 1.0.2, if the UID/GID of the ticket was different from the UID/GID of the container, write operations could fail. With this fix, if the UID/GID of the ticket is different from the UID/GID of the container, all operations are performed using the UID/GID of the ticket.
K8S-164	In version 1.0.1, SELinux relabeling on the pod-container volume mounts caused an issue with the flexvolume-mounted filesystem. (SELinux relabeling is enabled by default for the volume plug-in in version 1.0.1.) In version 1.0.2, the volume plug-in resolves the issue by opting out of SELinux relabeling, reporting `selinux Relabel:false` in its `init` call.

last-updated: Apr 11, 2022

MapR Data Fabric for Kubernetes Release 1.0.1
These notes describe version 1.0.1 of the MapR Data Fabric for Kubernetes.

You may also be interested in the Kubernetes documentation.

Version	1.0.1
Release Date	May 2018
MapR Version Interoperability	Compatible with MapR 5.2.2 or later. **Note:** Version 1.0.1 supports installing MapR and Kubernetes software on the same nodes. However, not all versions of the MapR Data Fabric for Kubernetes support this feature. To identify other versions that support this feature, see the MapR Data Fabric for Kubernetes release notes.
OS Compatibility	The operating system (OS) on a node where the volume plug-in is installed must be a supported OS for the MapR version. For a list of supported OS versions, see Operating System Support Matrix on page 5522.
Kubernetes Compatibility	Kubernetes 1.9 or later. **Note:** Kubernetes alpha features are not supported.
MapR Software Downloads	MapR installation (`.yaml`) files are located here: https://package.mapr.hpe.com/tools/KubernetesDataFabric
Source on GitHub	This repository contains Docker images, installation files, and examples: https://github.com/mapr/KubernetesDataFabric
Docker Hub	Docker containers for the MapR installation files are located here: • https://hub.docker.com/r/maprtech/kdf-provisioner/ • https://hub.docker.com/r/maprtech/kdf-plugin/
Documentation	MapR Data Fabric for Kubernetes FlexVolume Driver Overview on page 671
Related Resources	https://mapr.com/solutions/data-fabric/kubernetes/

New in This Release

Version 1.0.1 of the MapR Data Fabric for Kubernetes can be used:

- On cluster nodes that are installed with MapR software only (MapR 5.2.2 or later)
- On cluster nodes having both MapR software (MapR 6.0.1 or later) and Kubernetes software

This release of the MapR Data Fabric for Kubernetes includes a set of Docker containers and their respective `.yaml` configuration files that can be installed onto a Kubernetes cluster. Once installed, these containers provide a Kubernetes FlexVolume Driver for MapR File System and a Kubernetes Dynamic Volume Provisioner that permit static and dynamic provisioning of MapR storage from Kubernetes.

Fixes

None.

Known Issues and Limitations

Note these limitations:

- Version 1.0.1 supports installing MapR 6.0.1 or later and Kubernetes software on the same nodes. However, not all versions of the MapR Data Fabric for Kubernetes support this feature. To identify other versions that support coexistence, see the MapR Data Fabric for Kubernetes release notes.
- All nodes in the Kubernetes cluster must use the same Linux OS. Configuration files are available to support these Linux distributions:
  - CentOS
  - RedHat (use CentOS configuration file)
  - SSE (use CentOS configuration file)
  - Ubuntu
- Docker for Mac with Kubernetes is not supported as a development platform for containers used with the MapR Data Fabric for Kubernetes.
- Volume plug-in files are supported for:
  - CentOS
  - Ubuntu
  - Microsoft Azure AKS
  - Red Hat OpenShift**
  - Google Kubernetes Engine (GKE)
- Amazon EKS is not supported.
- The Basic POSIX client package is included by default when you install the MapR Data Fabric for Kubernetes. The Platinum POSIX client can be enabled by specifying a parameter in the Pod spec. Only the POSIX client is supported. NFSv3 is not supported.

**OpenShift Origin is supported because it supports Kubernetes 1.9. The OpenShift Container Platform (formerly known as OpenShift Enterprise) can be used only if it supports Kubernetes 1.9.

Resolved Issues

None

MapR Data Fabric for Kubernetes Release 1.0

These notes describe the first release of the MapR Data Fabric for Kubernetes.

You may also be interested in the Kubernetes documentation.

Version	1.0
Release Date	March 2018
MapR Version Interoperability	Compatible with MapR 5.2.2 or later.
OS Compatibility	The operating system (OS) on a node where the volume plug-in is installed must be a supported OS for the MapR version. For a list of supported OS versions, see Operating System Support Matrix on page 5522.
Kubernetes Compatibility	Kubernetes 1.9 and later.*

last-updated: Apr 11, 2022

MapR Software Downloads	MapR installation (`.yaml`) files are located here: https://package.mapr.hpe.com/tools/KubernetesDataFabric
Source on GitHub	This repository contains Docker images, installation files, and examples: https://github.com/mapr/KubernetesDataFabric
Docker Hub	Docker containers for the MapR installation files are located here: • https://hub.docker.com/r/maprtech/kdf-provisioner/ • https://hub.docker.com/r/maprtech/kdf-plugin/
Documentation	MapR Data Fabric for Kubernetes FlexVolume Driver Overview on page 671
Related Resources	https://mapr.com/solutions/data-fabric/kubernetes/

*Kubernetes alpha features are not supported.

New in This Release

This first release of the MapR Data Fabric for Kubernetes introduces a set of Docker containers and their respective `.yaml` configuration files that can be installed onto a Kubernetes cluster. Once installed, these containers provide a Kubernetes FlexVolume Driver for MapR File System and a Kubernetes Dynamic Volume Provisioner that permit static and dynamic provisioning of MapR storage from Kubernetes.

Fixes

None.

Known Issues and Limitations

Note these limitations:

- Version 1.0 does *not* support the coexistence of MapR and Kubernetes software on the same nodes. If your installation requires MapR and Kubernetes software to coexist on the same nodes, see the MapR Data Fabric for Kubernetes release notes to identify the latest version that supports coexistence.
- All nodes in the Kubernetes cluster must use the same Linux OS. Configuration files are available to support these Linux distributions:
  - CentOS
  - RedHat (use CentOS configuration file)
  - SSE (use CentOS configuration file)
  - Ubuntu
- Docker for Mac with Kubernetes is not supported as a development platform for containers used with the MapR Data Fabric for Kubernetes.
- Volume plug-in files are supported for:
  - CentOS
  - Ubuntu
  - Microsoft Azure AKS
  - Red Hat OpenShift**

  - Google Kubernetes Engine (GKE)
- Amazon EKS is not supported.
- The Basic POSIX client package is included by default when you install the MapR Data Fabric for Kubernetes. The Platinum POSIX client can be enabled by specifying a parameter in the Pod spec. Only the POSIX client is supported. NFSv3 is not supported.

**OpenShift Origin is supported because it supports Kubernetes 1.9. The OpenShift Container Platform (formerly known as OpenShift Enterprise) can be used only if it supports Kubernetes 1.9.

Resolved Issues

None

Patches for Known Issues

A notice of known issues is maintained on the MapR Support website. The website indicates if patches or workarounds are available for an issue.

For patch information, visit the Support notices of known issues, some with patches or workarounds.

To download patches, see Downloading a Patch on page 437.

MapR PACC Release Notes

This section contains release notes for the MapR Persistent Application Client Container (PACC).

PACC 6.2.0_7.0.0 Release Notes

These notes describe the MapR Persistent Application Container Client (PACC) for core 6.2.0 with EEP 7.0.0.

You may also be interested in the MapR Data Science Refinery Release Notes on page 5637.

Version	6.2.0_7.0.0
Release Date	October 2020
MapR Version Interoperability	Compatible with release 6.2.0 and EEP 7.0.0
Docker Hub	Docker containers for the data-fabric installation files are located here: • https://hub.docker.com/r/maprtech/pacc/tags/
Documentation	About the MapR Persistent Application Client Container (PACC) on page 403
Related Resources	HPE Ezmeral Data Fabric

New in This Release

This release provides support for core 6.2.0 with EEP 7.0.0 and compatibility with core 6.1.0.

This release provides an option during image configuration to specify additional packages. If you prefer to use existing images from the Docker hub, you can use an environmental variable to specify packages that are installed during startup. This option increases startup time and requires an Internet connection.

For more information, see Creating a MapR PACC Image Using mapr-setup.sh on page 409.

Fixes

CORE-497: mapr-thin-client-6.2.0*.tar.gz: kill_client_processes() function from mapr-fuse script invokes killproc with incorrect params.

Known Issues and Limitations

CORE-498: After installation of the `mapr-thin-client*.tar.gz`, the `mapr-posix-client-container` fails to start, and the following message appears in the `/opt/mapr/logs/posix-client-container.log`:

```
Failed to call init on library /tmp/libMapRClient_c.so.0
```

This issue is fixed in release 6.2 but might affect earlier PACC releases.

Workaround: Create the following file (the file can be empty), and retry the operation:

```
/opt/mapr/hadoop/hadoop-2.7.4/etc/hadoop/core-site.xml
```

For additional known issues, see MapR PACC Known Issues on page 417.

Resolved Issues

None.

PACC 6.1.0_6.0.0 Release Notes

These notes describe the MapR 6.1.0 EEP 6.0.0 release of the MapR PACC.

You may also be interested in the MapR Data Science Refinery Release Notes on page 5637.

Version	6.1.0_6.0.0
Release Date	September 2018
MapR Version Interoperability	Compatible with MapR 6.1.0 and EEP 6.0.0
Docker Hub	Docker containers for the MapR installation files are located here: • https://hub.docker.com/r/maprtech/pacc/tags/
Documentation	About the MapR Persistent Application Client Container (PACC) on page 403
Related Resources	MapR Persistent Application Client Container

New in This Release

This release provides support for MapR 6.1 with EEP 6.0.0. There are no significant new features in the PACC image itself.

Release notes for previous PACC images are not currently available.

Fixes

None.

Known Issues and Limitations

For known issues, see MapR PACC Known Issues on page 417.

Resolved Issues

None.

Security Vulnerabilities

This section describes potential security vulnerabilities in MapR software. Where necessary, appropriate workarounds are provided.

On the Support Portal, you can sign up to receive proactive notices about vulnerabilities. See MapR Support Portal: How do I sign-up for proactive email advisories on critical issues?

Web Browser Security Issues

This section describes security issues with web browsers.

Web browsers and web servers often need to update their security requirements and configurations to ensure secure communication. Sometimes when web browser security requirements change, the browser is no longer able to connect to the Control System or other web interfaces.

The following fixes are available to resolve browser connection issues caused by changes in browser security requirements, or by an organization's need to maintain legacy (insecure) protocols:

Issue	Affects MapR Version
Weak Ephemeral Diffie-Hellman Key	3.x, 4.x, and 5.0
Unable to Establish a Secure Connection	3.1.x, 4.0.x
Requirement to Enable Insecure Protocols (Not Recommended)	5.1

Note: Based on your MapR version, you may need to apply the fix for more than one issue.

Unable to Establish a Secure Connection

This section describes secure connection issues.

Recent versions of Safari and Chrome web browsers have removed support for older certificate cipher algorithms, including those used by some versions of MapR. Because of this, users of these new browser versions may lose the ability to log into the Control System.

A fix for this issue is available in MapR Versions 4.0.2 and later. Existing clusters can be patched to workaround this issue. Information and installation instructions for this patch are found later in this document. For additional fixes that you may also want to apply at this time, see Web Browser Security Issues.

Affected Versions

To determine whether you will be affected, your MapR version must be in the range listed in the MapR section below, and you must be accessing the Control System using a browser version listed in either the Safari or Chrome sections.

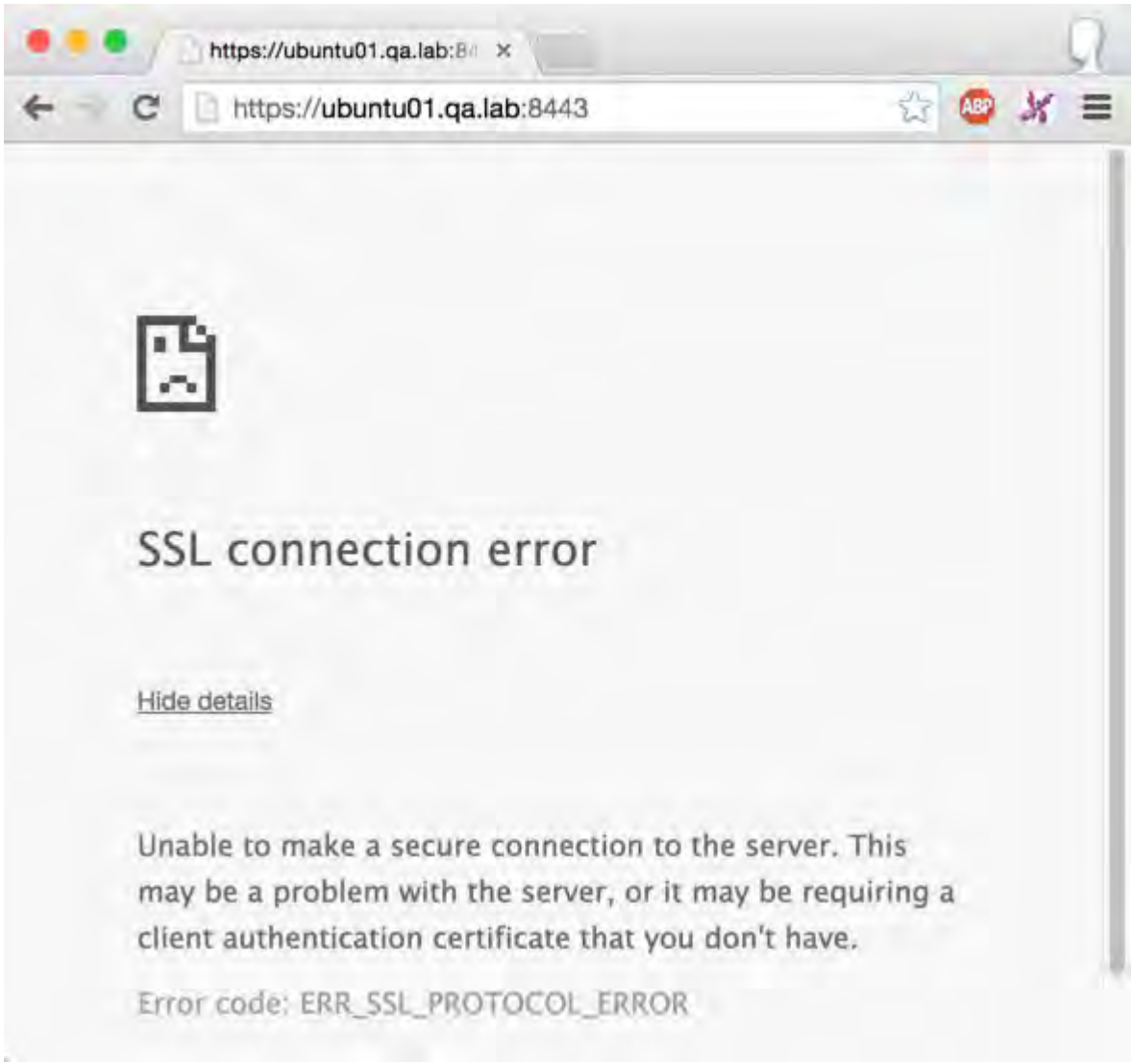
- MapR - Versions 3.1, 3.1.1, 4.0.0, and 4.0.1
- Safari - Versions 7.0 and higher.
- Chrome - Versions 39.0 and higher.

last-updated: Apr 11, 2022

Symptoms

Error message for Chrome:

```
          SSL connection error. Unable to make a secure connection to the
server.
          This maybe a problem with the server,
          or it may be requiring a client authentication certificate that
you don't have.
          Error code: ERR_SSL_PROTOCOL_ERROR
```



Error message for Safari

```
          Safari can't open the page <URL>
          because Safari can't establish a secure connection to the server
<server name>.
```


last-updated: Apr 11, 2022

Patching your Cluster

The steps to implement the fix for a secure cluster (cluster with wire-level security) differ from the steps to implement the fix on a non-secure cluster. However, in both cases, you will use the fixssl script to generate new versions of the ssl_keystore and ssl_truststore.

While you are implementing the fix on a non-secure cluster, the webserver will experience a brief downtime. The impact on a secure cluster will be greater, as more services will need to be restarted for the patch to take effect. You have a secure cluster if you use wire-level security to encrypt data transmission between the nodes in your cluster.

Patching a Non-Secure Cluster

The following steps show how to patch a secure cluster when you are unable to establish a secure connection. Once the fix is complete, no further action is required except to access the Control System and other web interfaces, such as the JobTracker UI and the ResourceManager UI.

1. Determine which nodes in the cluster run the webserver role. For example:

```
$ maprcli node list -columns configuredservice -filter
'[configuredservice==webserver]'
hostname  configuredservice                                    ip
centos21
webserver,nodemanager,cldb,fileserver,resourcemanager,hoststats
10.10.82.21
```

2. Perform the following steps on each webserver node:
   a) Download the script from the following location: https://package.mapr.hpe.com/scripts/mcs/

```
wget https://package.mapr.hpe.com/scripts/mcs/fixssl
```

   b) Run the following command to update the permissions on the file:

```
chmod 755 fixssl
```


last-updated: Apr 11, 2022

c) Run the following command to run the script:

```
sudo ./fixssl
```

Once you run the script, the following is displayed:

```
Webserver restarted. Issue should be resolved"
```

The fixssl script performs the following steps on a node in a secure cluster:

1. Updates manageSSLKeys.sh to use the new certificate cipher algorithm.
2. Backs up the existing certificates so that new versions can be generated with the new cipher algorithm:
   - `/opt/mapr/conf/ssl_keystore` is renamed to `/opt/mapr/conf/ssl_keystore_old`
   - `/opt/mapr/comf/ssl_truststore` is renamed to `/opt/mapr/comf/ssl_truststore_old`
3. Runs `/opt/mapr/server/configure.sh -R` to generate new versions of the keystore and truststore files.
4. Restarts the webserver.

Patching a Secure Cluster
Explains how to patch a secure cluster when you are unable to establish a secure connection.

Once the fix is complete, no further action is required except to access the Control System and other web interfaces, such as the JobTracker UI and the ResourceManager UI.

1. Perform the following steps on any cluster node:
   a) Download the script from the following location: https://package.mapr.hpe.com/scripts/mcs/ For example:

```
wget https://package.mapr.hpe.com/scripts/mcs/fixssl
```

   b) Run the following command to update the permissions on the file:

```
chmod 755 fixssl
```

c) Run the following command to run the script:

```
sudo ./fixssl
```

Once you run the script, the following is displayed

```
Creating 10 year self signed certificate with
subjectDN='CN=*.us-west-2.compute.internal'
Certificate stored in file </tmp/tmpfile-mapcert.3743>
Certificate was added to keystore

**********************************************************************
******************
* In order for your cluster to work, please copy the following files
in /opt/mapr/conf  *
* to all the nodes in the cluster, to the same directory:
ssl_keystore ssl_truststore   *
* After copying the files to the other nodes, please restart CLDB,
Webserver, and any   *
* other service that utilizes https (Jobtracker,
tasktracker)                                *
* (See doc for more details if you do not wish to have downtime in
your cluster)       *
**********************************************************************
******************
```

2. On each node in the cluster, back up existing certificates and copy the certificates to all other nodes in the cluster. For example:

```
$ maprcli node list -columns ip
 hostname ip
 ip-172-31-18-196.us-west-2.compute.internal 172.31.18.196
 ip-172-31-18-197.us-west-2.compute.internal 172.31.18.197
 ip-172-31-18-198.us-west-2.compute.internal 172.31.18.198
 ip-172-31-18-199.us-west-2.compute.internal 172.31.18.199
 ip-172-31-18-200.us-west-2.compute.internal 172.31.18.200

$ ssh 172.31.18.200 "mv /opt/mapr/conf/ssl_keystore /opt/mapr/conf/
ssl_keystoreold"

$ ssh 172.31.18.200 "mv /opt/mapr/conf/ssl_truststore /opt/mapr/conf/
ssl_truststoreeold"

 $ scp /opt/mapr/conf/ssl_keystore /opt/mapr/conf/ssl_truststore
mapr@172.31.18.200:/opt/mapr/conf
```


last-updated: Apr 11, 2022

3. Restart the CLDB secondary services. To do this, first you determine which cluster nodes are running the CLDB service and then determine which node is running the primary CLDB. The secondary instances are the non-primary CLDB nodes. For example:

```
$ maprcli node list -columns configuredservice -filter
'[configuredservice==cldb]'
 hostname
configuredservice                                    ip
 ip-172-31-18-198.us-west-2.compute.internal
webserver,cldb,fileserver,nfs,hoststats,jobtracker  172.31.18.198
 ip-172-31-18-199.us-west-2.compute.internal
webserver,cldb,fileserver,nfs,hoststats,jobtracker  172.31.18.199
 ip-172-31-18-200.us-west-2.compute.internal
webserver,cldb,fileserver,nfs,hoststats,jobtracker  172.31.18.200

 $ maprcli node cldbmaster

cldbmaster

 ServerID: 8868598593037642491 HostName:
ip-172-31-18-199.us-west-2.compute.internal

 $maprcli node services -cldb restart -nodes 172.31.18.198
172.31.18.200
```

4. Restart half of the TaskTracker and Nodemanager services.
   a) List all TaskTracker or NodeManager Hosts. For example:

```
 $ maprcli node list -columns configuredservice -filter
'[configuredservice==tasktracker]or[configuredservice==nodemanager]'
 hostname
configuredservice                       ip
 ip-172-31-18-196.us-west-2.compute.internal
fileserver,tasktracker,nfs,hoststats  172.31.18.196
 ip-172-31-18-197.us-west-2.compute.internal
fileserver,tasktracker,nfs,hoststats  172.31.18.197
```

   b) Restart TaskTracker and NodeManager services on half of the nodes that run those services. For example, the following command will restart both TaskTracker and NodeManager services on all nodes specified. If either service is not configured on that node, it will ignore it.

```
 $ maprcli node services -multi '[{ "name": "tasktracker", "action":
"restart"}, { "name": "nodemanager", "action": "restart"}]' -nodes
172.31.18.196
 ERROR (10002) -  Service: nodemanager is not configured on node:
ip-172-31-18-196.us-west-2.compute.internal
```

5. Restart JobTracker and ResourceManager services.

last-updated: Apr 11, 2022

a) List all nodes running JobTracker or ResourceManager. For example:

```
$ maprcli node list -columns configuredservice -filter
'[configuredservice==jobtracker]or[configuredservice==resourcemanager]
'
hostname
configuredservice                                      ip
ip-172-31-18-198.us-west-2.compute.internal
webserver,cldb,fileserver,nfs,hoststats,jobtracker  172.31.18.198
ip-172-31-18-199.us-west-2.compute.internal
webserver,cldb,fileserver,nfs,hoststats,jobtracker  172.31.18.199
ip-172-31-18-200.us-west-2.compute.internal
webserver,cldb,fileserver,nfs,hoststats,jobtracker  172.31.18.200
```

b) Restart JobTracker and ResourceManager services. For example, the following command will restart both JobTracker and ResourceManager services on the specified nodes. If either service is not configured on that node, it will ignore it.

```
$ maprcli node services -multi '[{ "name": "jobtracker",
"action": "restart"}, { "name": "resourcemanager", "action":
"restart"}]' -nodes 172.31.18.198 172.31.18.199 172.31.18.200
ERROR (10002) -  Service: resourcemanager is not configured on node:
ip-172-31-18-199.us-west-2.compute.internal
ERROR (10002) -  Service: resourcemanager is not configured on node:
ip-172-31-18-200.us-west-2.compute.internal
ERROR (10002) -  Service: resourcemanager is not configured on node:
ip-172-31-18-198.us-west-2.compute.internal
```

6. Restart remaining TaskTracker and NodeManager services. For example, the following command will restart both TaskTracker and NodeManager services on the specified nodes. If either service is not configured on that node, it will ignore it.

```
$ maprcli node services
-multi '[{ "name": "tasktracker", "action": "restart"}, { "name":
"nodemanager", "action": "restart"}]'
-nodes 172.31.18.197 ERROR (10002) -  Service: nodemanager is not
configured on node: ip-172-31-18-197.us-west-2.compute.internal
```

7. Restart additional secure services (Oozie, HistoryServer, Webserver, HiveServer2, Hue). For example, the following command can be run with the IPs or hostnames of all nodes in the cluster, as it will only restart the services that it finds:

```
$ maprcli node services
                    -multi '[{ "name": "hue", "action": "restart"},
                        { "name": "historyserver", "action": "restart"},
                        { "name": "webserver", "action": "restart"},
                        { "name": "oozie", "action": "restart"},
                        { "name": "hs2", "action": "restart"}]'
                    -nodes 172.31.18.198 172.31.18.199 172.31.18.200
172.31.18.196 172.31.18.197
```

8. Restart CLDB primary service. For example:

```
$ maprcli node cldbmaster
cldbmaster

ServerID: 8868598593037642491 HostName:
ip-172-31-18-199.us-west-2.compute.internal

$ maprcli node services -cldb restart -nodes 172.31.18.199
```

The fixssl script performs the following steps on a node in a secure cluster:

1. Updates manageSSLKeys.sh to use the new certificate cipher algorithm.
2. Backs up the existing certificates so that new versions can be generated with the new cipher algorithm:
   - `/opt/mapr/conf/ssl_keystore` is renamed to `/opt/mapr/conf/ssl_keystore_old`
   - `/opt/mapr/comf/ssl_truststore` is renamed to `/opt/mapr/comf/ssl_truststore_old`
3. Runs the following command to generate new versions of the keystore and truststore files:

```
/opt/mapr/manageSSLKey.sh  create -N <clustername> -ug
<maprusername>:<maprgroup>
```

   - The cluster name is retrieved from `/opt/mapr/conf/mapr-clusters.conf`.
   - The `mapr` user and `mapr` group is retrieved from `/opt/mapr/conf/daemon.conf`.

Weak Ephemeral Diffie-Hellman Key

Recently, some web browsers have updated their list of supported cipher algorithms which are used to ensure secure communication between the browser and web server. Due to this update, new browser versions may lose the ability to login to the Control System and other web interfaces since the ciphers supported by the web browser do not match the ciphers supported by the web servers.

Affected Versions

- MapR - Versions 3.x, 4.x, and 5.0
- Browsers - Latest versions such as Chrome 45 and Firefox 39

Symptoms

Users might see the following error messages if they encounter the issue:

Table

Browser	Error Message
Firefox	`An error occurred during a connection to <ip>:<port>. SSL received a weak ephemeral Diffie-Hellman key in Server Key Exchange handshake message. (Error code: ssl_error_weak_server_ephemeral_dh_key)`
Chrome	`Server has a weak ephemeral Dillie-Heffman public key` or `ERR_SSL_WEAK_EPHEMERAL_DH_KEY`

How to Fix the Issue

Based on the MapR Cluster version that you have, perform one of the following options to fix the issue:

last-updated: Apr 11, 2022

Table

MapR Version	Option(s)
4.x and 5.0	Apply the latest patch on every node in the cluster. -or- Edit the core-site.xml file on each node with a service that runs a web server.
3.x	Edit the core-site.xml file on each node with a service that runs a web server.

Applying Patches to Resolve Browser Connection Issues

With this option, you apply the latest patch on every node in the cluster and then restart all the services.

1. Download the latest patch from the following location: `https://package.mapr.hpe.com/patches/releases/v<version>/<os>/`

   For example: `wget https://package.mapr.hpe.com/patches/releases/v4.1.0/redhat/mapr-patch-4.1.<buildversion>.rpm`

2. Apply the patch on every node in the cluster.

Editing core-site.xml

With this option, you update the core-site.xml on each node with a service that runs a web server such as WebServer (Control System), ResourceManager, and HistoryServer nodes. Then, restart the services associated with the web servers. For example, you would need to restart the webserver service on the node that runs the Control System.

1. Add the following configuration to the core-site.xml on each node with a service that runs a web server:

```
<property>
  <name>hadoop.ssl.exclude.cipher.suites</name>
  <value>
SSL_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA,SSL_RSA_EXPORT_WITH_DES40_CBC_SHA,S
SL_RSA_EXPORT_WITH_RC4_40_MD5,TLS_DHE_RSA_WITH_AES_128_CBC_SHA,TLS_DHE_RS
A_WITH_AES_256_CBC_SHA,TLS_DHE_RSA_WITH_AES_256_CBC_SHA256,TLS_DHE_DSS_WI
TH_AES_256_CBC_SHA256,TLS_DHE_DSS_WITH_AES_256_CBC_SHA,TLS_DHE_RSA_WITH_A
ES_128_CBC_SHA256,TLS_DHE_DSS_WITH_AES_128_CBC_SHA256,TLS_DHE_DSS_WITH_AE
S_128_CBC_SHA
 </value>
</property>
```

   - For MapR 3.x clusters, the core-site.xml file is in the following location: /opt/mapr/hadoop/hadoop-0.20.2/conf/
   - For MapR 4.x and 5.x clusters, the core-site.xml file is in the following location: /opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop/

2. Restart services associated with web servers.

   For example:

   - To restart the Control System webserver: `maprcli node services -webserver restart -nodes <webserver nodes>`
   - To restart the ResourceManager service(s): `maprcli node services -name resourcemanager -action restart -nodes <space delimited list of resourcemanager nodes>`

last-updated: Apr 11, 2022

Requirement to Enable Insecure Protocols

MapR disables insecure protocols by default. For example, TLSv1 and SSLv3 are disabled by default due to their associated security risks. In the event that your client environment or crypto libraries cannot be upgraded, you can decide to enable insecure protocols.

Note: Enabling insecure protocols is not recommended as the security of communications between the browser and web server is put at risk.

To enable insecure protocols:

1. Based on your requirements, add one of the following configurations to the core-site.xml file on each node with a service that runs a web server:
   - To enable SSLv3:

     ```
     <property>
      <name>hadoop.ssl.exclude.insecure.protocols</name>
      <value>SSLV3</value>
     </property>
     ```

   - To enable TLSv1:

     ```
     <property>
      <name>hadoop.ssl.exclude.insecure.protocols</name>
      <value>TLSV1</value>
     </property>
     ```

   - To enable all insecure protocols that MapR disables by default:

     ```
     <property>
      <name>hadoop.ssl.exclude.insecure.protocols</name>
      <value></value>
     </property
     ```

   The core-site.xml is in the following location: `/opt/mapr/hadoop/hadoop-2.x.x/etc/hadoop/`
2. Restart services associated with web servers.

   Examples:
   - To restart the Control System webserver:

     ```
     maprcli node services -webserver restart -nodes <webserver nodes>
     ```

   - To restart the ResourceManager service(s):

     ```
     maprcli node services -name resourcemanager -action restart -nodes
     <space delimited list of resourcemanager nodes>
     ```

Previous Versions

This page contains links to the MapR documentation for releases that are currently supported or have recently reached end-of-life.

- MapR 6.0

last-updated: Apr 11, 2022

- MapR 5.2
- MapR 5.1

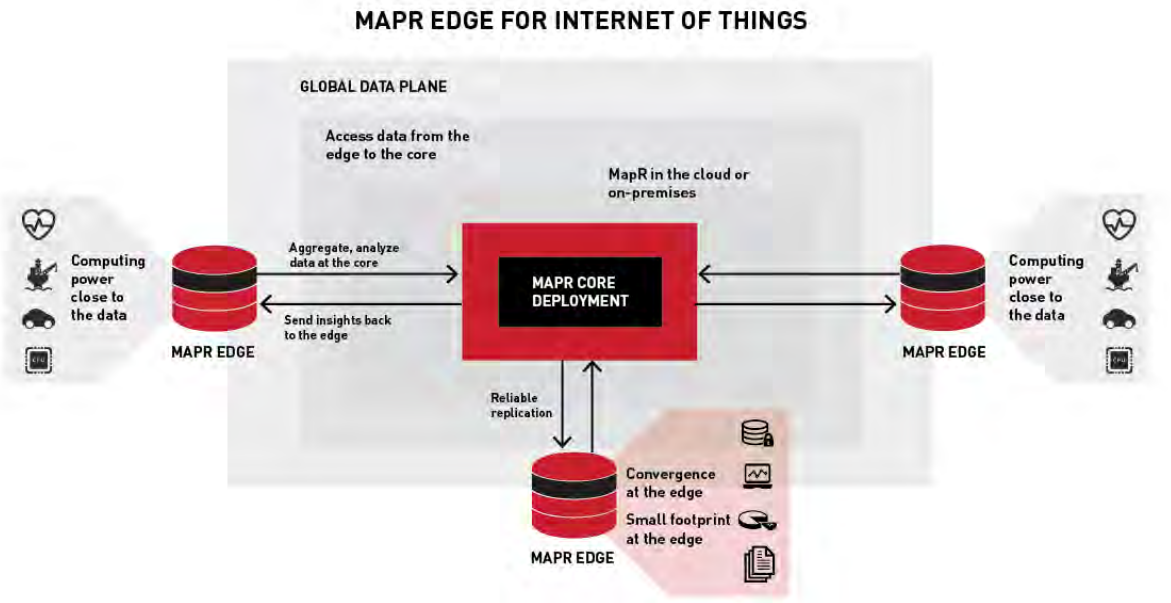
The "Other Docs" section contains MapR 6.1.0 documentation and documentation that applies to multiple releases (such as the interoperability matrix, release notes for ecosystem projects, user documentation for the MapR Installer, and information about security vulnerabilities).

MapR Edge

This section contains information about MapR Edge, which is a small footprint edition of the MapR Converged Data Platform designed to capture, process, and analyze IoT data close to the source.

MapR Edge is a small footprint edition of the MapR Converged Data Platform that you can use to capture, process, and analyze IoT data close to the source.

MapR Edge is a fully-functional MapR cluster that can be run on small form-factor commodity hardware, such as Intel NUCs. Edge clusters are supported in three- to five-node configurations. Each cluster supports the full capabilities of the MapR Converged Data Platform, including the capacity for files, tables, and streams, along with related data management and protection capabilities such as security, snapshots, mirroring, replication, and compression.



Installation, Configuration, and Management

You install, configure, and manage MapR Edge clusters and nodes the same way you handle traditional MapR clusters and nodes. Each cluster is managed and monitored independently.

When installing a MapR Edge cluster, you must ensure to configure the nodes according to the guidelines in the table below - MapR Edge Supported Cluster Configuration. Pay special attention to node hardware minimums and maximums, the number of supported storage pools, and caveats around upgrades and failure tolerance for different cluster sizes.

All MapR Edge clusters must be deployed in conjunction with a core MapR cluster. You must use one or more of the MapR Converged Data Platform's data replication features to synchronize data from Edge to Core, such as MapR mirroring, MapR Database table replication, or MapR Streams replication.

last-updated: Apr 11, 2022

MapR Edge Supported Cluster Configuration

Before you architect your system to use MapR Edge, consider these supported-configuration specifications:

Specification	Value		
Number of Nodes[1] Per Cluster	Three[2]	Four[2]	Five
Max No of Data Drives Per Node	Up to 4		
Storage Capacity (Usable)[3]	Min : 64GB Max : 10TB		
Replication Factor	Up to 3X		
No. of Storage Pools (per Node)	1		
Cluster Failure Tolerance [4]	1 node		2 nodes
Node Config Types	Homogeneous[5]		
Boot Disk Per Node	1 (Min 20GB)		
Processing Services	Spark, Drill		
Included Software	MapR File System, MapR Database, MapR Streams		
Node Hardware Specs	CPU-Type : x86(64Bit), Cores: 2 - 4, RAM: 16 - 32 GB, Disk-Type : SATA,SAS,SSD, vDisk Speed: 1Gb minimum, 10Gb		
Online Software Upgrade and Patching	Offline Upgrade or Rolling Upgrade		
Supported MapR Software Version	MapR 5.2 and later		
Supported Deployments	Bare Metal or Virtual Instances		

[1] Node is defined as “data node”, or node running a FileServer process, and responsible for storing data on behalf of the entire cluster. Nodes deployed with control-only services like CLDB and ZooKeeper do not count towards minimum node count, as they do not contribute to overall availability of data

[2] Clusters with less than 5 nodes may exhibit variable performance, especially during times of failure recovery when node resources are consumed with re-replication of data.

[3] Usable storage defined by total disk size divided by replication factor.

[4] This defines how many failures a cluster can sustain and still keep the cluster accessible to its clients/apps. Definition of failure includes anything that makes a node become unavailable, including hardware failure, software failure, disk failure, network failure, or power failure. MapR cannot assure data integrity for any additional failures beyond this count.

[5] All nodes must be exactly same in terms of capacity, including number of drives, amount of memory, type of cpu, etc.

Additional Design Considerations for Edge Clusters

The table above lists several unique considerations for clusters of less than 5 nodes. Carefully design your deployment to achieve a particular RPO/RTO, taking these considerations into account. Some strategies for increasing availability and RPO/RTO in case of smaller clusters include -

- Continuously moving critical data from the edge cluster to a core cluster using MapR replication features like mirroring, MapR Database table replication, and MapR Streams replication. This minimizes RPO/RTO in case of multi-failure scenarios.

last-updated: Apr 11, 2022

- Limiting reliance on any single point of failure infrastructure, such as chassis, power source, disk, or network device. Power and network redundancy are strongly recommended. This decreases the likelihood of a multiple failure scenario.

Installing the File Migration Service on the Edge Cluster

Describes how to install the File Migration service on CentOS nodes.

To install the package on CentOS, run the following command:

```
yum install mapr-filemigrate
```

Note: The File Migration service is only supported on CentOS. There are no packages for Debian-based OS.

Since the File Migration service provides for high availability, you can install it on as many nodes as you wish. Warden will ensure that it is always active on exactly one node.

After installation, a sample properties file, `FileMigrate.properties.default`, with default values is placed in the `/opt/mapr/filemigrate/filemigrate-1.0.0/conf` directory. To manually edit the properties file, follow the steps for Configuring the File Migration Service Using the Properties File on page 1009. Using the UI to configure the service, automatically creates the necessary file in the `maprfs:///var/mapr/filemigrate` directory. For more information, see Configuring File Migration Service Using the UI on page 1007.

Migrating Files From MapR Edge Cluster to AWS S3

The File Migration service for MapR Edge on page 6579 clusters monitors a set of configured directory trees (referred to as *policies* in the service UI) on the MapR cluster for new and changed files. When the service detects new or changed files, it automatically uploads them to AWS S3 (Amazon's storage environment) using the AWS SDK TransferManager object. The File Migration service is a long-running process that is controlled by Warden.

Note: For information on installing File Migration Service on your Edge cluster, see Installing the File Migration Service on the Edge Cluster on page 222.

Product Licensing

Provides information related to product licensing.

What's Included

Describes the products packaged with HPE Ezmeral Data Fabric.

HPE Ezmeral Data Fabric

HPE Ezmeral Data Fabric includes the following products and features:

HPE Ezmeral Data Fabric File Store (Formerly MapR-XD Enterprise Premier SSD and HDD)	• Auto-balancing • Cloud File Migration • Compression • Container Client including PACC, Flex Volume Plugin, CSI (unlimited client instances)* • Data Protection Replication

last-updated: Apr 11, 2022

	• Data tiering (Cold) to 3rd party external stores • Data tiering (Warm) to erasure coding • Data Topologies • Distributed datastore for files • External KMIP Keystore • Global Namespace • HDFS API and httpsfs • HPE Ezmeral Data Fabric Installer (Formerly MapR Installer) • HPE Ezmeral Data Fabric Management (Formerly MapR Management) using CLI, REST and GUI • HPE Ezmeral Data Fabric Monitoring (Formerly MapR Monitoring) • Mirroring of data (used for disaster recovery) • Multiple file server instances per node • Multi-site volume mirroring • Multi-tenancy • NFSv4 and NFSv3 • Platinum POSIX Client (unlimited client instances) • Policy-based security • POSIX Client (unlimited client instances) • Quotas • Resiliency and self-healing • Rolling upgrades • Snapshots • Support for HDD and SSD • Unified Security including authentication, authorization, encryption (wire and data-at-rest) and auditing
HPE Ezmeral Data Fabric Database (Formerly MapR Document Database Enterprise Premier)	(Requires purchase of HPE Ezmeral Data Fabric File Store) • Automatic compactions • Change Data Capture (CDC) • HPE Ezmeral Data Fabric DB Data Access Gateway (Formerly MapR-DB Data Access Gateway) • JSON document database using OJAI API

last-updated: Apr 11, 2022

	• Multi-master table replication • Multiple file servers instances per node • Multi-tenancy • Policy-based security • Resiliency and self-healing • Secondary indexes • Strong consistency • Table replication that can be used for disaster recovery and multi-site access • Unified Security including authentication, authorization, encryption (wire and data-at-rest) and auditing • Column-oriented database using HBase API
HPE Ezmeral Data Fabric Event Data Streams (Formerly MapR Event Data Streams Enterprise Premier)	(Requires purchase of HPE Ezmeral Data Fabric File Store) • Automatic partition balancing • Distributed publish-subscribe messaging infrastructure • Kafka API support • Kafka Connect • Kafka REST Proxy • Kafka Schema Registry • Kafka Streams • KSQL • Multi-site stream replication • Multi-tenancy • Unified Security including authentication, authorization, encryption (wire and data-at-rest) and auditing
HPE Ezmeral Data Fabric Analytics with Hadoop (Formerly MapR Analytics with Hadoop Enterprise Premier)	(Requires purchase of HPE Ezmeral Data Fabric File Store) • Apache Flume • Apache HBase • Apache Hive • Apache Hue • Apache MapReduce v2 (MapReduce v1 is not supported)

last-updated: Apr 11, 2022

	• Apache Oozie • Apache Pig • Apache Sqoop • Apache YARN • HPE Ezmeral Data Fabric DB OJAI Connector for Apache Hive (Formerly MapR-DB OJAI Connector for Apache Hive) • S3 Gateway on Ezmeral Data Fabric
HPE Ezmeral Data Fabric Advanced Analytics with Spark (Formerly MapR Advanced Analytics with Spark Enterprise Premier)	(Requires purchase of HPE Ezmeral Data Fabric File Store) • Apache Spark Core • Apache Spark MLlib • Apache Spark SQL • Apache Spark Streaming • Apache YARN • GraphX • HPE Ezmeral Data Fabric DB Binary Connector for Apache Spark (Formerly MapR-DB Binary Connector for Apache Spark) • HPE Ezmeral Data Fabric DB OJAI Connector for Apache Spark (Formerly MapR-DB OJAI Connector for Apache Spark) • HPE Ezmeral Data Fabric Streams Integration (Formerly MapR Streams Integration) • S3 Gateway on Ezmeral Data Fabric • SparkR • Spark Standalone and Spark on YARN support
HPE Ezmeral Data Fabric Interactive SQL Engine with Drill (Formerly MapR Interactive SQL Engine with Drill Enterprise Premier)	(Requires purchase of HPE Ezmeral Data Fabric File Store) • Drill Explorer • Drill Monitoring • Drill query and administration UI • Drill standalone and Drill-on-YARN support • File formats (Text,JSON,Parquet) • Impersonation • JDBC/ODBC drivers • Multiple data type support

last-updated: Apr 11, 2022

	• Queries on File • Queries on Hive tables and views • Queries on HPE Ezmeral Data Fabric Document Database tables and secondary indexes • REST API • S3 Gateway on Ezmeral Data Fabric • Schema-less ANSI-compliant distributed SQL query engine • SQLLine
HPE Ezmeral Data Fabric Platform Bundle (Formerly MapR Platform Bundle Enterprise Premier for SSD and HDD)	• HPE Ezmeral Data Fabric Advanced Analytics with Spark • HPE Ezmeral Data Fabric Analytics with Hadoop • HPE Ezmeral Data Fabric Database • HPE Ezmeral Data Fabric Event Data Streams • HPE Ezmeral Data Fabric File Store • HPE Ezmeral Data Fabric Interactive SQL Engine with Drill

HPE Ezmeral Container Platform

See What's Included in HPE Ezmeral Container Platform.

The HPE Ezmeral Data Fabric Converged Community Edition (Formerly MapR Converged Community Edition) is available free of cost with usage restrictions specified in the MapR End User License Agreement, and with community forum support.

HPE EZMERAL DATA FABRIC SOFTWARE LICENSING

Contains HPE Ezmeral Data Fabric software licensing information.

Your order includes both a license agreement and a quote. Detailed instructions for obtaining a software license key are available in the HPE support policy. Through the order package, HPE grants the licensee a nonexclusive license to use HPE Ezmeral Data Fabric software when the licensee lawfully obtains it, up to the level of authorized use specified in the customer contract.

SOFTWARE LICENSE KEYS

For each HPE Ezmeral Data Fabric installation, a software license key is created. This applies to both new and upgraded software. This license key is generated based on a cluster ID. The cluster ID is generated once the software is installed on a cluster.

HPE EZMERAL DATA FABRIC PRODUCT LICENSING

HPE licenses its software as a term-subscription for a fixed period of time that is outlined in the customer quote. Other terms that might be specific to your agreement will also be outlined in your quote. An HPE term-subscription typically authorizes the licensee to use the most current commercially available version, release, or update of HPE Ezmeral Data Fabric products.

last-updated: Apr 11, 2022

HPE Data Fabric products are sometimes sold based on capacity under management, which can be measured by terabyte or compute unit. The minimum for HPE Data Fabric File and Object Store is 250 terabytes of HDD or 100 terabytes of SSD, when purchased without other products. HPE Data Fabric requires a minimum of 5 compute units (or nodes) per cluster.

At the end of each fixed term [most commonly 36 months] the customer may choose to renew the licenses for an additional 36 months [at the prevailing price]. If the term-subscription is not renewed, the licensee will no longer have the rights to use the software, will no longer be entitled to the benefits of support, and must destroy all copies of the software.

DEFINITIONS

HDD Capacity Under Management Total hard disk capacity allocated to and managed by HPE Ezmeral Data Fabric products. Capacity Under Management is measured in terabytes 1TB.

SSD Capacity Under Management Total SSD capacity allocated to and managed by HPE Ezmeral Data Fabric products. Capacity Under Management is measured in TB. SSD is based on SATA and SAS interconnects and does not include PCIe-based NVME drives.

Client

A piece of computer software that embeds HPE Ezmeral Data Fabric software to access a service made available by an HPE Ezmeral Data Fabric server.

User

User means an individual authorized by the customer to use the software, regardless of whether the individual is actively using the programs at any given time.

Compute Unit

A compute unit is a server or virtual machine that does not exceed 1 motherboard, 4 CPU sockets, 32 total cores [including virtual cores) and 256GB of RAM. If a server or virtual machine exceeds any of these parameters, it will be counted as two or more compute units, depending on the factor by which the respective parameter[s) are exceeded.

Attribute	Quantity/Size
Motherboard	1
CPU Sockets	4
Sockets	32
Main memory	256GB

Motherboard

The motherboard is the main circuit board of your computer and is also known as the mainboard or logic board.

CPU Socket

The CPU socket is the connector on the motherboard that houses a CPU and forms the electrical interface and contact with the CPU.

CPU Core

Each physical processor contains smaller processing units called physical CPU cores. Some processors have two cores, some four, some six or eight, and so on.

Virtual Core

The unit of processing power in a virtual hardware system. A virtual core is the virtual representation of one or more hardware threads. The virtual Operating Systems Environments use one or more virtual cores. Note: for the purposes of licensing, 1 virtual cores will be counted as one physical core.

last-updated: Apr 11, 2022

Main Memory	The main memory is the area in a computer in which data is stored for quick access by the computer's processor. The term random access memory [RAM] often refers to this primary or main storage.
Node	A node is a server or virtual machine that does not exceed (a) one motherboard; (b) 4 CPU sockets; (c) 32 total cores; (d) 24 hard drives with up to 50 TB total hard drive capacity or 12 TB total flash or SSD capacity; (e) 2x10 GigE capacity; or (f) 256 GB of RAM. If a server or virtual machine exceeds any of these parameters, it will be counted as two or more nodes.
Per core licensing of HPE Ezmeral Data Fabric Platform Bundle	Each license allows the customer to deploy HPE Ezmeral Data Fabric for AI and Analytics on one Core and 2 terabytes of Storage Capacity. The customer must purchase more licenses if they exceed the allowable amount of Cores or Storage Capacity. Core means a part of a CPU that executes a single stream of compiled instruction code. Single-core processors can only process one instruction at a time. Multiple-core processors (CPUs) imply a processing system composed of two or more independent cores. Processing cores can be physical or virtual depending on whether the processor holding the cores is embedded in a physical machine or a virtual machine. For purposes of licensing, two virtual cores is equal to one physical core. Storage Capacity means the total storage capacity (HDD & SSD) allocated to and managed by HPE Products, measured in Terabytes (TB) of raw capacity. Includes space for data, data replication, erasure coding, snapshots, metadata, logs and other data that is stored in HPE Data Fabric.

HPE EZMERAL DATA FABRIC ADDITIONAL LICENSE AUTHORIZATION

Contains HPE Ezmeral Data Fabric additional licensing authorization information.

Last updated: January 17, 2020

THIS HPE EZMERAL DATA FABRIC ADDITIONAL LICENSE AUTHORIZATION ("ALA") IS BY AND BETWEEN HEWLETT PACKARD ENTERPRISE COMPANY ("HPE") AND ITS SUBSIDIARIES AND THE INDIVIDUAL OR LEGAL ENTITY USING THE APPLICABLE SOFTWARE MADE AVAILABLE BY HPE ("CUSTOMER") (WHETHER BY HPE OR AN AUTHORIZED HPE/MAPR PARTNER) AND GOVERNS ALL USE BY CUSTOMER OF THE HPE EZMERAL DATA FABRIC. IF LICENSED THROUGH AN AUTHORIZED HPE/MAPR PARTNER THIS ALA IS IN ADDITION TO AND SUPPLEMENTS THE HPE STANDARD EULA LOCATED AT https://www.hpe.com/us/en/software/licensing.html. THIS ALA (AND THE HPE STANDARD EULA IF LICENSED THROUGH AN AUTHORIZED HPE/MAPR PARTNER) ALSO SUPERSEDES ANY CLICKTHROUGH EULA EMBEDDED IN THE HPE EZMERAL DATA FABRIC. THIS ALA ALSO GOVERNS ALL USE BY CUSTOMER OF FREE SOFTWARE (AS DEFINED BELOW) PROVIDED BY HPE TO CUSTOMER. BY CLICKING ON THE "ACCEPT" BUTTON BELOW AND/OR A BUTTON OR CHECKBOX WITH SIMILAR DESIGNATION THAT DEMONSTRATES ACCEPTANCE OF THIS ALA (AND THE HPE STANDARD EULA IF LICENSED THROUGH AN AUTHORIZED HPE/MAPR PARTNER), OR BY DOWNLOADING, COPYING OR USING THE COMMERCIAL SOFTWARE OR FREE SOFTWARE, CUSTOMER EXPRESSLY ACCEPTS AND AGREES TO THE TERMS OF THIS ALA (AND THE HPE STANDARD EULA IF LICENSED THROUGH AN AUTHORIZED HPE/MAPR PARTNER), AND CONSENTS TO THE COLLECTION, USE AND TRANSFER OF DATA AS OUTLINED IN THE HPE PRIVACY STATEMENT (https://www.hpe.com/us/en/legal/privacy.html). CERTAIN PROVISIONS OF THIS ALA APPLY ONLY TO EITHER THE COMMERCIAL SOFTWARE OR THE FREE SOFTWARE, AS MORE PARTICULARLY SPECIFIED BELOW. BY WAY OF EXAMPLE, CUSTOMER MAY PURCHASE A COMMERCIAL SOFTWARE LICENSE KEY FROM HPE OR AN AUTHORIZED HPE/MAPR PARTNER AT ANY TIME AND CONVERT CUSTOMER'S COPY OF FREE SOFTWARE TO THE COMMERCIAL

SOFTWARE, IN WHICH CASE THE PROVISIONS APPLICABLE TO COMMERCIAL SOFTWARE WILL APPLY FROM THE TIME OF SUCH CONVERSION.

1. Definitions. The following capitalized terms shall have the meanings set forth below:

1.1. "Commercial Software" means the software identified in an order (either by HPE or an authorized HPE/MapR partner) and licensed for a fee, e.g., MapR Enterprise Edition or MapR Enterprise Database Edition software products when licensed for a fee. HPE may allow Customer to convert a copy of Free Software into Commercial Software by entering or installing a license Key for the Commercial Software purchased by Customer.

1.2. "Documentation" means the documentation and guides related to the Licensed Products freely available at https://docs.datafabric.hpe.com/home/.

1.3. "Feedback" means any comments or other feedback Customer may provide to HPE concerning the functionality and performance of the Licensed Products, including identification of potential errors and improvements.

1.4. "Free Software" means a software product that is provided by HPE to Customer free of charge for Customer's internal use for trial, evaluation, testing or similar non-production purposes, and is expressly identified by HPE as free, as evaluation software, as Not For Resale or NFR software, or any similar designation. For the purposes of this ALA, Free Software includes the MapR Community Edition software product or other MapR products made available by HPE on limited-time free, trial or Not For Resale basis.

1.5. "Free Software Term" means a thirty-day period of time that commences when Customer receives the applicable Free Software.

1.6. "Key" means the license key or similar control mechanism to help ensure compliance with the use and time limitations with respect to Licensed Products.

1.7. "Licensed Products" means the Commercial Software and Free Software.

1.8. License Metric: means the specific manner in which the applicable product(s), as defined in the MapR Licensing Data Sheet located at https://mapr.com/products/whats-included/assets/mapr-customerlicensing-01152020.pdf, are licensed.

1.9. "Open Source Software" means any third party software that is distributed as "free software", "open source software" or under a similar licensing or distribution model. Without limiting the generality of the foregoing, Apache Hadoop, Apache Solr and Apache Lucene are Open Source Software.

2. Standard Version. This Section 2 applies solely with respect to the Commercial Software, and not to Free Software:

2.1. License. Subject to the terms and conditions of this ALA, HPE hereby grants Customer a limited, non-exclusive, non-transferable, non-sublicensable license to install, copy and use the Commercial Software internally in the quantities set forth in the applicable License Metric quantity specified in the applicable order, during the applicable license term indicated in the order. For the avoidance of doubt, Customer may not install or use the Commercial Software on hardware which exceeds any License Metric quantities of the product component elements. For the further avoidance of doubt, Customer may not grant access to or transfer the use of the Commercial Software to any third party, whether on a standalone basis or as integrated into any other product, except with respect to third party consultants and service providers providing services to Customer.

2.2. Record Keeping. Customer shall establish and maintain complete and accurate records related to the location, access and use of the Commercial Software by Customer, its employees or its agents, and any such other information as reasonably necessary for HPE to verify compliance with the terms of this ALA. Such records shall be kept for at least 3 years following the end of the quarter to which they pertain.

3. Free Software.

3. This Section 3 applies solely with respect to Free Software, and not to the Commercial Software:

3.1. License. Subject to the terms and conditions of this ALA, HPE hereby grants Customer a limited, non-exclusive, non-transferable, non-sublicensable license to install, copy and use the Free Software

internally for trial, evaluation, testing or similar non-production purposes during the Free Software Term, subject to the use and time limitations specified by HPE, whether expressly or through the configuration of a Key. Customer may not grant access to or transfer the use of the Free Software to any third party, whether on a stand-alone basis or as integrated into any other product. If Customer decides to use the Free Software after the Free Software Term, Customer must obtain a license for the equivalent Commercial Software. If Customer decides not to obtain a license for the equivalent Commercial Software after the Free Software Term, Customer will cease using and will delete any such Free Software from its computer systems.

3.2. Termination of License. Either party may terminate the license granted in Section 3.1 for convenience upon 5 days notice.

3.3. No Support. Customer acknowledges that HPE is not obligated to provide any support, maintenance, updates or upgrades for and in connection with the Free Software.

3.4. Disclaimer. FREE SOFTWARE IS PROVIDED "AS-IS" AND WITHOUT ANY WARRANTY. CUSTOMER ACKNOWLEDGES AND AGREES THAT FREE SOFTWARE IS NOT SUITABLE FOR ANY PURPOSE OTHER THAN LIMITED INTERNAL TRIAL AND EVALUATION. HPE AND ITS LICENSORS AND SUPPLIERS SPECIFICALLY DISCLAIM ALL WARRANTIES, WHETHER IMPLIED, STATUTORY OR OTHERWISE, INCLUDING THE WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, TITLE, FITNESS FOR A PARTICULAR PURPOSE OR SATISFACTORY QUALITY, AND ANY AND ALL WARRANTIES ARISING FROM COURSE OF DEALING OR USAGE IN TRADE. NO ADVICE OR INFORMATION, WHETHER ORAL OR WRITTEN, OBTAINED FROM HPE OR ELSEWHERE SHALL CREATE ANY WARRANTY NOT EXPRESSLY STATED IN THIS ALA. HPE AND ITS LICENSORS AND SUPPLIERS DO NOT WARRANT THAT THE FREE SOFTWARE WILL OPERATE WITHOUT ERROR OR INTERRUPTION. CUSTOMER ASSUMES ALL RESPONSIBILITY FOR THE SELECTION OF THE FREE SOFTWARE OR A SPECIFIC VERSION THEREOF TO ACHIEVE CUSTOMER'S INTENDED RESULTS, AND FOR THE OPERATION, USE AND RESULTS OF THE FREE SOFTWARE. THE FREE SOFTWARE IS NOT DESIGNED, INTENDED OR WARRANTED FOR USE IN HAZARDOUS ENVIRONMENTS REQUIRING FAIL-SAFE CONTROLS, INCLUDING WITHOUT LIMITATION, OPERATION OF NUCLEAR FACILITIES, AIRCRAFT NAVIGATION OR COMMUNICATION SYSTEMS, AIR TRAFFIC CONTROL, AND LIFE SUPPORT OR WEAPONS SYSTEMS.

The following provisions of this ALA shall apply to all Licensed Products:

4. Keys, Other Restrictions and Data Collection Notice.

4.1. License Keys. Customer shall not destroy, disable or circumvent, or attempt to destroy, disable or circumvent in any way the Key and/or the use and time limitations set by the Key or any Licensed Products. Customer acknowledges and agrees that any attempt to exceed the use of the Licensed Products beyond the limits configured into the Key will automatically and immediately terminate the licenses granted under this ALA.

4.2 Information Collection. The Licensed Products may contain functionality that automatically collect information concerning the use and configuration of the HPE Ezmeral Data Fabric and overall capacity of your cluster and transmit that information to HPE. The Licensed Products will not access, collect, store or transmit any personally identifiable information or business data files residing in your computer environment as part of this functionality. The Licensed Products only initiate outbound communications to HPE and do not listen for inbound communications. The Licensed Products collect and transmit the information above by default. Unless otherwise provided in the applicable order, Customer has the ability to configure the Licensed Products to turn off the transmission of such information to HPE. HPE may use the information transmitted by the Licensed Products for activities such as determining usage for billing and license compliance and helping improve upon and market its product and service offerings. HPE may retain this information in perpetuity.

5. Open Source Software. The Licensed Products may incorporate or be provided together with Open Source Software. Copyrights and other proprietary rights to the Open Source Software are held by the copyright holders identified in the applicable distribution or the applicable help, notices, about or source files. All Open Source Software is distributed to Customer under the terms of the applicable open source

last-updated: Apr 11, 2022

license agreements referenced in the applicable distribution or the applicable help, notices, about or source files.

6. Feedback. Customer hereby assigns to HPE all right, title, and interest in and to the Feedback, if any.

7. Open Source Components. To the extent the Licensed Products includes open source licenses, such licenses shall control over this ALA with respect to the particular open source component. To the extent Licensed Products includes the GNU General Public License or the GNU Lesser General Public License: (a) the software includes a copy of the source code; or (b) if you downloaded the software from a website, a copy of the source code is available on the same website; or (c) if you send HPE written notice, HPE will send you a copy of the source code for a reasonable fee.

8. Australian Consumers. If you acquired the software as a consumer within the meaning of the 'Australian Consumer Law' under the Australian Competition and Consumer Act 2010 then despite any other provision of this ALA, the terms at this URL apply: http://www.hpe.com/software/SWLicensing.

9. Russian Consumers. If you are based in the Russian Federation and the rights to use the software are provided to you under a separate license and/or sublicense agreement concluded between you and a duly authorized HPE partner, then this ALA shall not be applicable.

HPE CUSTOMER PASS THROUGH TERMS FOR MAPR SOFTWARE AND SERVICES

Contains HPE customer pass through terms for MapR software and services information.

HPE's obligations with respect to products or services supplied by HPE and procured by an end-user customer (hereinafter “Customer”) from authorized HPE/MapR Business Partners are limited to the terms and conditions in these HPE CUSTOMER PASS THROUGH TERMS (“Terms”) and the specific Supporting Material included with the HPE supplied products and services. HPE is not responsible for the acts or omissions of HPE Business Partners, for any obligations undertaken by them or representations that they may make, or for any other products or services that they supply to Customer.

1. **Orders.** “Order” means the accepted order including any HPE/MapR-branded supporting material which is identified as incorporated either by attachment or reference (“Supporting Material”). Supporting Material may include (as examples) product lists, hardware or software specifications, end user license agreements, service descriptions, data sheets and their supplements and statements of work (SOWs), HPE Packaged Support Service Agreement, published warranties and service level agreements, and may be available to Customer in hard copy or by accessing a designated HPE/MapR website.

2. **Support Services.** HPE’s support services will be described in the applicable Supporting Material, which will cover the description of HPE’s offering, eligibility requirements, service limitations and Customer responsibilities, as well as the Customer systems supported.

3. **Professional Services.** HPE will deliver any ordered IT consulting, training, or other services as described in the applicable Supporting Material.

4. **Professional Services Acceptance.** The acceptance process (if any) will be described in the applicable Supporting Material, will apply only to the deliverables specified, and shall not apply to other products or services to be provided by HPE.

5. **Eligibility.** HPE’s service, support and warranty commitments do not cover claims resulting from:

    - 1. improper use, site preparation, or site or environmental conditions or other non-compliance with applicable Supporting Material;
    - 2. modifications or improper system maintenance or calibration not performed by HPE or authorized by HPE;
    - 3. failure or functional limitations of any non-HPE software or product impacting systems receiving HPE support or service;

last-updated: Apr 11, 2022

- 4. malware (e.g. virus, worm, etc.) not introduced by HPE; or
- 5. abuse, negligence, accident, fire or water damage, electrical disturbances, transportation by Customer, or other causes beyond HPE's control.

6. **Dependencies.** HPE's ability to deliver services will depend on Customer's reasonable and timely cooperation and the accuracy and completeness of any information from Customer needed to deliver the services.

7. **Services Performance.** Services are performed using generally recognized commercial practices and standards. Customer agrees to provide prompt notice of any such service concerns and HPE will re-perform any services that fail to meet this standard.

8. **Services with Deliverables.** If Supporting Material for services defines specific deliverables, HPE warrants those deliverables will conform materially to their written specifications for 30 days following delivery. If Customer notifies HPE of such non-conformity during the 30 day period, HPE will promptly remedy the impacted deliverables and Customer will return those deliverables to HPE.

9. **Remedies.** These Terms state all remedies for warranty claims. To the extent permitted by law, HPE disclaims all other warranties.

10. **Confidentiality.** Information exchanged under these Terms will be treated as confidential if identified as such at disclosure or if the circumstances of disclosure would reasonably indicate such treatment. Confidential information may only be used for the purpose of fulfilling obligations or exercising rights under these Terms, and shared with employees, agents or contractors with a need to know such information to support that purpose. Confidential information will be protected using a reasonable degree of care to prevent unauthorized use or disclosure for 3 years from the date of receipt or (if longer) for such period as the information remains confidential. These obligations do not cover information that: i) was known or becomes known to the receiving party without obligation of confidentiality; ii) is independently developed by the receiving party; or iii) where disclosure is required by law or a governmental agency.

11. **Limitation of Liability.** HPE's liability to Customer under these Terms is limited to $1,000,000. Neither Customer nor HPE will be liable for lost revenues or profits, downtime costs, loss or damage to data or indirect, special or consequential costs or damages. This provision does not limit either party's liability for: unauthorized use of intellectual property, death or bodily injury caused by their negligence; acts of fraud; willful repudiation of these Terms; nor any liability which may not be excluded or limited by applicable law.

12. **Force Majeure.** Neither party will be liable for performance delays nor for non-performance due to causes beyond its reasonable control.

13. **General.** These Terms represent our entire understanding with respect to its subject matter and supersede any previous communication or agreements that may exist. To the extent there is any conflict between these Terms and any Supporting Material, these Terms should apply. Modifications to these Terms will be made only through a written amendment signed by HPE and Customer. These Terms will be governed by the laws of the country of the HPE affiliate delivering services to the Customer the courts of that locale will have jurisdiction. Customer and HPE agree that the United Nations Convention on Contracts for the International Sale of Goods will not apply. Claims arising or raised in the United States will be governed by the laws of the state of California, excluding rules as to choice and conflict of law.

14. **Data Protection.** Each party shall comply with their respective obligations under applicable data protection legislation. To the extent HPE processes personal data on your behalf in the course of providing the services, the HPE Support Services – Data Privacy and Security Agreement found at www.hpe.com/info/customerprivacy.html shall apply.

last-updated: Apr 11, 2022

15. **Media Sanitization.** You are responsible for properly sanitizing or removing data from products that may be replaced or returned to HPE as part of the repair process to ensure the safeguarding of your data. For more information on your responsibilities, go to https://www.hpe.com/us/en/about/support-drivers/privacydataprotection.html.

Other Resources

This page provides links to additional resources such as on-demand training, videos, blogs, and the MapRMapR Data Platform community.

In addition to the product documentation, you may be interested in the following resources:

Training	https://learn.ezmeral.software.hpe.com/
Blogs and Videos	https://community.hpe.com/t5/hpe-ezmeral-uncut/bg-p/software#.XzXV2-hKg2w
Ezmeral Data Fabric Community	https://community.datafabric.hpe.com/s/
HPE Developer Community	https://developer.hpe.com/
Videos, Reports, and Case Studies	https://www.hpe.com/us/en/resource-library.html

Glossary

List of terms (with description) used in MapR documentation.

.dfs_attributes

A special file in every directory, for controlling the compression and chunk size used for the directory and its subdirectories.

.rw

A special mount point in the root-level volume (or read-only mirror) that points to the writable original copy of the volume.

.snapshot

A special directory in the top level of each volume, containing all the snapshots for that volume.

access control expression (ACE)

A Boolean expression that defines a combination of users, groups, or roles that have access to an object stored natively such as a directory, file, or MapR Database table.

Access Control Expression (ACE)

last-updated: Apr 11, 2022

Note: An ACE (up to 64KB in length) is a combination of users, groups, and/or roles for whom access (to volume data) is defined using boolean expressions and sub expressions within single quotes. When you pass in an access type that has already been set, the new value replaces the existing value for that access type. There is no change to access types that are not passed in with the command, whether or not they were set. For more information, see Managing Access Control Expressions on page 1448.

ACE

access control list (ACL)

A list of permissions attached to an object. An ACL specifies users or system processes that can perform specific actions on an object.

Access Control List (ACL)

Note: An Access Control List (ACL) is a list of users or groups. Each user or group in the list is paired with a defined set of permissions that limit the actions that the user or group can perform on the object secured by the ACL. In MapR, the objects secured by ACLs are the job queue, volumes, and the cluster itself.

ACL

accountable entity (AE)

In the Control System, a user or group whose use of a volume can be subject to quotas. Using the Control System, you can set or modify quotas that limit the space used by all the volumes owned by an accountable entity.

Related information

accounting entity (AE) on page 6593

quota on page 6599

accounting entity (AE)

In the CLI, a user or group whose use of a volume can be subject to quotas. Using the CLI, you can set or modify quotas that limit the space used by all the volumes owned by the accounting entity.

Related information

accountable entity (AE) on page 6593

quota on page 6599

administrator

A user or users with special privileges to administer the cluster or cluster resources. Administrative functions can include managing hardware resources, users, data, services, security, and availability.

For more information, see 6.1 Administration on page 752. See also *MapR user*.

advisory quota

An advisory disk capacity limit that can be set for a volume, user, or group. When disk usage exceeds the advisory quota, an alert is sent.

last-updated: Apr 11, 2022

air gap

Physical isolation between a computer system and unsecured networks. To enhance security, air-gapped computer systems are disconnected from other systems and networks.

application containers

Lightweight, stand-alone executables that include verything needed to run an application. Application containers are typically available for Linux and Windows applications.

bitmask

A binary number in which each bit controls a single toggle.

chunk

Files in the filesystem are split into chunks (similar to Hadoop blocks) that are normally 256 MB by default. Any multiple of 65,536 bytes is a valid chunk size, but tuning the size correctly is important. Files inherit the chunk size settings of the directory that contains them, as do subdirectories on which chunk size has not been explicitly set. Any files written by a Hadoop application, whether via the file APIs or over NFS, use chunk size specified by the settings for the directory where the file is written.

cluster admin

The MapR user.

For more information, see MapR user on page 6598.

coalesce

The interval of time during which READ, WRITE, or GETATTR operations on one file from one IP address or UID are logged only once for a particular operation, if auditing is enabled.
For example, suppose that a client application reads a single file three times in 6 minutes, so that there is one read at 0 minutes, another at 3 minutes, and a final read at 6 minutes. If the coalesce interval is at least 6 minutes, then only the first read operation is logged. However, if the interval is between 4 and 6 minutes, then only the first and third read operations are logged. If the interval is 2 minutes, all three read operations are logged.

Now however, if the client was also writing to the file, irrespective of the coalesce interval for the read operation in the example stated previously, the write operation is logged, as it is a different operation from reading.

container

The unit of shared storage in a MapR cluster. Every container is either a name container or a data container.
Related information
application containers on page 6594
Docker containers on page 6596
YARN resource containers on page 6602

container location database (CLDB)

A service, running on one or more MapR nodes, that maintains the locations of services, containers, and other cluster information.

Note:

The Container Location Database (CLDB) service tracks the following information about every container in the filesystem:

- The node where the container is located
- Size of the container
- The volume to which the container belongs
- The policies, quotas, and usage for that volume

custom resource (CR)

In Kubernetes, the plan or blueprint for building and maintaining an application. Custom resources are specified as `.yaml` files.

A custom resource is a valid instance of a *custom resource definition (CRD)*. Along with controllers, custom resources form a Kubernetes *operator*.

custom resource definition (CRD)

In Kubernetes, a list of valid fields that defines the shape of a custom resource (CR).

CRDs enforce validation of a *custom resource (CR)* and should not be modified. CRDs are specified as `.yaml` files.

data compaction

A process that enables users to remove empty or deleted space in the database and to compact the database to occupy contiguous space.

Related information

log compaction on page 6597

data container

One of the two types of containers in a MapR cluster. Data containers typically have a cascaded configuration (master replicates to replica1, replica1 replicates to replica2, and so on). Every data container is either a master container, an intermediate container, or a tail container depending on its replication role.

desired replication factor

The number of copies of a volume that should be maintained by the MapR cluster for normal operation.

When the number of copies falls below the desired replication factor, but remains equal to or above the *minimum replication factor*, re-replication occurs after the timeout specified in the `cldb.fs.mark.rereplicate.sec` parameter.

last-updated: Apr 11, 2022

developer preview

A label for a feature or collection of features that have usage restrictions. Developer previews are not tested for production environments, and should be used with caution.

disk space balancer

The disk space balancer is a tool that balances disk space usage on a cluster by moving containers between storage pools. Whenever a storage pool is over 70% full (or a threshold defined by the `cldb.balancer.disk.threshold.percentage` parameter), the disk space balancer distributes containers to other storage pools that have lower utilization than the average for that cluster. The disk space balancer aims to ensure that the percentage of space used on all of the disks in the node is similar.

disktab

A file on each node, containing a list of the node's disks that have been configured for use by the filesystem.

Docker containers

The application containers used by Docker software. Docker is a leading proponent of OS virtualization using application containers ("containerization").

dump file

A file containing data from a volume for distribution or restoration. There are two types of dump files: *full* dump files containing all data in a volume, and *incremental* dump files that contain changes to a volume between two points in time.

full dump file

entity

A user or group. Users and groups can represent accounting or accountable entities.
Related information
accounting entity (AE) on page 6593
accountable entity (AE) on page 6593

epoch

A sequence number that identifies all copies that have the latest updates for a container. The larger the number, the most up-to-date the copy of the container. The CLDB uses the epoch to ensure that an out-of-date copy cannot become the master for the container.

filelet

A filelet, also called an fid, is a 256MB shard of a file. A 1 GB file for instance is comprised of the following filelets: 64K (primary fid)+(256MB-64KB)+256MB+256MB+256MB.

full dump file

Related information

dump file on page 6596

HBase

A distributed storage system, designed to scale to a very large size, for managing massive amounts of structured data. Apache HBase is deprecated.

heartbeat

A signal sent by each FileServer and NFS node every second to provide information to the CLDB about the node's health and resource usage.

incremental dump file

Related information

dump file on page 6596

log compaction

A process that purges messages previously published to a topic partition, retaining the latest version.

Related information

data compaction on page 6595

MapR core

The minimum complement of software packages required to construct a MapR cluster. These packages include `mapr-core`, `mapr-core-internal`, `mapr-cldb`, `mapr-apiserver`, `mapr-fileserver`, `mapr-zookeeper`, and others. Note that ecosystem components are not part of MapR core.

To view the "core" packages on a MapR cluster, you can use the `yum list installed` command, as shown in Checking the MapR Core Version on page 5415.

MapR filesystem

The NFS-mountable, distributed, high-performance MapR data-storage system.

Related concepts

File System on page 452

Discusses the features of the MapR distributed file system and compares it to the Hadoop Distributed File System (HDFS).

MapR administrator

The "MapR user." The user that cluster services run as (typically named `mapr` or `hadoop`) on each node.

last-updated: Apr 11, 2022

See *MapR user*.

MapR Data Access Gateway

A service that acts as a proxy and gateway for translating requests between lightweight client applications and the MapR cluster.

For more information, see Administering the MapR Data Access Gateway on page 1492.

MapR Data Fabric for Kubernetes

A set of Docker containers that provide persistent storage for Kubernetes objects through the MapR Filesystem. Once the Docker containers are installed, both a Kubernetes FlexVolume Driver and a Kubernetes Dynamic Volume Provisioner are available for static and dynamic provisioning of MapR storage.

MapR Ecosystem Pack (EEP)

A selected set of stable, interoperable, and widely used components from the Hadoop Ecosystem that are fully supported on the MapR platform. EEPs can include connectors and developer APIs that provide common Hadoop Ecosystem interfaces to MapR components (for example, Kafka Connect).

MapR gateway

A gateway that supports table and stream replication. The MapR gateway mediates one-way communication between a source MapR cluster and a destination cluster. The MapR gateway also applies updates from JSON tables to their secondary indexes and propagates Change Data Capture (CDC) logs.

For more information, see Administering MapR Gateways on page 1150.

MapR user

The user that cluster services run as (typically named `mapr` or `hadoop`) on each node. The MapR user, also known as the "MapR admin," has full privileges to administer the cluster. The administrative privilege, with varying levels of control, can be assigned to other users as well.

For more information, see Managing Users and Groups on page 752.

MAST Gateway

A gateway that serves as a centralized entry point for all the operations that need to be performed on tiered storage.

For more information, see Overview of MAST Gateway on page 472.

minimum replication factor

The minimum number of copies of a volume that should be maintained by the MapR cluster for normal operation. When the replication factor falls below this minimum, re-replication occurs as aggressively as possible to restore the replication level. If any containers in the CLDB volume fall below the minimum replication factor, writes are disabled until aggressive re-replication restores the minimum level of replication.

mirror

A read-only physical copy of a volume.

name container

A container in a MapR cluster that holds a volume's namespace information and file chunk locations, and the first 64 KB of each file in the volume.

Network File System (NFS)

A protocol that allows a user on a client computer to access files over a network as though they were stored locally.

node

An individual server (physical or virtual machine) in a cluster.

NodeManager (NM)

A data service that works with the ResourceManager to host the YARN resource containers that run on each data node.

operator

In Kubernetes, a way to install and manage an application. Kubernetes operators handle not just application installation, but also the entire application lifecycle, including complex upgrades. An operator consists of a combination of two real Kubernetes objects: a controller and a custom resource.

Persistent Application Client Container (PACC)

A Docker-based application container image that includes a container-optimized MapR client. The PACC provides seamless access to cluster services, including the MapR File System, MapR Database, and MapR Event Store For Apache Kafka. The PACC makes it fast and easy to run containerized applications that access data in cluster.

quota

A disk capacity limit that can be set for a volume, user, or group. When disk usage exceeds the quota, no more data can be written.

recovery point objective (RPO)

The maximum allowable data loss as a point in time. If the recovery point objective is two hours, then the maximum allowable amount of data loss that is acceptable is two hours of work.

last-updated: Apr 11, 2022

recovery time objective (RTO)

The maximum allowable time to recovery after data loss. If the recovery time objective is five hours, then it must be possible to restore data up to the recovery point objective within five hours.

Related information

recovery point objective (RPO) on page 6599

replication factor

The number of copies of a volume.

replication role

The replication role of a container determines how that container is replicated to other storage pools in the cluster.

A *name container* may have one of two replication roles: master or replica. A *data container* may have one of three replication roles: master, intermediate, or tail.

replication role balancer

The replication role balancer is a tool that switches the replication roles of containers to ensure that every node has an equal share of of master and replica containers (for name containers) and an equal share of master, intermediate, and tail containers (for data containers).

re-replication

Re-replication occurs whenever the number of available replica containers drops below the number prescribed by that volume's replication factor. Re-replication may occur for a variety of reasons including replica container corruption, node unavailability, hard disk failure, or an increase in replication factor.

ResourceManager (RM)

A YARN service that manages cluster resources and schedules applications.

role

The service that the node runs in a cluster. You can use a node for one, or a combination of the following roles: CLDB, JobTracker, WebServer, ResourceManager, Zookeeper, FileServer, TaskTracker, NFS, and HBase.

secret

A Kubernetes object that holds sensitive information, such as passwords, tokens, and keys. Pods that require this sensitive information reference the secret in their pod definition. Secrets are the method Kubernetes uses to move sensitive data into pods.

last-updated: Apr 11, 2022

secure by default

The MapR platform and supported ecosystem components are designed to implement security unless the user takes specific steps to turn off security options.

Related concepts

Security for Ecosystem Components on page 702

Whether you install MapR software by using the MapR Installer or by using manual steps, the platform and its ecosystem components are installed with security ON by default.

schedule

A group of rules that specify recurring points in time at which certain actions are determined to occur.

snapshot

A read-only logical image of a volume at a specific point in time.

storage pool

A unit of storage made up of one or more disks. By default, MapR storage pools contain two or three disks. For high-volume reads and writes, you can create larger storage pools when initially formatting storage during cluster creation.

Note: Storage pool refers to the combined storage capacity that is obtained by combining one or more storage devices. Storage devices can be anything from a very small disk drive to large arrays of disk drives (each containing 20-30 drives).

A storage pool is created to get a very large capacity of GBs/TBs/PBs available, from which users are provided needed amounts of storage

For example, one can combine 10 hard disk drives of 4TB each, totaling to 40TBs. Now, one can either directly use the 40TB as a single device or partition the space out to many smaller storage capacities such as 100GB, 1TB and so on from this 40TB and provide that access to different users.

stripe width

The number of disks in a storage pool. See *storage pool*.

super group

The group that has administrative access to the MapR cluster.

super user

The user that has administrative access to the MapR cluster.

ticket

In the MapR platform, a file that contains keys used to authenticate users and cluster servers. Tickets are created using the `maprlogin` or `configure.sh` utilities and are encrypted to protect their contents.

last-updated: Apr 11, 2022

Different types of tickets are provided for users and services. For example, every user who wants to access a cluster must have a user ticket, and every node in a cluster must have a server ticket.

ticket secret

A Kubernetes secret that contains a ticket.

See also *secret* and *ticket*.

volume

A tree of files and directories grouped for the purpose of applying a policy or set of policies to all of them at once.

Warden

A MapR process that coordinates the starting and stopping of configured services on a node.

For more information, see Warden on page 677.

YARN resource containers

A unit of memory allocated for use by YARN to process each map or reduce task.

ZooKeeper

ZooKeeper is a coordination service for distributed applications. It provides a shared hierarchical namespace that is organized like a standard filesystem.

For more information, see ZooKeeper on page 675.

Landing Page Nav Version 2

This file is a resource file used for creating a persona based overflow for common tasks.

HPE Ezmeral Data Fabric is a platform for data-driven analytics, ML, and AI workloads. The platform serves as a secure data store and provides file storage, NoSQL databases, object storage, and event streams. The patented filesystem architecture was designed and built for performance, reliability, and scalability. Learn more

File Store

Distributed filesystem for structured and unstructured data. Patented capabilities protect data and make data storage extremely reliable and scalable. Learn more

File System on page 452 Administer Files and Directories Administrator's Reference File Store APIs

NoSQL Databases

Two NoSQL databases for analytics and operational applications - a key-value database with HBase API and a JSON document database with OJAI API. Learn more

Key-Value Database Document Database Developing Applications

Event Streams

last-updated: Apr 11, 2022

Publish-and-subscribe messaging system (built into the Data Fabric platform) that supports Kafka APIs. No additional process management required. Learn more

Supported Apache Kafka Streams APIs on page 3856 Getting Started with Event Streams Develop Streams Applications

EEP

Ezmeral Ecosystem Packs (EEP, previously MEP) provide a set of ecosystem components that work together on one or more Data Fabric cluster versions. Learn more

MapR Ecosystem Packs on page 3174 EEP Release Notes Interoperability Matrices Ecosystem Component Release Notes

Apache Hadoop

HPE Ezmeral Data Fabric provides a full Hadoop distribution that is API-compatible with all versions of Hadoop and leverages the capabilities of the HPE Ezmeral Data Fabric filesystem. Learn more

MapReduce and Applications Managing Jobs and Applications on page 1267 Copying Data from Apache Hadoop to a MapR Cluster on page 2375

Apache Spark

A multi-language engine for data engineering, data science, and machine-learning workloads. Runs on single-node machines or on multi-node clusters. Learn more

Spark Standalone on page 4028 Spark on YARN on page 4032 Spark Streaming

Apache Drill

Schema-free SQL query engine for Hadoop, NoSQL databases, cloud, and file storage. Connect to data sources and run ANSI SQL queries from the CLI, UI, or standard BI tools. Learn more

Drill Drivers on page 3333Connecting Drill to Data Sources on page 3251Optimizing Queries with Indexes

last-updated: Apr 11, 2022